Exploratory analysis

March 18, 2021

[80]:
```
%reload_ext autoreload
%autoreload 2
default_figsize=(14,12)
```

[81]:
```
import datasets
import numpy as np
import pandas as pd
import seaborn as sn
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['figure.figsize'] = (14, 12)

dataset_name = "hou"
dataset_module = datasets.datasets_by_name_all[dataset_name]
x,y,metadata = dataset_module.load(dropna=True,verbose=True)
y = datasets.map_y_em(y,dataset_name)

# generate dataframe with both x and y
xy = pd.concat([x,y],axis=1)
xy.describe()
```

```
Warning loading data from Hou2016_VPHAS-SDSS-IPHAS-2MASS.csv:
Dropped 27 rows with missing values.
Rows (original):   1034
Rows (after drop): 1007
```

[81]:

	umag	gmag	rmag	imag	Hamag
count	1007.000000	1007.000000	1007.000000	1007.000000	1007.000000
mean	17.947877	16.366036	15.557746	15.048451	15.347805
std	1.660195	1.368795	1.418495	1.370818	1.440670
min	13.616000	12.398000	12.100000	11.590000	11.450000
25%	16.505000	15.296000	14.365000	13.825000	14.125000
50%	18.217000	16.618000	15.950000	15.430000	15.750000
75%	19.226000	17.470500	16.755000	16.225000	16.560000
max	24.651000	21.633000	19.330000	18.290000	18.890000

	Jmag	Hmag	Kmag	em
count	1007.000000	1007.000000	1007.000000	1007.0

```
mean     14.248893    13.983537    13.843248    1.0
std       1.329480     1.331519     1.341729    0.0
min      10.501000     9.331000     8.578000    1.0
25%      13.083000    12.900500    12.767000    1.0
50%      14.586000    14.294000    14.133000    1.0
75%      15.405500    15.085000    14.954000    1.0
max      17.013000    16.700000    17.150000    1.0
```

