# pestpp-glm

May 1, 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 | \ |
|---|---|---|---|---|---|
| cn_strt6 | cn_strt6 | log | 1 | 0 | |
| flow | flow | log | 1 | 0 | |
| cn_hk7 | cn_hk7 | log | 1 | 0 | |
| drncond_k00 | drncond_k00 | log | 10 | 0 | |
| gr_vka3 | gr_vka3 | log | 705 | 0 | |
| gr_prsity5 | gr_prsity5 | log | 705 | 0 | |
| cn_rech4 | cn_rech4 | log | 1 | 0 | |
| pp_sy0 | pp_sy0 | log | 32 | 0 | |
| pp_strt1 | pp_strt1 | log | 32 | 0 | |
| gr_sy5 | gr_sy5 | log | 705 | 0 | |
| pp_ss2 | pp_ss2 | log | 32 | 0 | |
| pp_vka0 | pp_vka0 | log | 32 | 0 | |
| cn_prsity7 | cn_prsity7 | log | 1 | 0 | |

| | | | | |
|---|---|---|---|---|
| gr_prsity4 | gr_prsity4 | log | 705 | 0 |
| gr_vka5 | gr_vka5 | log | 705 | 0 |
| pp_hk0 | pp_hk0 | log | 32 | 0 |
| gr_sy4 | gr_sy4 | log | 705 | 0 |
| gr_ss5 | gr_ss5 | log | 705 | 0 |
| pp_hk1 | pp_hk1 | log | 32 | 0 |
| gr_rech2 | gr_rech2 | log | 705 | 0 |
| gr_strt4 | gr_strt4 | log | 705 | 0 |
| pp_prsity0 | pp_prsity0 | log | 32 | 0 |
| strk | strk | log | 40 | 0 |
| pp_ss1 | pp_ss1 | log | 32 | 0 |
| gr_sy3 | gr_sy3 | log | 705 | 0 |
| welflux | welflux | log | 2 | 0 to 0.176091 |
| pp_prsity1 | pp_prsity1 | log | 32 | 0 |
| gr_rech3 | gr_rech3 | log | 705 | 0 |
| cn_ss6 | cn_ss6 | log | 1 | 0 |
| cn_prsity6 | cn_prsity6 | log | 1 | 0 |
| ... | ... | ... | ... | ... |
| pp_rech1 | pp_rech1 | log | 32 | 0 |
| pp_vka2 | pp_vka2 | log | 32 | 0 |
| cn_rech5 | cn_rech5 | log | 1 | -0.39794 |
| pp_rech0 | pp_rech0 | log | 32 | 0 |
| cn_strt7 | cn_strt7 | log | 1 | 0 |
| welflux_k02 | welflux_k02 | log | 6 | 0 |
| gr_strt5 | gr_strt5 | log | 705 | 0 |
| cn_sy6 | cn_sy6 | log | 1 | 0 |
| gr_hk5 | gr_hk5 | log | 705 | 0 |
| pp_vka1 | pp_vka1 | log | 32 | 0 |
| gr_ss4 | gr_ss4 | log | 705 | 0 |
| cn_sy8 | cn_sy8 | log | 1 | 0 |
| pp_sy2 | pp_sy2 | log | 32 | 0 |
| cn_hk8 | cn_hk8 | log | 1 | 0 |
| pp_sy1 | pp_sy1 | log | 32 | 0 |
| cn_prsity8 | cn_prsity8 | log | 1 | 0 |
| cn_vka7 | cn_vka7 | log | 1 | 0 |
| cn_sy7 | cn_sy7 | log | 1 | 0 |
| cn_vka8 | cn_vka8 | log | 1 | 0 |
| cn_vka6 | cn_vka6 | log | 1 | 0 |
| pp_ss0 | pp_ss0 | log | 32 | 0 |
| pp_prsity2 | pp_prsity2 | log | 32 | 0 |
| gr_strt3 | gr_strt3 | log | 705 | 0 |
| pp_strt0 | pp_strt0 | log | 32 | 0 |
| pp_strt2 | pp_strt2 | log | 32 | 0 |
| cn_ss8 | cn_ss8 | log | 1 | 0 |
| pp_hk2 | pp_hk2 | log | 32 | 0 |
| gr_hk4 | gr_hk4 | log | 705 | 0 |
| cn_ss7 | cn_ss7 | log | 1 | 0 |
| gr_hk3 | gr_hk3 | log | 705 | 0 |

| | upper bound | lower bound | standard deviation |
|---|---|---|---|
| cn_strt6 | 0.0211893 | -0.0222764 | 0.0108664 |
| flow | 0.09691 | -0.124939 | 0.0554622 |
| cn_hk7 | 1 | -1 | 0.5 |
| drncond_k00 | 1 | -1 | 0.5 |
| gr_vka3 | 1 | -1 | 0.5 |
| gr_prsity5 | 0 | -1 | 0.25 |
| cn_rech4 | 0.0791812 | -0.09691 | 0.0440228 |
| pp_sy0 | 0.243038 | -0.60206 | 0.211275 |
| pp_strt1 | 0.0211893 | -0.0222764 | 0.0108664 |
| gr_sy5 | 0.243038 | -0.60206 | 0.211275 |
| pp_ss2 | 1 | -1 | 0.5 |
| pp_vka0 | 1 | -1 | 0.5 |
| cn_prsity7 | 0 | -1 | 0.25 |
| gr_prsity4 | 0 | -1 | 0.25 |
| gr_vka5 | 1 | -1 | 0.5 |
| pp_hk0 | 1 | -1 | 0.5 |
| gr_sy4 | 0.243038 | -0.60206 | 0.211275 |
| gr_ss5 | 1 | -1 | 0.5 |
| pp_hk1 | 1 | -1 | 0.5 |
| gr_rech2 | 0.0413927 | -0.0457575 | 0.0217875 |
| gr_strt4 | 0.0211893 | -0.0222764 | 0.0108664 |
| pp_prsity0 | 0 | -1 | 0.25 |
| strk | 2 | -2 | 1 |
| pp_ss1 | 1 | -1 | 0.5 |
| gr_sy3 | 0.243038 | -0.60206 | 0.211275 |
| welflux | 0.176091 to 0.30103 | -0.30103 to 0 | 0.0752575 to 0.11928 |
| pp_prsity1 | 0 | -1 | 0.25 |
| gr_rech3 | 0.0413927 | -0.0457575 | 0.0217875 |
| cn_ss6 | 1 | -1 | 0.5 |
| cn_prsity6 | 0 | -1 | 0.25 |
| ... | ... | ... | ... |
| pp_rech1 | 0.0413927 | -0.0457575 | 0.0217875 |
| pp_vka2 | 1 | -1 | 0.5 |
| cn_rech5 | -0.09691 | -1 | 0.225772 |
| pp_rech0 | 0.0413927 | -0.0457575 | 0.0217875 |
| cn_strt7 | 0.0211893 | -0.0222764 | 0.0108664 |
| welflux_k02 | 1 | -1 | 0.5 |
| gr_strt5 | 0.0211893 | -0.0222764 | 0.0108664 |
| cn_sy6 | 0.243038 | -0.60206 | 0.211275 |
| gr_hk5 | 1 | -1 | 0.5 |
| pp_vka1 | 1 | -1 | 0.5 |
| gr_ss4 | 1 | -1 | 0.5 |
| cn_sy8 | 0.243038 | -0.60206 | 0.211275 |
| pp_sy2 | 0.243038 | -0.60206 | 0.211275 |
| cn_hk8 | 1 | -1 | 0.5 |
| pp_sy1 | 0.243038 | -0.60206 | 0.211275 |

```
cn_prsity8          0          -1        0.25
cn_vka7             1          -1         0.5
cn_sy7       0.243038    -0.60206    0.211275
cn_vka8             1          -1         0.5
cn_vka6             1          -1         0.5
pp_ss0              1          -1         0.5
pp_prsity2          0          -1        0.25
gr_strt3    0.0211893  -0.0222764   0.0108664
pp_strt0    0.0211893  -0.0222764   0.0108664
pp_strt2    0.0211893  -0.0222764   0.0108664
cn_ss8              1          -1         0.5
pp_hk2              1          -1         0.5
gr_hk4              1          -1         0.5
cn_ss7              1          -1         0.5
gr_hk3              1          -1         0.5

[65 rows x 7 columns]
```

### 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]: 719

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
```

Out[6]: 719

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

Out[7]:
```
array(['pp_hk0', 'pp_hk1', 'pp_hk2', 'pp_prsity0', 'pp_prsity1',
       'pp_prsity2', '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]: 719

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

Out[9]:
```
cn_strt6        1
cn_strt8        1
cn_prsity7      1
flow            1
cn_ss8          1
cn_vka6         1
cn_rech4        1
cn_ss7          1
cn_sy7          1
cn_vka8         1
cn_vka7         1
cn_prsity8      1
cn_rech5        1
cn_hk8          1
cn_hk7          1
cn_sy8          1
cn_sy6          1
cn_strt7        1
cn_hk6          1
cn_ss6          1
cn_prsity6      1
welflux         2
welflux_k02     6
drncond_k00    10
pp_hk2         32
pp_rech1       32
pp_sy1         32
pp_vka2        32
pp_rech0       32
pp_hk1         32
pp_prsity2     32
pp_prsity0     32
pp_ss2         32
pp_sy2         32
pp_hk0         32
pp_ss1         32
pp_vka0        32
pp_strt1       32
```

```
pp_vka1        32
pp_ss0         32
pp_strt0       32
pp_strt2       32
pp_sy0         32
pp_prsity1     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]: 527
```

