pestpp-glm

April 29, 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 [2]:
```
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 [3]:
```
t_d = "template"
m_d = "master_glm"
```

In [4]:
```
pst = pyemu.Pst(os.path.join(t_d,"freyberg.pst"))
pst.write_par_summary_table(filename="none")
```

Out[4]:
```
                  type transform  count      initial value  \
grsy5            grsy5       log    705                  0
pp_ss2          pp_ss2       log     32                  0
ss6_cn          ss6_cn       log      1                  0
ss8_cn          ss8_cn       log      1                  0
vka8_cn        vka8_cn       log      1                  0
pp_ss1          pp_ss1       log     32                  0
grhk5            grhk5       log    705                  0
pp_hk1          pp_hk1       log     32                  0
strt7_cn      strt7_cn       log      1                  0
grsy4            grsy4       log    705                  0
grss4            grss4       log    705                  0
pp_strt2      pp_strt2       log     32                  0
pp_hk2          pp_hk2       log     32                  0
```

strt6_cn	strt6_cn	log	1	0
pp_rech0	pp_rech0	log	32	0
sy8_cn	sy8_cn	log	1	0
grsy3	grsy3	log	705	0
flow	flow	log	1	0
pp_strt0	pp_strt0	log	32	0
vka7_cn	vka7_cn	log	1	0
vka6_cn	vka6_cn	log	1	0
hk8_cn	hk8_cn	log	1	0
sy6_cn	sy6_cn	log	1	0
grvka5	grvka5	log	705	0
grvka4	grvka4	log	705	0
drncond_k00	drncond_k00	log	10	0
hk7_cn	hk7_cn	log	1	0
grvka3	grvka3	log	705	0
pp_strt1	pp_strt1	log	32	0
pp_sy2	pp_sy2	log	32	0
strk	strk	log	40	0
grhk4	grhk4	log	705	0
grstrt5	grstrt5	log	705	0
pp_rech1	pp_rech1	log	32	0
rech5_cn	rech5_cn	log	1	-0.39794
grss5	grss5	log	705	0
pp_vka2	pp_vka2	log	32	0
pp_ss0	pp_ss0	log	32	0
grss3	grss3	log	705	0
pp_sy1	pp_sy1	log	32	0
grstrt3	grstrt3	log	705	0
sy7_cn	sy7_cn	log	1	0
strt8_cn	strt8_cn	log	1	0
grrech3	grrech3	log	705	0
pp_vka1	pp_vka1	log	32	0
ss7_cn	ss7_cn	log	1	0
welflux_k02	welflux_k02	log	6	0
hk6_cn	hk6_cn	log	1	0
rech4_cn	rech4_cn	log	1	0
welflux	welflux	log	2	0 to 0.176091
grhk3	grhk3	log	705	0
pp_hk0	pp_hk0	log	32	0
pp_sy0	pp_sy0	log	32	0
pp_vka0	pp_vka0	log	32	0
grstrt4	grstrt4	log	705	0
grrech2	grrech2	log	705	0

	upper bound	lower bound	standard deviation
grsy5	0.243038	-0.60206	0.211275
pp_ss2	1	-1	0.5
ss6_cn	1	-1	0.5

ss8_cn	1	-1	0.5
vka8_cn	1	-1	0.5
pp_ss1	1	-1	0.5
grhk5	1	-1	0.5
pp_hk1	1	-1	0.5
strt7_cn	0.0211893	-0.0222764	0.0108664
grsy4	0.243038	-0.60206	0.211275
grss4	1	-1	0.5
pp_strt2	0.0211893	-0.0222764	0.0108664
pp_hk2	1	-1	0.5
strt6_cn	0.0211893	-0.0222764	0.0108664
pp_rech0	0.0413927	-0.0457575	0.0217875
sy8_cn	0.243038	-0.60206	0.211275
grsy3	0.243038	-0.60206	0.211275
flow	0.09691	-0.124939	0.0554622
pp_strt0	0.0211893	-0.0222764	0.0108664
vka7_cn	1	-1	0.5
vka6_cn	1	-1	0.5
hk8_cn	1	-1	0.5
sy6_cn	0.243038	-0.60206	0.211275
grvka5	1	-1	0.5
grvka4	1	-1	0.5
drncond_k00	1	-1	0.5
hk7_cn	1	-1	0.5
grvka3	1	-1	0.5
pp_strt1	0.0211893	-0.0222764	0.0108664
pp_sy2	0.243038	-0.60206	0.211275
strk	2	-2	1
grhk4	1	-1	0.5
grstrt5	0.0211893	-0.0222764	0.0108664
pp_rech1	0.0413927	-0.0457575	0.0217875
rech5_cn	-0.09691	-1	0.225772
grss5	1	-1	0.5
pp_vka2	1	-1	0.5
pp_ss0	1	-1	0.5
grss3	1	-1	0.5
pp_sy1	0.243038	-0.60206	0.211275
grstrt3	0.0211893	-0.0222764	0.0108664
sy7_cn	0.243038	-0.60206	0.211275
strt8_cn	0.0211893	-0.0222764	0.0108664
grrech3	0.0413927	-0.0457575	0.0217875
pp_vka1	1	-1	0.5
ss7_cn	1	-1	0.5
welflux_k02	1	-1	0.5
hk6_cn	1	-1	0.5
rech4_cn	0.0791812	-0.09691	0.0440228
welflux	0.176091 to 0.30103	-0.30103 to 0	0.0752575 to 0.11928
grhk3	1	-1	0.5

pp_hk0	1	-1	0.5
pp_sy0	0.243038	-0.60206	0.211275
pp_vka0	1	-1	0.5
grstrt4	0.0211893	-0.0222764	0.0108664
grrech2	0.0413927	-0.0457575	0.0217875

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]: 620
```

```
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)),"p
        par.loc[s_pars,"partrans"] = "fixed"
        pst.npar_adj
```

```
Out[8]: 620
```

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

```
Out[9]: hk7_cn          1
        vka6_cn         1
        ss6_cn          1
        ss7_cn          1
```

```
hk6_cn          1
rech4_cn        1
rech5_cn        1
vka8_cn         1
strt7_cn        1
ss8_cn          1
flow            1
strt8_cn        1
sy7_cn          1
vka7_cn         1
hk8_cn          1
sy8_cn          1
strt6_cn        1
sy6_cn          1
welflux         2
welflux_k02     6
drncond_k00    10
pp_hk0         32
pp_strt1       32
pp_sy2         32
pp_hk2         32
pp_ss1         32
pp_strt2       32
pp_hk1         32
pp_strt0       32
pp_vka1        32
pp_vka2        32
pp_vka0        32
pp_rech0       32
pp_rech1       32
pp_ss2         32
pp_ss0         32
pp_sy0         32
pp_sy1         32
strk           40
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]: 428
```

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

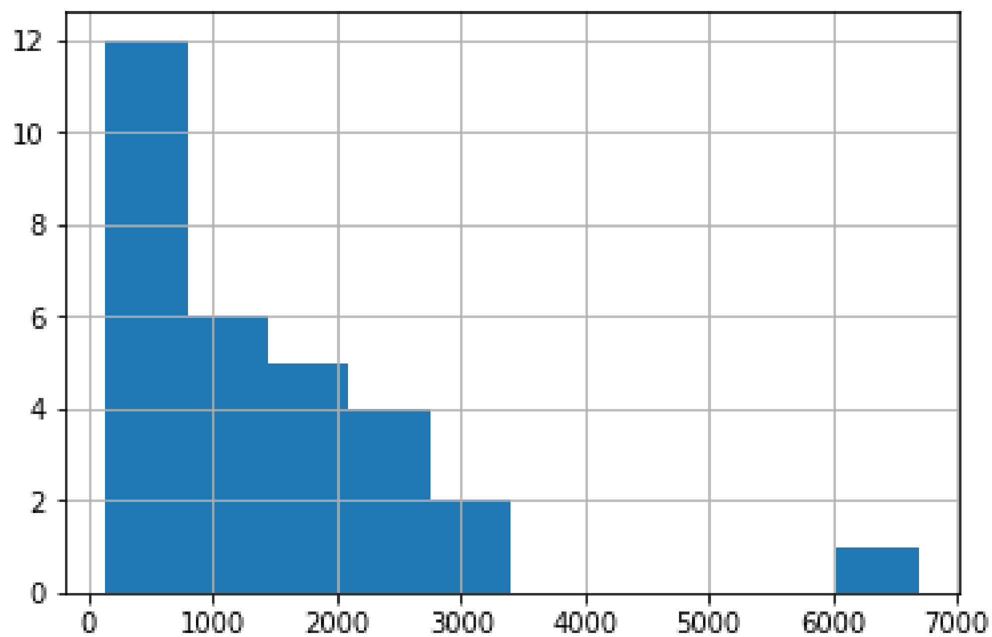
FOSM posterior parameter covariance matrix and run those 100 realizations to get the psoterior forecast PDFs

```
In [11]: pst.control_data.noptmax = 3
         pst.pestpp_options["n_iter_base"] = -1
         pst.pestpp_options["n_iter_super"] = 3
         pst.pestpp_options["num_reals"] = 50 # this is how many ies uses
         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).

