ABU 量化系统 简介（版本 0.1）

- **author = 'BBFamily'**
- **email = 'bbfamily@126.com'**
- **weixin = 'aaaabbbuu'**

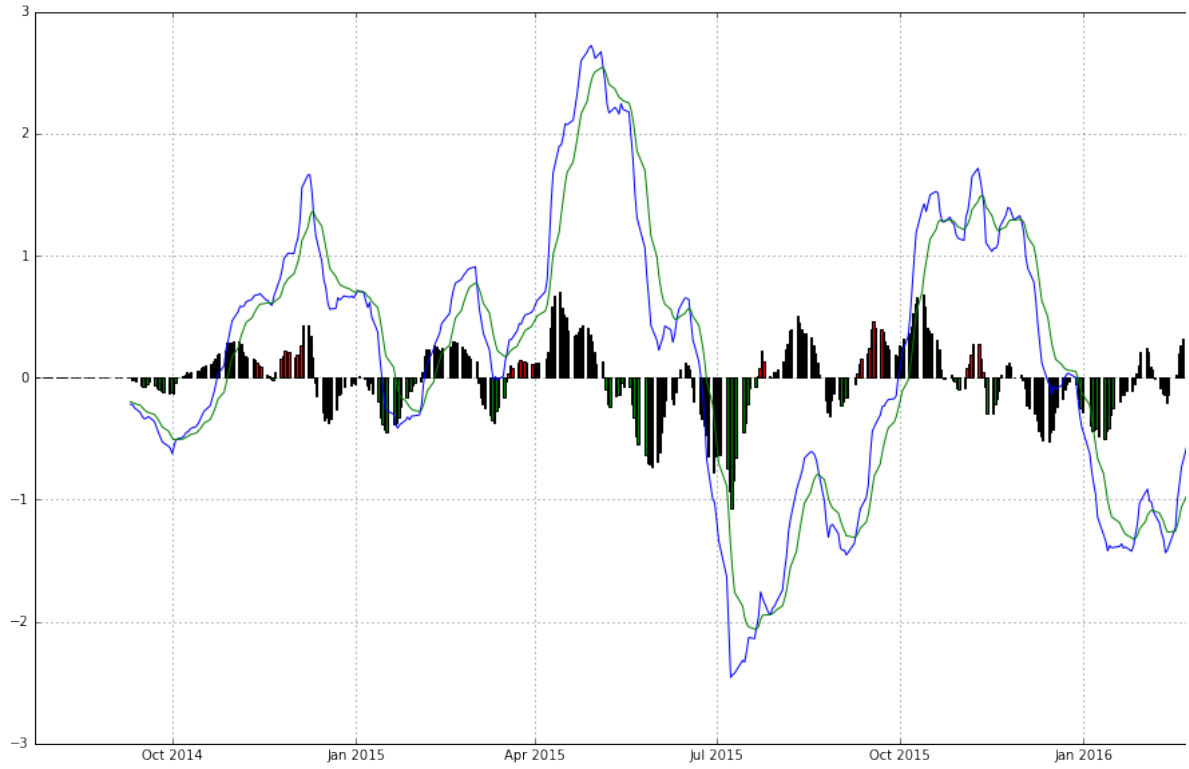
第二部分 相关与指标

`python import ZEnv import ZLog import ZCommonUtil %matplotlib inline`

```python import SymbolPd

import AtrIndicator import BollIndicator import MacdIndicator import ObvIndicator import RsiIndicator kl_pd = SymbolPd.make_kfold_pd('usNOAH') ```

## 1. 指标

**atr指标**

`python AtrIndicator.plot_atr_from_klpd(kl_pd)`


**布林带**

`python BollIndicator.plot_boll_from_klpd(kl_pd)`
```

