pestpp-glm

April 28, 2019

1 PESTPP-GLM

In this notebook, we will run PESTPP-GLM in standard parameter estimation mode and regularization mode. In both cases, we will use the baked-in bayes-linear posterior monte carlo analysis to get posterior forecast PDFs. We will use the prior monte carlo outputs as the prior forecast PDF.

In [1]:
```
import os
import shutil
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import flopy
import pyemu
```

```
flopy is installed in /Users/jeremyw/Dev/gw1876/activities_2day_mfm/notebooks/flopy
```

In [2]:
```
t_d = "template"
m_d = "master_glm"
```

In [3]:
```
pst = pyemu.Pst(os.path.join(t_d,"freyberg.pst"))
pst.write_par_summary_table(filename="none")
```

Out[3]:
```
                       type transform  count      initial value  \
grstrt3             grstrt3       log    705                  0
grsy3                 grsy3       log    705                  0
grhk5                 grhk5       log    705                  0
pp_vka1             pp_vka1       log     67                  0
strt8_cn           strt8_cn       log      1                  0
welflux_k02    welflux_k02       log      6                  0
grvka5               grvka5       log    705                  0
pp_rech0           pp_rech0       log     67                  0
rech4_cn           rech4_cn       log      1                  0
pp_strt0           pp_strt0       log     67                  0
grstrt5             grstrt5       log    705                  0
grvka3               grvka3       log    705                  0
pp_strt2           pp_strt2       log     67                  0
```

grss5	grss5	log	705	0
pp_ss2	pp_ss2	log	67	0
pp_hk2	pp_hk2	log	67	0
pp_ss1	pp_ss1	log	67	0
sy6_cn	sy6_cn	log	1	0
ss8_cn	ss8_cn	log	1	0
pp_hk1	pp_hk1	log	67	0
ss6_cn	ss6_cn	log	1	0
grstrt4	grstrt4	log	705	0
drncond_k00	drncond_k00	log	10	0
sy8_cn	sy8_cn	log	1	0
pp_sy0	pp_sy0	log	67	0
grsy4	grsy4	log	705	0
vka8_cn	vka8_cn	log	1	0
vka6_cn	vka6_cn	log	1	0
pp_vka0	pp_vka0	log	67	0
hk6_cn	hk6_cn	log	1	0
grsy5	grsy5	log	705	0
welflux	welflux	log	2	0 to 0.176091
grss3	grss3	log	705	0
hk8_cn	hk8_cn	log	1	0
sy7_cn	sy7_cn	log	1	0
strt7_cn	strt7_cn	log	1	0
pp_sy2	pp_sy2	log	67	0
strk	strk	log	40	0
grrech3	grrech3	log	705	0
ss7_cn	ss7_cn	log	1	0
grss4	grss4	log	705	0
flow	flow	log	1	0
pp_rech1	pp_rech1	log	67	0
grhk4	grhk4	log	705	0
grvka4	grvka4	log	705	0
strt6_cn	strt6_cn	log	1	0
grrech2	grrech2	log	705	0
pp_hk0	pp_hk0	log	67	0
rech5_cn	rech5_cn	log	1	-0.39794
pp_ss0	pp_ss0	log	67	0
grhk3	grhk3	log	705	0
pp_sy1	pp_sy1	log	67	0
pp_strt1	pp_strt1	log	67	0
hk7_cn	hk7_cn	log	1	0
vka7_cn	vka7_cn	log	1	0
pp_vka2	pp_vka2	log	67	0

	upper bound	lower bound	standard deviation
grstrt3	0.0211893	-0.0222764	0.0108664
grsy3	0.243038	-0.60206	0.211275
grhk5	1	-1	0.5

pp_vka1	1	-1	0.5
strt8_cn	0.0211893	-0.0222764	0.0108664
welflux_k02	1	-1	0.5
grvka5	1	-1	0.5
pp_rech0	0.0413927	-0.0457575	0.0217875
rech4_cn	0.0791812	-0.09691	0.0440228
pp_strt0	0.0211893	-0.0222764	0.0108664
grstrt5	0.0211893	-0.0222764	0.0108664
grvka3	1	-1	0.5
pp_strt2	0.0211893	-0.0222764	0.0108664
grss5	1	-1	0.5
pp_ss2	1	-1	0.5
pp_hk2	1	-1	0.5
pp_ss1	1	-1	0.5
sy6_cn	0.243038	-0.60206	0.211275
ss8_cn	1	-1	0.5
pp_hk1	1	-1	0.5
ss6_cn	1	-1	0.5
grstrt4	0.0211893	-0.0222764	0.0108664
drncond_k00	1	-1	0.5
sy8_cn	0.243038	-0.60206	0.211275
pp_sy0	0.243038	-0.60206	0.211275
grsy4	0.243038	-0.60206	0.211275
vka8_cn	1	-1	0.5
vka6_cn	1	-1	0.5
pp_vka0	1	-1	0.5
hk6_cn	1	-1	0.5
grsy5	0.243038	-0.60206	0.211275
welflux	0.176091 to 0.30103	-0.30103 to 0	0.0752575 to 0.11928
grss3	1	-1	0.5
hk8_cn	1	-1	0.5
sy7_cn	0.243038	-0.60206	0.211275
strt7_cn	0.0211893	-0.0222764	0.0108664
pp_sy2	0.243038	-0.60206	0.211275
strk	2	-2	1
grrech3	0.0413927	-0.0457575	0.0217875
ss7_cn	1	-1	0.5
grss4	1	-1	0.5
flow	0.09691	-0.124939	0.0554622
pp_rech1	0.0413927	-0.0457575	0.0217875
grhk4	1	-1	0.5
grvka4	1	-1	0.5
strt6_cn	0.0211893	-0.0222764	0.0108664
grrech2	0.0413927	-0.0457575	0.0217875
pp_hk0	1	-1	0.5
rech5_cn	-0.09691	-1	0.225772
pp_ss0	1	-1	0.5
grhk3	1	-1	0.5

```
pp_sy1                  0.243038       -0.60206         0.211275
pp_strt1               0.0211893     -0.0222764        0.0108664
hk7_cn                         1             -1              0.5
vka7_cn                        1             -1              0.5
pp_vka2                        1             -1              0.5
```

1.0.1 reduce the number of adjustable parameters

This is the painful part: we cant use 10K+ pars because we cant wait around for that many runs and then the linear algebra of factoring a 10k+ by 10K+ matrix is also difficult. So that means we need to fix a lot a parameters #frownyface

```
In [4]: par = pst.parameter_data
```

```
In [5]: # grid-scale pars
        gr_pars = par.loc[par.pargp.apply(lambda x: "gr" in x),"parnme"]
        par.loc[gr_pars,"partrans"] = "fixed"
        pst.npar_adj
```

```
Out[5]: 1215
```

```
In [6]: # these are the sfr conductance parameters - Ive left all 40 adjustable
        # but if you uncomment this, it will tie them into 1 parameter effectively
        # strk_pars = par.loc[par.pargp=="strk","parnme"]
        # p1 = strk_pars.iloc[0]
        # par.loc[strk_pars.iloc[1:],"partrans"] = "tied"
        # par.loc[strk_pars.iloc[1:],"partied"] = p1
        # pst.npar_adj
```

```
In [7]: par.loc[par.pargp.apply(lambda x: "pp" in x),"pargp"].unique()
```

```
Out[7]: array(['pp_hk0', 'pp_hk1', 'pp_hk2', 'pp_rech0', 'pp_rech1', 'pp_ss0',
               'pp_ss1', 'pp_ss2', 'pp_strt0', 'pp_strt1', 'pp_strt2', 'pp_sy0',
               'pp_sy1', 'pp_sy2', 'pp_vka0', 'pp_vka1', 'pp_vka2'], dtype=object)
```