To get a "posterior" ensemble, we need to throw out the realizations with large phi - lets just take the 20 best:

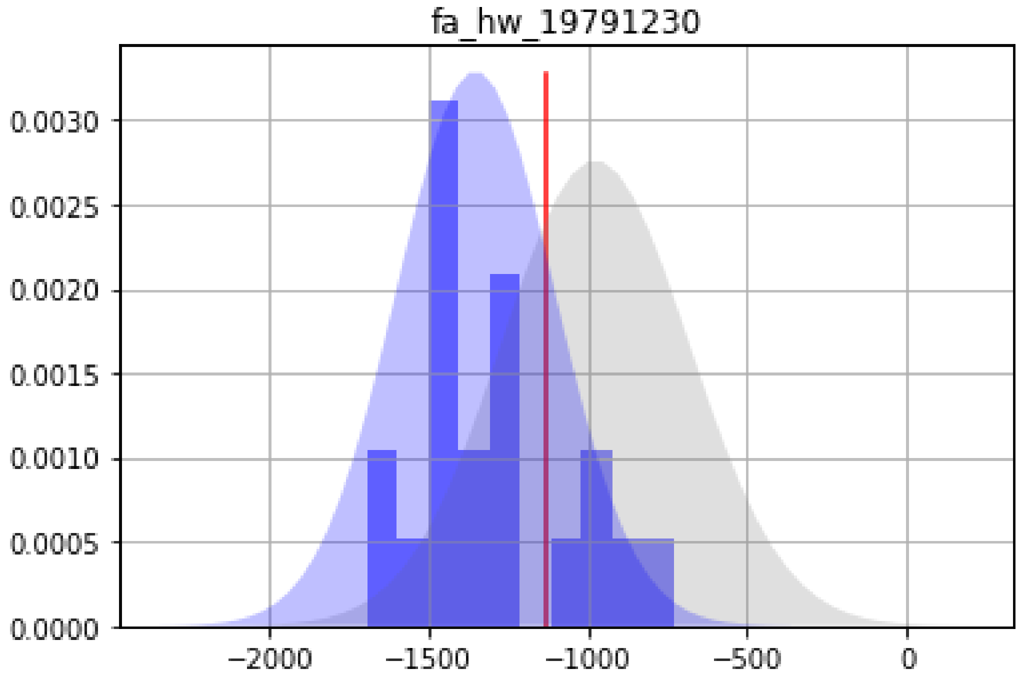
```
In [15]: oe_pt = oe.loc[oe.phi_vector.sort_values().index[:20],:] #just take the 20 lowest phi
```

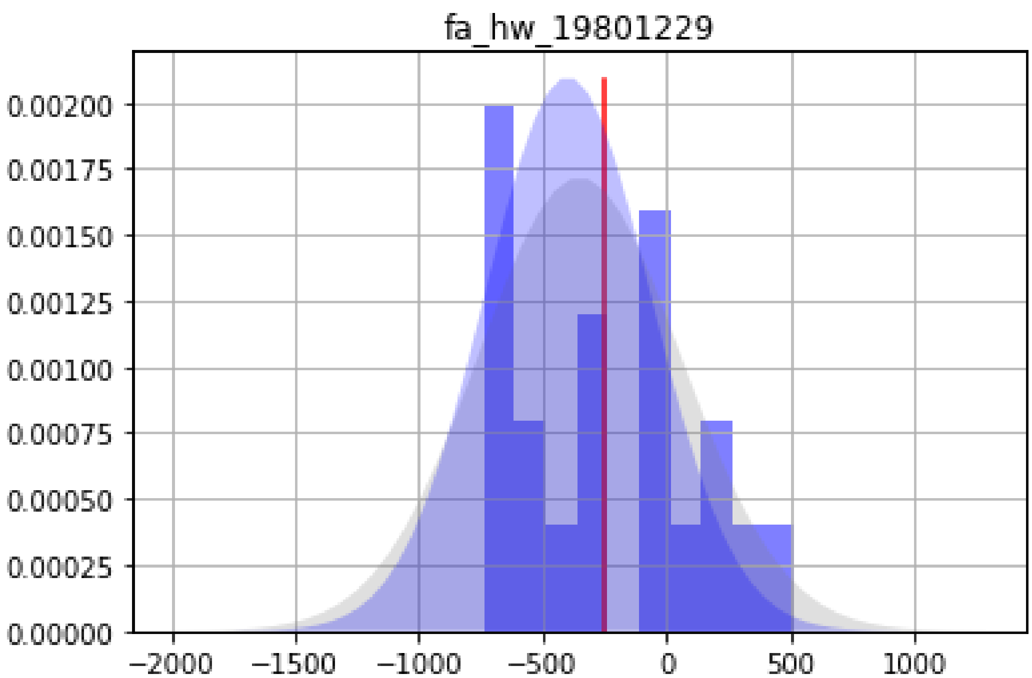
We can also load and plot the FOSM forecast results along side of the ensemble results:

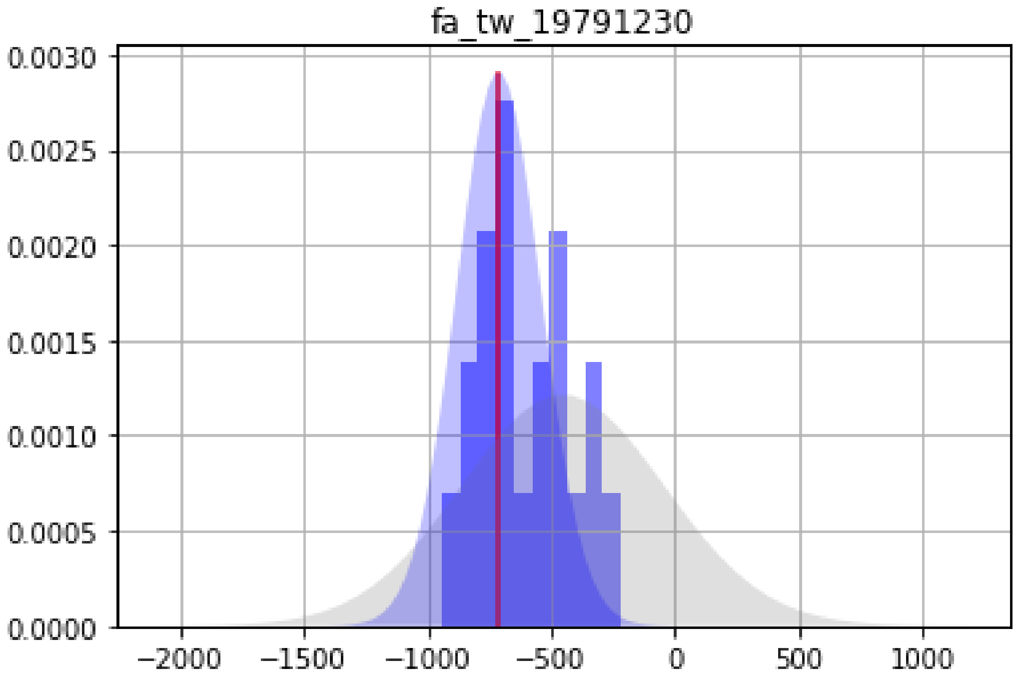
```
In [17]: f_df = pd.read_csv(os.path.join(m_d,"freyberg_pp.pred.usum.csv"),index_col=0)
         f_df.index = f_df.index.map(str.lower)
         f_df
```

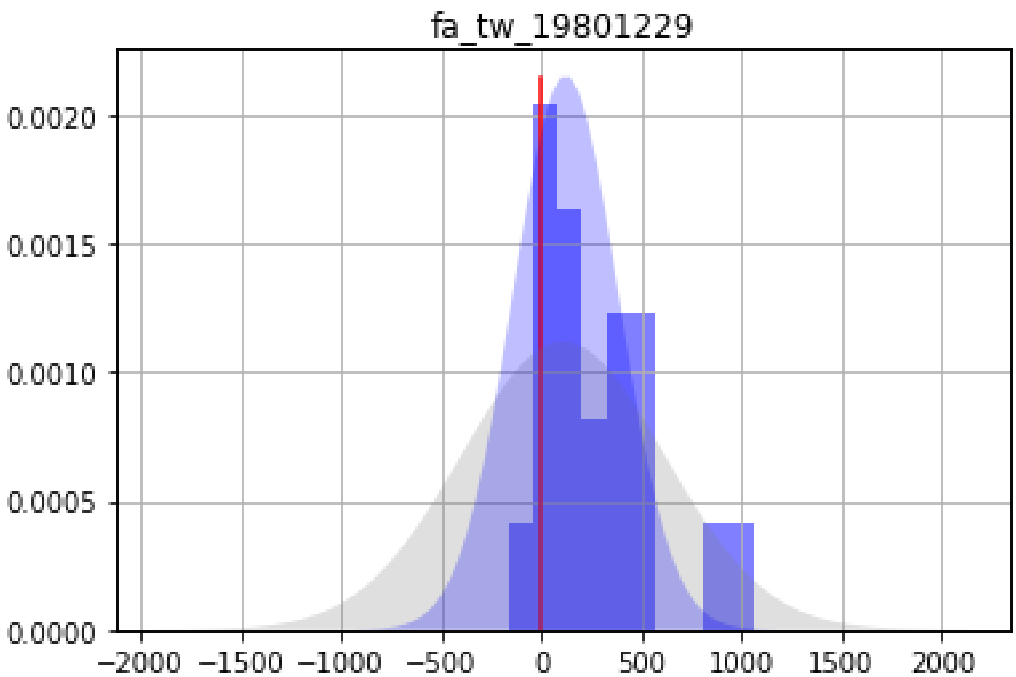
```
Out[17]:                      prior_mean  prior_stdev  prior_lower_bound  \
         name
         fa_hw_19791230        -977.2390    295.32800         -1567.8900
         fa_hw_19801229        -351.2160    409.77000         -1170.7600
         fa_tw_19791230        -453.0330    409.35100         -1271.7400
         fa_tw_19801229         108.9600    506.73200          -904.5040
         hds_00_013_002_000      39.6102      3.96314            31.6840
         hds_00_013_002_001      38.3838      4.05782            30.2681

                              prior_upper_bound  post_mean  post_stdev  \
         name
         fa_hw_19791230               -386.5840 -1353.4000  247.839000
         fa_hw_19801229                468.3240  -399.0650  335.565000