macd指标

`python MacdIndicator.plot_macd_from_klpd(kl_pd)`



Obv指标

`python ObvIndicator.plot_obv_from_klpd(kl_pd)`

RSI指标

```
python RsiIndicator.plot_rsi_from_klpd(kl_pd)
```

2. 相关性

支持的几种相关性：具体参看SearchCorrTask 1. 皮尔逊相关性 2. 斯皮尔曼秩相关性 3. 皮尔逊sign相关性 4. 时间线性加权相关性

2.1 实时比较操作

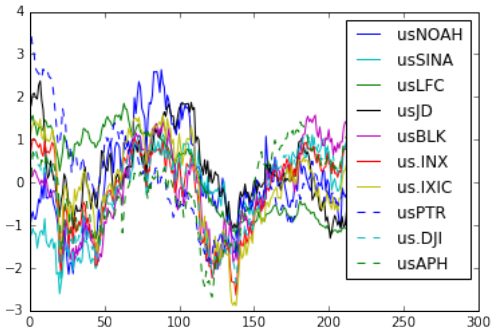
> 费时，不适合张任务的回归测试，但可以在实盘中使用

`python from SimilarHelper import find_similar_with_cnt, CoreCorrType, pd_list`

皮尔逊相关，非时间线性加权，默认使用多进程方式工作，默认采用8 process，how的参数也可以是thread，或单进程模式

`python net_cg_ret = SimilarHelper.find_similar_with_cnt('usNOAH', 252, show_cnt=10, how='process', rolling=False, show=True, corr_type=CoreCorrType.E_CORE_TYPE_PEARS.value)`

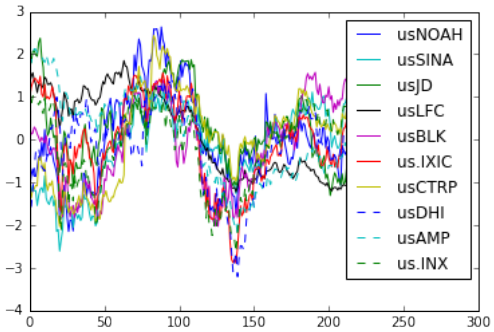
```
[('usNOAH', 1.0), ('usSINA', 0.61013592968789809), ('usLFC', 0.55992779447230523), ('usJD', 0.55271668899289317), ('usBLK', 0.53034567009321276), ('us.INX', 0.52618164897697028), ('us.IXIC', 0.52453610276882712), ('usPTR', 0.51194546819474251), ('us.DJI', 0.51092251519517629), ('usAPH', 0.50891702667480554)]


**********************************************************************************************************
show net cg ret...
```



皮尔逊相关，**时间线性加权** 默认加权使用60日时间加权

`python net_cg_ret = SimilarHelper.find_similar_with_cnt('usNOAH', 252, show_cnt=10, how='process', rolling=True, show=True, corr_type=CoreCorrType.E_CORE_TYPE_PEARS.value)`

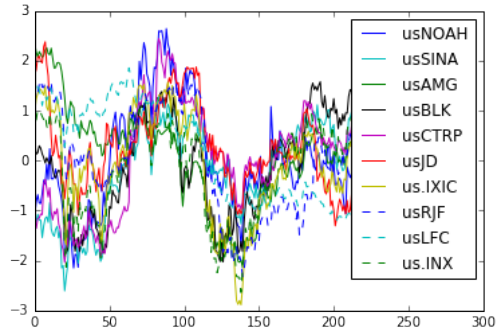
```
[('usNOAH', 0.99999999999999967), ('usSINA', 0.56870000412170252), ('usJD', 0.53765903965465123), ('usLFC', 0.5294377349137569), ('usBLK', 0.49923974059865167), ('us.IXIC', 0.49692869829420766), ('usCTRP', 0.48716122745665835), ('usDHI', 0.48561410297294422), ('usAMP', 0.48251781042029129), ('us.INX', 0.47890377037887744)]


**********************************************************************************************************
show net cg ret...
```