Fix the storage pilot points - we still have layer-scale storage pars adjustable

```
In [8]: s_pars = par.loc[par.pargp.apply(lambda x: "pp" in x and ("ss" in x or "sy" in x)),"pa
        par.loc[s_pars,"partrans"] = "fixed"
        pst.npar_adj
```

```
Out[8]: 813
```

```
In [9]: adj_par = par.loc[par.partrans=="log",:]
        adj_par.pargp.value_counts().sort_values()
```

```
Out[9]: sy6_cn          1
        rech5_cn        1
        ss7_cn          1
        strt7_cn        1
```

```
sy7_cn          1
hk6_cn          1
strt6_cn        1
vka6_cn         1
flow            1
ss6_cn          1
ss8_cn          1
sy8_cn          1
vka7_cn         1
hk8_cn          1
strt8_cn        1
vka8_cn         1
hk7_cn          1
rech4_cn        1
welflux         2
welflux_k02     6
drncond_k00    10
strk           40
pp_strt1       67
pp_vka1        67
pp_rech1       67
pp_hk0         67
pp_hk1         67
pp_rech0       67
pp_vka0        67
pp_strt2       67
pp_hk2         67
pp_vka2        67
pp_strt0       67
Name: pargp, dtype: int64
```

fix the future recharge pilot points, vka in layers 1 and 3 and the initial condition pilot points (we still have layer-scale pars for each of these types)

```
In [10]: fi_grps = ["pp_rech1","pp_vka0","pp_vka2","pp_strt0","pp_strt1","pp_strt2"]
         par.loc[par.pargp.apply(lambda x: x in fi_grps),"partrans"] = "fixed"
         pst.npar_adj

Out[10]: 411
```

Ok, thats better…so lets run PESTPP-GLM. We will use a single "base parameter" jacobian matrix as the basis for 6 super parameter iterations. Then we will draw 100 realizations from the FOSM posterior parameter covariance matrix and run those 100 realizations to get the psoterior forecast PDFs

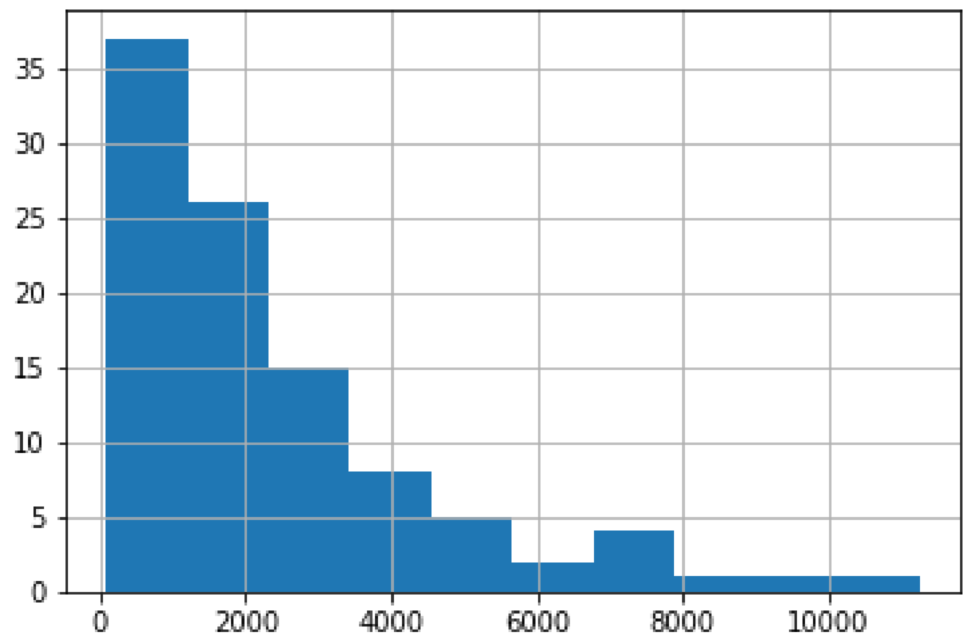
```
In [11]: pst.control_data.noptmax = 6
         pst.pestpp_options["n_iter_base"] = -1
         pst.pestpp_options["n_iter_super"] = 6
         pst.pestpp_options["num_reals"] = 100
```

```
pst.pestpp_options["parcov"] = "prior_cov.jcb"
pst.write(os.path.join(t_d,"freyberg_pp.pst"))
```

In [12]:
```
pyemu.os_utils.start_slaves(t_d,"pestpp-glm","freyberg_pp.pst",num_slaves=20,slave_roo
                            master_dir=m_d)
```

In [13]:
```
df = df=pd.read_csv(os.path.join(m_d,"freyberg_pp.post.obsen.csv"),index_col=0)
oe = pyemu.ObservationEnsemble.from_dataframe(pst=pst,df=df)
```

In [14]:
```
ax = oe.phi_vector.hist()#bins=np.linspace(0,100,20))
```

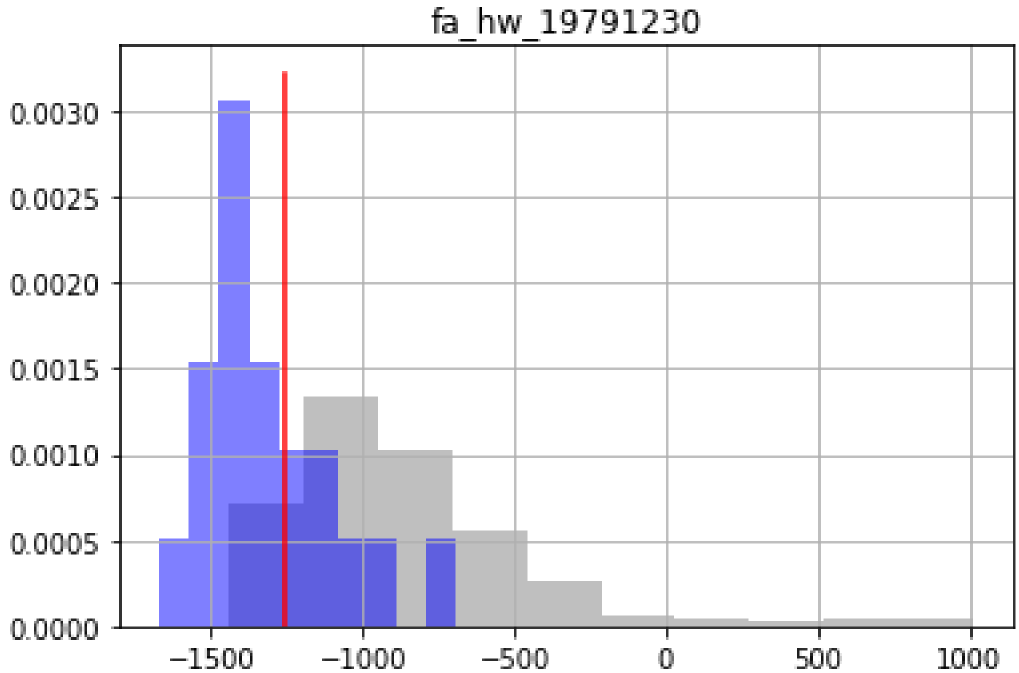


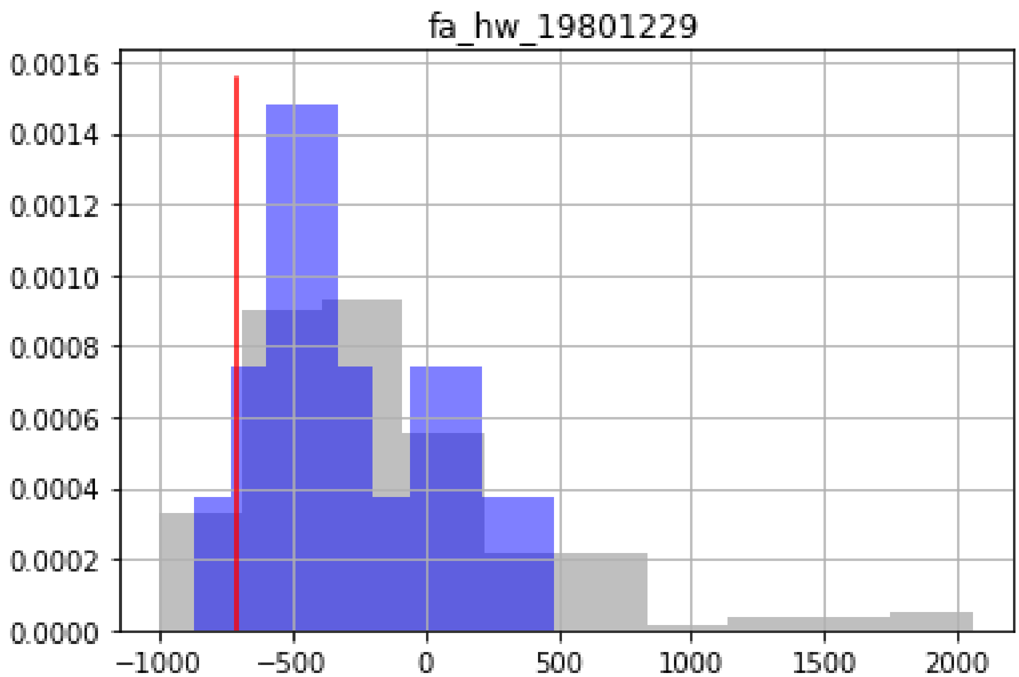
Here we see the distribution of phi values across the 100 posterior realizations. Should we accept all of these??? The theoretical phi we should accept is number of nonzero obs (14).

