Exploratory analysis

March 18, 2021

[116]:
```
%reload_ext autoreload
%autoreload 2
default_figsize=(14,12)
```

[117]:
```
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 = "all_em"
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 Mohr-Smith_2017.csv:
Dropped 38 rows with missing values.
Rows (original):   5915
Rows (after drop): 5877
Warning loading data from McSwain2005-2009_VPHAS-2MASS.csv:
Dropped 2313 rows with missing values.
Rows (original):   5455
Rows (after drop): 3142
Warning loading data from Hou2016_VPHAS-SDSS-IPHAS-2MASS.csv:
Dropped 27 rows with missing values.
Rows (original):   1034
Rows (after drop): 1007
```

[117]:

	umag	gmag	rmag	imag	Hamag	\
count	10307.000000	10307.000000	10307.000000	10307.000000	10307.000000	
mean	16.952936	16.452857	15.257591	14.554945	14.953842	
std	2.135863	1.835733	1.548815	1.438887	1.522385	

min	10.980000	10.130000	11.820000	11.081000	11.160000
25%	15.340000	15.050000	14.020000	13.400000	13.730000
50%	16.940000	16.420000	15.240000	14.534000	14.940000
75%	18.625000	17.730000	16.404500	15.633000	16.100000
max	24.651000	21.633000	19.330000	18.290000	18.890000

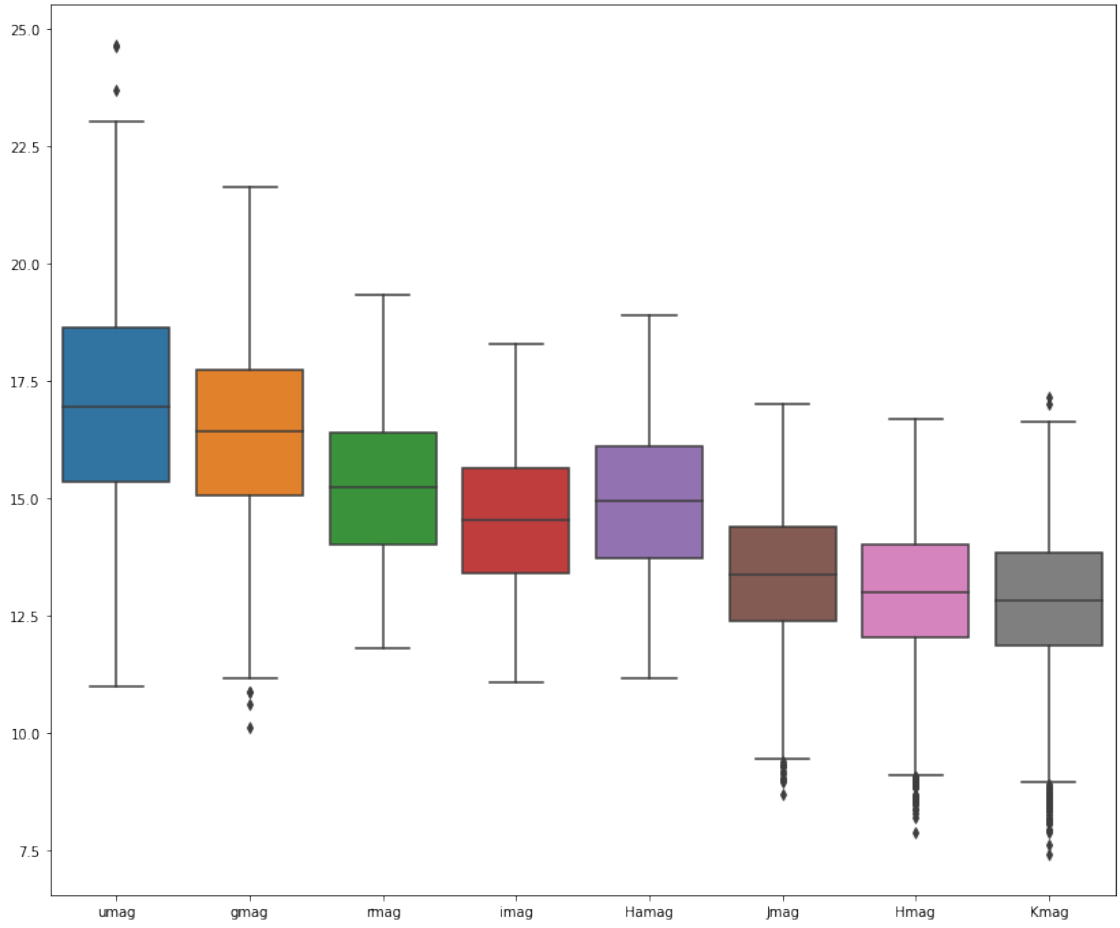
	Jmag	Hmag	Kmag	em
count	10307.000000	10307.000000	10307.000000	10307.000000
mean	13.386535	12.990008	12.802591	0.139129
std	1.354593	1.371860	1.394660	0.346098
min	8.693000	7.870000	7.414000	0.000000
25%	12.396000	12.040000	11.868000	0.000000
50%	13.374000	13.010000	12.833000	0.000000
75%	14.386000	14.003000	13.831500	0.000000
max	17.013000	16.700000	17.150000	1.000000