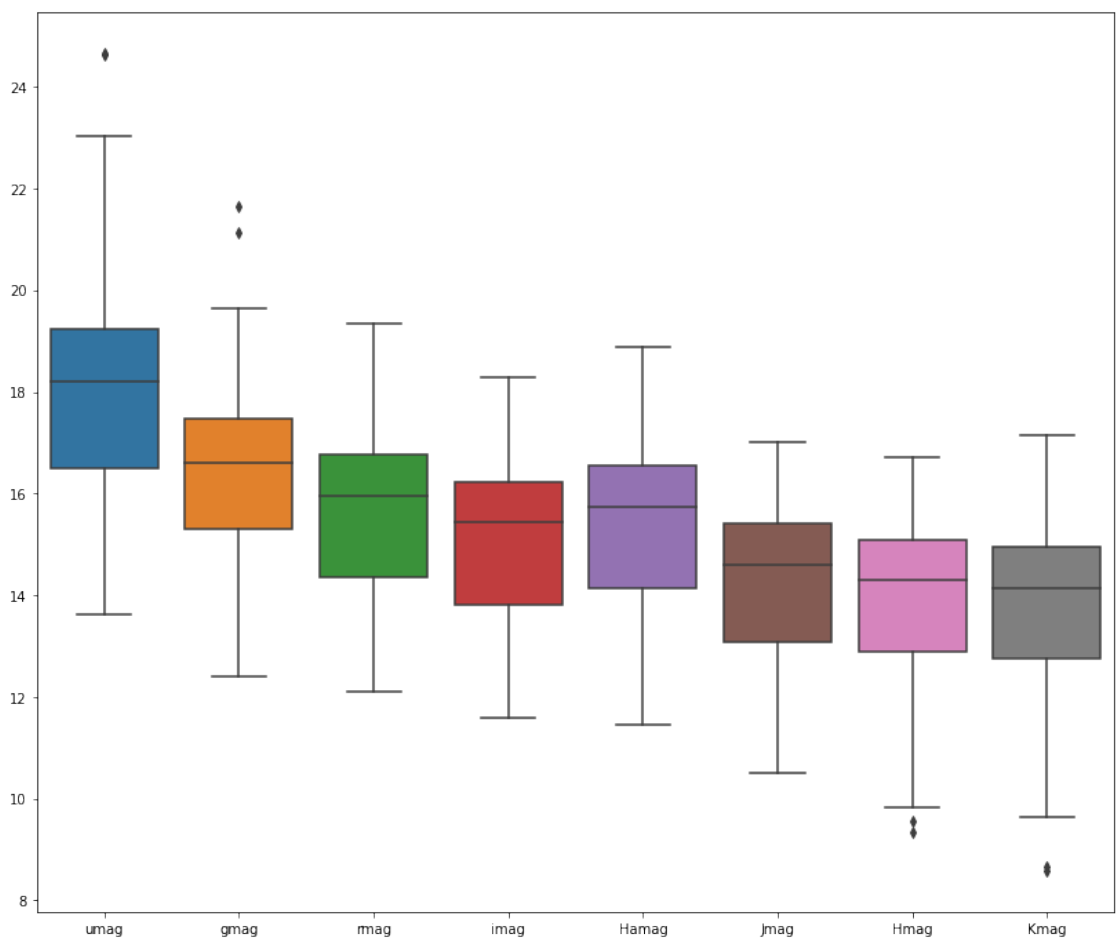
1 Variable visualization

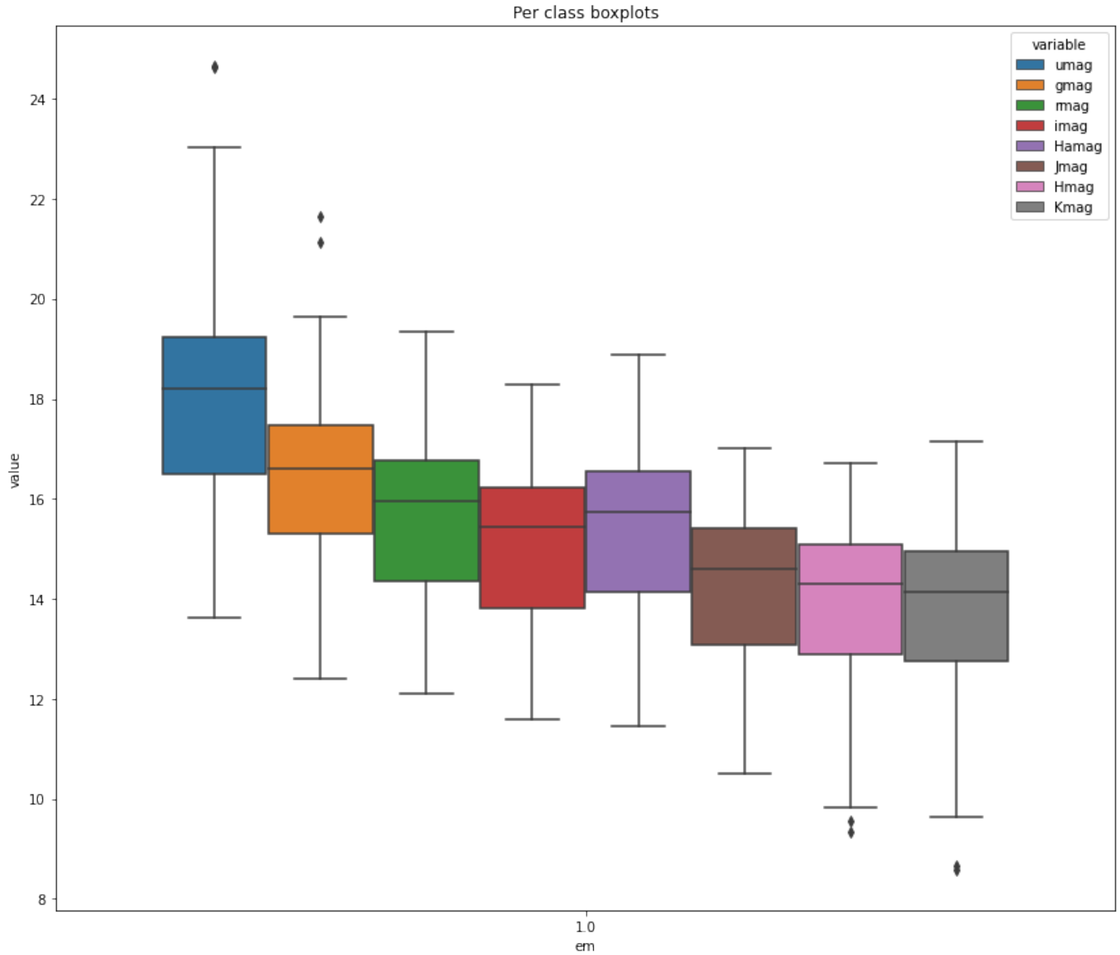
[82]:

```
sn.boxplot(data=x)

plt.figure()
xy_long = pd.melt(xy, id_vars='em')
sn.boxplot(x='em', y='value', hue='variable', data=xy_long)
plt.title("Per class boxplots")
```

[82]:

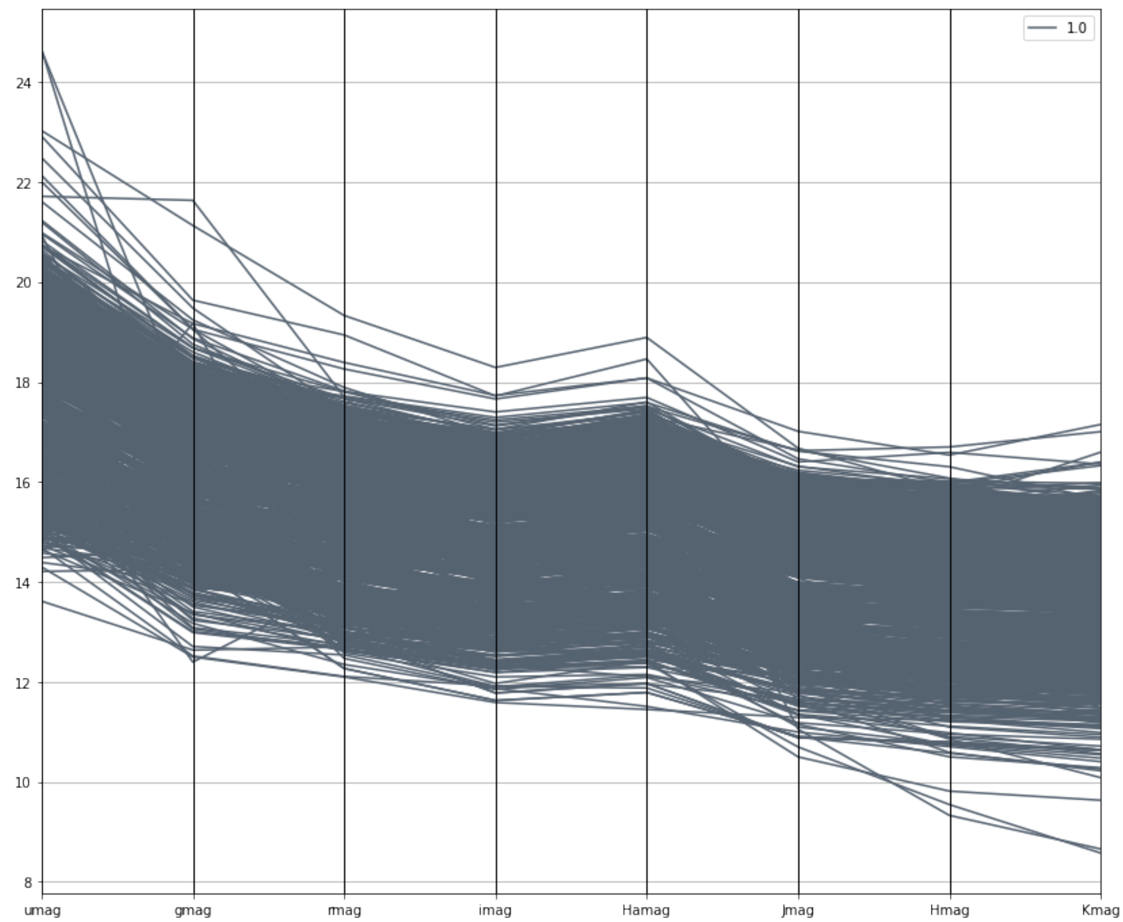
```
Text(0.5, 1.0, 'Per class boxplots')
```





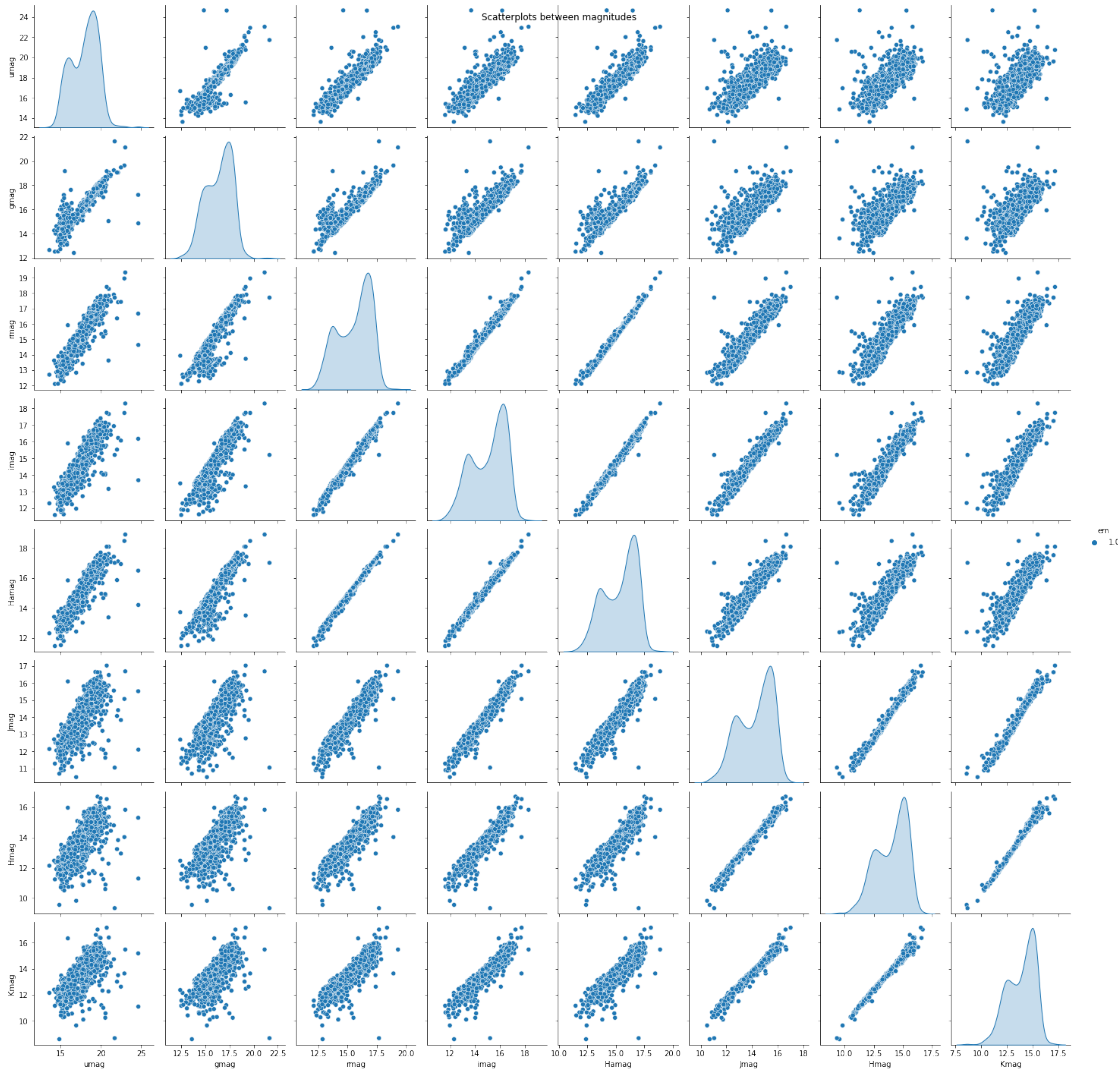
[83]:
```
pd.plotting.parallel_coordinates(xy,"em",color=('#556270','#C7F464'))
```