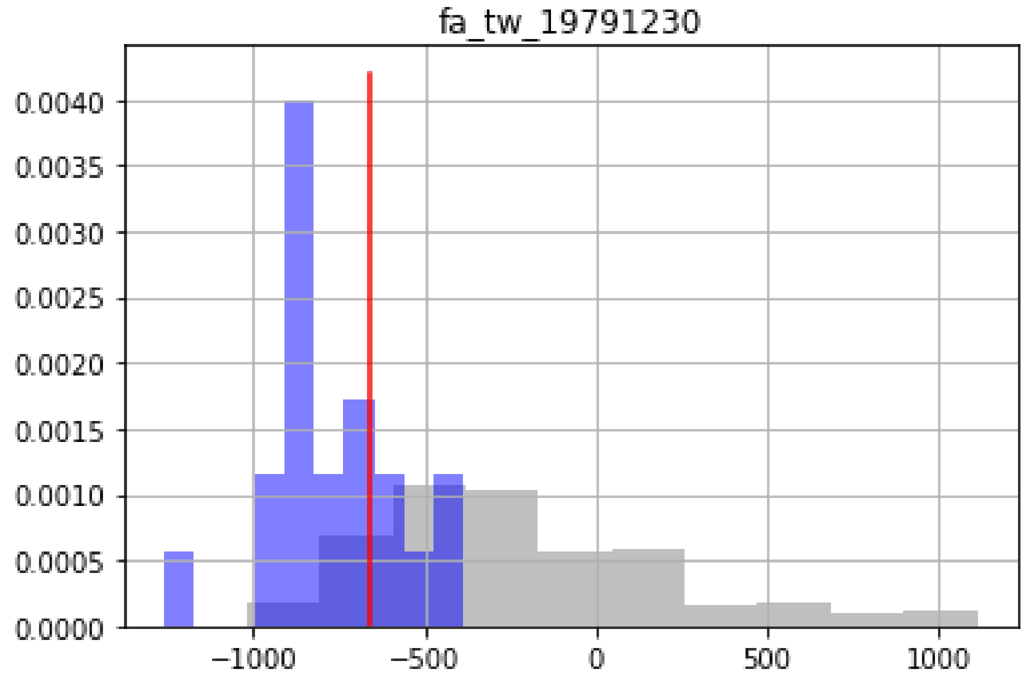
In [15]:
```
oe_pt = oe.loc[oe.phi_vector.sort_values().index[:20],:]
```

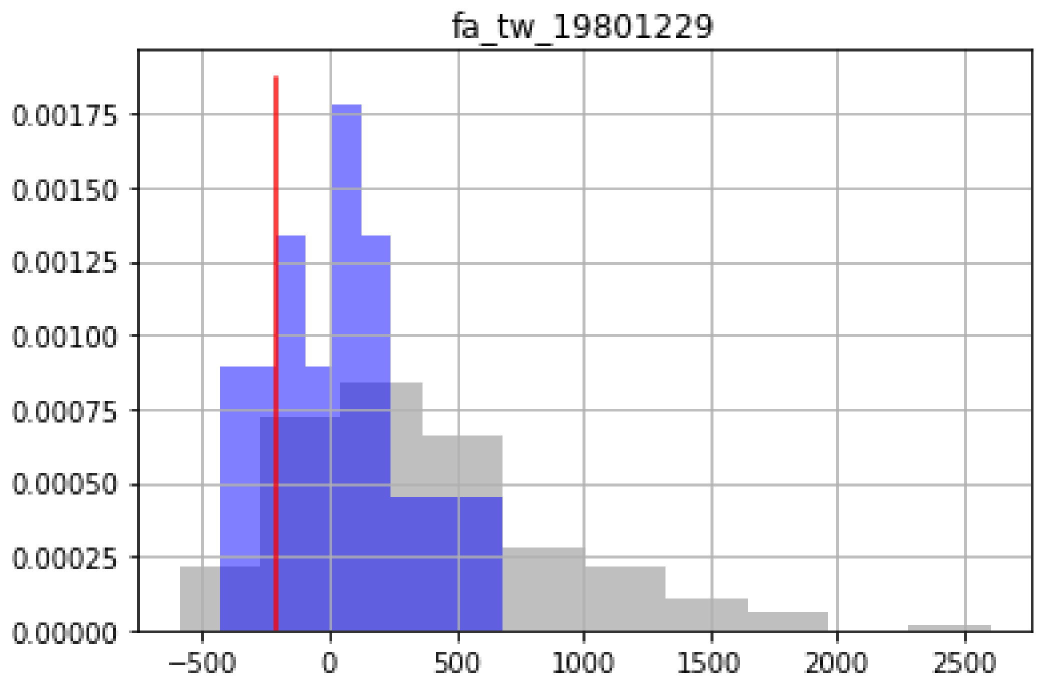
In [16]:
```
oe_pr = pd.read_csv(os.path.join("master_prior_sweep","sweep_out.csv"),index_col=0)
```

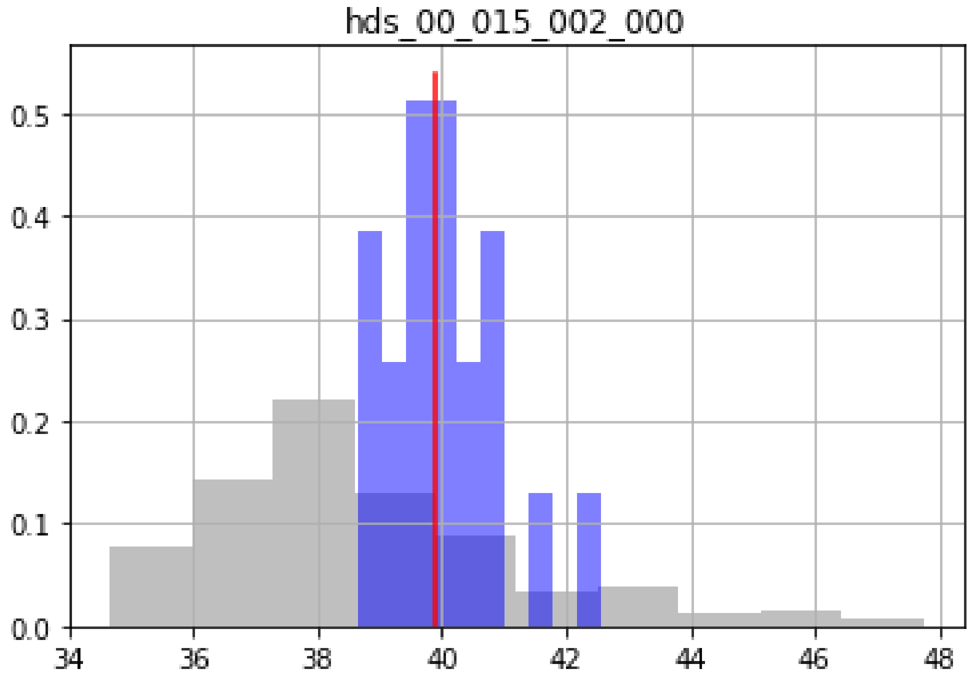
In [17]:
```
obs = pst.observation_data
fnames = pst.pestpp_options["forecasts"].split(",")
for forecast in fnames:
    ax = plt.subplot(111)
    oe_pr.loc[:,forecast].hist(ax=ax,color="0.5",alpha=0.5,normed=True)
    oe_pt.loc[:,forecast].hist(ax=ax,color="b",alpha=0.5,normed=True)

    ax.plot([obs.loc[forecast,"obsval"],obs.loc[forecast,"obsval"]],ax.get_ylim(),"r"
    ax.set_title(forecast)
    plt.show()
```


fa_hw_19791230
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
−1500
−1000
−500
0
500
1000