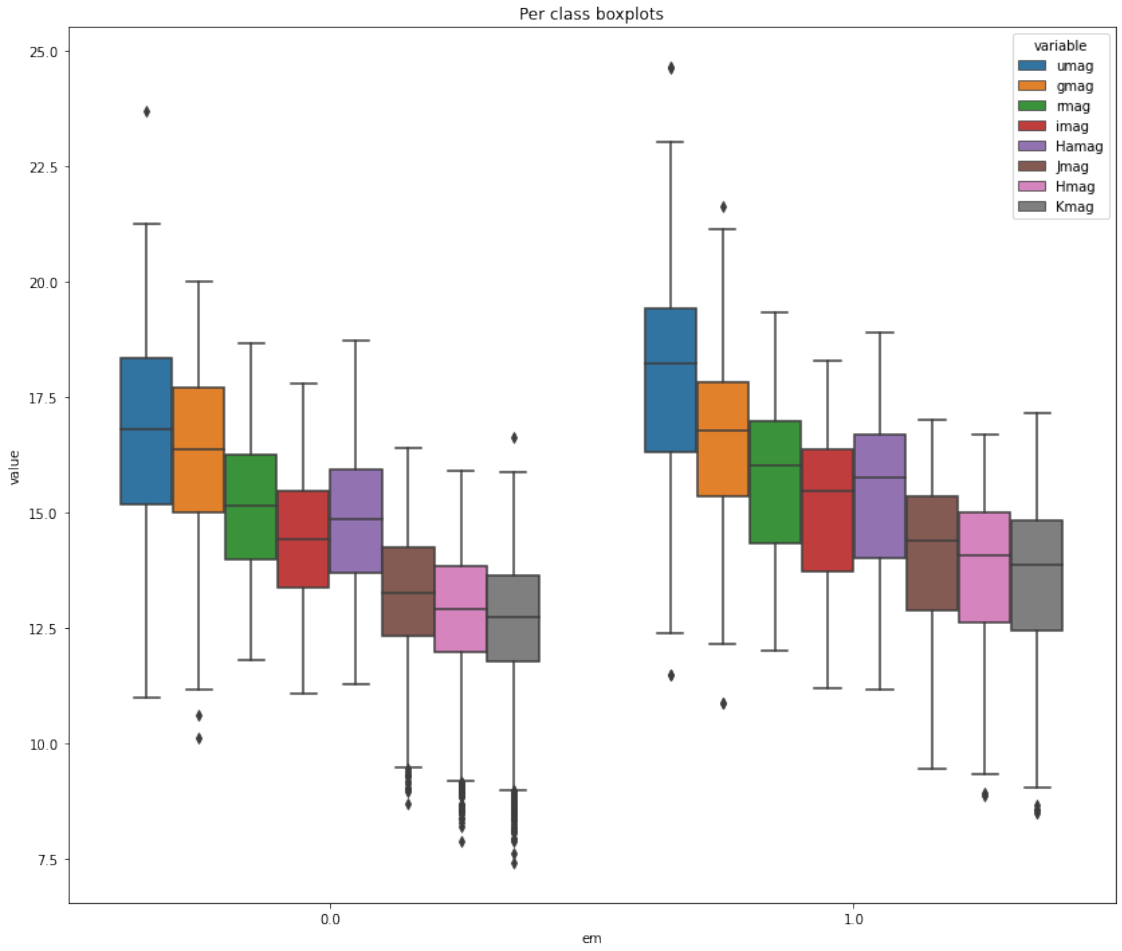
1 Variable visualization

[118]:
```
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")
```

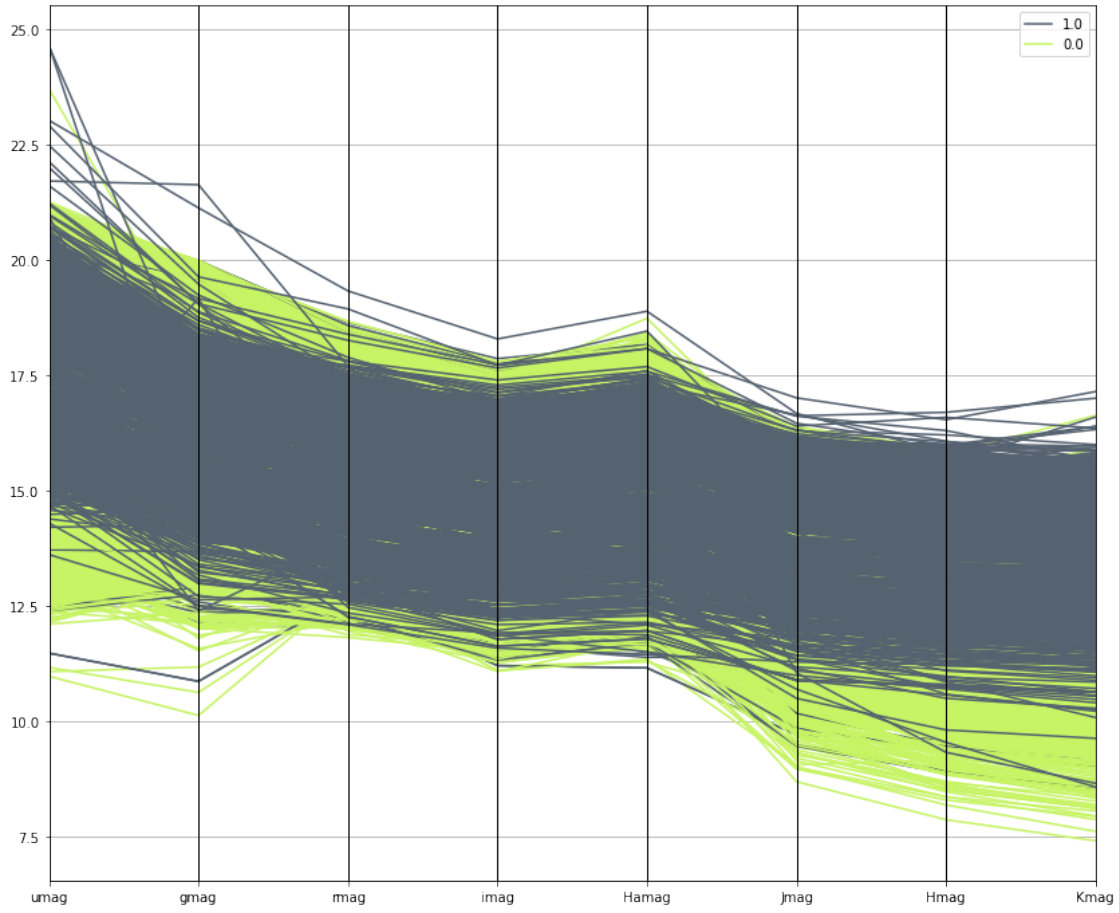
[118]:
```
Text(0.5, 1.0, 'Per class boxplots')
```