斯皮尔曼轶相关性加权

`python net_cg_ret = SimilarHelper.find_similar_with_cnt('usNOAH', 252, show_cnt=10, how='process', rolling=False, show=True, corr_type=CoreCorrType.E_CORE_TYPE_SPERM.value)`

```
[('usNOAH', 0.99999999999999978), ('usSINA', 0.54008994510319208), ('usAMG', 0.53164919563837731), ('usBLK',
0.52406920780532629), ('usCTRP', 0.52284033856443368), ('usJD', 0.51989268437538194), ('us.IXIC', 0.51808009750564055),
('usRJF', 0.51543054576671066), ('usLFC', 0.5120318940826416), ('us.INX', 0.5116959640305754 9)]


**********************************************************************************************************
show net cg ret...
```



net_cg_ret按照相关性排序

`python net_cg_ret[-5:], net_cg_ret[:5]`

```
([('usCIZN', -0.10865662217854094),
  ('usELON', -0.1110607757823851),
  ('usAIII', -0.11893515495825376),
  ('usCHCI', -0.1275198806866982),
  ('usCBFV', -0.15845990225407669)],
 [('usNOAH', 0.99999999999999978),
  ('usSINA', 0.54008994510319208),
  ('usAMG', 0.53164919563837731),
  ('usBLK', 0.52406920780532629),
  ('usCTRP', 0.52284033856443368)])
```

2.2 使用缓存数据进行高效相关分析，适合长任务回归测试

```
    SimilarHelper -》 K_TASK_PROCESS_CNT_LIST = np.array([2, 2, 4, 8])
    控制制作缓存数据分配的进程数默认如下:
    1. 皮尔逊相关性:  2 process
    2. 斯皮尔曼秩相关性: 2 process
    3. 皮尔逊sign相关性: 4 process
    4. 时间线性加权相关性: 8 process
    可调整参数，支持在每个进程中使用多线程继续优化加速，默认关闭的，需要可以打开g_enable_mul_thread

    由于时间线性加权相关性全量太过费时，一般可只做前三个
```

使用SimilarHelper.pd_list()加载数据, 如果不存在制作开始按照预设分配的进程数

`python import SimilarHelper corr_pd_list = SimilarHelper.pd_list()`

`python corr_pd_list[1]`

	usA	usAA	usAAC	usAAMC	usAAME	usAAN	usAAOI	usAAON	usAAP	usAAT
usA	1.000000	0.454448	0.274068	0.308121	0.055074	0.485181	0.431542	0.563792	0.382550	0.404967
usAA	0.454448	1.000000	0.187478	0.254320	-0.002108	0.376553	0.425732	0.432990	0.219294	0.220109
usAAC	0.274068	0.187478	1.000000	0.055484	0.106134	0.216095	0.227468	0.253837	0.171094	0.195180
usAAMC	0.308121	0.254320	0.055484	1.000000	-0.075133	0.349098	0.325843	0.243445	0.201342	0.217744
usAAME	0.055074	-0.002108	0.106134	-0.075133	1.000000	-0.097836	-0.097696	-0.012328	-0.102436	-0.067698
usAAN	0.485181	0.376553	0.216095	0.349098	-0.097836	1.000000	0.310072	0.475825	0.392844	0.389914
usAAOI	0.431542	0.425732	0.227468	0.325843	-0.097696	0.310072	1.000000	0.392092	0.236215	0.283324
usAAON	0.563792	0.432990	0.253837	0.243445	-0.012328	0.475825	0.392092	1.000000	0.284862	0.432501
usAAP	0.382550	0.219294	0.171094	0.201342	-0.102436	0.392844	0.236215	0.284862	1.000000	0.402522
usAAT	0.404967	0.220109	0.195180	0.217744	-0.067698	0.389914	0.283324	0.432501	0.402522	1.000000