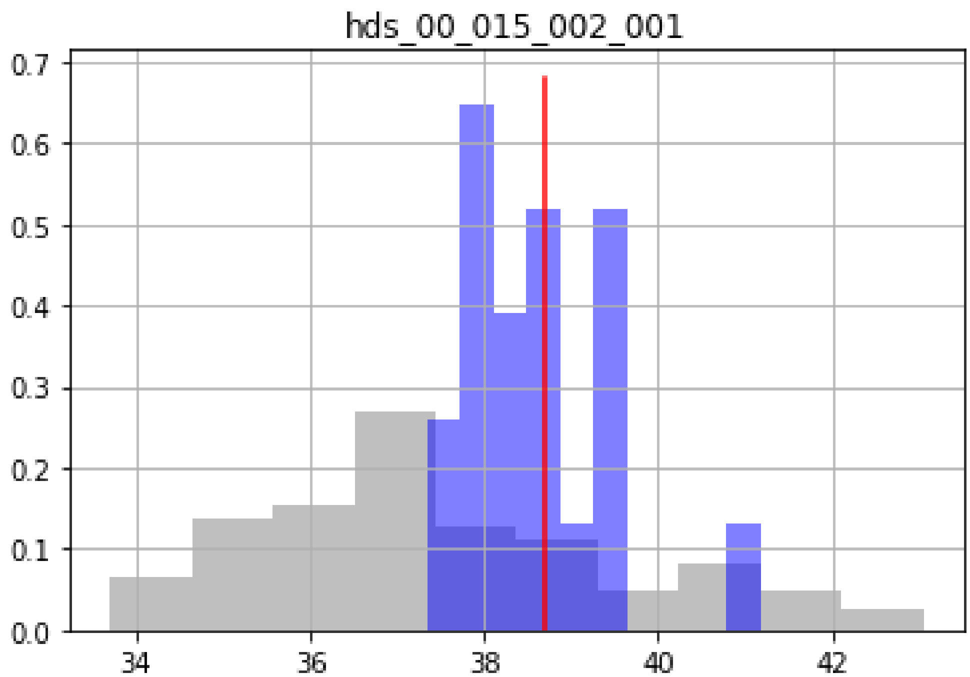

fa_hw_19801229
0.0016
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
−1000
−500
0
500
1000
1500
2000


fa_tw_19791230
0.0040
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
−1000
−500
0
500
1000


fa_tw_19801229
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
−500
0
500
1000
1500
2000
2500



hds_00_015_002_000
0.5
0.4
0.3
0.2
0.1
0.0
34
36
38
40
42
44
46
48



hds_00_015_002_001
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
34
36
38
40
42