[83]: <AxesSubplot:>



[84]:
```
sn.heatmap(xy.corr().abs())
plt.title("Correlation between magnitudes")
plt.show()

sn.pairplot(xy,hue="em")
plt.suptitle("Scatterplots between magnitudes")
# axes=pd.plotting.scatter_matrix(x,c=y["em"],alpha=0.
 ↪9,grid=False,figsize=(14,12))
```

[84]: Text(0.5, 0.98, 'Scatterplots between magnitudes')



2 Outlier detection via confidence interval

[85]:
```
from scipy import stats
m = len(x.columns) # number of columns = number of hypothesis
confidence= 0.98
adjusted_confidence = 1- (1-confidence)/m  # bonferroni-adjusted confidence
max_zscore = stats.norm.ppf(adjusted_confidence)
print(f"Confidence  (desired): {confidence}")
print(f"Confidence (adjusted): {adjusted_confidence}")
print(f"Z-score    (adjusted): {max_zscore}")

indices = (np.abs(stats.zscore(x-x.mean())) > max_zscore).any(axis=1)
outliers_x = x[indices]
```

```python
if dataset_name != "all_em":
    outliers_metadata = metadata[indices]
    outliers_x = outliers_x.
 ↪merge(outliers_metadata,left_index=True,right_index=True)
outliers_x
```

```
Confidence  (desired): 0.98
Confidence (adjusted): 0.9975
Z-score    (adjusted): 2.807033768343811
```

[85]:

	umag	gmag	rmag	imag	Hamag	Jmag	Hmag	Kmag	Fe_type	\
72	22.918	19.636	18.94	17.72	18.46	15.065	14.027	13.628	NaN	
94	23.028	21.130	19.33	18.29	18.89	16.676	15.830	15.471	NaN	
132	24.635	17.203	16.66	16.17	16.48	15.515	15.300	15.175	NaN	
331	14.300	12.500	12.10	11.59	11.45	11.290	11.231	11.149	NaN	
622	16.941	15.160	12.83	11.97	12.42	10.501	9.816	9.634	NaN	
629	24.651	14.845	14.63	13.68	14.19	12.102	11.286	11.082	NaN	
662	14.853	13.601	12.86	12.31	12.37	10.700	9.547	8.578	NaN	
683	21.713	21.633	17.70	15.20	17.00	11.054	9.331	8.658	NaN	
691	14.690	12.519	12.11	11.92	11.96	11.622	11.604	11.546	NaN	
775	16.656	12.398	13.93	13.49	13.71	12.687	12.464	12.359	NaN	

	h_err	…	Halpha_type	w1_err	e_Hmag	e_rmag	objtype_SIMBAD	\
72	0.092	…	II	0.027	0.092	0.02	NaN	
94	0.120	…	II	0.047	0.120	0.04	NaN	
132	0.105	…	II	0.042	0.105	0.01	NaN	
331	0.022	…	II	0.023	0.022	0.00	Star	
622	0.026	…	VI	0.023	0.026	0.00	NaN	
629	0.030	…	VI	0.062	0.030	0.00	NaN	
662	0.029	…	V	0.031	0.029	0.00	NaN	
683	0.024	…	II	0.023	0.024	0.01	NaN	
691	0.025	…	II	0.022	0.025	0.00	Star in Cluster	
775	0.024	…	II	0.023	0.024	0.00	NaN	

	_RA2000	e_Kmag	k	e_gmag	_DEC2000
72	87.994208	0.080	13.628	0.029	22.254083
94	83.549950	0.146	15.471	0.033	29.150907
132	81.378157	0.118	15.175	0.005	29.639252
331	92.968107	0.017	11.149	0.001	23.729066
622	97.414520	0.026	9.634	0.005	0.917894
629	88.095161	0.044	11.082	0.004	20.697878
662	87.728220	0.020	8.578	0.003	20.246568
683	100.284660	0.023	8.658	0.047	10.402258
691	102.043940	0.024	11.546	0.002	9.644880
775	99.229276	0.024	12.359	0.002	9.463165