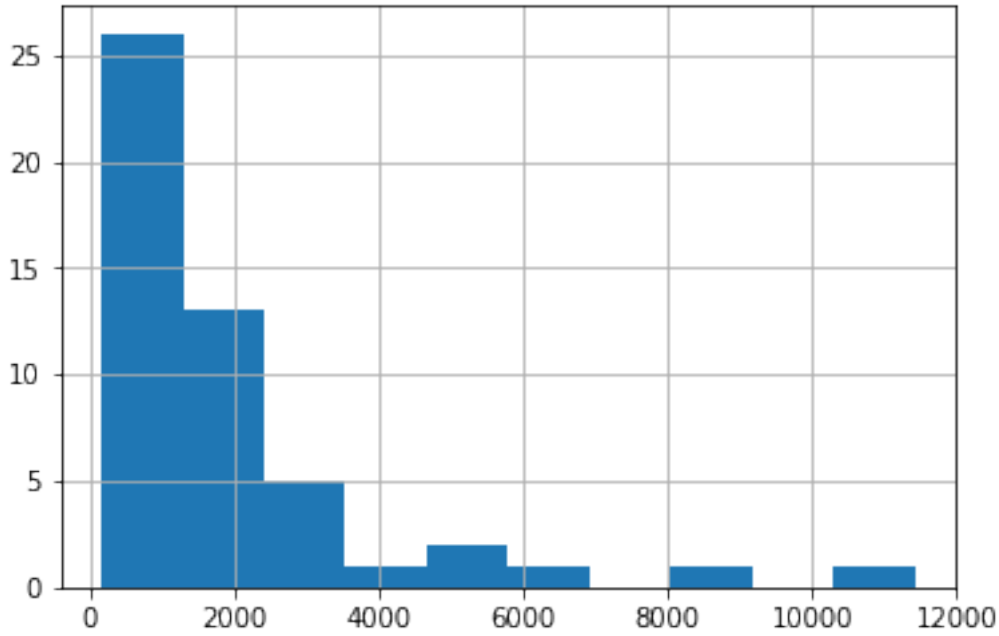
Ok, thats better…so lets run PESTPP-GLM. We will use a single "base parameter" jacobian matrix as the basis for 6 super parameter iterations. Then we will draw 100 realizations from the FOSM posterior parameter covariance matrix and run those 100 realizations to get the psoterior forecast PDFs

```
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_ro
                                     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:

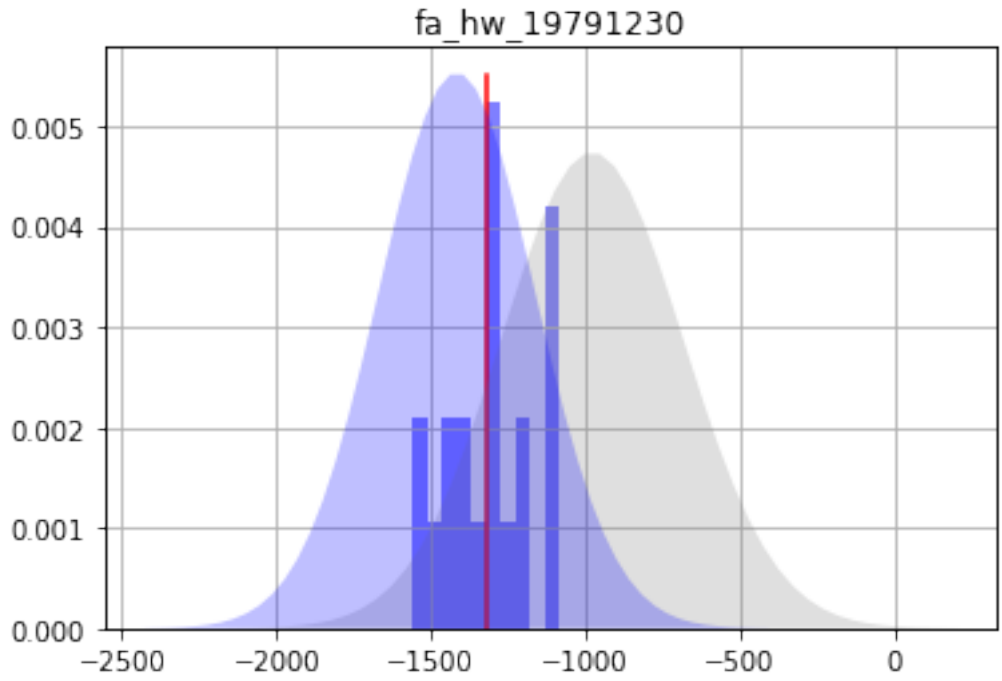
```
In [16]: f_df = pd.read_csv(os.path.join(m_d,"freyberg_pp.pred.usum.csv"),index_col=0)
         f_df.index = f_df.index.map(str.lower)
         f_df
```

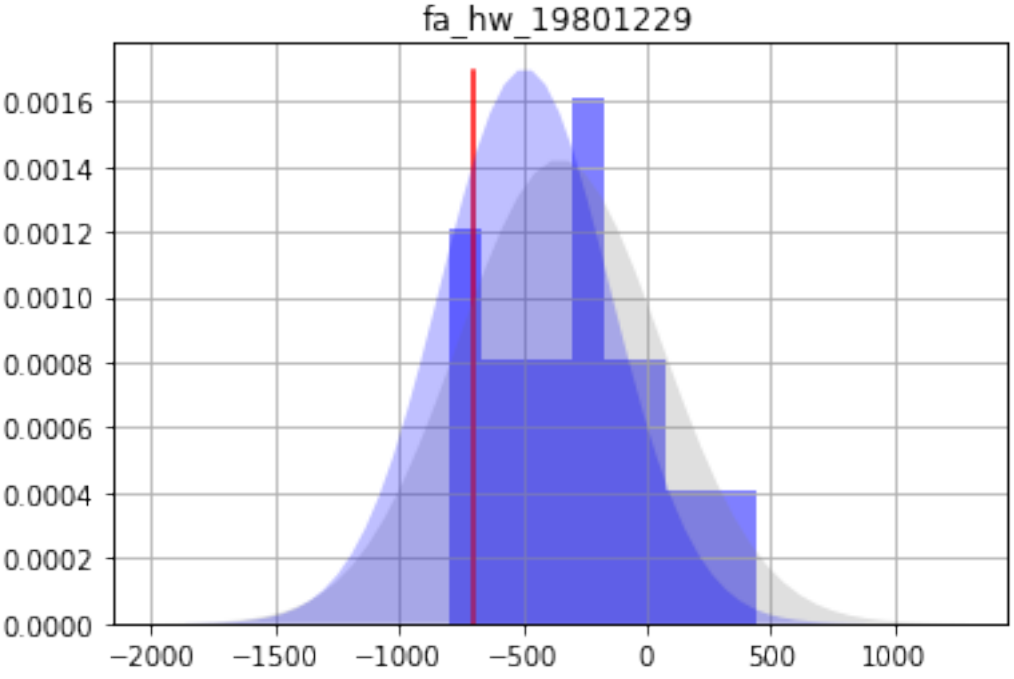
Out[16]:

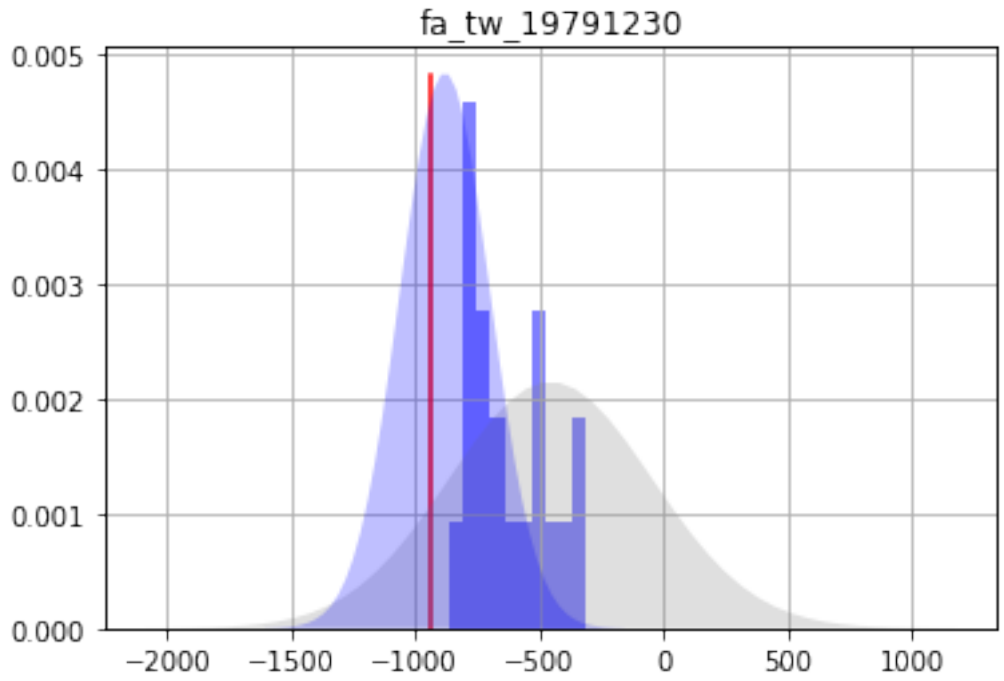
| name | prior_mean | prior_stdev | prior_lower_bound | prior_upper_bound | post_mean | post_stdev |
|---|---|---|---|---|---|---|
| fa_hw_19791230 | -977.2390 | 295.32800 | -1567.8900 | -386.5840 | -1415.6100 | 252.974000 |
| fa_hw_19801229 | -351.2160 | 409.77000 | -1170.7600 | | | |
| fa_tw_19791230 | -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 | | | |
| part_status | 2.0000 | 0.00000 | 2.0000 | | | |
| part_time | 907.7020 | 704.75100 | -501.8010 | | | |

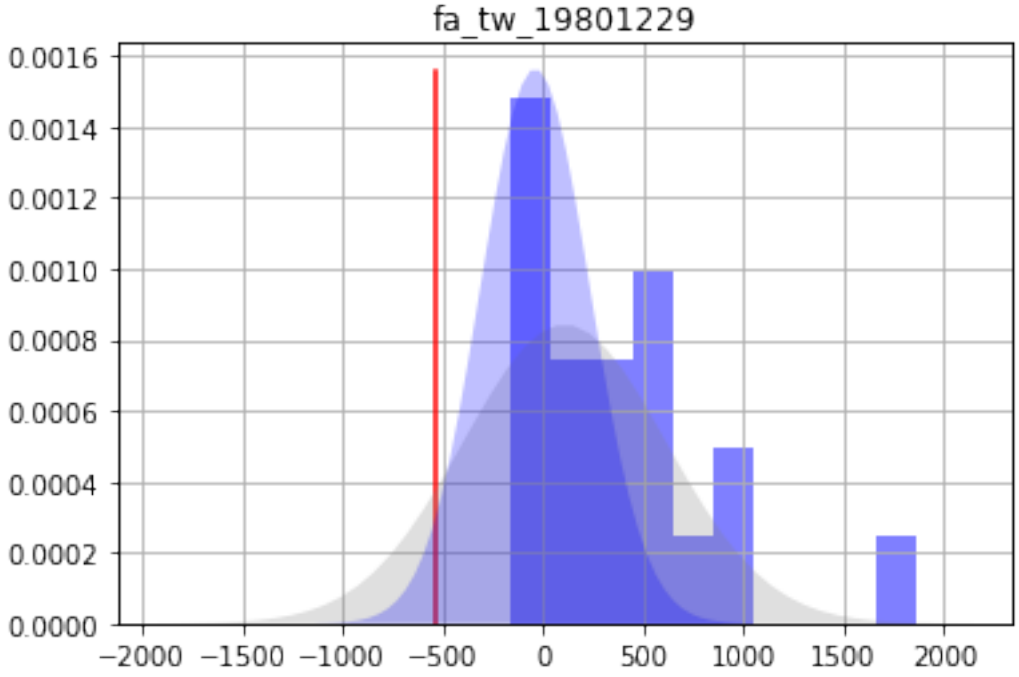
```
fa_hw_19801229              468.3240  -491.6660  342.574000
fa_tw_19791230              365.6690  -878.9190  181.462000
fa_tw_19801229             1122.4200   -38.3736  272.738000
hds_00_013_002_000           47.5365    40.1763    0.501236
hds_00_013_002_001           46.4994    38.5735    0.802533
part_status                   2.0000     2.0000    0.000000
part_time                  2317.2000   916.9650  606.568000