25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
umag
gmag
rmag
imag
Hamag
Jmag
Hmag
Kmag



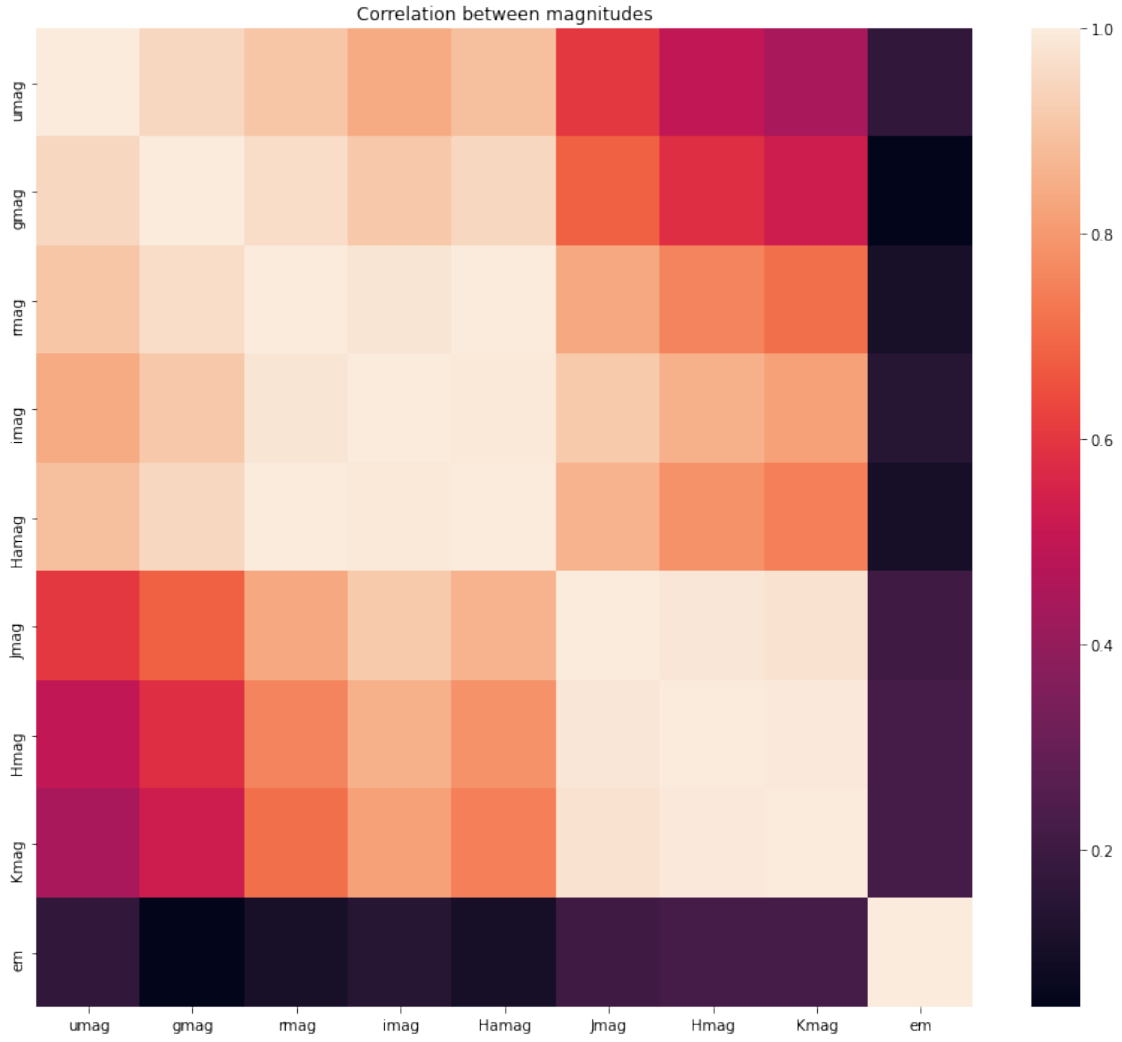
[119]:
```
pd.plotting.parallel_coordinates(xy,"em",color=('#556270','#C7F464'))
```