         fa_tw_19791230                365.6690  -712.0900  170.592000
         fa_tw_19801229               1122.4200   122.4170  263.672000
         hds_00_013_002_000             47.5365    39.0899    0.273990
         hds_00_013_002_001             46.4994    37.5112    0.687837

                              post_lower_bound  post_upper_bound
         name
         fa_hw_19791230             -1849.0800         -857.7250
         fa_hw_19801229             -1070.2000          272.0660
         fa_tw_19791230             -1053.2700         -370.9060
         fa_tw_19801229              -404.9260          649.7600
         hds_00_013_002_000            38.5420           39.6379
         hds_00_013_002_001            36.1355           38.8868
```

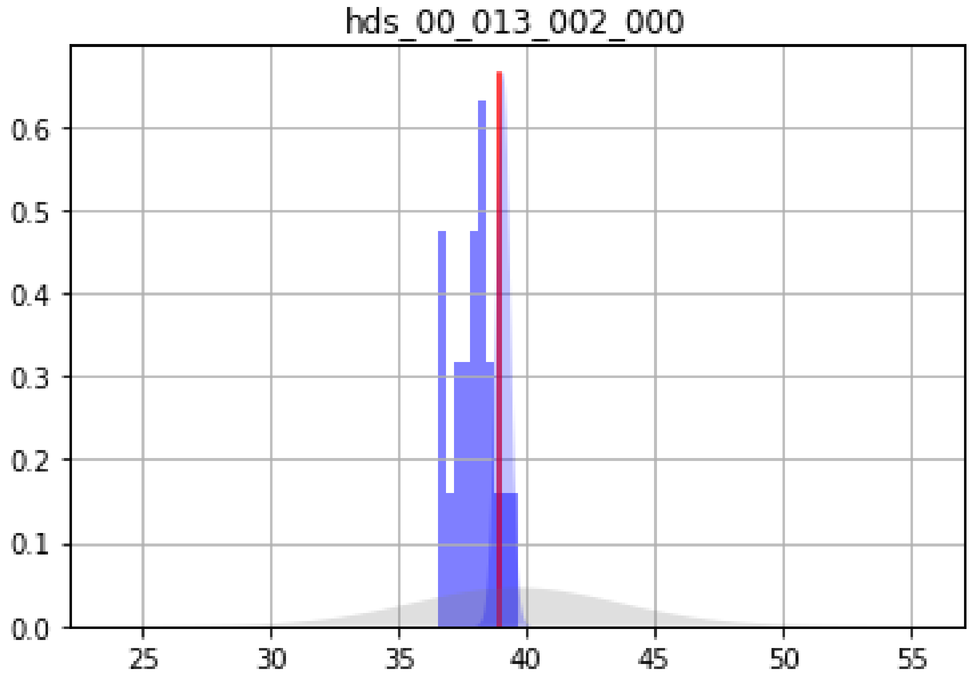
```
In [18]: obs = pst.observation_data
         fnames = pst.pestpp_options["forecasts"].split(",")
         for forecast in fnames:
             ax = plt.subplot(111)
             oe_pt.loc[:,forecast].hist(ax=ax,color="b",alpha=0.5,normed=True)
             ax.plot([obs.loc[forecast,"obsval"],obs.loc[forecast,"obsval"]],ax.get_ylim(),"r"
             axt = plt.twinx()
             x,y = pyemu.plot_utils.gaussian_distribution(f_df.loc[forecast,"prior_mean"],f_df
             axt.fill_between(x,0,y,facecolor="0.5",alpha=0.25)
             x,y = pyemu.plot_utils.gaussian_distribution(f_df.loc[forecast,"post_mean"],f_df.