[10 rows x 28 columns]

3 Outlier detection via IQR

[86]:
```
iqr_factor=1.5
q25,q75=x.quantile(0.25),x.quantile(0.75)
iqr=q75-q25
min_values = q25-iqr_factor*iqr
max_values = q75+iqr_factor*iqr
# ou
indices = (np.logical_or(x<min_values,x>max_values)).any(axis=1)
outliers_x = x[indices]
if dataset_name != "all_em":
    outliers_metadata = metadata[indices]
    outliers_x = outliers_x.
 ↪merge(outliers_metadata,left_index=True,right_index=True)
outliers_x
```

[86]:
```
       umag    gmag   rmag   imag  Hamag    Jmag    Hmag    Kmag Fe_type  \
94   23.028  21.130  19.33  18.29  18.89  16.676  15.830  15.471     NaN
132  24.635  17.203  16.66  16.17  16.48  15.515  15.300  15.175     NaN
629  24.651  14.845  14.63  13.68  14.19  12.102  11.286  11.082     NaN
662  14.853  13.601  12.86  12.31  12.37  10.700   9.547   8.578     NaN
683  21.713  21.633  17.70  15.20  17.00  11.054   9.331   8.658     NaN

     h_err  … Halpha_type  w1_err  e_Hmag  e_rmag  objtype_SIMBAD  \
94   0.120  …          II   0.047   0.120    0.04             NaN
132  0.105  …          II   0.042   0.105    0.01             NaN
629  0.030  …          VI   0.062   0.030    0.00             NaN
662  0.029  …           V   0.031   0.029    0.00             NaN
683  0.024  …          II   0.023   0.024    0.01             NaN

        _RA2000  e_Kmag       k  e_gmag   _DEC2000
94    83.549950   0.146  15.471   0.033  29.150907
132   81.378157   0.118  15.175   0.005  29.639252
629   88.095161   0.044  11.082   0.004  20.697878
662   87.728220   0.020   8.578   0.003  20.246568
683  100.284660   0.023   8.658   0.047  10.402258

[5 rows x 28 columns]
```

4 Analysis of q-features (q_3) (all magnitudes)

[87]:
```
x_np=x.to_numpy()
import qfeatures
coefficients = dataset_module.coefficients
systems = dataset_module.systems
coefficients_np = np.array([coefficients[k] for k in x.columns])
```

```
systems = [systems[k] for k in x.columns]
q=qfeatures.calculate(x_np,coefficients_np,x.columns,systems,combination_size=3)
m = q.magnitudes

q_df = pd.DataFrame(m, columns = q.column_names)
q_df.describe()
```