[119]: <AxesSubplot:>

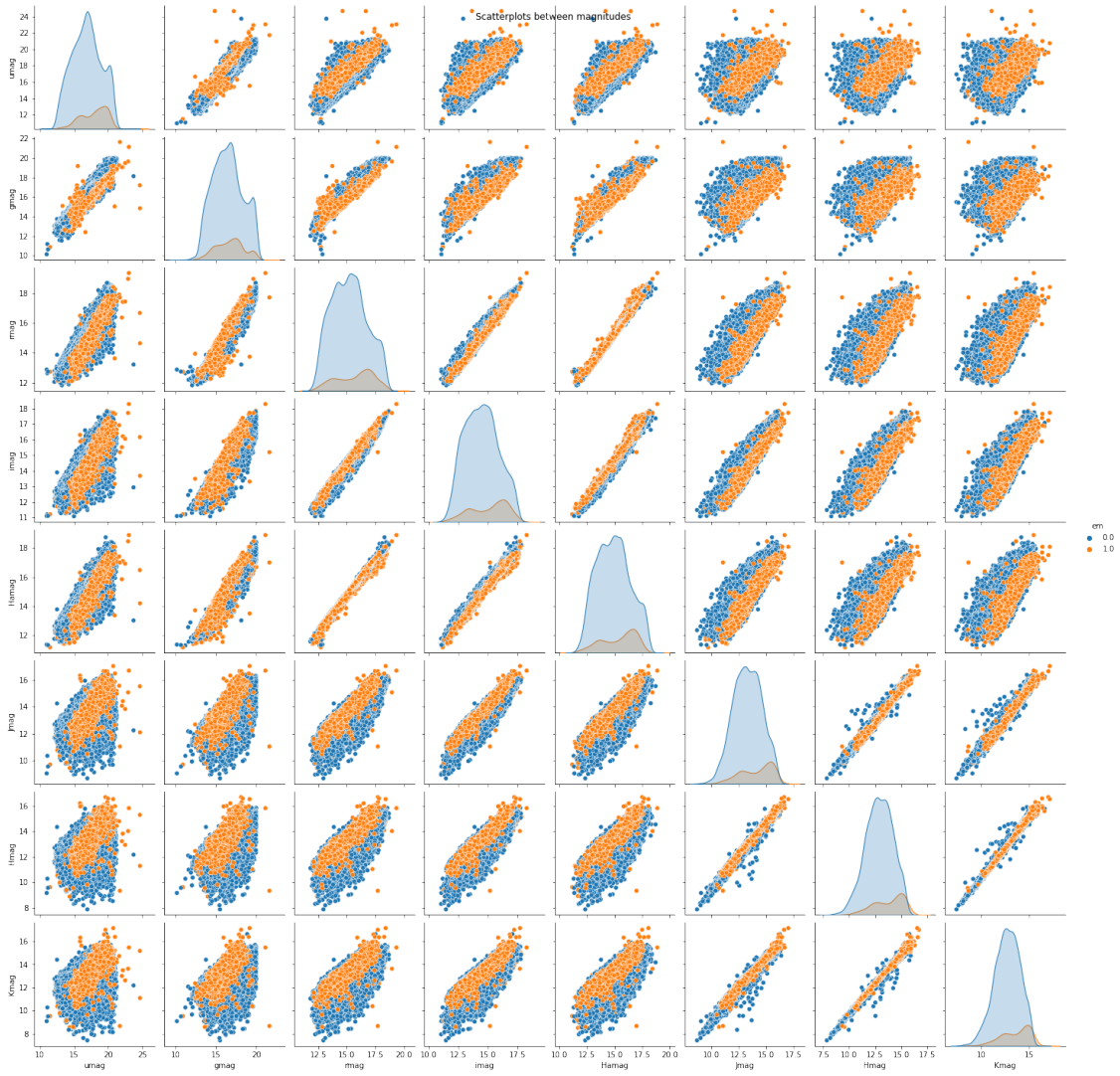


[120]:
```
sn.heatmap(xy.corr().abs())
plt.title("Correlation between magnitudes")
plt.show()

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))
```



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



2 Outlier detection via confidence interval

[121]:
```
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]
```

```
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
```

[121]:

	umag	gmag	rmag	imag	Hamag	Jmag	Hmag	Kmag
40	23.710	18.130	13.210	12.940	13.010	12.250	12.170	12.180
43	11.080	11.180	12.630	11.210	11.290	10.360	10.280	10.270
76	10.980	10.130	12.870	11.100	11.350	9.060	9.160	9.080
89	11.480	10.870	12.730	11.210	11.160	9.710	9.610	9.520
90	11.480	10.870	12.730	11.210	11.160	9.710	9.610	9.520
144	11.170	10.630	12.740	11.110	11.320	9.570	9.560	9.490
264	19.450	17.230	14.220	12.360	13.630	9.650	8.910	8.510
275	17.560	16.460	13.810	12.200	13.270	9.780	9.140	8.750
299	17.129	15.965	13.610	12.248	13.113	9.942	9.267	8.845
314	17.230	15.975	13.523	12.100	13.031	9.670	8.930	8.536
321	17.566	15.969	13.405	11.925	12.928	9.317	8.560	8.115
327	17.929	16.214	13.487	11.917	12.970	9.144	8.367	7.875
338	17.728	16.367	13.978	12.543	13.478	10.023	9.224	8.802
379	16.304	15.184	12.878	11.698	12.404	9.383	8.702	8.270
412	16.936	15.480	12.953	11.486	12.341	8.693	7.870	7.414
433	20.597	18.508	15.374	13.514	14.826	10.056	9.057	8.424
466	16.588	15.619	13.410	12.131	12.934	9.722	9.025	8.567
522	17.558	16.161	13.570	12.095	13.049	9.327	8.474	7.953
582	14.975	14.257	12.236	11.081	11.762	8.968	8.297	7.940
587	14.794	14.270	12.475	11.404	12.034	9.516	8.969	8.667
718	19.141	17.146	14.245	12.551	13.746	9.479	8.656	8.113
736	15.700	14.928	12.883	11.681	12.432	9.315	8.620	8.200
750	17.844	16.488	14.034	12.574	13.513	9.870	9.100	8.600
830	14.600	14.320	12.575	11.566	12.225	9.273	8.673	8.371
839	14.593	14.293	12.563	11.542	12.169	9.450	8.950	8.520
1401	14.979	14.243	12.437	11.366	11.680	9.488	8.925	8.498
1994	15.168	14.395	12.444	11.318	11.885	9.005	8.365	8.059
2064	14.353	13.961	12.394	11.538	11.903	9.465	8.881	8.555
2218	17.744	16.371	13.813	12.309	13.302	9.674	8.837	8.406
2526	16.736	16.161	13.962	12.592	13.489	9.756	8.880	8.334
2642	15.297	14.605	12.680	11.556	12.232	9.689	9.120	8.725
5034	16.710	15.442	13.140	11.778	12.642	9.577	8.856	8.469
5451	16.262	15.748	13.700	12.411	13.154	9.975	9.167	8.738
5640	16.351	15.423	13.115	11.700	12.603	9.018	8.190	7.617
5844	14.949	14.260	12.473	11.405	12.035	9.485	8.892	8.531
5886	15.297	14.487	12.594	11.352	12.067	9.205	8.511	8.126