             axt.fill_between(x,0,y,facecolor="b",alpha=0.25)
             axt.set_ylim(0,axt.get_ylim()[1])
             axt.set_yticks([])
             ax.set_title(forecast)
             plt.show()
```

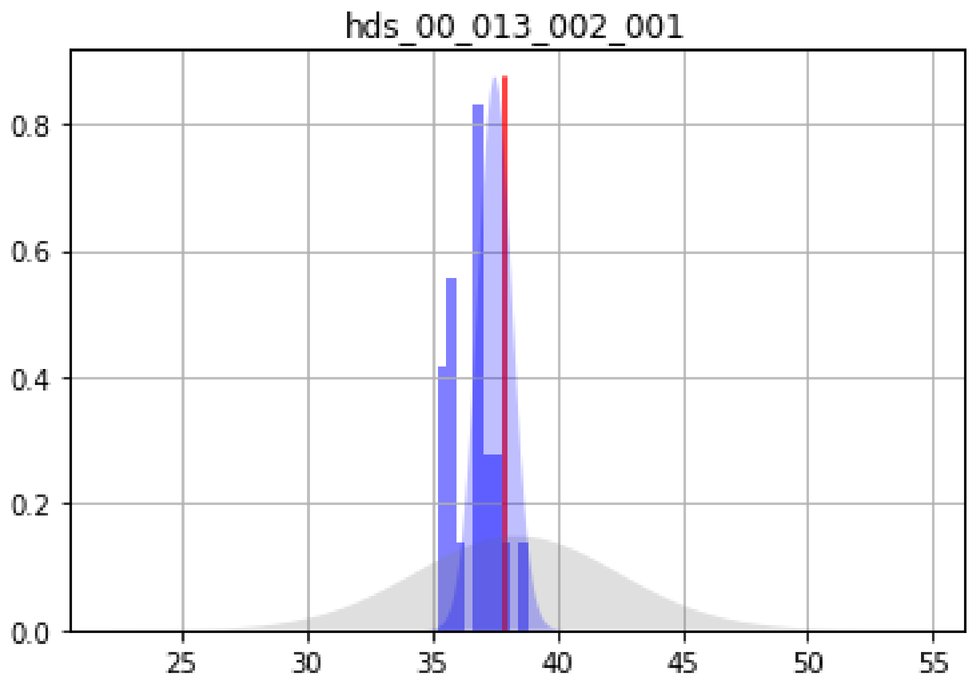

fa_hw_19791230
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
−2000
−1500
−1000
−500
0


fa_hw_19801229
0.00200
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
−2000
−1500
−1000
−500
0
500
1000


fa_tw_19791230
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
−2000
−1500
−1000
−500
0
500
1000


fa_tw_19801229
0.0020
0.0015
0.0010
0.0005
0.0000
−2000
−1500
−1000
−500
0
500
1000
1500
2000


hds_00_013_002_000
0.6
0.5
0.4
0.3
0.2
0.1
0.0
25
30
35
40
45
50
55


hds_00_013_002_001
0.8
0.6
0.4
0.2
0.0
25
30
35
40
45
50
55

1.0.2 Setup of Tikhonov regularization

Now 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 [19]: cov = pyemu.Cov.from_binary(os.path.join(t_d,"prior_cov.jcb"))
```

```
In [20]: cnames = set(cov.row_names)
         pnames = set(pst.adj_par_names)
         cnames.symmetric_difference(pnames)
```

```
Out[20]: {'ss3024002',
          'sy3015015',
          'sy3006002',
          'vka4005004',
          'ss4016011',
          'strt3021007',
          'hk3025007',
          'strt5038014',
          'ss5036017',
          'sy5005001',
          'sy3028008',
          'sy3013001',
          'rech2028002',
          'ss3035012',
          'vka5029019',
          'rech3008007',
          'vka3012010',
          'hk4027011',
          'vka3003005',
          'vka5024013',
          'vka5000015',
          'ss4024018',
          'vka5021008',
          'hk5012010',
          'ss4009001',
          'sy3032008',
          'sy4008008',
          'ss5019016',
          'ss3020015',
          'hk3014010',
          'vka5026009',
          'hk5033014',
          'vka3006010',
          'sy5005018',
          'hk3039007',
          'strt5006015',
          'hk5010008',
          'strt4024002',
```

```
'sy5012019',
'strt4013003',
'ss3016017',
'sy4000008',
'vka4032002',
'ss3005002',
'hk4028007',
'vka3037011',
'strt5013015',
'strt4017012',
'hk4033018',
'hk5008010',
'strt5021015',
'sy5029001',
'strt4001006',
'vka5033002',
'rech2001018',
'sy3001007',
'hk3023011',
'rech3023018',
'strt016',
'hk5023008',
'strt108',
'vka3017018',
'sy5003013',
'sy5036012',
'ss5038010',
'strt3029003',
'strt4029002',
'hk5018003',
'sy4029017',
'sy5009018',
'hk3030005',
'sy5023012',
'rech3032012',
'vka5008010',
'vka3011013',
'hk5005014',
'sy4031012',
'strt5019012',
'vka4023018',
'sy3010018',
'vka4008003',
'sy5027005',
'rech2033010',
'sy5022009',
'rech122',
'rech2001011',
```

```
'strt3026016',
'sy3002014',
'sy5027007',
'strt3034011',
'strt3007003',
'ss4033011',
'strt5002019',
'vka4037017',
'hk5008017',
'rech3031007',
'vka3032013',
'ss3026017',
'hk5008002',
'rech2036004',
'strt5007016',
'hk5030004',
'vka3015012',
'ss4036009',
'vka3005007',
'ss4005009',
'rech2036013',
'ss5022016',
'ss5002012',
'ss4035016',
'sy3015011',
'sy4019011',
'rech2037010',
'vka5024009',
'sy5031001',
'ss3004008',
'hk5029010',
'strt4004006',
'hk5028016',
'vka5011017',
'sy3001004',
'hk4024009',
'vka3028000',
'hk5022010',
'ss3017018',
'sy3024017',
'ss5024015',
'hk5031011',
'vka3006005',
'ss4026001',
'strt3015012',
'hk4020019',
'vka5018003',
'ss4039010',
```

```
'vka3036006',
'strt3039007',
'vka3014018',
'vka5030008',
'hk4029017',
'strt218',
'ss3019018',
'sy3001012',
'sy5003004',
'strt3017002',
'strt4011001',
'ss5018000',
'vka4023002',
'vka3001011',
'hk4023017',
'sy5020012',
'hk3027010',
'strt4023005',
'sy3037007',
'sy4025009',
'strt3009010',
'vka3019002',
'rech2003011',
'sy5025010',
'hk5033008',
'strt009',
'sy4031007',
'hk5005003',
'sy4016014',
'sy3019013',
'ss3038006',
'ss5016009',
'strt5009002',
'hk5005008',
'rech3014015',
'vka3013008',
'ss4023004',
'sy3032014',
'sy3036016',
'vka3000005',
'vka3037015',