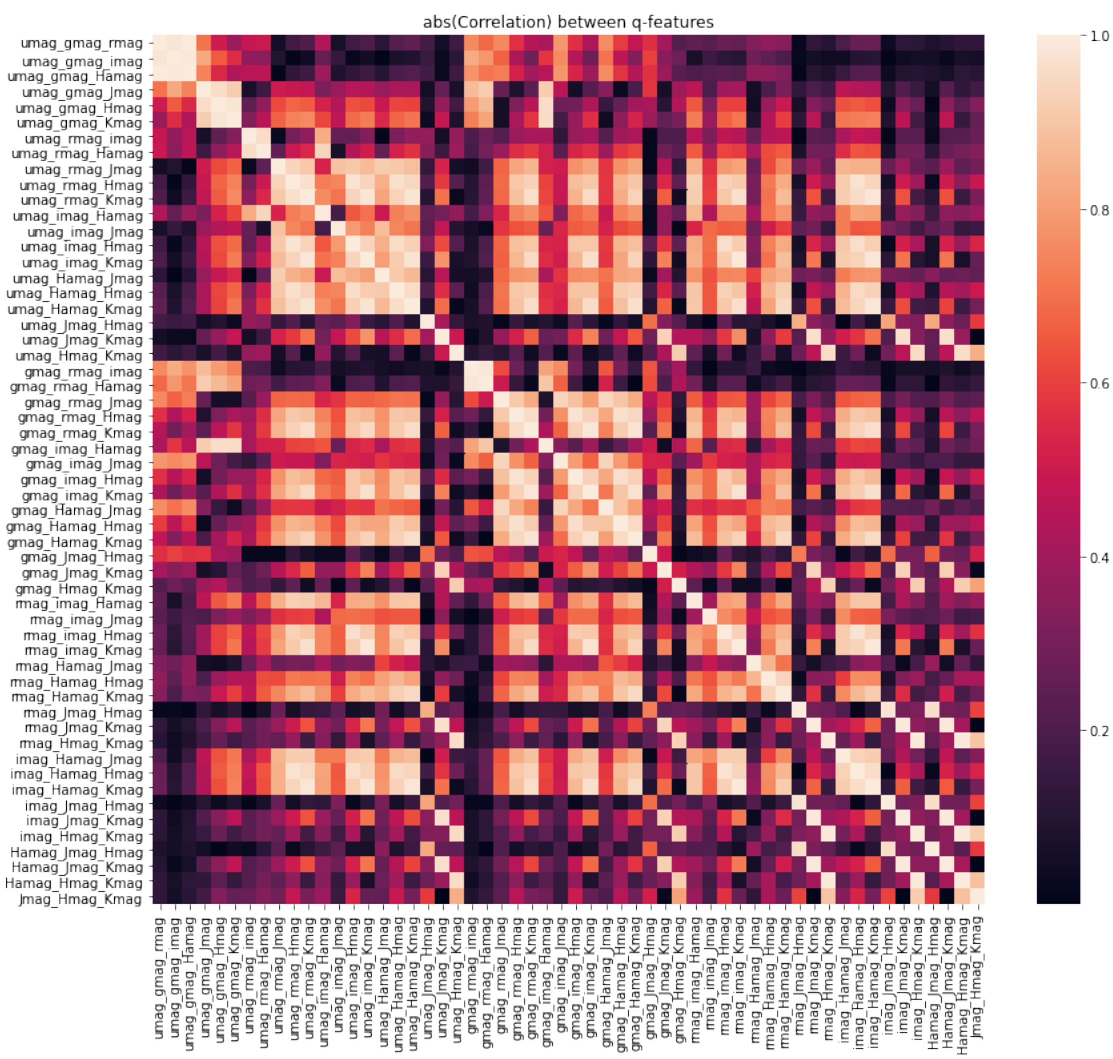
[87]:

```
       umag_gmag_rmag  umag_gmag_imag  umag_gmag_Hamag  umag_gmag_Jmag  \
count     1007.000000     1007.000000      1007.000000     1007.000000
mean         1.200440        0.741977         1.063210       -1.623278
std          0.823664        0.872624         0.843513        1.436653
min         -6.198398       -7.366959        -6.524593      -15.935431
25%          1.178662        0.741553         1.024215       -2.055917
50%          1.369199        0.963304         1.265187       -1.360764
75%          1.489747        1.051404         1.369208       -0.828021
max          9.704550        9.063398         9.472379        5.653403

       umag_gmag_Hmag  umag_gmag_Kmag  umag_rmag_imag  umag_rmag_Hamag  \
count     1007.000000     1007.000000     1007.000000      1007.000000
mean        -4.063644       -7.404559        1.770638         2.186077
std          2.257856        3.456072        0.516441         0.622566
min        -29.070391      -46.138137       -0.026327        -0.079757
25%         -4.958109       -8.856859        1.531942         1.868495
50%         -3.670370       -6.819732        1.732561         2.143327
75%         -2.723609       -5.253641        1.956249         2.447827
max          4.414391        4.119078        8.865444         9.593336

       umag_rmag_Jmag  umag_rmag_Hmag  …  imag_Hamag_Jmag  imag_Hamag_Hmag  \
count     1007.000000     1007.000000  …      1007.000000      1007.000000
mean        -1.391000       -4.728029  …         0.356941         0.975940
std          1.015766        2.267895  …         0.134663         0.385626
min        -15.186556      -33.829435  …        -0.094083        -0.067130
25%         -1.894056       -5.835587  …         0.273833         0.739326
50%         -1.311667       -4.451217  …         0.339347         0.922130
75%         -0.854889       -3.377304  …         0.421736         1.161500
max          4.667222        1.825435  …         1.751083         5.368848

       imag_Hamag_Kmag  imag_Jmag_Hmag  imag_Jmag_Kmag  imag_Hmag_Kmag  \
count      1007.000000     1007.000000     1007.000000     1007.000000
mean          1.814892        0.228467       -0.512821        0.491838
std           0.720126        0.231544        0.514158        0.610817
min          -0.639804       -0.871457       -5.255294       -2.465464
25%           1.356389        0.099630       -0.738353        0.206157
50%           1.719804        0.231152       -0.444235        0.462778
75%           2.188350        0.353446       -0.242647        0.749595
max           9.922418        1.406913        2.277294        5.374222
```

	Hamag_Jmag_Hmag	Hamag_Jmag_Kmag	Hamag_Hmag_Kmag	Jmag_Hmag_Kmag
count	1007.000000	1007.000000	1007.000000	1007.000000
mean	0.279772	-0.783490	0.594054	0.146156
std	0.341645	0.754679	0.813179	0.185327
min	-1.889261	-8.177190	-3.191216	-0.707641
25%	0.092609	-1.085876	0.228765	0.047830
50%	0.280348	-0.676980	0.558020	0.132556
75%	0.472978	-0.394304	0.937892	0.221882
max	1.874522	3.231967	7.073667	1.382222

[8 rows x 56 columns]

[88]:
```python
sn.heatmap(q_df.corr().abs())
plt.title("abs(Correlation) between q-features")
plt.show()
```



5 Analysis of q-features (q_3) (calculated by system)

[89]:
```
x_np=x.to_numpy()
import qfeatures
coefficients = dataset_module.coefficients
systems = dataset_module.systems
coefficients_np = np.array([coefficients[k] for k in x.columns])
systems = [systems[k] for k in x.columns]
q= qfeatures.calculate(x_np,coefficients_np,x.
 ↪columns,systems,combination_size=3,by_system=True)

m = q.magnitudes

q_df = pd.DataFrame(m, columns = q.column_names)
q_df.describe()
```