usAAU	0.090275	0.159105	0.059951	0.104212	0.006994	0.131901	0.154099	0.013636	0.116574	-0.012216
usAAV	0.299074	0.464863	0.183979	0.244326	0.037983	0.346940	0.295155	0.245856	0.172848	0.106784
usAAVL	0.460946	0.343482	0.246605	0.172560	0.058241	0.332850	0.281272	0.387769	0.118523	0.185027
usAAWW	0.549689	0.511748	0.300822	0.231094	-0.054949	0.471911	0.392038	0.476789	0.330528	0.301207
usAB	0.524226	0.439046	0.106088	0.256513	-0.035333	0.413320	0.365664	0.365089	0.320084	0.357058
usABAC	0.069879	0.132819	0.010925	-0.007673	-0.067837	-0.033568	0.030410	0.125273	0.170593	0.031762
usABAX	0.447674	0.209852	0.118695	0.198642	-0.018452	0.348538	0.302488	0.412130	0.214052	0.337991
usABB	0.578691	0.519576	0.163079	0.393712	0.005483	0.468175	0.500296	0.425108	0.343830	0.353729
usABBV	0.555465	0.280976	0.277051	0.174658	0.069059	0.348778	0.303197	0.347684	0.279087	0.330945
usABC	0.529042	0.235207	0.239536	0.172946	-0.018286	0.355721	0.275164	0.307936	0.226937	0.357537
usABCB	0.475052	0.342009	0.211363	0.157182	-0.036168	0.318717	0.378797	0.517792	0.354050	0.302414
usABCD	0.354736	0.290120	0.147932	0.187004	-0.042498	0.322558	0.262014	0.273638	0.231879	0.316499
usABCO	0.396089	0.237458	0.141335	0.166912	0.012578	0.378912	0.300450	0.405622	0.271899	0.282061
usABEV	0.403559	0.306684	0.128369	0.225910	-0.034657	0.344967	0.217230	0.242848	0.282973	0.325563
usABG	0.550430	0.437030	0.320961	0.242265	-0.018087	0.510831	0.346124	0.469174	0.432710	0.348313
usABGB	0.131156	0.074578	0.062067	0.029161	0.047082	0.044418	0.014763	-0.030705	-0.040545	0.053948
usABIO	0.250981	0.258947	0.034687	0.231376	-0.008111	0.276710	0.177802	0.175909	0.091684	0.076951
usABM	0.474329	0.261511	0.218154	0.221335	-0.068930	0.494608	0.239745	0.502469	0.346443	0.469951
usABMD	0.462585	0.179917	0.179309	0.245769	-0.017807	0.332748	0.315565	0.372108	0.274189	0.300099
usABR	0.232418	0.122577	0.188977	0.124278	0.090227	0.271181	0.179305	0.263117	0.095628	0.244742
...	...	...	...	...	...	...	...	...	...	...
usYHOO	0.569589	0.315519	0.197819	0.271629	-0.005471	0.368495	0.364323	0.362053	0.288955	0.340730
usEVH	0.389634	0.257354	0.199764	0.214047	0.085129	0.218687	0.260083	0.392008	0.174974	0.255529
usGI	0.316709	0.275658	0.140576	0.164743	-0.099628	0.251890	0.283881	0.302175	0.161479	0.191164
usDTEA	0.284310	0.251952	0.153737	0.131513	0.027239	0.217612	0.127556	0.232763	0.072712	0.166824
usPSG	0.317306	0.434420	0.225205	0.273092	-0.022342	0.379699	0.348245	0.246246	0.195166	0.217840
usQVCA	0.558049	0.340614	0.240380	0.326454	-0.014908	0.524353	0.407633	0.484439	0.451765	0.465532
usAYA	0.364391	0.433319	0.214026	0.188849	0.009975	0.329815	0.304226	0.301947	0.308580	0.233615
usTEDU	0.331351	0.251101	0.153393	0.182075	0.079115	0.277623	0.304044	0.283202	0.133715	0.138087
usBPMX	-0.023579	0.015340	0.090173	-0.018458	0.015553	0.066045	0.053967	0.066632	0.027794	0.052820
usAXON	0.339576	0.096327	0.200350	0.211688	-0.036692	0.264798	0.108883	0.230999	0.168519	0.162368
usWING	0.375607	0.259010	0.126423	0.325191	0.060404	0.410260	0.273141	0.278381	0.207025	0.317478
usAGN	0.450831	0.152522	0.217112	0.199542	-0.076309	0.299923	0.272233	0.237015	0.322057	0.318542
usCYNA	0.245070	0.189420	0.084588	0.138561	-0.089123	0.169813	0.212136	0.079870	0.165048	0.139062
usFIT	0.371107	0.278178	0.103519	0.214491	0.028199	0.299482	0.228014	0.226467	0.241972	0.204270
usINFY	0.496916	0.435122	0.211575	0.283410	-0.036155	0.410039	0.450674	0.397472	0.379555	0.394253
usUNVR	0.447825	0.468338	0.226963	0.326363	-0.008633	0.396387	0.417752	0.375852	0.296985	0.263691
usALRM	0.458248	0.372059	0.159829	0.187691	-0.079083	0.394311	0.294063	0.404661	0.324234	0.261349
usAPPF	0.389969	0.321819	0.068792	0.232718	-0.040292	0.291501	0.270549	0.326899	0.111500	0.264680
usCNXC	0.298068	0.309068	0.105643	0.165096	0.050371	0.211987	0.235728	0.171029	0.111885	0.163588
usGKOS	0.339527	0.156565	0.173740	0.131183	-0.021536	0.232621	0.212223	0.175413	0.196056	0.228032
usGNRT	0.343969	0.375890	0.186795	0.195938	0.028350	0.307620	0.313758	0.303029	0.217177	0.220750
usGPP	0.208322	0.253523	0.128787	0.164123	0.013668	0.208800	0.136328	0.137359	0.065988	0.109269
usLNTH	0.247155	0.165498	0.130132	0.081825	-0.075762	0.203492	0.139172	0.293002	0.171665	0.307344
usMCRB	0.384040	0.266243	0.190672	0.244506	-0.014354	0.359491	0.330061	0.354221	0.156847	0.171041