1.0.2 Setup of Tikhonov regularization

So lets setup and use some formal regularization to bring the final phi up to around 14. We will use first-order regularization based on the covariance matrix we build earlier:

```
In [18]: cov = pyemu.Cov.from_binary(os.path.join(t_d,"prior_cov.jcb"))
```

```
In [19]: cnames = set(cov.row_names)
         pnames = set(pst.adj_par_names)
         cnames.symmetric_difference(pnames)
```

```
Out[19]: {'hk4006002',
          'strt4017015',
          'ss4008010',
          'vka5006014',
          'strt3017010',
          'ss3020013',
          'hk5029007',
          'ss5005018',
          'ss5034007',
          'vka5023010',
          'vka3026001',
          'vka4029017',
          'vka5009018',
          'ss3026010',
          'hk4014012',
          'ss3000007',
          'hk5007007',
          'rech3023006',
          'ss4004017',
          'strt3018017',
          'hk4003017',
          'vka4016008',
          'sy4016015',
          'sy113',
          'sy5001013',
          'ss5010015',
          'sy3024015',
          'ss3003001',
          'ss3000013',
          'sy5010015',
          'hk4020011',
          'sy5010000',
          'sy5023018',
          'vka3024000',
          'vka5022013',
          'ss5027002',
          'ss4035017',
          'sy5025017',
```

```
'vka5021011',
'strt4017003',
'vka3021011',
'ss4025001',
'rech2033018',
'strt3016011',
'vka4012001',
'sy5011003',
'vka5038013',
'ss3016016',
'hk5003007',
'vka5016003',
'strt5015001',
'hk5001016',
'rech3005017',
'rech3005018',
'hk3017009',
'strt4000005',
'vka4029016',
'ss4029001',
'vka4007017',
'hk5023009',
'sy103',
'ss5036008',
'strt4003011',
'strt4032017',
'sy4009013',
'ss5010012',
'sy4030006',
'vka5026003',
'vka5016009',
'ss4020017',
'hk5002012',
'sy3010002',
'vka3006009',
'rech3018014',
'strt3019002',
'sy5030003',
'sy4034006',
'hk3022013',
'strt5001006',
'strt3037005',
'sy4016000',
'strt152',
'ss3000017',
'strt5027009',
'ss3030003',
'sy5030009',
```

```
'vka4004009',
'strt5005015',
'ss3006019',
'vka3016015',
'vka3029010',
'ss5003016',
'sy5030011',
'ss5015016',
'hk3000019',
'vka3029007',
'rech3029014',
'vka3011017',
'hk4021003',
'sy3012016',
'vka3031014',
'rech3000004',
'strt5001018',
'ss3003008',
'vka4028017',
'hk4023002',
'sy3013003',
'strt5005011',
'sy4010014',
'ss5036007',
'ss5017000',
'ss5019015',
'ss3029009',
'strt3020010',
'rech2005000',
'vka4007015',
'ss4002001',
'vka3009019',
'strt5005008',
'hk5014016',
'rech2002006',
'vka4005009',
'vka4023001',
'sy4037007',
'rech3037016',
'hk4002001',
'hk5033018',
'hk4036005',
'rech2012017',
'strt4014015',
'sy3023002',
'sy5029015',
'vka5021013',
'ss221',
```

```
'strt5036010',
'hk4016014',
'rech3008015',
'hk5023013',
'sy4004011',
'sy5033007',
'strt5035014',
'rech3029010',
'ss3012019',
'hk3009016',
'strt3008001',
'sy3005003',
'hk4006010',
'ss146',
'strt4031017',
'ss036',
'hk3033008',
'ss5030010',
'vka5005000',
'strt5000006',
'vka4003018',
'strt146',
'sy5017011',
'sy3005010',
'sy4032007',
'strt3003010',
'hk5011001',
'ss5034004',
'strt3025018',
'ss3015014',
'strt5000011',
'ss5029016',
'sy4035013',
'sy4007007',
'hk4019013',
'strt5002009',
'strt4016008',
'strt4026007',
'ss3012016',
'strt3032015',
'ss4035014',
'rech3000016',
'sy3012013',
'sy3037005',
'vka3022013',
'rech3035018',
'strt4004006',
'sy3003004',
```

```
'rech3033003',
'strt3006007',
'strt4028000',
'hk5023014',
'sy5006004',
'ss5019019',
'vka3003012',
'rech3035015',
'ss039',
'rech2035004',
'sy3006003',
'strt3016015',
'vka5032007',
'ss4014012',
'strt3030000',
'sy5010011',
'ss3021014',
'vka3036008',
'ss3034014',
'strt4017018',
'ss4027009',
'hk4013011',
'hk3013002',
'vka4032007',
'sy5013018',
'hk4004001',
'sy3006009',
'strt5008010',
'ss4017017',
'strt5008011',
'sy5026005',
'hk3030009',
'sy3029015',
'hk3015012',
'ss4006008',
'strt3020008',
'vka3037017',
'strt5008002',
'hk3016010',
'vka4039012',
'strt3026011',
'hk3003015',
'ss4016009',
'vka4021019',
'vka5004011',
'ss4034011',
'vka5023015',
'vka3011018',
```

```
'hk3013011',
'hk3038009',
'sy037',
'ss4021007',
'hk3023017',
'vka4033008',
'ss132',
'strt5003004',
'hk4027006',
'strt5038013',
'hk3031005',
'rech3004011',
'ss3009009',
'vka4004003',
'vka4028007',
'hk5017008',
'ss3000008',
'sy3038015',
'strt3036008',
'sy4006016',
'rech3000011',
'ss5021001',
'vka3006011',
'vka4025000',
'rech3002013',
'sy125',
'vka5005005',
'strt4033002',
'strt3006004',
'ss5002010',
'hk4002014',
'ss3030005',
'strt3017011',
'hk3010010',
'sy5018012',
'hk4026016',
'vka3033014',
'vka5006009',
'sy5018014',
'hk4026006',
'ss5004004',
'rech3001005',
'vka4034004',
'strt3005005',
'ss3035010',
'strt5016001',
'strt3029005',
'strt4015013',
```

```
'vka5026017',
'strt4007017',
'vka4038008',
'sy3002015',
'vka5019014',
'ss3010013',
'rech2005017',
'strt3022001',
'vka025',
'sy3028012',
'hk5037005',
'sy3020011',
'strt5039011',
'strt5022002',
'hk3003016',
'hk4027001',
'hk3005008',
'ss4017009',
'strt5018018',
'strt5012010',
'vka3030015',
'hk3038016',
'strt5036005',
'sy3033004',
'strt4039008',
'ss5027016',
'strt005',
'rech2032003',
'sy3023015',
'strt5029016',
'strt5020015',
'strt3035005',
'sy3032008',
'hk5022008',
'vka3007019',
'sy5023000',
'rech3017003',
'sy5025016',
'vka5031009',
'sy222',
'vka3026005',
'strt3029009',
'rech2015018',
'ss3017016',
'vka4001005',
'strt3005019',
'sy120',
'ss5014013',
```

```
'hk5009013',
'hk3007010',
'sy3026007',
'strt4021001',
'ss4037010',
'sy5015010',
'vka212',
'strt3003011',
'sy3017008',
'ss5012017',
'ss057',
'vka3024009',
'hk3015014',
'sy4023003',
'vka3023015',
'vka3015001',
'sy5030001',
'hk4002012',
'sy3029010',
'rech3005012',
'sy3024019',
'strt4009016',
'vka3005005',
'vka4007009',
'ss3017014',
'rech3033004',
'vka4007010',
'strt3000011',
'hk3004017',
'ss5011000',
'vka5032015',
'ss3014017',
'sy223',
'rech2022010',
'sy4020017',
'vka4014013',
'vka4001012',
'vka4011002',
'vka051',
'ss5010009',
'sy3027018',
'ss4016011',
'ss233',
'sy3016010',
'hk5009000',
'ss5028014',
'hk5026000',
'strt4006001',
```

```
'strt5015010',
'strt3029007',
'strt5007006',
'vka3014017',
'ss4015009',
'vka3014014',
'hk4025009',
'ss5005013',
'ss3023009',
'ss4022011',
'vka266',
'hk5012000',
'hk4034017',
'hk3014019',
'hk3027003',
'sy4038008',
'vka5025003',
'ss4023001',
'hk3004013',
'vka5013017',
'vka5037014',
'hk4020003',
'vka4015000',
'ss3038015',
'strt202',
'ss3031010',
'strt4039011',
'ss5001016',
'hk3037012',
'strt4038015',
'ss5026016',
'strt4004019',
'ss3039009',
'strt3004007',
'sy3030003',
'sy5036007',
'vka3015017',
'strt5039013',
'rech3006016',
'vka5021018',
'sy3026018',
'sy4001005',
'sy4009009',
'ss5022010',
'hk3008013',
'sy3022009',
'strt3028015',
'sy259',
```

```
'hk3033005',
'ss027',
'sy4005016',
'hk5027005',
'strt5019015',
'vka3035010',
'strt3023016',
'sy4035012',
'rech2013019',
'vka262',
'hk3036004',
'rech3006006',
'ss5000013',
'strt5029000',
'ss4031015',
'hk4038014',
'rech3001016',
'strt5006005',
'ss4000018',
'strt5015002',
'ss4015013',
'sy3009003',
'sy3008000',
'sy5029011',
'hk3037006',
'rech2035008',
'hk5030016',
'sy3009013',
'ss3007010',
'sy3020019',
'ss3005016',
'strt5023004',
'rech3002008',
'sy3031008',
'sy249',
'vka3022011',
'hk4038010',
'rech109',
'sy4016014',
'strt4001009',
'sy5017008',
'strt4023004',
'strt3016000',
'ss3007011',
'vka3015000',
'ss3031012',
'hk5024018',
'vka4010015',
```

```
'ss4023007',
'ss5016016',
'hk4031005',
'ss4019016',
'sy5021002',
'sy5033003',
'sy4004000',
'sy5003012',
'ss5020016',
'ss4026013',
'sy4030000',
'rech2010018',
'hk5016009',
'rech2014011',
'rech2016010',
'vka3002006',
'strt3000019',
'ss3003014',
'hk5019016',
'ss3036007',
'ss5010017',
'ss3021017',
'strt014',
'hk5027017',
'hk5007008',
'sy233',
'ss5032003',
'ss3000014',
'rech3023000',
'sy5032010',
'rech3039011',
'vka4002002',
'ss5014015',
'rech2000013',
'strt5028014',
'rech3010009',
'strt5010015',
'sy4005009',
'vka4026015',
'vka4008000',
'ss4017013',
'strt160',
'vka4008019',
'strt4024007',
'ss5017014',
'sy4021002',
'vka3004018',
'strt4002013',
```

```
'strt3028005',
'sy4007002',
'rech2026011',
'hk5032012',
'ss3024009',
'hk3020015',
'ss4014011',
'ss4025003',
'strt4038006',
'sy3021011',
'ss5002012',
'sy4007006',
'vka5024007',
'rech3009011',
'sy5023014',
'sy4022003',
'sy216',
'sy3036014',
'hk3016008',
'rech2038011',
'strt5006009',
'sy5032015',
'ss3017013',
'hk4022017',
'sy4038005',
'sy4020016',
'sy4006000',
'ss5016015',
'hk5023007',
'ss4000019',
'vka5005004',
'ss3018017',
'ss5030006',
'strt5026016',
'ss024',
'hk4035004',
'strt4030000',
'ss3035005',
'vka3000013',
'strt4012017',
'hk5034007',
'strt4018010',
'ss4033010',
'rech3029004',
'sy3005019',
'ss3025001',
'ss5009008',
'sy4000005',
```

```
'vka4024019',
'sy3003015',
'hk4013009',
'vka4007001',
'ss3033015',
'vka4003010',
'hk5023011',
'rech3019012',
'ss4036011',
'sy163',
'sy4014013',
'strt227',
'sy3017010',
'sy5016014',
'vka3025004',
'rech2027006',
'ss4026005',
'strt045',
'vka4031008',
'sy4014000',
'sy5010010',
'strt3013015',
'hk4020000',
'ss5026009',
'strt4016016',
'ss049',
'strt5000002',
'ss4023014',
'hk5024008',
'rech3038009',
'strt059',
'vka5025001',
'strt3030016',
'strt4025002',
'hk5015015',
'rech2029012',
'ss5018013',
'hk3005002',
'vka4001007',
'strt3003003',
'hk5032014',
'ss5033010',
'strt4024014',
'vka5025000',
'rech118',
'vka4022019',
'ss3022011',
'rech2030000',
```

```
'hk5005018',
'strt062',
'sy234',
'ss5006009',
'strt3024011',
'hk3017013',
'strt3032018',
'rech144',
'vka5006006',
'ss4036012',
'ss5032011',
'hk5000003',
'sy5030007',
'ss3006011',
'hk3029016',
'vka4022012',
'strt228',
'sy3025010',
'vka5012009',
'hk5016019',
'hk3003007',
'rech3008013',
'vka5031003',
'strt021',
'sy5027017',
'sy4004002',
'strt3024008',
'strt5015016',
'hk5036014',
'ss3016001',
'vka3013018',
'ss001',
'rech2023000',
'vka3018016',
'strt3038015',
'vka4031001',
'strt4036018',
'sy5030019',
'rech3007008',
'vka4013003',
'vka5003005',
'rech3027007',
'sy5011012',
'strt4003000',
'strt4011016',
'vka4012010',
'rech2022007',
'ss3035009',
```

```
'sy5019018',
'hk3005011',
'rech3022002',
'ss4030019',
'ss4024000',
'sy4038014',
'ss3020010',
'rech2034019',
'rech3030007',
'ss3033013',
'vka3033008',
'strt5023002',
'rech3019017',
'vka5016017',
'sy3017000',
'hk5031019',
'sy5015008',
'rech2011018',
'vka4032006',
'hk3028018',
'sy015',
'hk4004008',
'strt3035016',
'ss4020016',
'hk5012019',
'vka218',
'rech3007001',
'ss4024010',
'vka4035011',
'hk3010008',
'vka3005018',
'vka3014010',
'strt161',
'ss5025002',
'sy3005004',
'sy3013015',
'hk4020019',
'vka5013010',
'sy5034009',
'rech2024019',
'vka5008019',
'hk4001011',
'hk3011001',
'rech2036012',
'hk4027000',
'strt4011000',
'sy5026011',
'hk4016016',
```

```
'ss3028012',
'sy161',
'rech3035011',
'ss5000007',
'sy4020010',
'rech2010015',
'sy3026014',
'rech3023014',
'sy204',
'strt4007008',
'strt5021010',
'sy226',
'hk5011002',
'sy4010019',
'sy3039012',
'vka4027019',
'vka4028015',
'strt5032009',
'vka5027006',
'ss5002000',
'strt5020009',
'strt5022001',
'hk4015016',
'vka5006015',
'sy4018018',
'hk3030003',
'vka3008016',
'ss5017001',
'vka3024019',
'hk5033004',
'strt3006012',
'sy3032014',
'vka3026016',
'sy3000014',
'rech3026014',
'sy3031011',
'sy3022013',
'vka3014015',
'ss056',
'ss3027017',
'vka4031004',
'vka3000000',
'strt4015017',
'hk3003013',
'sy4021015',
'sy5002009',
'ss3001012',
'hk5014002',
```

```
'ss3002008',
'strt3008003',
'strt4039007',
'strt3024002',
'strt3026019',
'ss5025017',
'sy3022012',
'sy4003003',
'hk4015002',
'vka5032002',
'vka4022014',
'hk3010003',
'strt4020009',
'sy3023018',
'strt4019001',
'sy3014001',
'ss3038006',
'vka4023016',
'ss3020019',
'strt5026006',
'ss4028015',
'vka5005006',
'hk4027018',
'ss3001014',
'ss5022006',
'sy3011016',
'hk4011001',
'vka5023006',
'strt141',
'strt3035009',
'hk5012013',
'sy3020015',
'vka3004014',
'ss3017009',
'strt5037007',
'ss5008016',
'hk3034009',
'vka3001001',
'rech3006015',
'rech3027019',
'ss5009010',
'strt3011013',
'vka4009001',
'sy4003005',
'hk3018003',
'ss3038012',
'hk3022002',
'hk4014008',
```

```
'sy5008002',
'strt4013000',
'rech3032010',
'sy4019000',
'ss3000019',
'vka5001014',
'strt4005015',
'vka3017003',
'sy5008007',
'ss5004018',
'vka4005006',
'strt158',
'rech2010014',
'strt3015009',
'hk4025003',
'hk3006013',
'vka3012012',
'ss5024018',
'ss066',
'ss4022009',
'hk5008018',
'ss4017018',
'sy4023011',
'ss3039010',
'strt4005012',
'hk4006014',
'vka4030011',
'sy4024009',
'strt028',
'sy3003003',
'hk3004005',
'ss5031010',
'rech2032008',
'ss3029003',
'strt3015014',
'strt5001005',
'sy3023016',
'hk3037005',
'rech2025009',
'strt4034008',
'sy3035012',
'hk3025009',
'vka4024009',
'strt4026003',
'sy4014014',
'vka3020009',
'rech3031005',
'strt5038005',
```

```
'vka4006013',
'ss3025013',
'rech2020017',
'hk5004018',
'sy5032009',
'rech2016013',
'vka3002015',
'vka036',
'strt4033011',
'sy4021001',
'hk3019019',
'hk4026011',
'ss4009015',
'sy3022002',
'sy3017001',
'ss3028005',
'strt5016002',
'sy3010016',
'vka4002001',
'rech2005005',
'rech2002018',
'ss4005019',
'vka3012019',
'strt3026013',
'strt5029003',
'strt3025001',
'strt4009000',
'hk3014002',
'sy5023017',
'sy3007014',
'strt3032010',
'sy3016012',
'ss4019015',
'strt4032016',
'strt4013009',
'sy3030008',
'sy3039006',
'vka3000010',
'vka5025017',
'hk4032010',
'vka4021012',
'ss4027018',
'strt5026014',
'hk4033014',
'ss5014003',
'hk4029019',
'strt5005018',
'vka5011003',
```

```
'ss5007007',
'hk4038005',
'sy5003011',
'strt4027002',
'strt4008000',
'ss4029011',
'hk5011016',
'hk5014018',
'sy5017013',
'strt3000002',
'rech3029018',
'hk5031015',
'vka3029009',
'sy5037008',
'sy3029006',
'rech2018019',
'vka3001019',
'hk5015019',
'ss5022002',
'sy5031011',
'sy5028006',
'sy3002011',
'vka3001005',
'rech3023004',
'hk4013019',
'rech2011015',
'vka3012016',
'hk3028000',
'ss5027000',
'sy4031008',
'strt5026000',
'rech3036007',
'ss4018011',
'strt5004007',
'strt5033013',
'strt4022004',
'vka5010018',
'ss4000003',
'vka5004009',
'vka4014010',
'ss5016019',
'sy4030007',
'rech2024015',
'rech2000016',
'strt4006005',
'hk5031009',
'ss3023012',
'vka4037011',
```

```
'hk3004001',
'strt5010003',
'hk4001001',
'vka3027000',
'ss3032002',
'ss5001002',
'vka5011014',
'vka3025016',
'strt5027017',
'vka3031005',
'hk3038014',
'rech3030014',
'strt4000018',
'hk4013000',
'strt5004004',
'strt4019000',
'hk3003011',
'strt5035008',
'ss5005006',
'sy4001002',
'rech2021001',
'ss5025009',
'rech3025018',
'vka5009010',
'hk3037010',
'strt4000017',
'ss5036010',
'vka4011019',
'strt3036010',
'hk5011010',
'ss5037010',
'rech2006002',
'sy155',
'sy3031000',
'ss5002014',
'strt5031002',
'sy5024019',
'rech3036018',
'strt4011011',
'ss4013011',
'sy5000012',
'vka4022017',
'strt3013011',
'vka3025006',
'strt5018014',
'ss3024003',
'strt3018013',
'hk4006017',
```

```
          'sy5000008',
          'vka4037005',
          ...}
```

```
In [20]: pyemu.helpers.first_order_pearson_tikhonov(pst,cov)
```

```
getting CC matrix
processing
```

```
In [21]: pst.prior_information.head()
```

```
Out[21]:                                                       equation    obgnme  \
         pilbl
         pcc_1  1.0 * log(dc0000390005) - 1.0 * log(dc0000390006) = 0.0  regul_cc
         pcc_2  1.0 * log(dc0000390005) - 1.0 * log(dc0000390007) = 0.0  regul_cc
         pcc_3  1.0 * log(dc0000390005) - 1.0 * log(dc0000390008) = 0.0  regul_cc
         pcc_4  1.0 * log(dc0000390005) - 1.0 * log(dc0000390009) = 0.0  regul_cc
         pcc_5  1.0 * log(dc0000390005) - 1.0 * log(dc0000390010) = 0.0  regul_cc

                pilbl    weight
         pilbl
         pcc_1  pcc_1  0.904837
         pcc_2  pcc_2  0.818731
         pcc_3  pcc_3  0.740818
         pcc_4  pcc_4  0.670320
         pcc_5  pcc_5  0.606531
```

```
In [22]: shutil.copy2(os.path.join(m_d,"freyberg_pp.jcb"),os.path.join(t_d,"restart_pp.jcb"))
```