6003	20.484	18.317	15.149	13.372	14.604	9.970	9.007	8.402
7430	16.870	14.440	12.510	11.620	12.130	9.924	9.015	8.747
7620	19.190	15.990	13.720	12.080	13.140	9.836	8.866	8.505
8152	18.690	15.600	13.520	12.090	13.010	10.053	9.082	8.692
8321	20.270	16.750	14.240	12.350	13.660	9.725	8.603	8.161
8641	14.210	13.950	13.340	12.930	13.150	12.348	9.971	8.329
8841	18.880	15.780	13.350	12.230	12.900	9.957	8.990	8.636
8876	20.660	16.880	14.000	12.590	13.460	9.700	8.521	8.118
9173	20.290	16.890	14.100	12.640	13.600	10.106	9.070	8.703
9186	20.140	16.660	14.010	12.480	13.430	9.984	8.943	8.580
9394	23.028	21.130	19.330	18.290	18.890	16.676	15.830	15.471
9432	24.635	17.203	16.660	16.170	16.480	15.515	15.300	15.175
9610	19.594	18.110	17.700	17.240	17.510	16.625	16.700	17.006
9854	20.728	19.168	18.390	17.730	18.080	17.013	16.539	17.150
9929	24.651	14.845	14.630	13.680	14.190	12.102	11.286	11.082
9962	14.853	13.601	12.860	12.310	12.370	10.700	9.547	8.578
9983	21.713	21.633	17.700	15.200	17.000	11.054	9.331	8.658

3 Outlier detection via IQR

[122]:
```
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
```

[122]:

	umag	gmag	rmag	imag	Hamag	Jmag	Hmag	Kmag
40	23.710	18.130	13.210	12.940	13.010	12.250	12.170	12.180
76	10.980	10.130	12.870	11.100	11.350	9.060	9.160	9.080
89	11.480	10.870	12.730	11.210	11.160	9.710	9.610	9.520
90	11.480	10.870	12.730	11.210	11.160	9.710	9.610	9.520
144	11.170	10.630	12.740	11.110	11.320	9.570	9.560	9.490
264	19.450	17.230	14.220	12.360	13.630	9.650	8.910	8.510
275	17.560	16.460	13.810	12.200	13.270	9.780	9.140	8.750
299	17.129	15.965	13.610	12.248	13.113	9.942	9.267	8.845
314	17.230	15.975	13.523	12.100	13.031	9.670	8.930	8.536
321	17.566	15.969	13.405	11.925	12.928	9.317	8.560	8.115
327	17.929	16.214	13.487	11.917	12.970	9.144	8.367	7.875

338	17.728	16.367	13.978	12.543	13.478	10.023	9.224	8.802
379	16.304	15.184	12.878	11.698	12.404	9.383	8.702	8.270
412	16.936	15.480	12.953	11.486	12.341	8.693	7.870	7.414
433	20.597	18.508	15.374	13.514	14.826	10.056	9.057	8.424
466	16.588	15.619	13.410	12.131	12.934	9.722	9.025	8.567
522	17.558	16.161	13.570	12.095	13.049	9.327	8.474	7.953
582	14.975	14.257	12.236	11.081	11.762	8.968	8.297	7.940
587	14.794	14.270	12.475	11.404	12.034	9.516	8.969	8.667
718	19.141	17.146	14.245	12.551	13.746	9.479	8.656	8.113
736	15.700	14.928	12.883	11.681	12.432	9.315	8.620	8.200
750	17.844	16.488	14.034	12.574	13.513	9.870	9.100	8.600
830	14.600	14.320	12.575	11.566	12.225	9.273	8.673	8.371
839	14.593	14.293	12.563	11.542	12.169	9.450	8.950	8.520
1401	14.979	14.243	12.437	11.366	11.680	9.488	8.925	8.498
1994	15.168	14.395	12.444	11.318	11.885	9.005	8.365	8.059
2064	14.353	13.961	12.394	11.538	11.903	9.465	8.881	8.555
2218	17.744	16.371	13.813	12.309	13.302	9.674	8.837	8.406
2526	16.736	16.161	13.962	12.592	13.489	9.756	8.880	8.334
2642	15.297	14.605	12.680	11.556	12.232	9.689	9.120	8.725
4675	16.992	15.882	13.628	12.295	13.172	10.076	9.316	8.894
5034	16.710	15.442	13.140	11.778	12.642	9.577	8.856	8.469
5085	14.948	14.293	12.512	11.497	12.127	9.699	9.197	8.898
5451	16.262	15.748	13.700	12.411	13.154	9.975	9.167	8.738
5640	16.351	15.423	13.115	11.700	12.603	9.018	8.190	7.617
5844	14.949	14.260	12.473	11.405	12.035	9.485	8.892	8.531
5886	15.297	14.487	12.594	11.352	12.067	9.205	8.511	8.126
6003	20.484	18.317	15.149	13.372	14.604	9.970	9.007	8.402
7430	16.870	14.440	12.510	11.620	12.130	9.924	9.015	8.747
7620	19.190	15.990	13.720	12.080	13.140	9.836	8.866	8.505
8152	18.690	15.600	13.520	12.090	13.010	10.053	9.082	8.692
8321	20.270	16.750	14.240	12.350	13.660	9.725	8.603	8.161
8641	14.210	13.950	13.340	12.930	13.150	12.348	9.971	8.329
8841	18.880	15.780	13.350	12.230	12.900	9.957	8.990	8.636
8876	20.660	16.880	14.000	12.590	13.460	9.700	8.521	8.118
9173	20.290	16.890	14.100	12.640	13.600	10.106	9.070	8.703
9186	20.140	16.660	14.010	12.480	13.430	9.984	8.943	8.580
9432	24.635	17.203	16.660	16.170	16.480	15.515	15.300	15.175
9610	19.594	18.110	17.700	17.240	17.510	16.625	16.700	17.006
9854	20.728	19.168	18.390	17.730	18.080	17.013	16.539	17.150
9929	24.651	14.845	14.630	13.680	14.190	12.102	11.286	11.082
9962	14.853	13.601	12.860	12.310	12.370	10.700	9.547	8.578
9983	21.713	21.633	17.700	15.200	17.000	11.054	9.331	8.658