                    post_lower_bound  post_upper_bound
name
fa_hw_19791230           -1921.5600         -909.6600
fa_hw_19801229           -1176.8100          193.4820
fa_tw_19791230           -1241.8400         -515.9940
fa_tw_19801229            -583.8500          507.1030
hds_00_013_002_000          39.1738           41.1787
hds_00_013_002_001          36.9685           40.1786
part_status                  2.0000            2.0000
part_time                 -296.1710         2130.1000
```

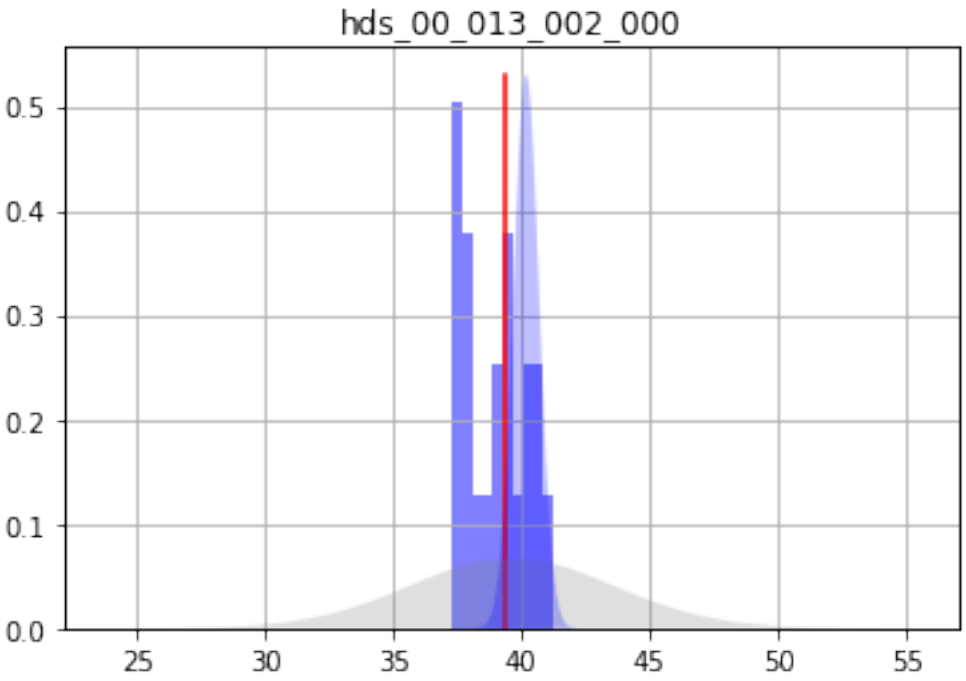
In [17]:
```
obs = pst.observation_data
fnames = pst.pestpp_options["forecasts"].split(",")
for forecast in fnames:
    ax = plt.subplot(111)
    oe_pt.loc[:,forecast].hist(ax=ax,color="b",alpha=0.5,normed=True)
    ax.plot([obs.loc[forecast,"obsval"],obs.loc[forecast,"obsval"]],ax.get_ylim(),"r"
  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

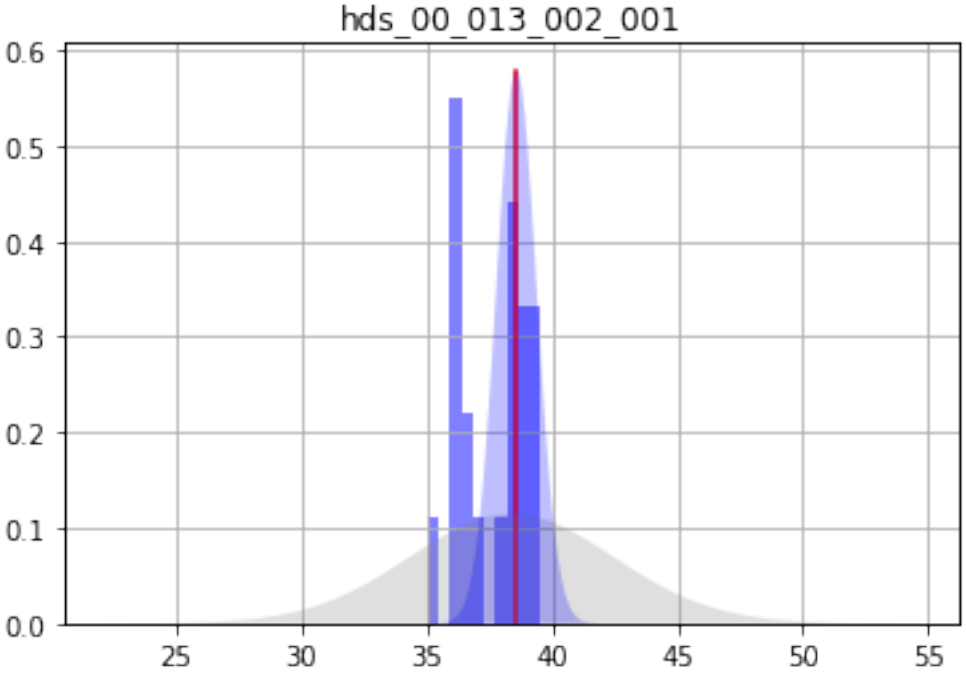

fa_hw_19791230
0.005
0.004
0.003
0.002
0.001
0.000
−2500
−2000
−1500
−1000
−500
0