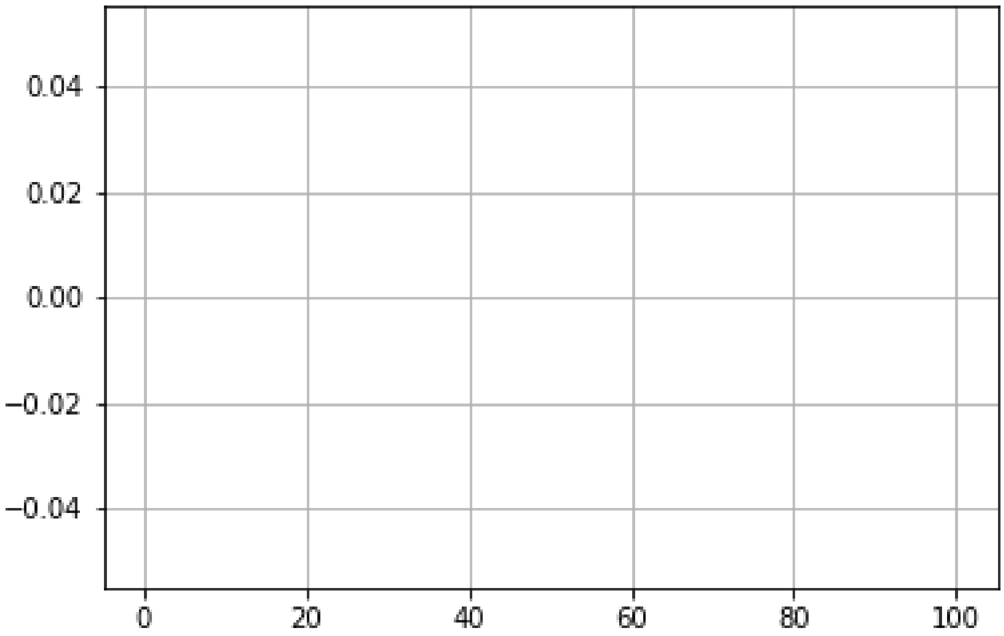
```
Out[22]: 'template/restart_pp.jcb'
```

```
In [23]: pst.pestpp_options["base_jacobian"] = "restart_pp.jcb"
         pst.reg_data.phimlim = pst.nnz_obs
         pst.reg_data.phimaccept = pst.reg_data.phimlim * 1.1
         pst.write(os.path.join(t_d,"freyberg_pp.pst"))
```

```
In [24]: pyemu.os_utils.start_slaves(t_d,"pestpp-glm","freyberg_pp.pst",num_slaves=20,slave_roo
                                     master_dir=m_d)
```