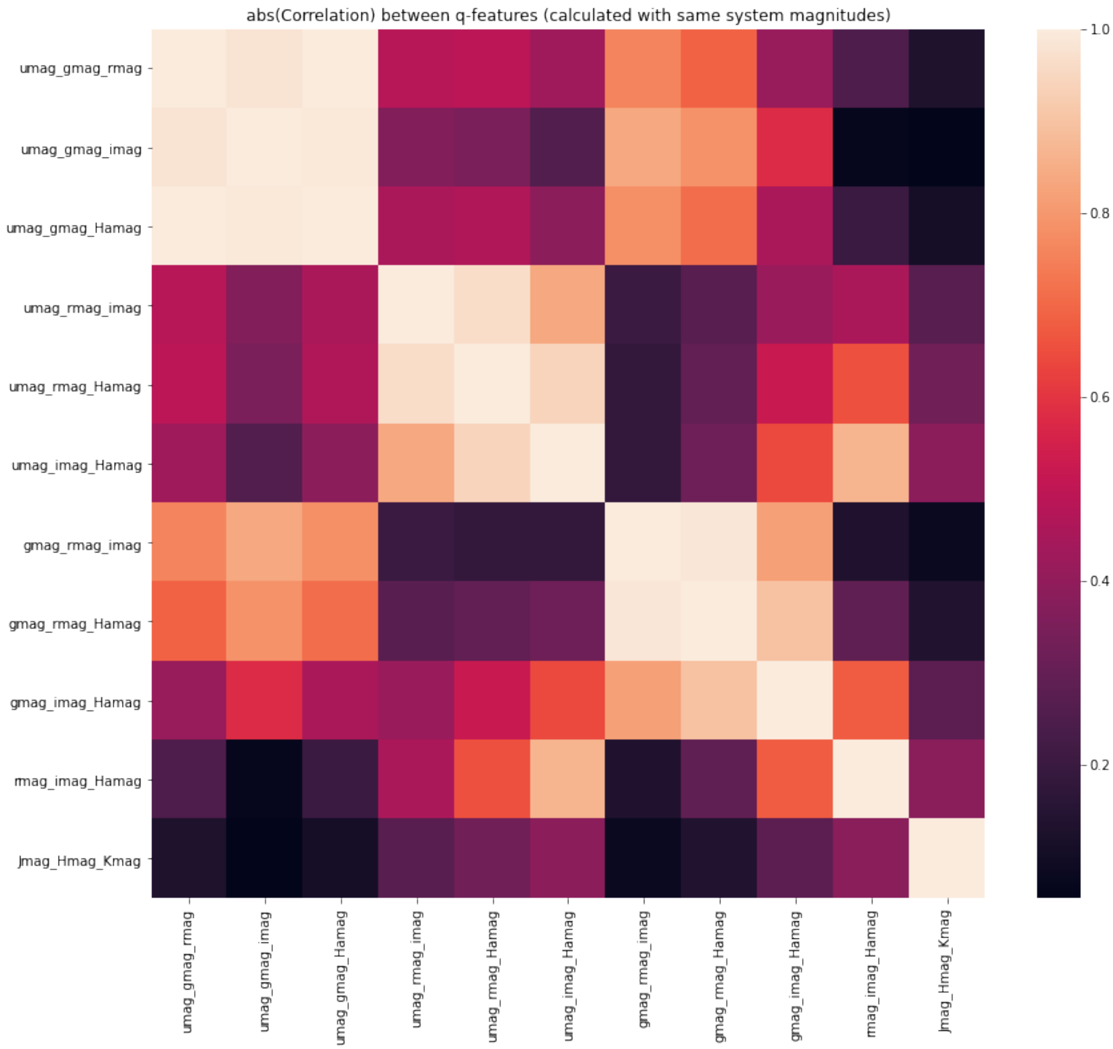
[89]:

	umag_gmag_rmag	umag_gmag_imag	umag_gmag_Hamag	umag_rmag_imag
count	1007.000000	1007.000000	1007.000000	1007.000000
mean	1.200440	0.741977	1.063210	1.770638
std	0.823664	0.872624	0.843513	0.516441
min	-6.198398	-7.366959	-6.524593	-0.026327
25%	1.178662	0.741553	1.024215	1.531942
50%	1.369199	0.963304	1.265187	1.732561
75%	1.489747	1.051404	1.369208	1.956249
max	9.704550	9.063398	9.472379	8.865444