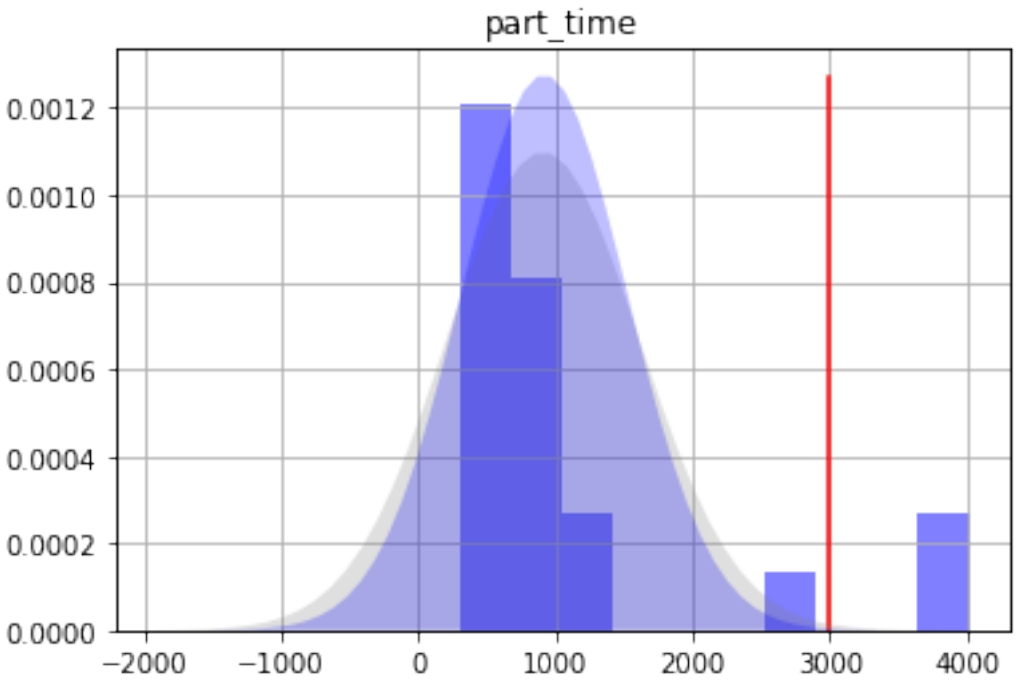

fa_hw_19801229
0.0016
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
−2000
−1500
−1000
−500
0
500
1000

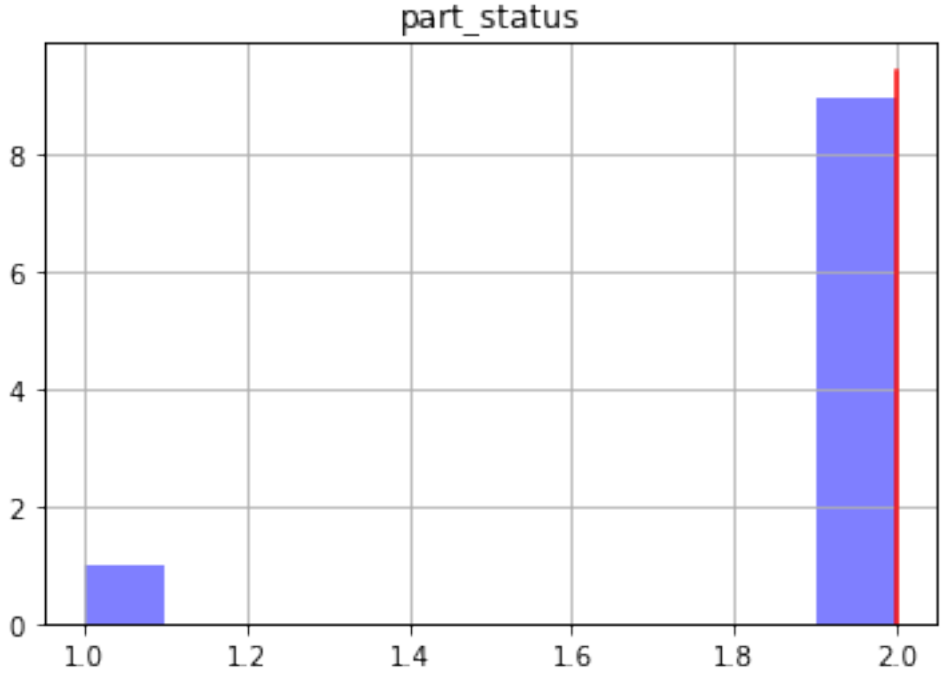

fa_tw_19791230
0.005
0.004
0.003
0.002
0.001
0.000
−2000
−1500
−1000
−500
0
500
1000


fa_tw_19801229
0.0016
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
0.6
0.5
0.4
0.3
0.2
0.1
0.0
25
30
35
40
45
50
55


part_time
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
−2000
−1000
0
1000
2000
3000
4000