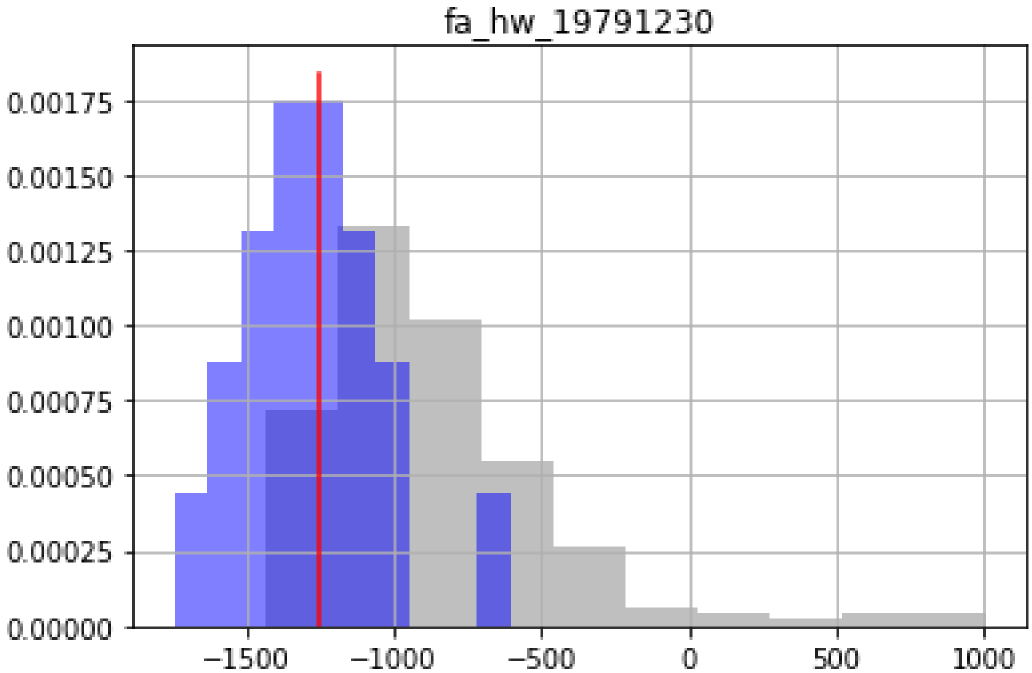
```
In [25]: df = df=pd.read_csv(os.path.join(m_d,"freyberg_pp.post.obsen.csv"),index_col=0)
         oe = pyemu.ObservationEnsemble.from_dataframe(pst=pst,df=df)
```

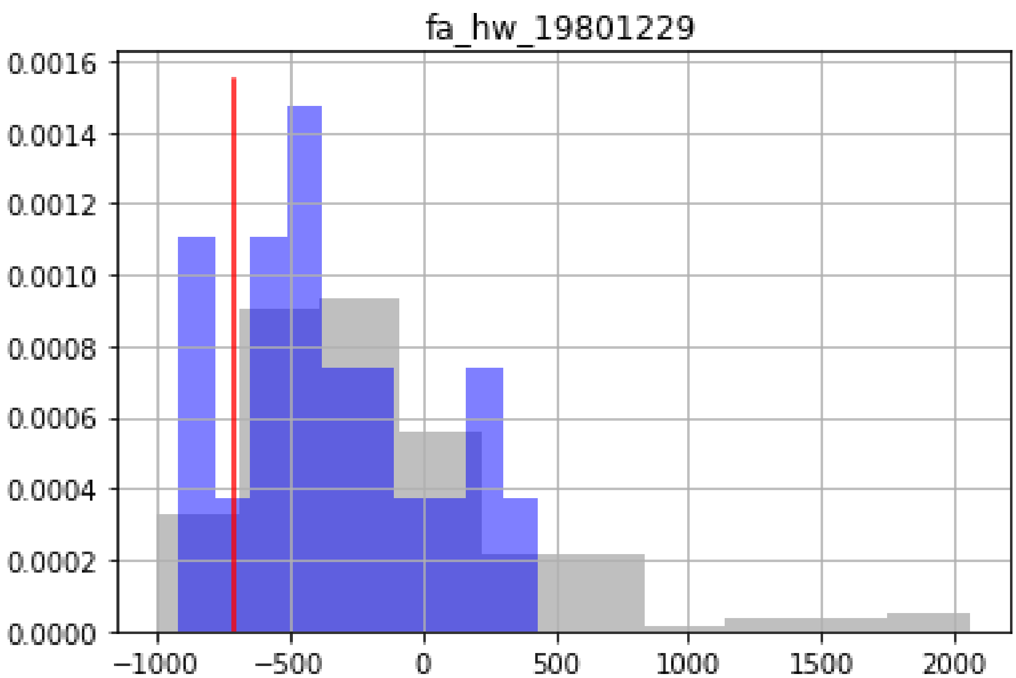
```
In [26]: ax = oe.phi_vector.hist(bins=np.linspace(0,100,20))
```