'vka5005018',
'hk4035018',
'rech3001015',
'rech3033006',
'sy3006000',
'strt4006014',
'hk3032002',
```

```
'strt4030010',
'ss5028016',
'vka4024014',
'strt5020012',
'vka212',
'sy4005012',
'hk3011000',
'rech3032010',
'ss5031017',
'hk3038010',
'hk3007017',
'rech2037013',
'sy4033012',
'ss5004003',
'vka4030014',
'strt4004004',
'vka5024017',
'strt3013017',
'hk5010009',
'vka5033009',
'rech3019003',
'vka4023019',
'hk3032016',
'ss5034004',
'vka4034009',
'sy4033003',
'ss5020009',
'sy3035014',
'vka3029013',
'vka5005007',
'sy3029006',
'vka5008018',
'sy4001001',
'ss3021007',
'sy4023010',
'ss3009012',
'strt3003015',
'vka3033018',
'vka201',
'vka4026006',
'ss3007002',
'sy4017002',
'hk3027000',
'ss5021018',
'hk3010010',
'ss3002003',
'sy4020016',
'vka5008001',
```

```
'strt4015018',
'sy5005015',
'vka4023008',
'ss4035007',
'sy3004016',
'rech3023004',
'strt5004017',
'hk4022007',
'rech3024004',
'strt5031009',
'sy3014008',
'strt4008014',
'hk4013000',
'vka5019008',
'rech2000014',
'sy4012016',
'strt4028008',
'ss4014019',
'ss3021000',
'strt3032004',
'hk5000008',
'ss3022016',
'rech3027005',
'hk3016017',
'hk3008008',
'strt4025008',
'sy4006000',
'hk4022015',
'strt4015009',
'ss5018011',
'hk4027006',
'hk5005013',
'hk5027015',
'ss5012014',
'strt5014018',
'ss5037008',
'strt5002015',
'vka4034011',
'sy3005014',
'strt4019017',
'strt5031017',
'sy3036014',
'vka3031016',
'strt3023013',
'hk5027010',
'strt5037013',
'vka3017016',
'hk3038015',
```

```
'ss4027007',
'sy5016008',
'rech2027004',
'strt4033007',
'rech2028007',
'vka4019012',
'strt4036014',
'hk4002002',
'sy5034012',
'hk4003015',
'hk5038016',
'rech3034019',
'hk3031015',
'ss3002008',
'hk4026008',
'ss3013003',
'strt5017019',
'rech2019003',
'strt4014002',
'vka5004011',
'hk3004005',
'rech2018013',
'hk5030014',
'rech2013017',
'ss5007011',
'strt3013013',
'sy5032010',
'vka5020010',
'rech105',
'sy4022008',
'ss3018018',
'vka5007008',
'vka3035010',
'vka5031006',
'sy5024003',
'ss4000006',
'ss4005011',
'sy5008013',
'strt3015013',
'strt4015016',
'sy4028009',
'hk3023001',
'rech2032009',
'strt3031004',
'hk5028006',
'hk4024016',
'strt5022009',
'hk4029015',
```

```
'hk3001008',
'ss3010019',
'hk3005017',
'ss5001006',
'hk5021001',
'ss3031019',
'sy3009019',
'vka4038014',
'strt5025016',
'sy5032019',
'ss4017018',
'sy5019011',
'hk4007015',
'strt5020018',
'ss3029019',
'sy3005013',
'ss5034016',
'ss3035018',
'strt3005004',
'hk3011010',
'strt5012012',
'hk3002012',
'rech3035004',
'rech2032013',
'ss4005006',
'vka3023007',
'hk4024005',
'strt3026015',
'strt5008014',
'sy5037017',
'strt5006002',
'rech2032008',
'sy4030002',
'strt5023011',
'sy3016017',
'hk3015001',
'strt5034008',
'sy4002014',
'hk3011002',
'sy5023002',
'strt4023002',
'sy5030011',
'ss5037005',
'rech2001000',
'strt4008000',
'sy5032016',
'strt5002014',
'sy3038012',
```

```
'vka3033005',
'sy4004000',
'sy5004011',
'vka4027018',
'ss5000003',
'rech3008000',
'vka3023006',
'ss3035016',
'strt3021010',
'vka5005013',
'ss5006006',
'sy4001014',
'ss3006008',
'rech2000011',
'hk4025015',
'rech3015003',
'sy4012008',
'sy4008003',
'hk4026010',
'vka3030009',
'ss3037008',
'ss5027016',
'vka3031014',
'sy5011009',
'ss4002019',
'hk3006005',
'sy4008012',
'strt5029001',
'vka5025018',
'hk3032005',
'rech3000019',
'sy4017018',
'ss3026003',
'sy5024018',
'hk3024011',
'sy5004000',
'sy5020015',
'strt4002014',
'sy3003010',
'ss5005017',
'rech3019011',
'vka5020003',
'rech2021011',
'vka5025009',
'sy5038014',
'hk5009010',
'hk4013015',
'sy3034003',
```

```
'sy3002004',
'vka5028011',
'sy3018018',
'rech2003013',
'ss3002005',
'hk5024015',
'sy4037017',
'hk5036012',
'rech2029002',
'vka4008014',
'ss3006018',
'vka4028003',
'sy3007017',
'sy4024010',
'ss3036018',
'vka3011008',
'rech3003003',
'rech3039014',
'vka4004000',
'strt029',
'sy3033012',
'vka4025018',
'hk3026012',
'ss5024011',
'sy3019019',
'strt4016002',
'strt4018009',
'sy4013009',
'ss5010018',
'ss5029003',
'strt5003019',
'rech3020003',