part_status
8
6
4
2
0
1.0
1.2
1.4
1.6
1.8
2.0

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

new binary format detected...
```

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

Out[19]: {'ss3021007',
          'sy4017000',
          'ss5008013',
          'sy5028013',
          'rech3012017',
          'prsity5026003',
          'vka4035017',
          'hk4029016',
          'strt4007006',
          'ss4007003',
          'vka3039013',
          'vka4009018',
          'vka3030005',
          'sy4004011',
          'sy4014014',
          'strt5018010',
          'rech2010011',
          'vka229',
          'vka5026016',
          'ss5004013',
          'sy4011018',
          'hk5014017',
          'hk4009013',
          'strt4011008',
          'prsity3009015',
          'vka3002010',
          'vka3021017',
          'prsity5019018',
          'ss5007008',
          'ss3026012',
          'strt4028003',
          'vka3035006',
          'vka5002001',
          'ss5030012',
          'ss5006008',
```

```
'hk3014015',
'vka4035007',
'prsity3015019',
'hk4018008',
'rech3014002',
'sy3030008',
'hk3026004',
'prsity4038010',
'vka3030013',
'strt5003011',
'hk4000005',
'strt3038011',
'prsity4009016',
'prsity3014015',
'ss4011015',
'hk3021001',
'prsity4039008',
'strt4020014',
'rech2010008',
'rech2032006',
'hk4035014',
'strt5031011',
'prsity5011019',
'vka3035005',
'hk3011008',
'prsity4025015',
'sy3018015',
'prsity4035005',
'sy5016009',
'sy3024015',
'prsity3020012',
'hk4002010',
'rech2000013',
'hk4027010',
'ss5003019',
'strt5036012',
'hk4004018',
'ss4003010',
'ss4010011',
'prsity3035016',
'rech2021018',
'prsity5036005',
'ss5026009',
'hk4031004',
'rech3000001',
'ss4005008',
'rech3017008',
'ss4033011',
```

```
'strt4016018',
'strt5030006',
'sy5015008',
'strt4007005',
'prsity4028017',
'sy4020001',
'hk3014009',
'strt3027002',
'hk5030007',
'prsity3029016',
'sy4029017',
'vka5028016',
'strt3014001',
'strt3037013',
'sy4019011',
'sy4024002',
'vka4011018',
'vka228',
'strt5036006',
'prsity3002000',
'sy5033009',
'vka5037008',
'hk3031004',
'strt3036017',
'prsity3039008',
'sy3000004',
'prsity4025009',
'hk4005013',
'strt4002007',
'rech2037013',
'sy5021017',
'rech3013018',
'sy4002004',
'hk3037014',
'prsity4037005',
'ss3016017',
'rech3028010',
'vka3003019',
'prsity3024014',
'strt3032009',
'hk3008018',
'sy3028011',
'rech2014001',
'rech2022007',
'sy4009013',
'vka5034011',
'hk5001014',
'vka5007004',
```

```
'ss4019012',
'strt4034006',
'rech3005005',
'sy3012014',
'ss3003012',
'sy3022013',
'strt4039014',
'vka5005000',
'vka3025011',
'rech3024016',
'ss3035015',
'prsity4036012',
'ss3033009',
'hk4037014',
'hk5034010',
'strt4038012',
'ss3022008',
'sy3009001',
'ss3031019',
'vka4026016',
'vka4004004',
'hk3000011',
'sy5003011',
'ss5031017',
'vka3003006',
'rech2017000',
'rech3035013',
'sy5021001',
'rech3016017',
'strt5000013',
'prsity4039009',
'rech2028007',
'hk4002013',
'rech3014018',
'vka5037006',
'vka4017019',
'sy3025002',
'sy3013010',
'strt3002012',
'hk5026011',
'ss4025010',
'prsity4015015',
'strt4030009',
'strt3024013',
'hk5023016',
'strt225',
'prsity5006005',
'prsity4003012',
```

```
'hk4017010',
'hk4025001',
'hk3038008',
'prsity3035013',
'ss5031001',
'strt4000002',
'sy3001014',
'prsity3010010',
'sy3028002',
'sy5000018',
'rech3033007',
'prsity3019002',
'vka3000015',
'strt5025006',
'rech2025013',
'ss5002016',
'sy5008002',
'vka5010014',
'strt5033007',
'ss4032004',
'hk5004004',
'hk4017008',
'prsity3002013',
'rech3026017',
'hk4016013',
'hk4007017',
'ss5017014',
'ss3024000',
'hk5004002',
'strt5025008',
'vka4004010',
'prsity4001005',
'hk3001010',
'vka3011018',
'hk5027011',
'prsity4007011',
'ss3028002',
'strt3022009',
'prsity3021007',
'prsity5033013',
'sy3024008',
'rech2037016',
'hk3010003',
'rech3018015',
'rech3011011',
'strt5012010',
'sy5036009',
'rech3025001',
```

```
'vka3011016',
'strt5017013',
'prsity5026018',
'vka4029001',
'hk4004001',
'vka5009000',
'prsity4009017',
'strt4028010',
'prsity5020011',
'hk5021015',
'strt4013014',
'ss4015015',
'sy4033014',
'prsity4030008',
'ss5001004',
'prsity3033016',
'rech3012014',
'sy5013016',
'rech2030005',
'sy3012010',
'strt131',
'prsity3029012',
'rech3004015',
'sy3025016',
'rech3022006',
'sy5032017',
'vka4014003',
'hk3007012',
'rech3023019',
'vka3005008',
'hk5035017',
'hk3022018',
'strt3039006',
'vka4000016',
'strt3014019',
'strt4013000',
'ss5035005',
'hk4026004',
'ss4012009',
'rech2021000',
'sy4029004',
'prsity5033009',
'sy5033008',
'strt3009014',
'rech2030003',
'sy5008019',
'strt4031015',
'ss4030016',
```

```
'vka4022001',
'prsity4027015',
'vka5037012',
'vka5030014',
'rech2024000',
'rech2016016',
'ss4024008',
'vka5025007',
'prsity5024000',
'prsity4014008',
'hk5024012',
'rech2036017',
'hk4018018',
'hk4023012',
'prsity4025000',
'rech2001015',
'hk3021013',
'prsity3017002',
'vka4022009',
'rech3002016',
'sy4022012',
'vka3014002',
'hk3028005',
'vka5033005',
'strt5019016',
'sy3003006',
'strt5004011',
'sy5032014',
'ss4027016',
'prsity4035018',
'prsity3034011',
'rech3038006',
'vka3007000',
'vka4020002',
'ss3033014',
'strt4017014',
'vka3036004',
'strt3025005',
'prsity3002016',
'vka4005001',
'ss5025018',
'sy4007008',
'sy4019001',
'strt5029001',
'strt5017014',
'prsity3031003',
'sy5000012',
'hk5024008',
```

```
'prsity5036007',
'sy5010019',
'prsity5013015',
'ss4011011',
'vka4005014',
'hk4025000',
'strt4017003',
'vka4009016',
'vka3012016',
'vka4012012',
'sy5033016',
'strt3000016',
'strt5012001',
'strt4013008',
'hk3010010',
'strt012',
'strt5030018',
'prsity4010008',
'sy3008012',
'rech2009002',
'ss3014003',
'sy5012012',
'ss4007017',
'prsity5025014',
'sy5002003',
'hk3003014',
'strt4003012',
'ss4015012',
'prsity4020001',
'vka3024005',
'ss4021003',
'prsity5033002',
'vka4021012',
'hk3013008',
'prsity4025016',
'rech2031006',
'strt5018001',
'sy5035005',
'vka3021010',
'hk5025002',
'sy4025004',
'vka4026003',
'sy3011010',
'vka5032003',
'prsity5032010',
'rech2028011',
'strt5025010',
'strt5016014',
```

```
'vka5009002',
'vka5028009',
'prsity3020019',
'vka5028018',
'vka5039008',
'ss5039008',
'ss4017003',
'strt3039014',
'strt4010011',
'strt5004002',
'sy4008017',
'vka5028007',
'vka3029004',
'hk5008002',
'ss4005000',
'strt5028005',
'strt5009016',
'rech2027010',
'ss5001006',
'vka5038015',
'ss4014011',
'vka4025011',
'prsity3029001',
'hk4026013',
'strt4006014',
'ss4012019',
'ss4039011',
'rech3015015',
'ss4033003',
'prsity3007000',
'vka004',
'sy5007017',
'vka3019016',
'sy3016000',
'vka4026015',
'strt3030009',
'strt3033008',
'sy3036013',
'rech2016010',
'ss5024014',
'hk4033006',
'hk3007019',
'sy4007004',
'sy3002013',
'hk4029009',
'rech3016016',
'ss3004007',
'ss4012013',
```

```
'vka3019018',
'vka5034008',
'rech3009009',
'strt3003005',
'strt5033011',
'vka4004012',
'ss4039007',
'strt5034010',
'hk4029000',
'ss5021014',
'strt3016001',
'rech3031019',
'strt5034019',
'sy5036006',
'vka3011008',
'sy3011014',
'rech3007006',
'vka4007013',
'prsity3006001',
'rech3022003',
'strt4016009',
'strt5027009',
'prsity3019001',
'sy5025008',
'prsity3027013',
'strt5032012',
'sy4020003',
'prsity3018012',
'sy5002002',
'hk3005018',
'ss3031016',
'vka4033002',
'rech3001008',
'rech3015000',
'prsity3000011',
'rech108',
'strt4022016',
'vka5035010',
'ss4034013',
'hk5017002',
'rech2035018',
'ss3013003',
'ss3000006',
'sy4009002',
'sy5024014',
'ss5001009',
'sy3005000',
'prsity4023017',
```

```
'ss4028001',
'strt4000015',
'strt4031012',
'ss3024005',
'strt3021014',
'sy5037017',
'rech2037017',
'ss3010003',
'ss5038014',
'vka021',
'rech2033013',
'rech3030006',
'sy3034010',
'ss5007012',
'vka4035004',
'hk4016014',
'strt3029008',
'hk5024016',
'hk4027009',
'rech3022012',
'prsity3018008',
'ss4032011',
'vka5013009',
'ss3021013',
'sy3032018',
'ss5007019',
'ss3026006',
'vka3022007',
'vka5032015',
'prsity4034006',
'strt4020013',
'rech3012015',
'sy4022018',
'strt3029014',
'hk4005012',
'ss5005017',
'sy5027006',
'ss5039009',
'vka4006007',
'rech2005006',
'vka5008015',
'ss4011001',
'strt3036016',
'vka4005004',
'prsity4006004',
'hk5002009',
'ss3008012',
'vka5034009',
```

```
'strt3007017',
'rech3036015',
'prsity5023012',
'prsity4011018',
'sy5033019',
'vka4039010',
'ss3002009',
'ss3025003',
'vka3000007',
'prsity3001014',
'vka5007016',
'strt4026019',
'strt5019012',
'sy4018002',
'sy5032005',
'vka3034010',
'rech3024012',
'ss3030005',
'sy4035007',
'ss4032013',
'rech2008014',
'prsity3011017',
'rech3019002',
'sy5014014',
'hk4015012',
'vka4030005',
'strt3025003',
'ss4009012',
'rech2019003',
'vka5003017',
'vka5027014',
'vka215',
'hk3029009',
'prsity4031017',
'rech2015001',
'vka3023008',
'strt3012003',
'vka5006005',
'strt5000007',
'rech3002017',
'hk4001008',
'prsity3022002',
'prsity3030000',
'rech2006001',
'vka4026011',
'strt4024011',
'hk3007017',
'hk4037012',
```

```
'prsity3001013',
'prsity5008016',
'prsity5032007',
'rech3014017',
'ss3002000',
'prsity3003019',
'sy4004018',
'strt4024012',
'prsity5031011',
'ss4012014',
'strt5025019',
'ss5015015',
'vka3029017',
'ss5028008',
'vka4016017',
'rech3000009',
'ss3029008',
'ss4021014',
'strt4015017',
'sy3006015',
'sy3029012',
'ss3020008',
'ss3027000',
'prsity3015010',
'ss5007006',
'prsity5037008',
'vka5037007',
'ss3003005',
'hk3024012',
'hk3020013',
'strt222',
'sy4019014',
'vka4000008',
'strt4033009',
'vka4027008',
'prsity5007004',
'prsity5030004',
'prsity3024001',
'strt3012000',
'prsity4000016',
'vka5024013',
'ss3020009',
'strt4002018',
'hk3004018',
'hk5026008',
'rech3018016',
'strt4001006',
'strt4015011',
```

```
'strt5001012',
'strt5019002',
'rech2036012',
'prsity5005002',
'sy5010001',
'vka5030018',
'vka3028018',
'vka5010015',
'rech2033015',
'rech3030013',
'vka5010002',
'ss3032004',
'strt3026018',
'prsity3004017',
'rech3008000',
'prsity3021014',
'strt3021015',
'rech131',
'hk4017011',
'strt4023006',
'sy3018001',
'sy3021017',
'sy3035007',
'vka3023010',
'vka4034017',
'ss5024007',
'sy3015014',
'ss5009019',
'prsity3026015',
'ss4020015',
'vka4011015',
'hk3020001',
'hk4031008',
'strt4016010',
'vka4025016',
'strt4039006',
'ss4010012',
'sy4032014',
'vka3034006',
'hk4028017',
'hk5012010',
'ss5002013',
'rech3026014',
'strt3017012',
'strt3011011',
'hk4014003',
'vka4023012',
'vka5023008',
```

```
'sy4021016',
'sy3018003',
'rech3002000',
'prsity4026019',
'prsity4011001',
'sy4028013',
'rech3027007',
'ss5011015',
'prsity3005000',
'ss3021000',
'strt3018001',
'hk5033002',
'strt3009008',
'sy5033004',
'ss5031008',
'prsity5012013',
'ss4005003',
'strt3013003',
'ss4028018',
'rech3012009',
'vka4035009',
'hk4031000',
'sy3029017',
'prsity4018018',
'rech2017015',
'strt3003007',
'hk5017017',
'strt4022018',
'strt3027008',
'vka3013008',
'strt5004012',
'prsity3000010',
'prsity3037012',
'vka3028016',
'prsity4000003',
'ss3018001',
'rech2027007',
'ss4037006',
'ss5024000',
'hk5022003',
'ss3013001',
'strt4028017',
'prsity5001000',
'vka4012018',
'ss3025018',
'ss5006002',
'ss4018008',
'strt3022011',
```

```
'hk4020009',
'vka4000013',
'vka4009019',
'sy4011012',
'rech2006013',
'hk5007010',
'prsity3009016',
'sy4034008',
'strt4032003',
'ss5030009',
'hk5009012',
'sy3007018',
'strt4019013',
'ss3014014',
'prsity5002010',
'ss4008007',
'prsity4031019',
'hk3014002',
'sy4027009',
'vka4036011',
'sy4018000',
'vka4019010',
'hk5030002',
'sy5025004',
'hk3018010',
'ss5025007',
'sy5002004',
'vka5005002',
'vka5032005',
'prsity3029017',
'strt3032008',
'strt4014013',
'vka3017016',
'hk5018013',
'strt4037015',
'prsity4033007',
'sy5027003',
'hk5018011',
'hk5029016',
'hk5035010',
'strt5027015',
'ss3039013',
'sy5022006',
'hk5005007',
'vka3033007',
'vka4008007',
'rech3005011',
'ss3037008',
```

```
'rech2014002',
'ss4013012',
'hk3001015',
'sy5008001',
'ss4039012',
'strt3003006',
'rech3016000',
'rech3028001',
'sy4000014',
'strt5036017',
'strt3021002',
'vka4003015',
'vka4031000',
'hk5015008',
'hk3024015',
'prsity4020010',
'sy4008011',
'sy4022008',
'sy4023011',
'strt4022011',
'prsity5017009',
'sy4008000',
'vka4023000',
'vka5022014',
'rech3001017',
'ss4017014',
'rech3029019',
'ss5031000',
'prsity3023019',
'vka4026008',
'prsity4002001',
'rech2012009',
'rech2020012',
'hk5018002',
'strt5022019',
'prsity4034014',
'strt4005003',
'prsity3002005',
'prsity4020019',
'prsity5026005',
'vka5023013',
'strt5029003',
'prsity5014009',
'sy4016015',
'vka3024001',
'ss4009016',
'vka4009008',
'rech2026001',
```

```
'vka5002019',
'prsity4037016',
'strt3025019',
'strt3019008',
'vka3005011',
'prsity4005015',
'strt3007011',
'ss3012018',
'vka5032017',
'vka3017017',
'hk5000016',
'strt4019000',
'vka4021006',
'ss4014012',
'vka4034006',
'hk4034013',
'strt4006001',
'ss4032001',
'hk3023007',
'hk5026018',
'prsity3000008',
'hk4034018',
'vka4037005',
'sy3021014',
'rech2030014',
'rech3032010',
'prsity4036015',
'vka5017012',
'prsity4018000',
'rech2026011',
'sy3031006',
'ss5008014',
'strt5038007',
'vka4006010',
'hk4004011',
'sy3006009',
'hk4019002',
'vka4001014',
'vka4007017',
'prsity3022015',
'prsity3018002',
'prsity4026018',
'sy4016000',
'prsity5008019',
'strt4023015',
'sy4002018',
'sy5015013',
'strt5025004',
```

```
'sy3030017',
'sy4033017',
'vka3033019',
'hk3007007',
'hk3022017',
'rech2006010',
'strt5000014',
'vka3001016',
'prsity3009009',
'vka5031006',
'sy4022016',
'hk4029017',
'hk5029006',
'hk5025012',
'prsity5034009',
'rech3035014',
'strt4033010',
'hk4009012',
'ss4033012',
'strt5037010',
'strt3003016',
'strt5036004',
'sy4026010',
'prsity3003018',
'hk3025009',
'prsity5015012',
'sy4026014',
'vka4004002',
'prsity5020000',
'ss3024019',
'hk3023015',
'prsity4028001',
'rech3019001',
'rech2002002',
'vka5010018',
'vka5033015',
'sy5030002',
'hk4008014',
'prsity4028011',
'vka5006003',
'prsity4012003',
'sy3033008',
'rech3007011',
'rech2010014',
'vka5015016',
'strt3003010',
'ss3029017',
'sy4007018',
```

```
'strt5007007',
'strt5000017',
'strt3012017',
'rech2016003',
'hk4026008',
'rech2000019',
'ss4016001',
'sy3021015',
'ss4016014',
'vka5029006',
'ss5035017',
'ss5021018',
'sy4006018',
'strt4013013',
'hk3006012',
'ss5007010',
'hk3008013',
'hk5028019',
'hk4030006',
'ss4001017',
'hk5030004',
'prsity3026011',
'ss4020002',
'strt213',
'strt3035015',
'sy5028009',
'ss5013003',
'vka4006002',
'sy5012014',
'ss5020009',
'ss5028005',
'vka4017002',
'hk5005003',
'rech2006003',
'prsity5023007',
'rech2021014',
'vka5001009',
'rech3023017',
'hk3027016',
'sy4001006',
'ss5013001',
'hk4035008',
'ss4035005',
'ss4001014',
'ss4011002',
'sy5003009',
'ss5030005',
'hk4023018',
```

```
'ss5000018',
'prsity3014016',
'hk4004000',
'prsity4004005',
'ss4026003',
'sy5024010',
'strt4006004',
'ss4027004',
'sy3033014',
'rech113',
'ss4026018',
'hk5028004',
'ss5038013',
'strt4029001',
'rech3005000',
'strt3033002',
'ss3006016',
'hk4014001',
'prsity3032012',
'strt3026006',
'sy4010003',
'strt4021003',
'strt5017001',
'rech2009000',
'hk3025001',
'vka3026007',
'sy5022010',
'rech2021012',
'hk4014016',
'strt5032006',
'rech2019009',
'sy4003015',
'vka4004018',
'sy5025005',
'strt4010009',
'rech3011015',
'vka5019012',
'sy5035008',
'sy3010002',
'rech2024018',
'strt3012018',
'vka5019016',
'prsity4015013',
'ss3028018',
'vka5008017',
'strt4009002',
'ss3012013',
'prsity4033018',
```

```
 'sy3020011',
 'prsity5006006',
 'hk3005014',
 'prsity3036011',
 'rech3002001',
 ...}
```

```
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"))
```