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

[123]:
```
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()
```

[123]:

	umag_gmag_rmag	umag_gmag_imag	umag_gmag_Hamag	umag_gmag_Jmag	\
count	10307.000000	10307.000000	10307.000000	10307.000000	
mean	-0.063921	-0.709701	-0.263438	-4.141992	
std	0.658024	0.703104	0.668365	1.920013	
min	-6.198398	-7.366959	-6.524593	-15.935431	
25%	-0.483868	-1.148602	-0.695596	-5.547403	
50%	-0.202892	-1.020696	-0.427495	-4.001958	
75%	0.079946	-0.450000	-0.102033	-2.759167	
max	9.704550	9.063398	9.472379	5.653403	

	umag_gmag_Hmag	umag_gmag_Kmag	umag_rmag_imag	umag_rmag_Hamag	\
count	10307.000000	10307.000000	10307.000000	10307.000000	
mean	-7.705369	-12.502502	0.840664	1.400112	
std	3.425802	5.545913	0.771068	0.943564	
min	-29.070391	-46.138137	-4.042982	-3.367383	
25%	-10.231500	-16.576010	0.330058	0.764346	
50%	-7.360348	-11.786373	0.722339	1.295589	
75%	-5.125587	-8.249484	1.310099	1.996963	
max	4.414391	4.119078	10.171579	10.305607	

	umag_rmag_Jmag	umag_rmag_Hmag	…	imag_Hamag_Jmag	imag_Hamag_Hmag	\
count	10307.000000	10307.000000	…	10307.000000	10307.000000	
mean	-3.709929	-8.558077	…	0.537134	1.436861	
std	1.897399	3.997383	…	0.273023	0.694547	
min	-15.186556	-33.829435	…	-0.156500	-0.411696	
25%	-5.096611	-11.436261	…	0.314993	0.895750	
50%	-3.488778	-8.058565	…	0.497042	1.315304	
75%	-2.278667	-5.460239	…	0.729924	1.944413	
max	7.726667	5.797391	…	1.753681	5.368848	

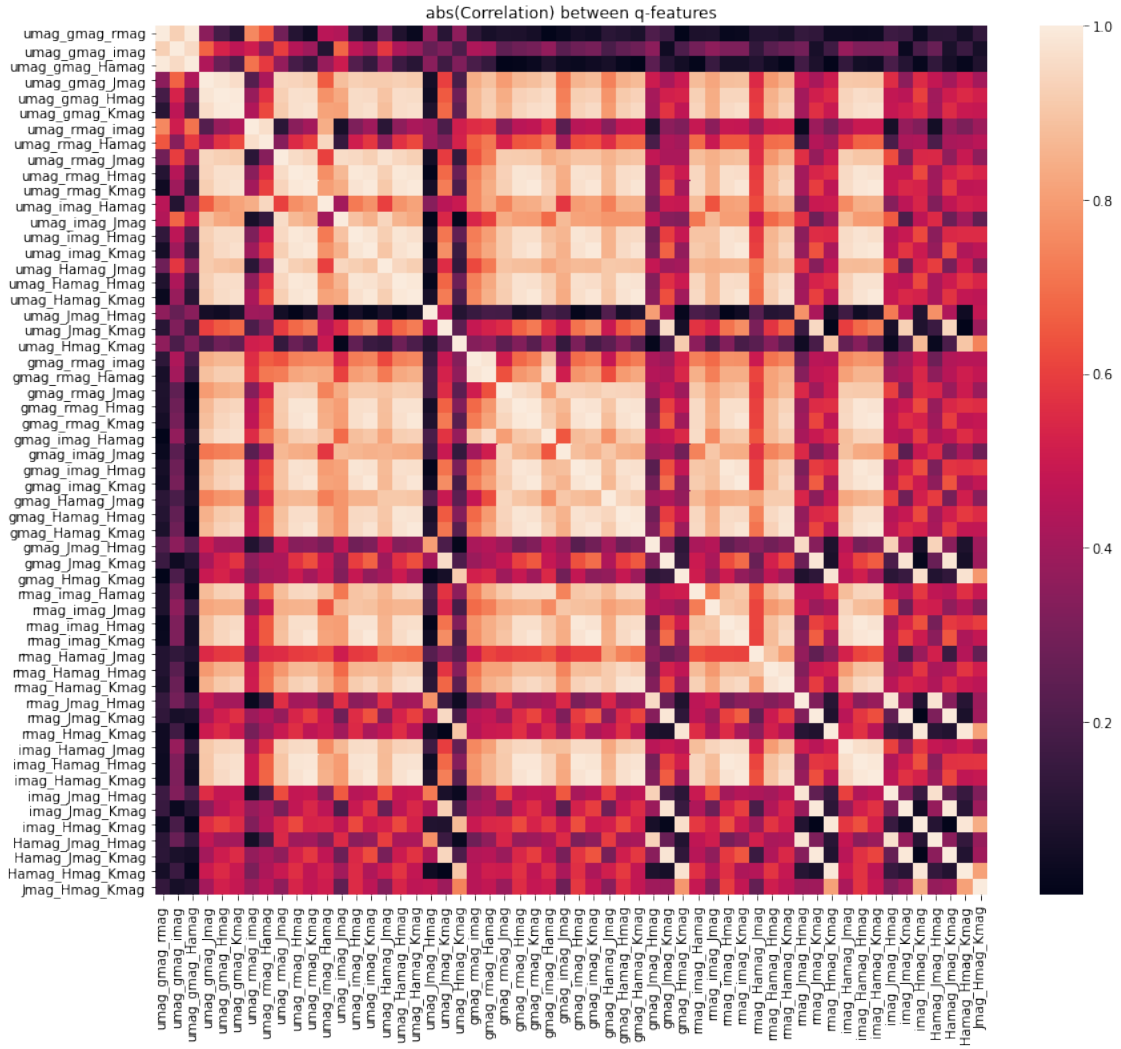
	imag_Hamag_Kmag	imag_Jmag_Hmag	imag_Jmag_Kmag	imag_Hmag_Kmag	\
count	10307.000000	10307.000000	10307.000000	10307.000000	
mean	2.624102	0.315014	-0.720819	0.799348	

```
std          1.275666       0.375150       0.627362       0.622198
min         -0.772131      -6.482065     -13.751588     -11.218902
25%          1.635833       0.089261      -0.989706       0.460582
50%          2.406157       0.310435      -0.644588       0.783667
75%          3.540333       0.544804      -0.366000       1.161458
max          9.922418       4.936783       4.142765       7.507065

       Hamag_Jmag_Hmag  Hamag_Jmag_Kmag  Hamag_Hmag_Kmag  Jmag_Hmag_Kmag
count     10307.000000     10307.000000     10307.000000     10307.000000
mean          0.343245        -1.142496         0.934883         0.237285
std           0.506861         0.925879         0.801468         0.194781
min          -9.400609       -19.827601       -15.024588        -2.622131
25%           0.064652        -1.545882         0.516039         0.126850
50%           0.348696        -1.020052         0.925157         0.223863
75%           0.641043        -0.602605         1.386402         0.336935
max           6.374304         5.383026         9.521608         2.715843

[8 rows x 56 columns]
```