'strt4002012',
'hk3035017',
'vka4020016',
'hk4026019',
'ss5033007',
'ss4025013',
'sy4009014',
'strt3032009',
'sy4007012',
'sy3000004',
'sy3001005',
'ss3006004',
'vka5036007',
'sy4022002',
'vka5000017',
'ss4014000',
```

```
'rech2008008',
'strt3035010',
'hk4014018',
'vka3032007',
'hk5019003',
'hk5005004',
'ss4014014',
'hk3004013',
'strt3008001',
'hk3027008',
'vka3030019',
'vka030',
'strt3003011',
'sy4024000',
'vka4015016',
'rech2009016',
'sy5030009',
'hk3016019',
'strt4005009',
'strt4018002',
'rech3035017',
'sy4026001',
'rech2014017',
'hk4026001',
'hk3030001',
'hk5010003',
'rech3033019',
'vka4028016',
'hk3002005',
'rech3027019',
'hk4009017',
'strt3026018',
'vka3033007',
'ss5037006',
'vka4026008',
'sy3030001',
'sy5002005',
'ss3025017',
'ss5036018',
'sy5023013',
'ss5027008',
'strt4030008',
'sy4031006',
'hk3026003',
'sy5023011',
'strt4019012',
'vka5036010',
'hk3017003',
```

```
'ss3036006',
'rech3009000',
'strt4016013',
'vka3035012',
'vka4023016',
'sy3033014',
'hk5029013',
'sy4024011',
'sy3016018',
'vka4005018',
'rech2015015',
'strt5007001',
'strt5013017',
'sy3027007',
'vka3032005',
'ss4012000',
'ss4006012',
'strt3030009',
'strt3002012',
'sy4036012',
'vka4033010',
'hk5010001',
'hk3005014',
'ss3031006',
'sy5025008',
'rech2031013',
'ss4007015',
'strt3001016',
'strt5015009',
'strt4030009',
'ss3030004',
'rech2009010',
'sy4034015',
'hk5005015',
'vka3009003',
'vka4006019',
'vka4019003',
'strt3000016',
'vka3030004',
'sy5035017',
'hk5023014',
'ss3031012',
'strt3000015',
'rech3011019',
'vka5027004',
'sy3033003',
'sy3015002',
'rech2031009',
```

```
'ss4020019',
'vka5025001',
'hk3017013',
'hk4007006',
'strt5008008',
'hk4015017',
'vka4029005',
'hk5012011',
'sy3017001',
'ss4027005',
'strt5010011',
'rech2033002',
'rech2036008',
'sy5005004',
'ss5030012',
'sy5032007',
'rech3023019',
'ss3026012',
'hk3032007',
'ss3038015',
'strt4029001',
'strt4003002',
'strt3030019',
'ss4039009',
'vka3017001',
'vka015',
'strt4025007',
'hk5036016',
'sy5031000',
'vka3035004',
'hk3011001',
'hk4000012',
'rech3001010',
'vka5039011',
'ss5032005',
'ss5024002',
'strt4019014',
'sy5014013',
'sy3029017',
'rech3014000',
'strt3018010',
'sy3004004',
'ss5031009',
'strt3004004',
'sy5008000',
'vka3023015',
'vka4005009',
'sy4000017',
```

```
'hk3024005',
'vka4036006',
'vka4010003',
'strt5006019',
'sy4007004',
'vka3027016',
'vka4003006',
'sy3022000',
'sy5037015',
'hk5030000',
'vka220',
'ss3006003',
'hk4001009',
'ss3036011',
'vka3016010',
'hk3017002',
'sy5032002',
'hk4016013',
'vka4019019',
'strt3033005',
'hk3000007',
'strt4010015',
'sy5020009',
'sy5035009',
'vka5022012',
'hk4006009',
'sy5016011',
'vka5032007',
'rech2021018',
'vka4019015',
'sy5002001',
'strt4003015',
'ss5006016',
'rech2008011',
'strt5009009',
'sy4037005',
'ss4024000',
'hk5009000',
'vka4031012',
'strt4000018',
'strt5011010',
'rech131',
'rech3014008',
'ss4020003',
'ss4009010',
'rech2002001',
'strt3003008',
'ss5009014',
```

```
'sy5017002',
'vka5026000',
'rech2024016',
'sy3026003',
'strt5010015',
'hk4039012',
'sy4028007',
'vka3016013',
'ss5031006',
'hk5002008',
'sy5032011',
'rech2013014',
'rech3028008',
'hk5029005',
'strt018',
'ss3027005',
'hk3008010',
'strt126',
'strt5033010',
'rech3006003',
'vka4019018',
'sy5039007',
'strt5017017',
'ss3029000',
'vka3004012',
'rech3004008',
'ss4028005',
'rech2020012',
'ss3011003',
'hk5009017',
'ss4024003',
'hk3023006',
'hk3036008',
'strt4024001',
'rech2017002',
'sy5015003',
'hk4005015',
'hk3018019',
'sy5018017',
'strt3006010',
'hk4019001',
'strt5025009',
'ss5028015',
'strt3033017',
'sy4005003',
'vka4003010',
'vka5037010',
'ss5025012',
```

```
'rech2006002',
'sy5012017',
'hk5019014',
'ss4034003',
'hk5021010',
'vka4008011',
'sy4030016',
'rech2026007',
'strt3005017',
'strt4035005',
'vka5010015',
'ss4021015',
'sy3031014',
'hk5027017',
'sy3038016',
'rech2020003',
'hk3007019',
'strt3032006',
'sy5008009',
'hk3026010',
'hk4017003',
'sy4003006',
'hk3017009',
'hk5007006',