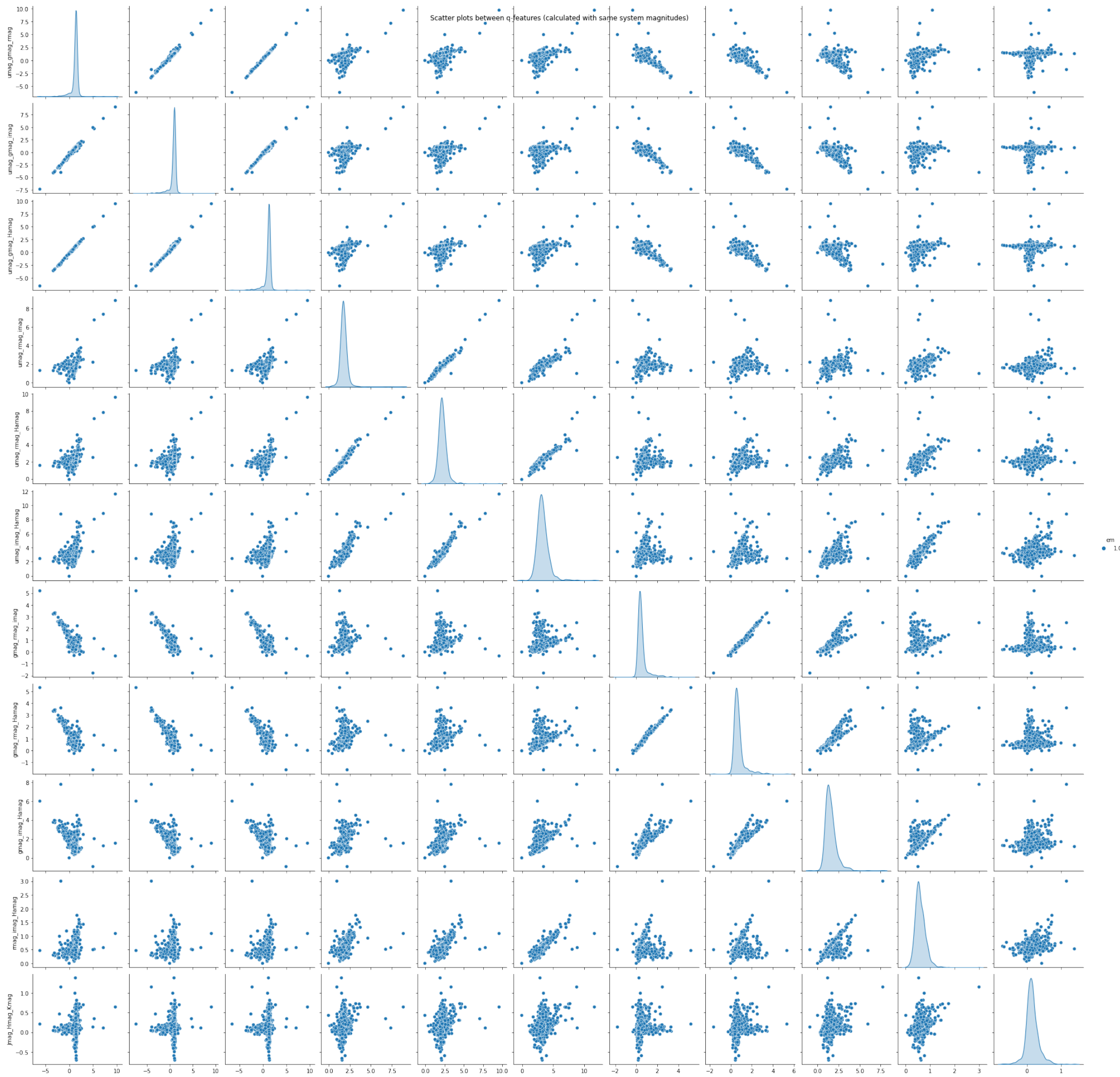
	umag_rmag_Hamag	umag_imag_Hamag	gmag_rmag_imag	gmag_rmag_Hamag
count	1007.000000	1007.000000	1007.000000	1007.000000
mean	2.186077	3.274319	0.513435	0.711168
std	0.622566	0.933284	0.502015	0.513900
min	-0.079757	-0.057140	-1.786737	-1.633776
25%	1.868495	2.703061	0.264421	0.430860
50%	2.143327	3.166897	0.384263	0.595234
75%	2.447827	3.692626	0.550079	0.796355
max	9.593336	11.609692	5.197842	5.329972

	gmag_imag_Hamag	rmag_imag_Hamag	Jmag_Hmag_Kmag
count	1007.000000	1007.000000	1007.000000
mean	1.540003	0.593226	0.146156
std	0.671485	0.221377	0.185327
min	-0.928542	0.003178	-0.707641
25%	1.103883	0.454486	0.047830
50%	1.399318	0.556916	0.132556
75%	1.821056	0.708364	0.221882
max	7.770383	3.004673	1.382222

[90]:
```
sn.heatmap(q_df.corr().abs())
plt.title("abs(Correlation) between q-features (calculated with same system␣
 ↪magnitudes)")
plt.show()
q_dfy=pd.concat([q_df,y],axis=1)
sn.pairplot(q_dfy,hue="em")
plt.suptitle("Scatter plots between q-features (calculated with same system␣
 ↪magnitudes)")
```



[90]: Text(0.5, 0.98, 'Scatter plots between q-features (calculated with same system magnitudes)')



Scatter plots between q-features (calculated with same system magnitudes)
umag_gmag_rmag
umag_gmag_imag
umag_gmag_Hamag
umag_rmag_imag
umag_rmag_Hamag
umag_imag_Hamag
gmag_rmag_imag
gmag_rmag_Hamag
gmag_imag_Hamag
rmag_imag_Hamag
jmag_Hmag_Kmag
em
1.0