usMCRN	0.406527	0.369982	0.187153	0.210304	-0.026441	0.298955	0.299980	0.389835	0.290215	0.222804
usTRU	0.475249	0.423885	0.184030	0.219624	-0.076183	0.328488	0.365291	0.408678	0.369126	0.292982
usXTLY	0.209666	0.246674	0.184329	0.091924	0.008961	0.175945	0.260839	0.309748	0.098571	0.175062
usMB	0.184388	0.129717	0.180827	0.001713	0.008354	0.191048	0.101319	0.190441	0.060116	0.242622
usRTTR	0.106476	0.136337	0.083411	0.073129	0.004266	0.195342	0.109586	0.082093	0.092234	0.089450
usYECO	0.051904	0.003792	-0.005672	0.035617	0.044880	0.059244	0.011793	0.045998	-0.059229	-0.041723

4421 rows × 4421 columns

应用

`python import TLineSimilar`

calc_similar_top

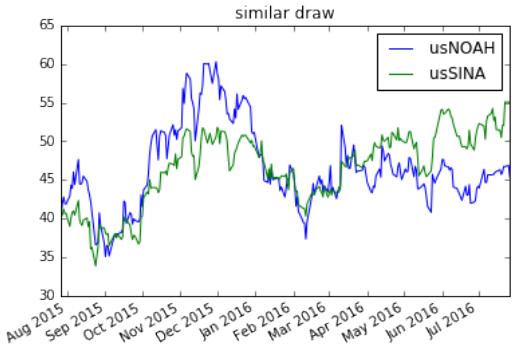
按照等权重的方式计算相关性排名, 显示对比中会按照look_max方式对齐数据

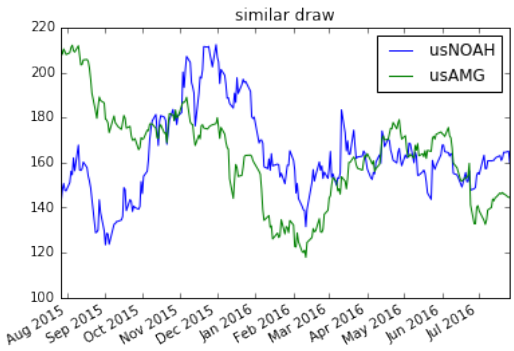
```
sc=slice 采用那几种相关性度量方式
slice(1, 2): sperman + pers sign
slice(0, 1): pers +  sperman
```