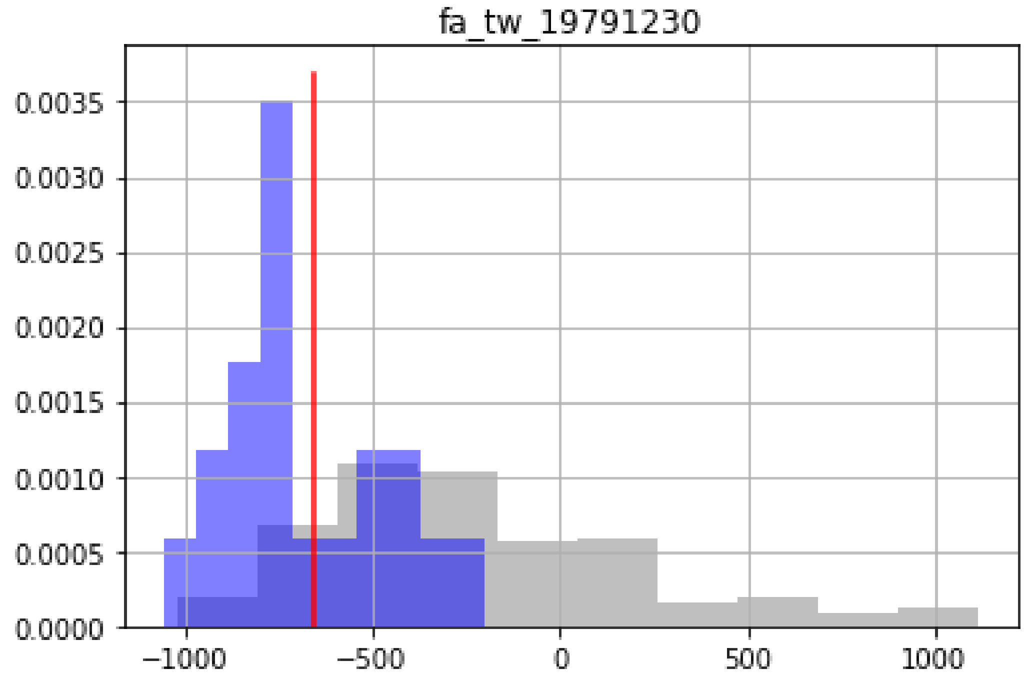


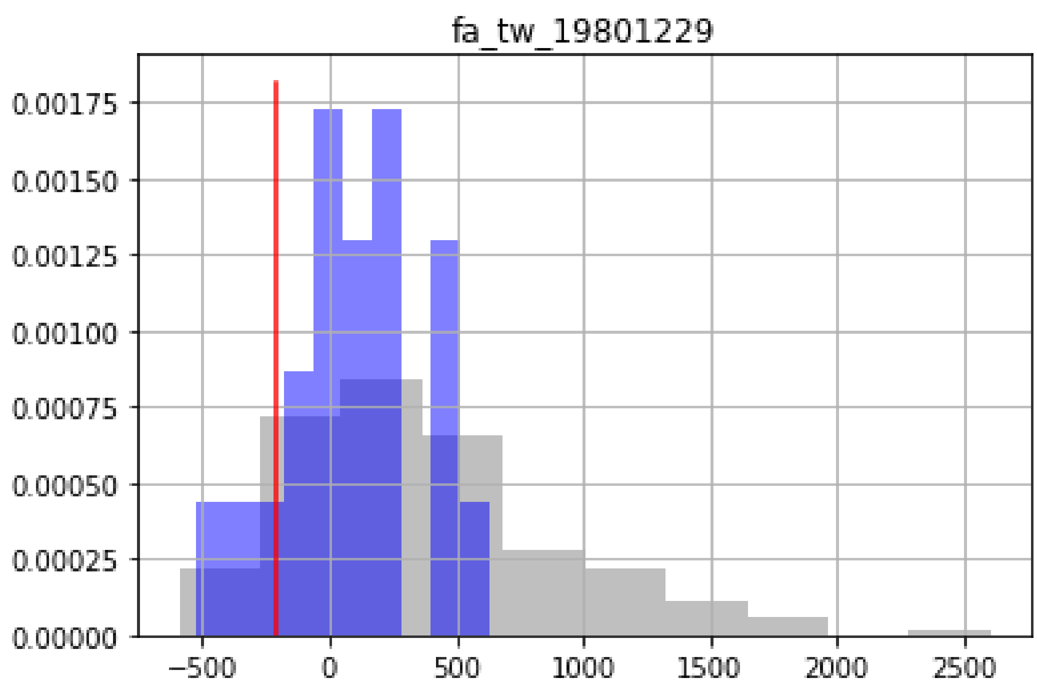

```
In [27]: oe_pt = oe.loc[oe.phi_vector.sort_values().index[:20],:]
```

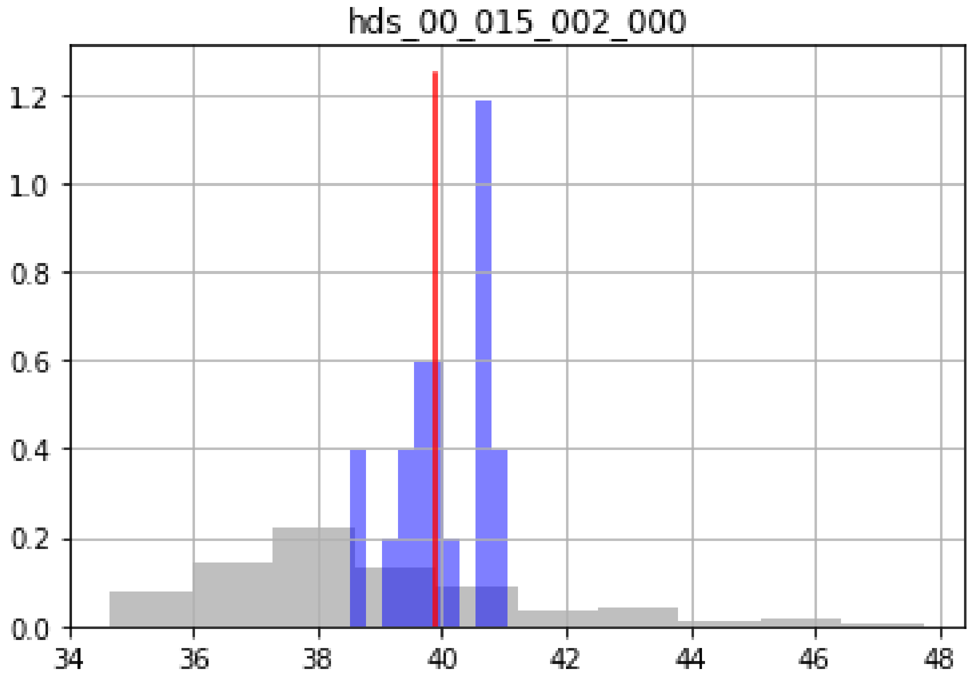
```
In [28]: obs = pst.observation_data
         fnames = pst.pestpp_options["forecasts"].split(",")
         for forecast in fnames:
             ax = plt.subplot(111)
             oe_pr.loc[:,forecast].hist(ax=ax,color="0.5",alpha=0.5,normed=True)
             oe_pt.loc[:,forecast].hist(ax=ax,color="b",alpha=0.5,normed=True)

             ax.plot([obs.loc[forecast,"obsval"],obs.loc[forecast,"obsval"]],ax.get_ylim(),"r"
             ax.set_title(forecast)
             plt.show()
```


fa_hw_19791230
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
−1500
−1000
−500
0
500
1000

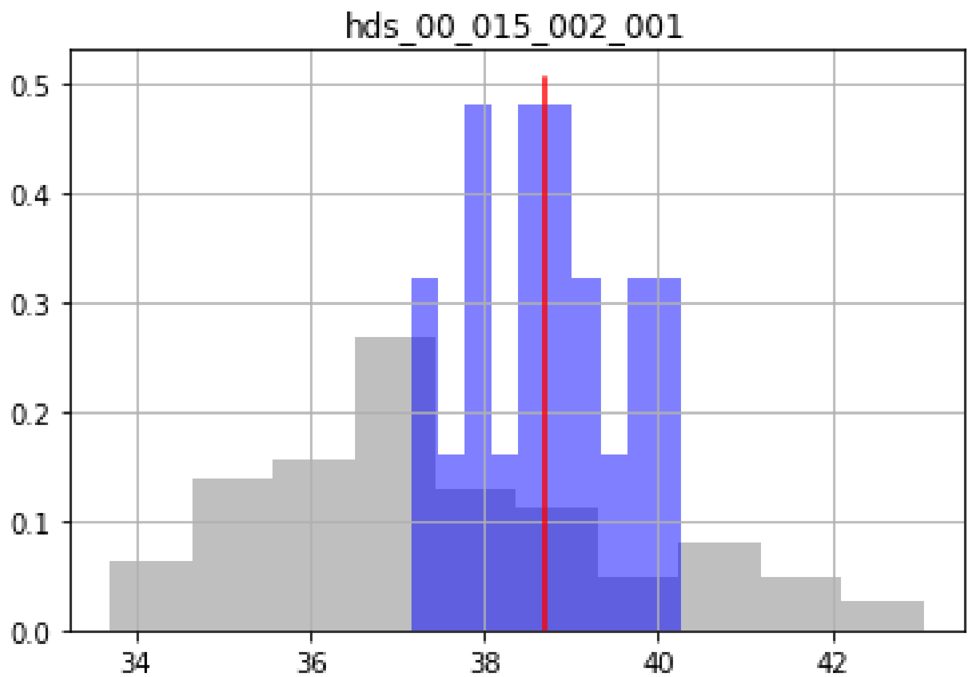

fa_hw_19801229
0.0016
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
−1000
−500
0
500
1000
1500
2000


fa_tw_19791230
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
−1000
−500
0
500
1000


fa_tw_19801229
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
−500
0
500
1000
1500
2000
2500



hds_00_015_002_000
1.2
1.0
0.8
0.6
0.4
0.2
0.0
34
36
38
40
42
44
46
48



hds_00_015_002_001
0.5
0.4
0.3
0.2
0.1
0.0
34
36
38
40
42

Damn!