'hk3015008',
'strt3036007',
'sy4027007',
'ss4033013',
'hk3000016',
'vka3024004',
'ss5028010',
'strt3038010',
'rech3031002',
'ss4022009',
'hk4009015',
'strt4034016',
'hk4005014',
'vka4037012',
'vka5013012',
'hk3035007',
'strt3001005',
'vka5034013',
'rech3005009',
'rech2038012',
'hk4003011',
'vka3013017',
'ss3005014',
'vka4001001',
```

```
'hk5024017',
'hk3012016',
'sy5021018',
'strt3012018',
'ss4022018',
'ss3024000',
'hk3021000',
'strt3028007',
'vka5021003',
'rech119',
'vka5028018',
'hk5001006',
'strt111',
'vka3008000',
'strt5014011',
'vka5025002',
'vka5004019',
'ss5033010',
'vka3032009',
'ss5001001',
'vka4024012',
'strt4009011',
'vka4029000',
'strt3033007',
'sy5027019',
'hk4025016',
'vka001',
'vka5034014',
'rech3010003',
'sy5036017',
'strt4000019',
'sy4023009',
'hk4015015',
'sy3003011',
'rech2012002',
'ss5026011',
'strt3009003',
'ss5028000',
'ss3003008',
'vka3030007',
'hk5001016',
'vka3012016',
'hk3029015',
'sy5028014',
'vka4003019',
'ss4002003',
'strt3002018',
'strt5024010',
```

```
'ss3005015',
'vka5037012',
'sy3005011',
'strt3020011',
'rech2034004',
'ss4015009',
'strt5007012',
'strt4007004',
'sy4000000',
'vka4031001',
'ss4002011',
'sy3031006',
'vka5023015',
'vka4019001',
'vka5010003',
'ss5011003',
'strt4027006',
'rech2038007',
'vka3032016',
'rech2002004',
'hk5032006',
'rech3020001',
'rech2027008',
'ss3009014',
'hk3032004',
'hk5033009',
'rech2029000',
'ss5003004',
'strt5013019',
'rech3000004',
'rech3022000',
'vka4013000',
'vka5021006',
'sy5032015',
'strt3001002',
'ss3030018',
'hk5007017',
'sy3026013',
'vka3020016',
'strt5004019',
'strt5013013',
'sy5016015',
'vka5002007',
'ss5033006',
'sy3018001',
'rech2007013',
'sy3035011',
'ss5005004',
```

```
'ss5025008',
'strt3000019',
'rech2031016',
'hk5038013',
'rech3022014',
'vka5011010',
'strt3011016',
'hk4002004',
'hk4010009',
'ss5004015',
'rech3002008',
'ss3012018',
'sy4007009',
'rech2007001',
'rech2029011',
'sy4037010',
'ss5011018',
'ss4029004',
'strt5028006',
'ss4003003',
'strt3013019',
'strt4024009',
'ss3005007',
'hk3010019',
'ss3034014',
'sy3022015',
'rech2010000',
'sy4006017',
'sy4025003',
'strt3024014',
'ss4011001',
'ss5004002',
'sy4021014',
'vka3026013',
'sy4037008',
'hk5015016',
'ss3006014',
'hk3002010',
'sy3027017',
'strt5010008',
'rech3021016',
'ss4032003',
'ss5021019',
'sy4002018',
'rech3005000',
'rech2014010',
'ss3000016',
'rech2030018',
```

```
'hk3037015',
'ss4025005',
'hk3007011',
'sy3008008',
'vka3021014',
'strt3008015',
'sy4037006',
'vka200',
'sy3031000',
'sy3039009',
'ss5018016',
'hk3019003',
'sy4022011',
'hk5004004',
'hk5002007',
'sy4015017',
'ss4030007',
'ss5030018',
'vka3013009',
'sy3017009',
'sy5036004',
'strt4026000',
'vka5031011',
'ss5031010',
'vka4035012',
'hk5035014',
'strt3001008',
'vka5026013',
'strt4024000',
'sy5008014',
'sy5026015',
'strt4014001',
'ss3019011',
'ss5028014',
'sy3031009',
'vka3000007',
'ss4013008',
'rech2027015',
'hk4001006',
'rech2035015',
'sy5011018',
'vka3013018',
'vka225',
'hk5029003',
'hk5024007',
'hk5021002',
'hk5031009',
'sy4015002',
```

```
'rech3006004',
'strt4022002',
'rech2030001',
'strt5035011',
'strt5020015',
'ss3014013',
'sy4018019',
'vka4022006',
'sy4030010',
'strt5024004',
'hk3008015',
'sy3011010',
'sy5015019',
'rech2025006',
'hk4010003',
'hk3022001',
'rech3017008',
'ss5032003',
'rech2031000',
'strt5026018',
'vka3029014',
'hk5018013',
'vka4002008',
'vka3027010',
'ss5037014',
'ss4015003',
'hk4005009',
'hk5019010',
'ss5012017',
'vka4035017',
'strt4031012',
'sy3030009',
'rech2027010',
'strt4008010',
'hk4033005',
'sy4001008',
'rech3002016',
'sy4026008',
'rech2006013',
'vka3018014',
'strt3003010',
'sy3017016',
'rech3032009',
'sy4007001',
'hk4003007',
'hk5015002',
'ss5024000',
'rech3021015',
```

```
'vka3031011',
'hk3027005',
...}
```

In [21]:
```
pyemu.helpers.first_order_pearson_tikhonov(pst,cov)
```

```
getting CC matrix
processing
```

In [22]:
```
pst.prior_information.head()
```

Out[22]:
```