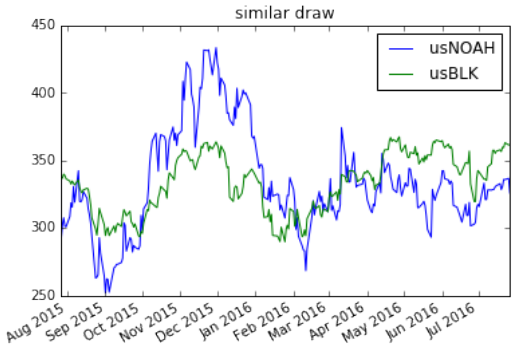
最后返回的结果是rank的结果，从2开始，1是symbol自身

slice(1, 2): sperman ＋ pers sign

`python TLineSimilar.calc_similar_top('usNOAH', sc=slice(1, 2), show_cnt=3)`



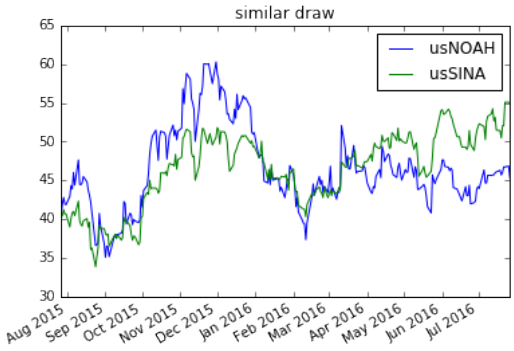


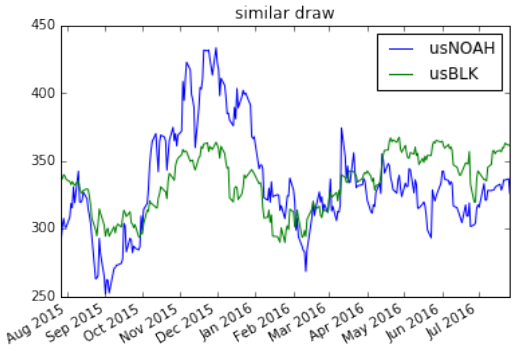


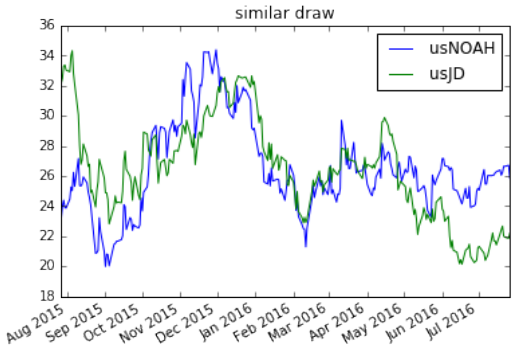

```
usSINA    2.0
usAMG     3.0
usBLK     4.0
dtype: float64
```

slice(0, 2): pers ＋ sperman + pers_sign 可以看到结果京东的rank上来了

```
python TLineSimilar.calc_similar_top('usNOAH', sc=slice(0, 2), show_cnt=3)
```







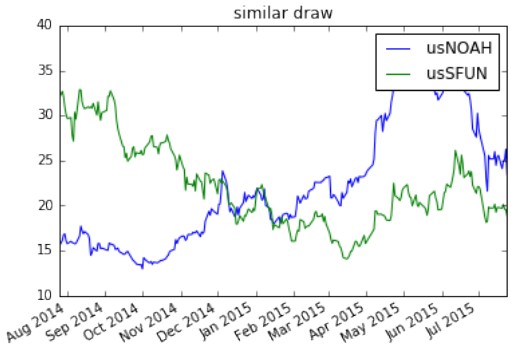
```
usSINA     4.0
usBLK      9.0
usJD      10.0
dtype: float64
```

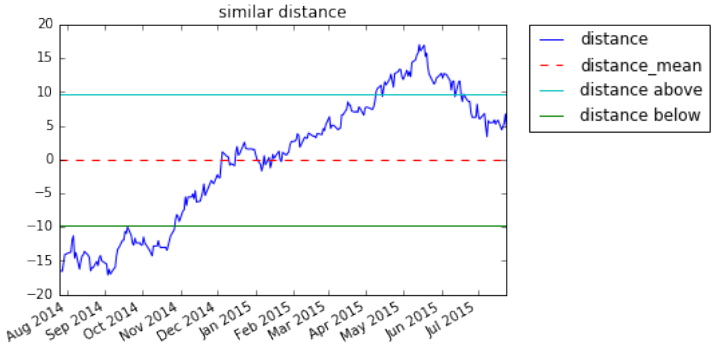
得到对比的股票在rank中的位置分量:

```
搜房的相关性于诺亚在总体所有股票0.84的位置上
```