[124]:
```
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)

[125]:
```
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()
```

[125]:

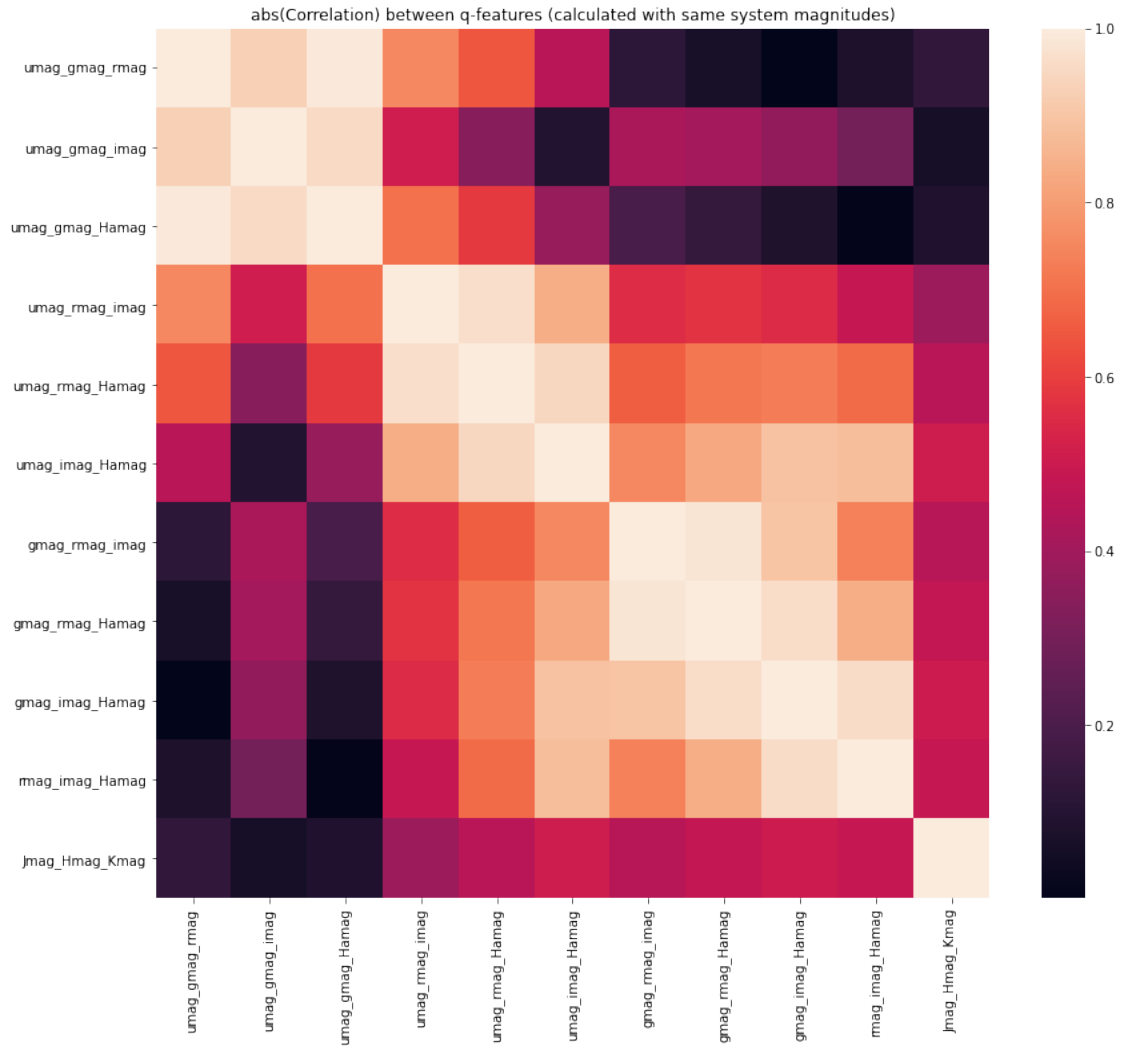
	umag_gmag_rmag	umag_gmag_imag	umag_gmag_Hamag	umag_rmag_imag
count	10307.000000	10307.000000	10307.000000	10307.000000
mean	-0.063921	-0.709701	-0.263438	0.840664
std	0.658024	0.703104	0.668365	0.771068
min	-6.198398	-7.366959	-6.524593	-4.042982
25%	-0.483868	-1.148602	-0.695596	0.330058
50%	-0.202892	-1.020696	-0.427495	0.722339
75%	0.079946	-0.450000	-0.102033	1.310099
max	9.704550	9.063398	9.472379	10.171579

	umag_rmag_Hamag	umag_imag_Hamag	gmag_rmag_imag	gmag_rmag_Hamag
count	10307.000000	10307.000000	10307.000000	10307.000000
mean	1.400112	2.897545	0.788471	1.054746
std	0.943564	1.441601	0.397424	0.500724
min	-3.367383	-1.556037	-3.764737	-3.443178
25%	0.764346	1.904645	0.504553	0.697477
50%	1.295589	2.703551	0.756842	0.998224
75%	1.996963	3.818360	1.066368	1.407572