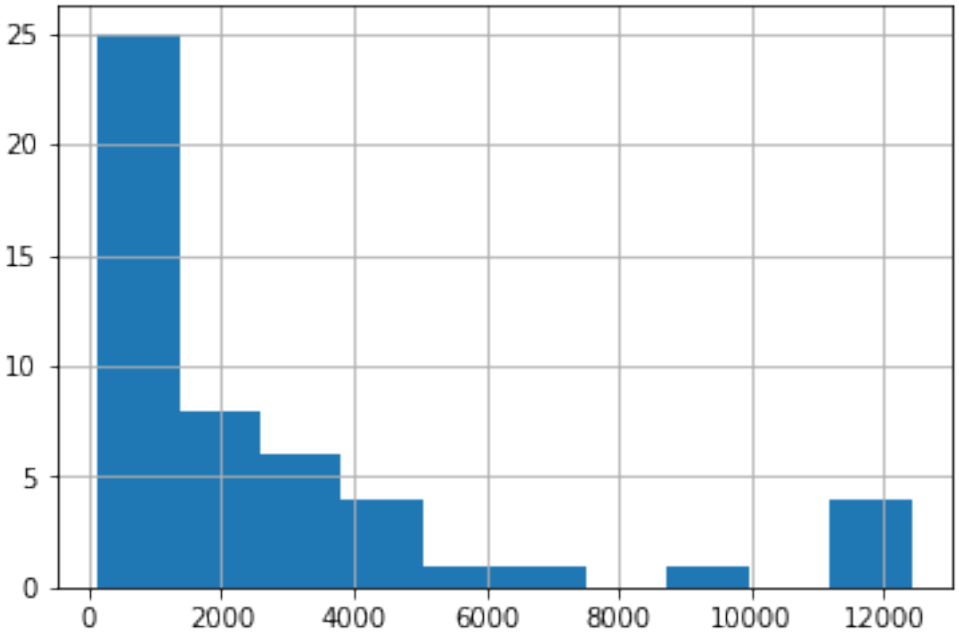
```
Out[22]: 'template/restart_pp.jcb'
```

```
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_ro
                                    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))
```




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

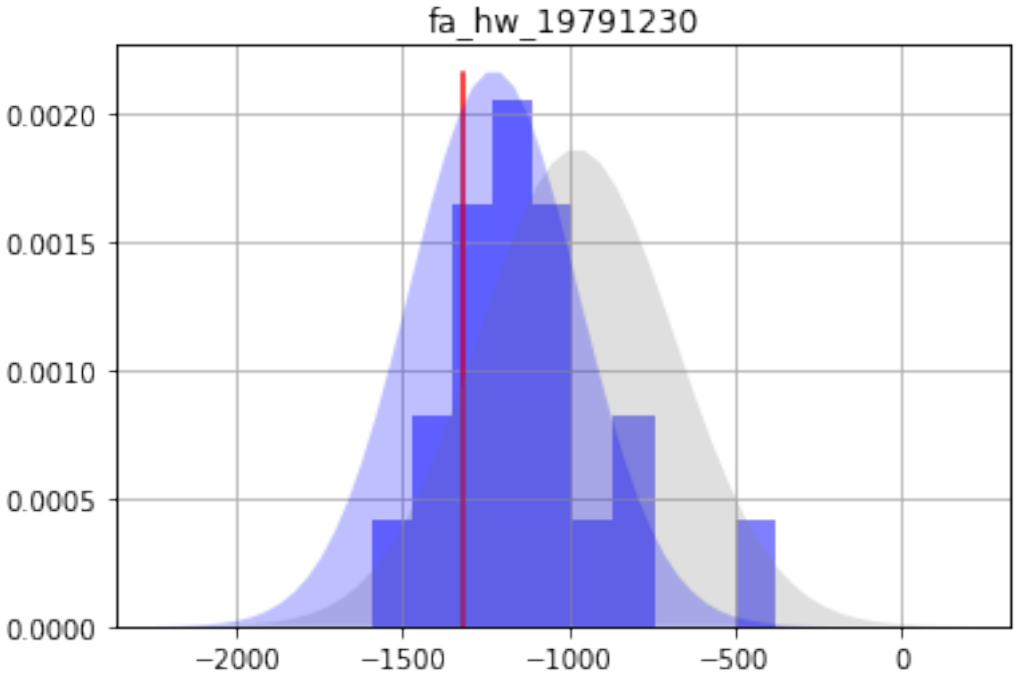
```
In [28]: f_df = pd.read_csv(os.path.join(m_d,"freyberg_pp.pred.usum.csv"),index_col=0)
         f_df.index = f_df.index.map(str.lower)
         f_df
```