`python TLineSimilar.calc_similar('usNOAH', 'usSFUN')`

```
usNOAH similar rank scoreusSFUN :0.840986202217
```






```
0.8409862022166931
```

相关与协整技术分析

相关不一定协整 协整不一定相关

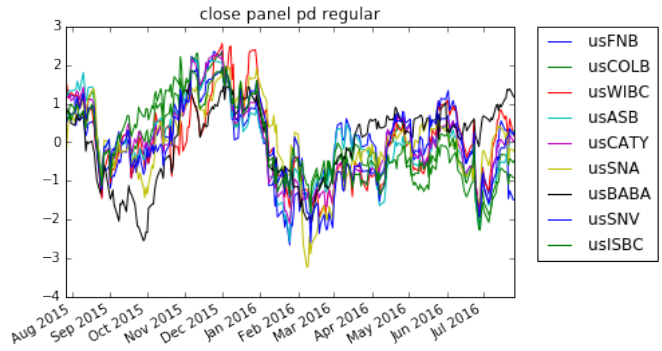
技术分析思路 从最相关的top n中寻找m个最协整的股票，现在的策略是从top 100多个 相关等权重中阀值0.08计算寻找最协整的10个股票，10个股票标准化后做 diff之后axis=1 轴上做sum，类似等权重投票机制成如

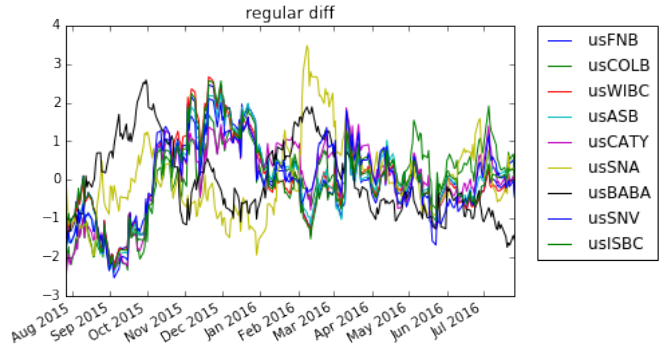
```
    distance_votes
    2015-07-27   -14.724491
    2015-07-28   -12.712066
    2015-07-29   -11.945266
    2015-07-30   -13.801350
    2015-07-31   -13.520431
    2015-08-03   -11.381343
    2015-08-04    -9.486645
    2015-08-05   -11.319338
    2015-08-06    -6.517725
    2015-08-07    -9.103014
    2015-08-10    -5.025694
    ......................
```

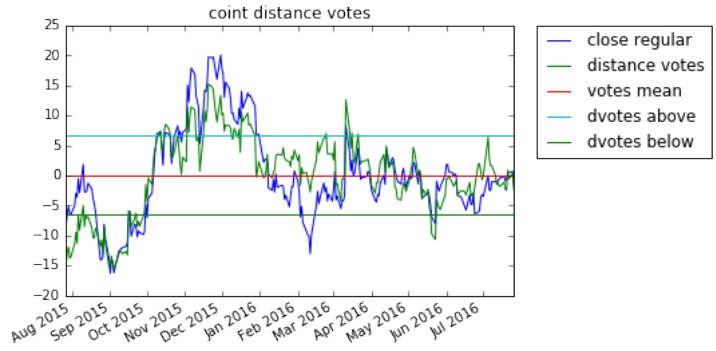
之后计算vote diff的above std， below std，mean, 可以根据above std， below std等作为交易信号，形成 策略因子的基础组成部份

更多技术分析阅读技术分析部分，在TLine模块下

`python TLineSimilar.coint_similar('usNOAH')`








```python
```