                                                    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 [23]:
```
shutil.copy2(os.path.join(m_d,"freyberg_pp.jcb"),os.path.join(t_d,"restart_pp.jcb"))
```

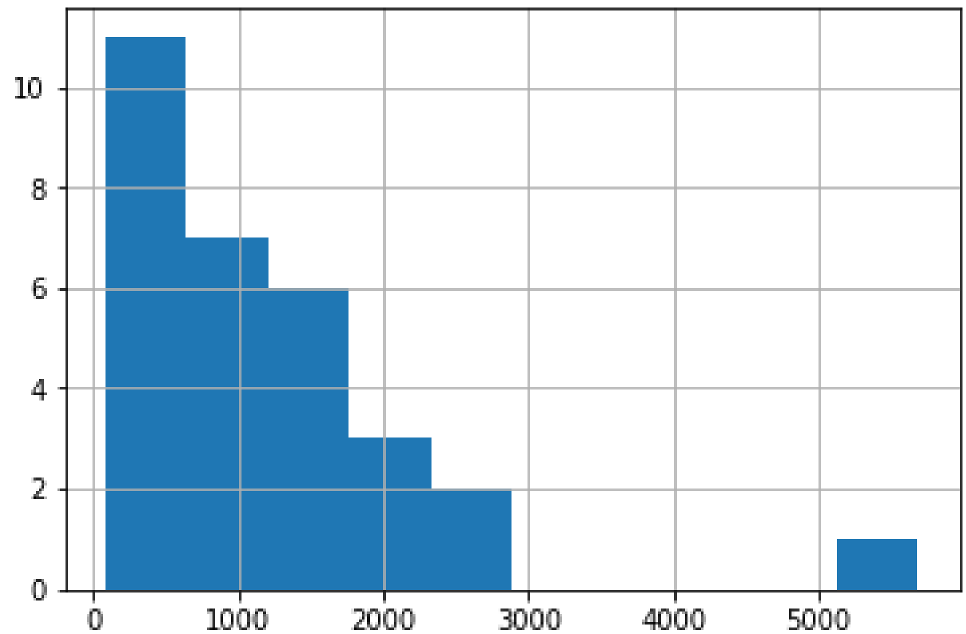
Out[23]:
```
'template/restart_pp.jcb'
```

In [24]:
```
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 [25]:
```
pyemu.os_utils.start_slaves(t_d,"pestpp-glm","freyberg_pp.pst",num_slaves=20,slave_roo
                           master_dir=m_d)
```

In [26]:
```
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 [27]:
```
ax = oe.phi_vector.hist()#bins=np.linspace(0,100,20))
```



Same as before, to get a "posterior" ensemble, we need to throw out the realizations with large phi - lets just take the 20 best:

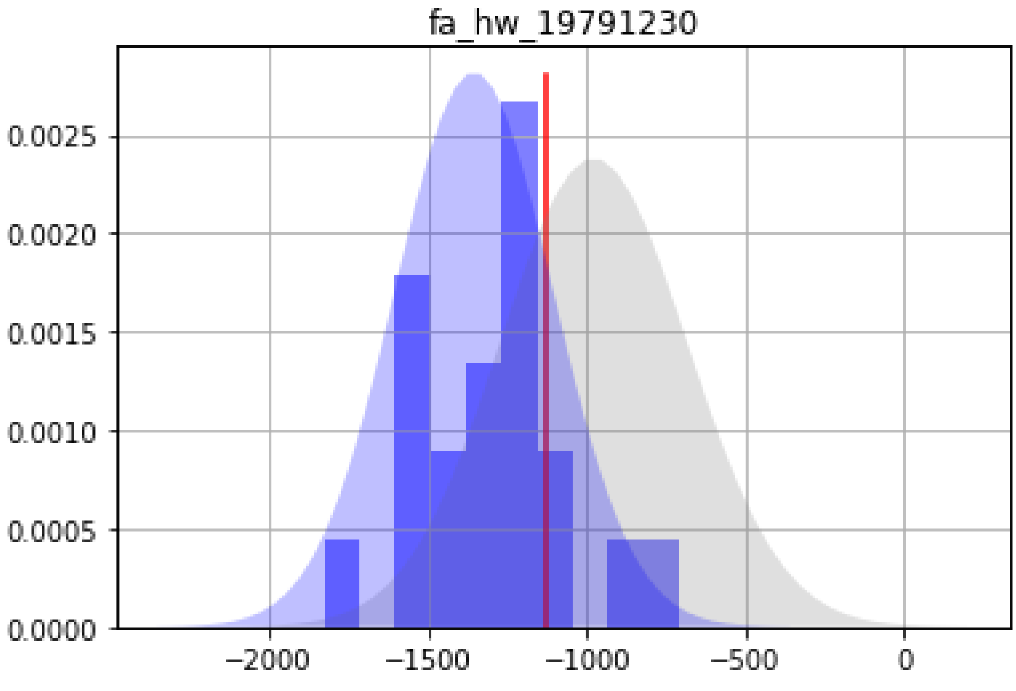
In [28]:
```
oe_pt = oe.loc[oe.phi_vector.sort_values().index[:20],:]
```

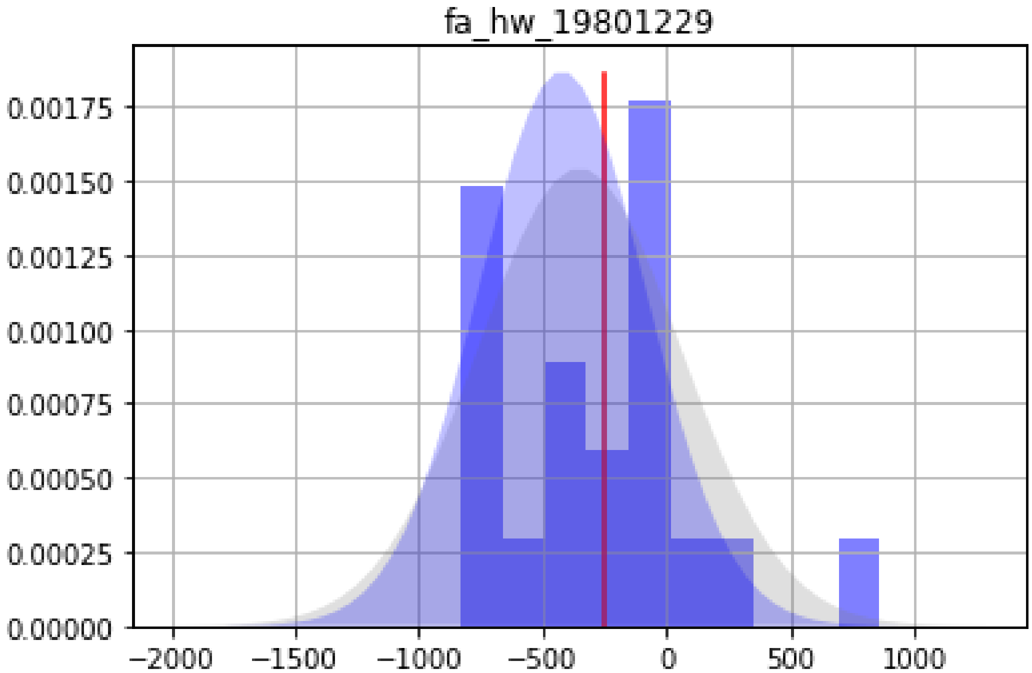
In [29]:
```
f_df = pd.read_csv(os.path.join(m_d,"freyberg_pp.pred.usum.csv"),index_col=0)
f_df.index = f_df.index.map(str.lower)
f_df
```

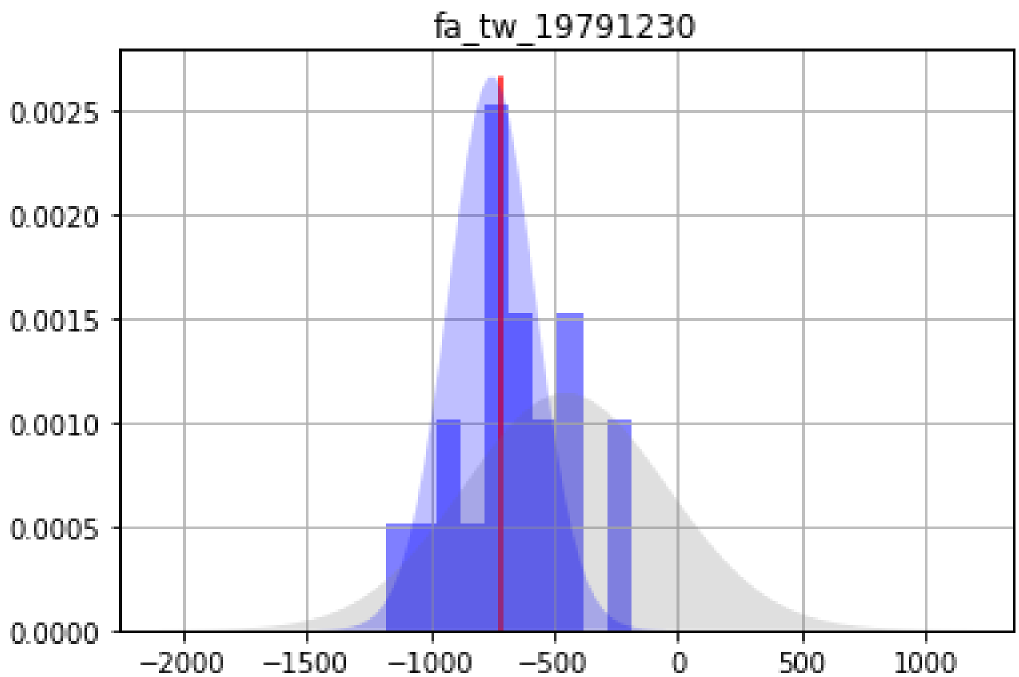
Out[29]:
```
                     prior_mean  prior_stdev  prior_lower_bound  \
name
fa_hw_19791230        -977.2390    295.32800         -1567.8900
fa_hw_19801229        -351.2160    409.77000         -1170.7600
fa_tw_19791230        -453.0330    409.35100         -1271.7400
fa_tw_19801229         108.9600    506.73200          -904.5040
hds_00_013_002_000      39.6102      3.96314            31.6840
hds_00_013_002_001      38.3838      4.05782            30.2681

                     prior_upper_bound  post_mean  post_stdev  \
name
fa_hw_19791230               -386.5840 -1354.3600  249.469000
fa_hw_19801229                468.3240  -421.9530  338.242000
fa_tw_19791230                365.6690  -751.4970  175.174000
fa_tw_19801229               1122.4200    72.8314  267.254000
hds_00_013_002_000             47.5365    39.1835    0.281947
hds_00_013_002_001             46.4994    37.6265    0.691057
```

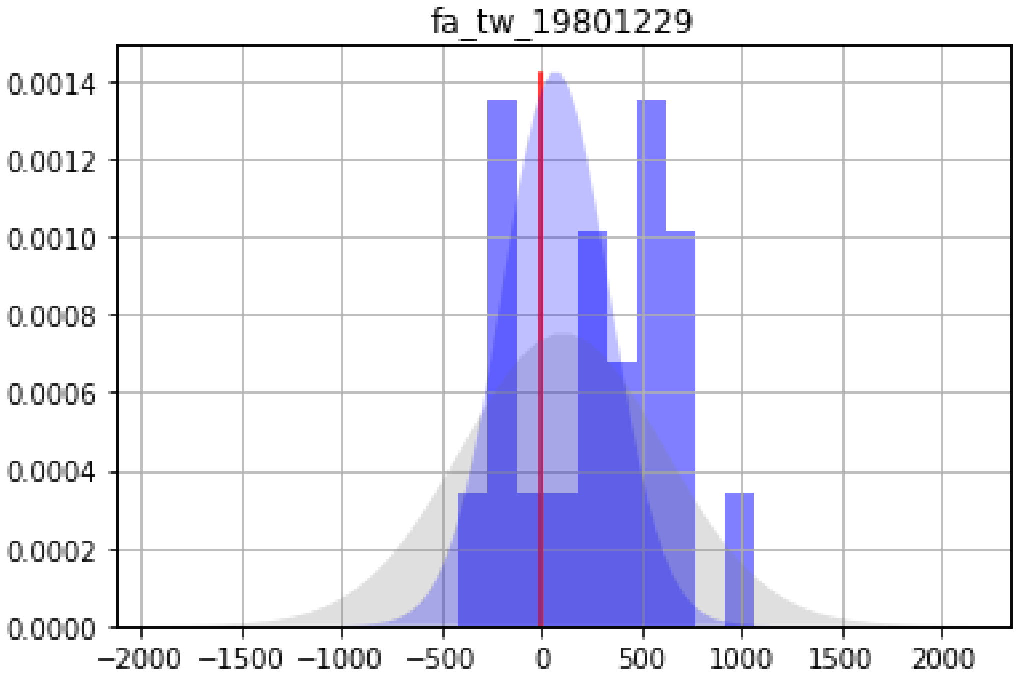
```
                       post_lower_bound  post_upper_bound
name
fa_hw_19791230              -1853.2900         -855.4170
fa_hw_19801229              -1098.4400          254.5310
fa_tw_19791230              -1101.8400         -401.1500
fa_tw_19801229               -461.6770          607.3400
hds_00_013_002_000             38.6196           39.7474
hds_00_013_002_001             36.2444           39.0086
```

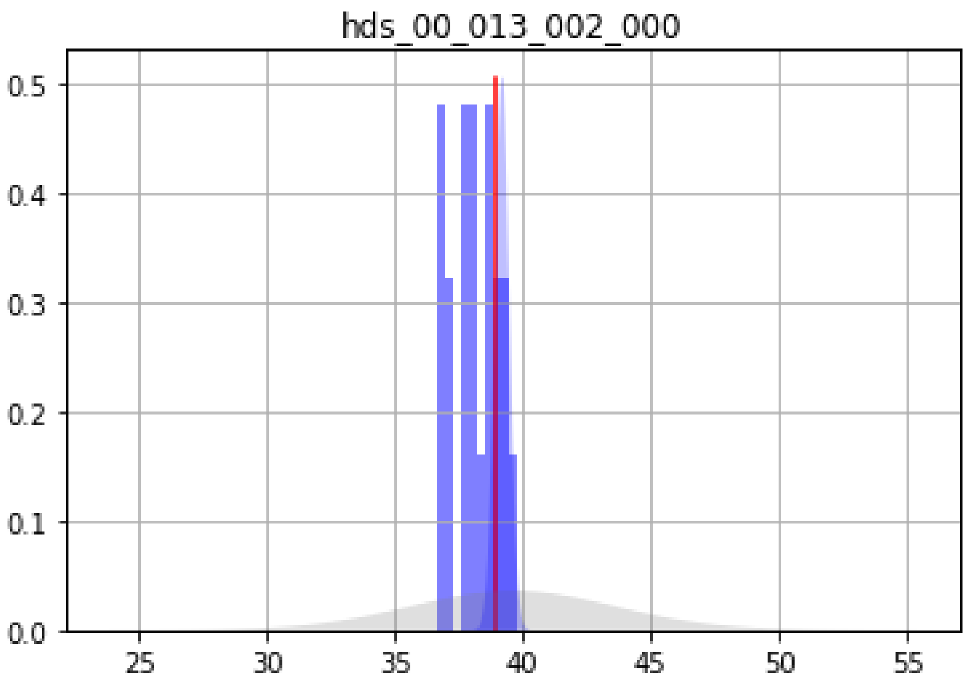
```
In [30]: obs = pst.observation_data
         fnames = pst.pestpp_options["forecasts"].split(",")
         for forecast in fnames:
             ax = plt.subplot(111)
             oe_pt.loc[:,forecast].hist(ax=ax,color="b",alpha=0.5,normed=True)
             ax.plot([obs.loc[forecast,"obsval"],obs.loc[forecast,"obsval"]],ax.get_ylim(),"r"
             axt = plt.twinx()
             x,y = pyemu.plot_utils.gaussian_distribution(f_df.loc[forecast,"prior_mean"],f_df
             axt.fill_between(x,0,y,facecolor="0.5",alpha=0.25)
             x,y = pyemu.plot_utils.gaussian_distribution(f_df.loc[forecast,"post_mean"],f_df.
             axt.fill_between(x,0,y,facecolor="b",alpha=0.25)
             axt.set_ylim(0,axt.get_ylim()[1])
             axt.set_yticks([])
             ax.set_title(forecast)
             plt.show()
```

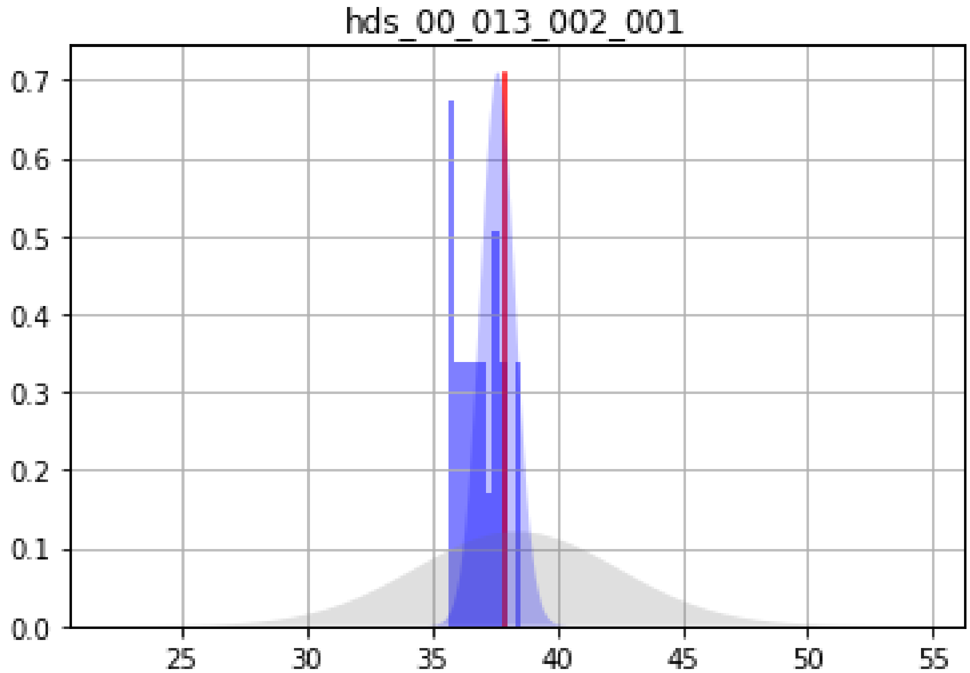



fa_hw_19801229
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
−2000
−1500
−1000
−500
0
500
1000


fa_tw_19791230
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
−2000
−1500
−1000
−500
0
500
1000


fa_tw_19801229
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
−2000
−1500
−1000
−500
0
500
1000
1500
2000


hds_00_013_002_000
0.5
0.4
0.3
0.2
0.1
0.0
25
30
35
40
45
50
55


hds_00_013_002_001
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
25
30
35
40
45
50
55