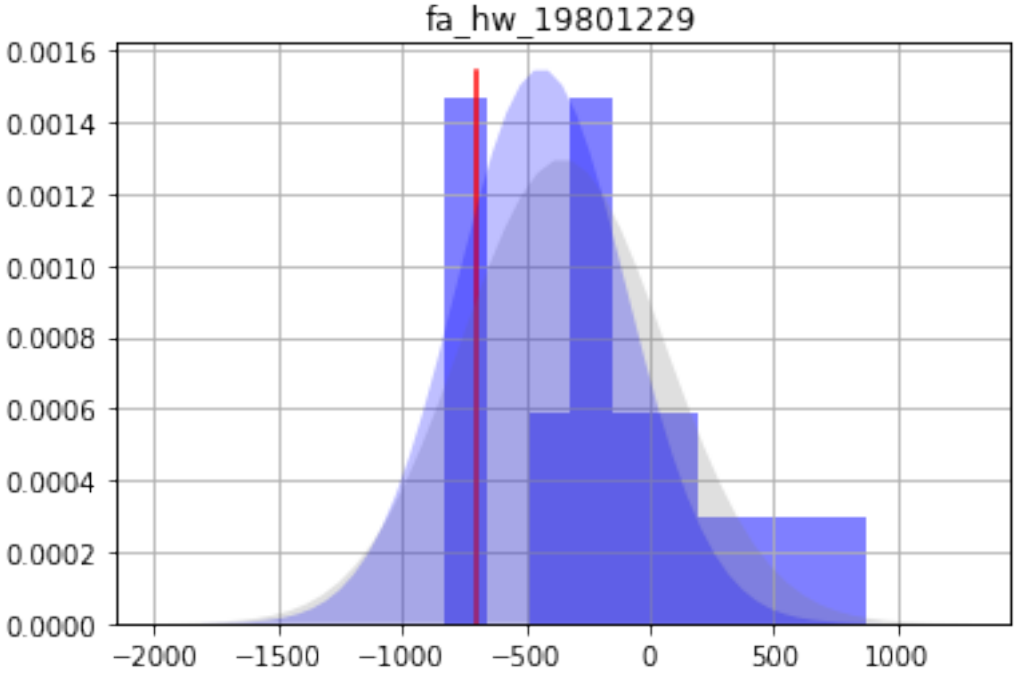
Out[28]:

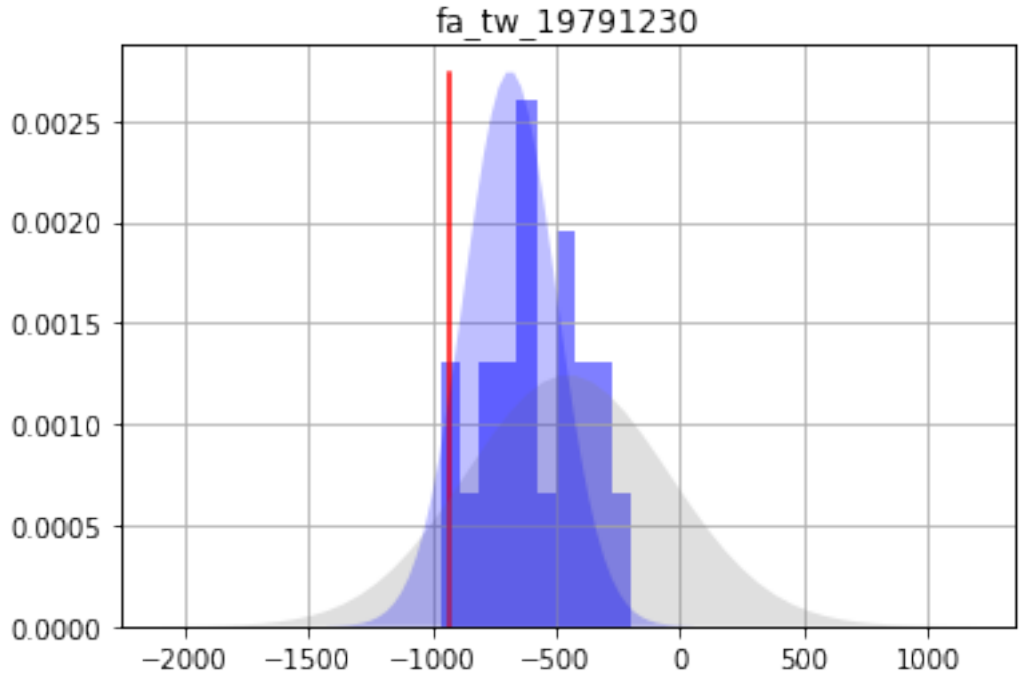
| | prior_mean | prior_stdev | prior_lower_bound | prior_upper_bound | post_mean | post_stdev |
|---|---|---|---|---|---|---|
| name | | | | | | |
| fa_hw_19791230 | -977.2390 | 295.32800 | -1567.8900 | -386.5840 | -1226.1600 | 253.662000 |
| fa_hw_19801229 | -351.2160 | 409.77000 | -1170.7600 | 468.3240 | -438.0650 | 342.835000 |
| fa_tw_19791230 | -453.0330 | 409.35100 | -1271.7400 | 365.6690 | -685.1120 | 185.061000 |
| fa_tw_19801229 | 108.9600 | 506.73200 | -904.5040 | 1122.4200 | 25.8356 | 275.087000 |
| hds_00_013_002_000 | 39.6102 | 3.96314 | 31.6840 | 47.5365 | 40.1357 | 0.528593 |
| hds_00_013_002_001 | 38.3838 | 4.05782 | 30.2681 | | | |
| part_status | 2.0000 | 0.00000 | 2.0000 | | | |
| part_time | 907.7020 | 704.75100 | -501.8010 | | | |

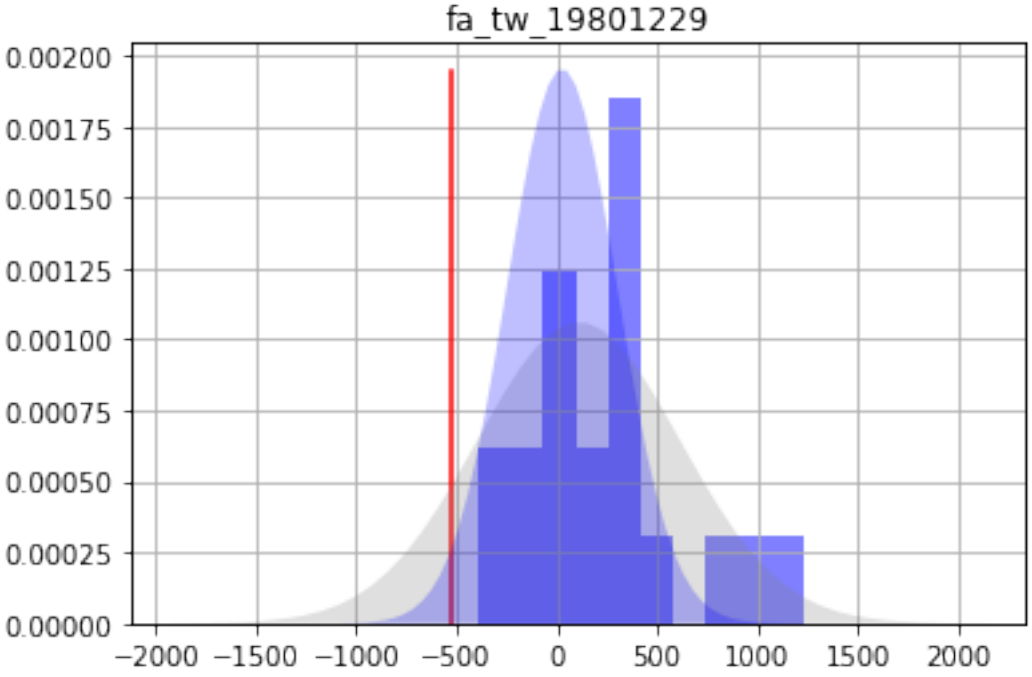
```
hds_00_013_002_001            46.4994    38.6689    0.821579
part_status                    2.0000     2.0000    0.000000
part_time                   2317.2000   864.5700  609.653000

                    post_lower_bound  post_upper_bound
name
fa_hw_19791230           -1733.4900         -718.8380
fa_hw_19801229           -1123.7400          247.6060
fa_tw_19791230           -1055.2300         -314.9900
fa_tw_19801229            -524.3380          576.0100
hds_00_013_002_000          39.0785           41.1929
hds_00_013_002_001          37.0257           40.3120
part_status                  2.0000            2.0000
part_time                 -354.7370         2083.8800
```