max	10.305607	11.609692	5.197842	5.329972

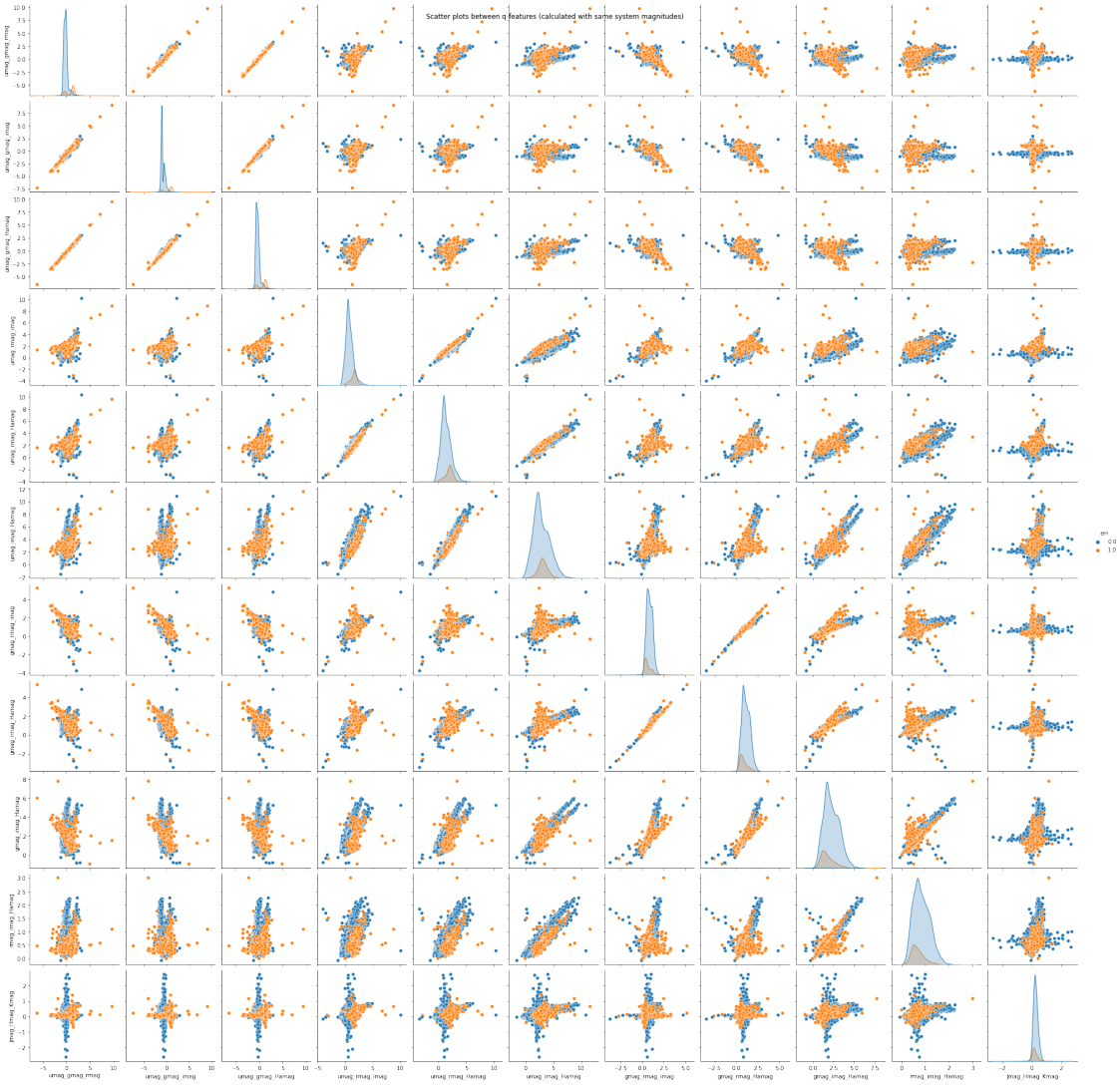
	gmag_imag_Hamag	rmag_imag_Hamag	Jmag_Hmag_Kmag
count	10307.000000	10307.000000	10307.000000
mean	2.194289	0.814487	0.237285
std	0.937865	0.344062	0.194781
min	-0.928542	-0.071486	-2.622131
25%	1.505748	0.550846	0.126850
50%	2.058925	0.762523	0.223863
75%	2.856493	1.050168	0.336935
max	7.770383	3.004673	2.715843

[126]:
```
sn.heatmap(q_df.corr().abs())
_=plt.title("abs(Correlation) between q-features (calculated with same system␣
 ↪magnitudes)")
```

[126]:
```
Text(0.5, 1.0, 'abs(Correlation) between q-features (calculated with same system
magnitudes)')
```



[127]:
```
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)")
```


Scatter plots between q features (calculated with same system magnitudes)
umag_gmag_rmag
umag_gmag_imag
umag_gmag_llamag
umag_rmag_imag
umag_rmag_llamag
umag_imag_llamag
gmag_rmag_imag
gmag_rmag_llamag
gmag_imag_llamag
rmag_imag_llamag
imag_lmag_Kmag
em
0.0
1.0