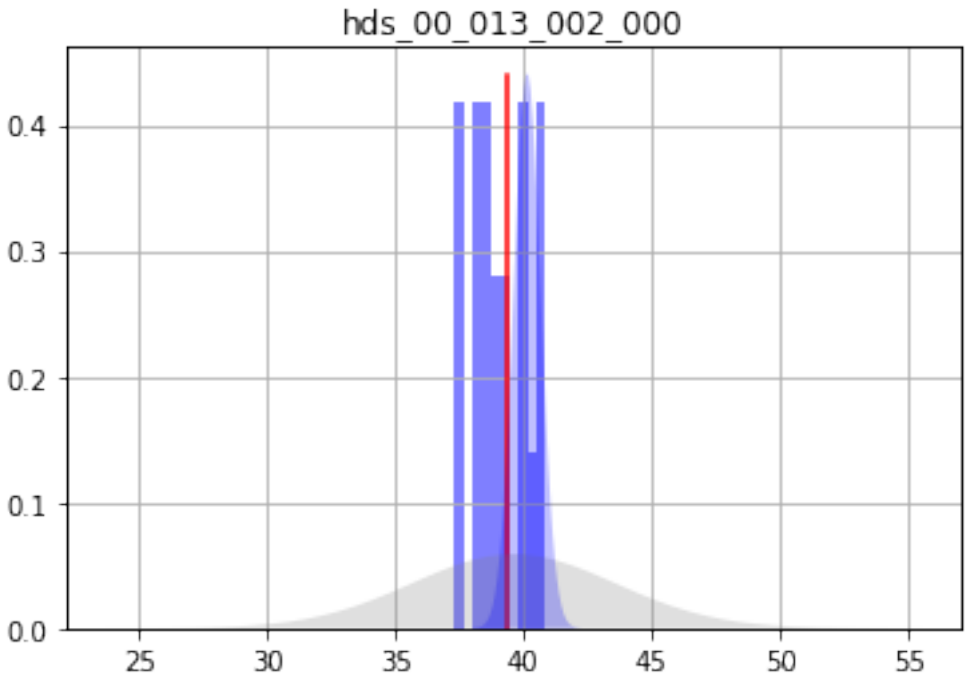
In [29]:
```
obs = pst.observation_data
fnames = pst.pestpp_options["forecasts"].split(",")
for forecast in fnames:
    ax = plt.subplot(111)
    oe_pt.loc[:,forecast].hist(ax=ax,color="b",alpha=0.5,normed=True)
    ax.plot([obs.loc[forecast,"obsval"],obs.loc[forecast,"obsval"]],ax.get_ylim(),"r"
    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()
```

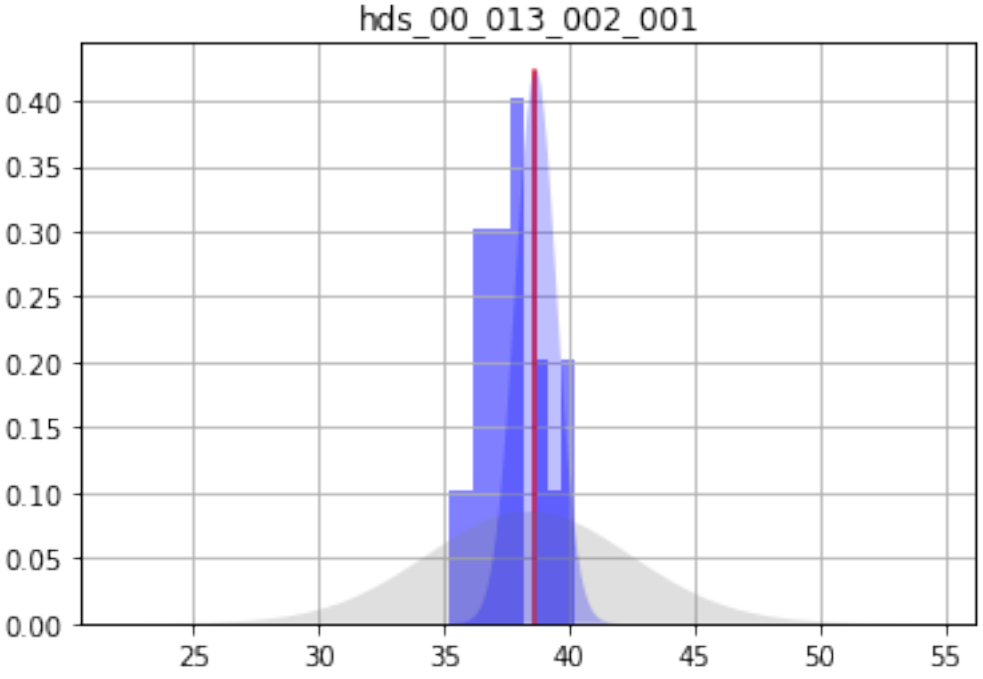

fa_hw_19791230
0.0020
0.0015
0.0010
0.0005
0.0000
−2000
−1500
−1000
−500
0

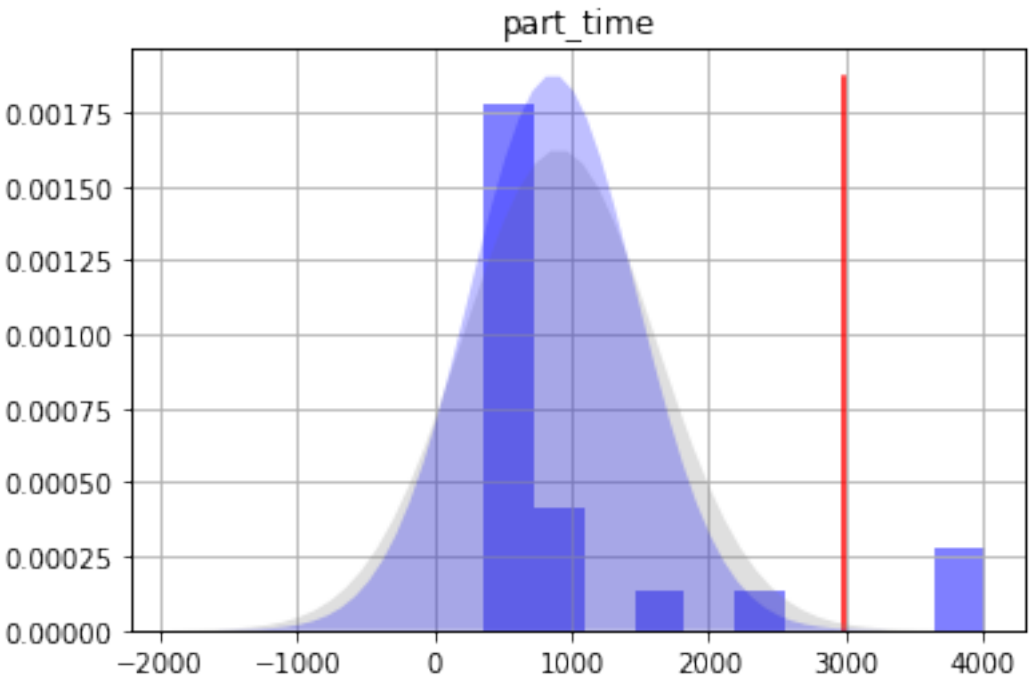

fa_hw_19801229
0.0016
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
−2000
−1500
−1000
−500
0
500
1000

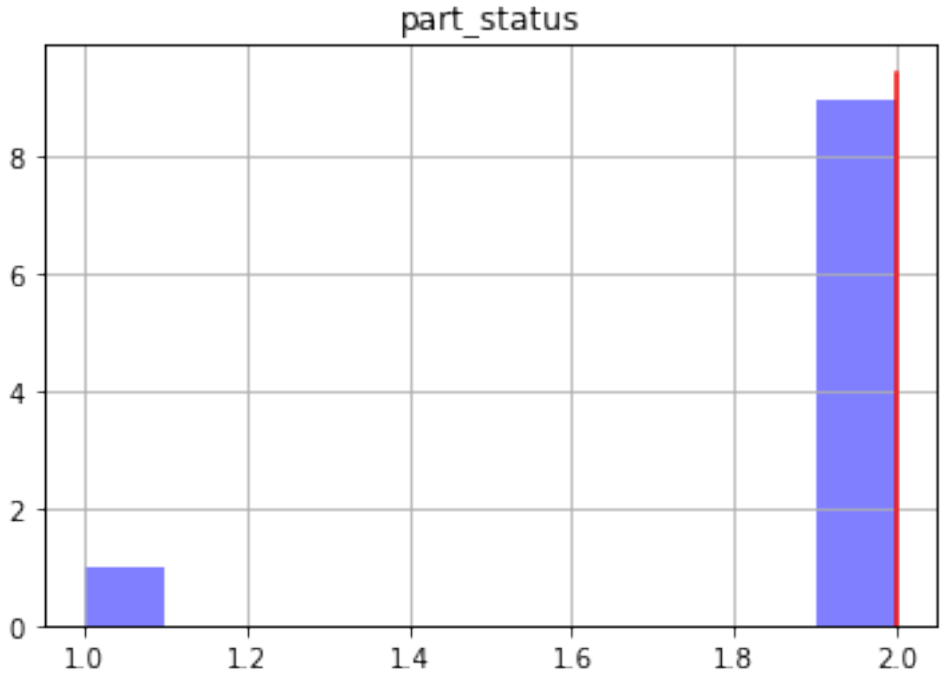

fa_tw_19791230
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
−2000
−1500
−1000
−500
0
500
1000


fa_tw_19801229
0.00200
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
−2000
−1500
−1000
−500
0
500
1000
1500
2000


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


hds_00_013_002_001
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
25
30
35
40
45
50
55


part_time
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
−2000
−1000
0
1000
2000
3000
4000


part_status
8
6
4
2
0
1.0
1.2
1.4
1.6
1.8
2.0