setup_pest_interface

April 28, 2019

1 Setup the PEST(++) interface around the enhanced Freyberg model

In this notebook, we will construct a complex model independent (non-intrusive) interface around an existing MODFLOW-NWT model using the python/flopy/pyemu stack.

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

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

In [2]:
```
b_d = os.path.join("..","base_model_files")
nam_file = "freyberg.nam"
```

1.0.1 load the model and run once to make sure everything is good-to-go

In [3]:
```
m = flopy.modflow.Modflow.load(nam_file,model_ws=b_d,check=False,forgive=False)
```

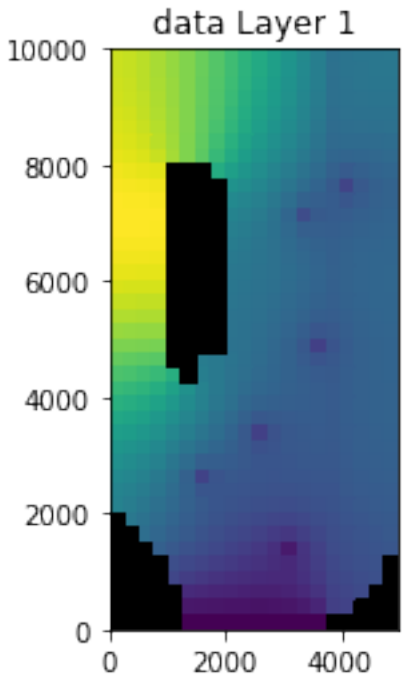
In [4]:
```
m.exe_name = "mfnwt"
m.change_model_ws("temp",reset_external=True)
m.write_input()
prep_deps.prep_template(t_d="temp")
pyemu.os_utils.run("{0} {1}".format("mfnwt",m.name+".nam"),cwd=m.model_ws)
```

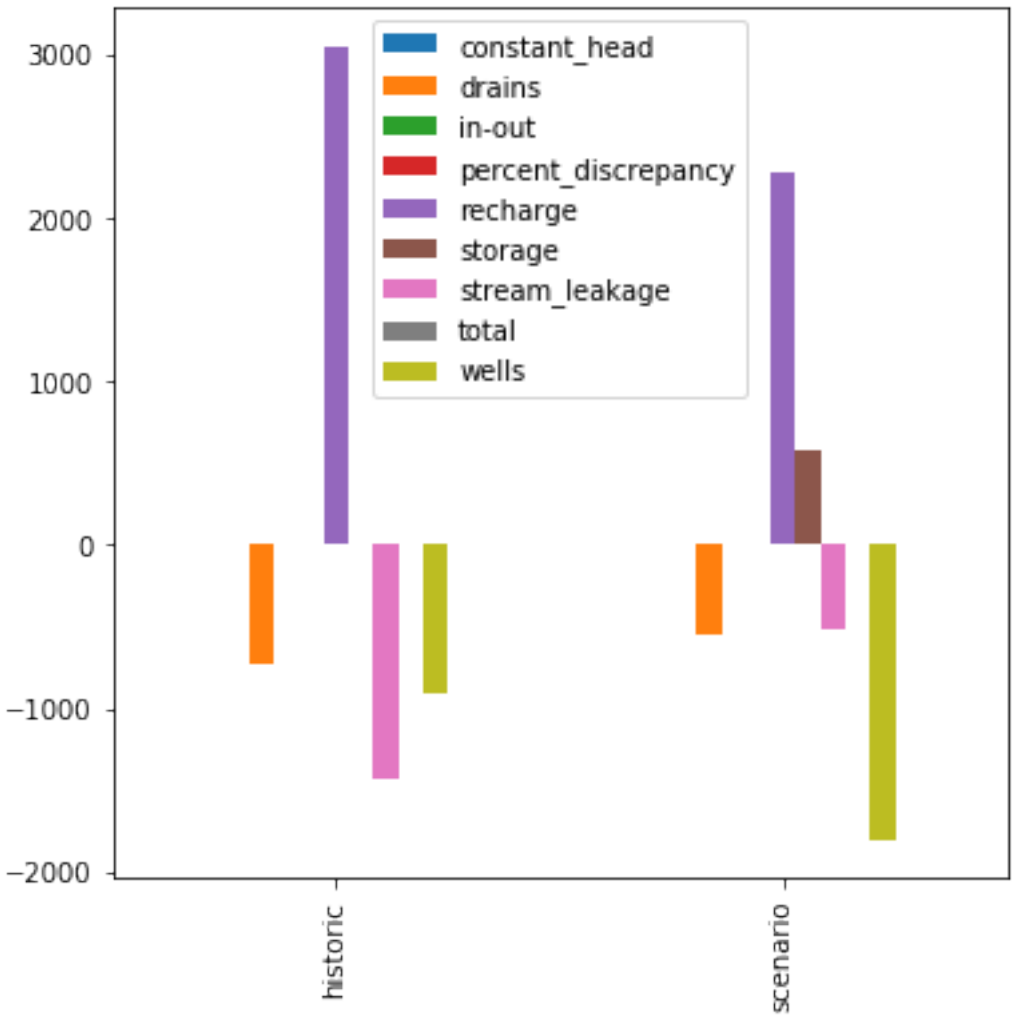
```
changing model workspace...
   temp
```

In [5]:
```
hds = flopy.utils.HeadFile(os.path.join(m.model_ws,m.name+".hds"),model=m)
hds.plot(mflay=0)
lst = flopy.utils.MfListBudget(os.path.join(m.model_ws,m.name+".list"))
```

```
df = lst.get_dataframes(diff=True)[0]
ax = df.plot(kind="bar",figsize=(6,6))
ax.set_xticklabels(["historic","scenario"])
```

```
Out[5]: [Text(0, 0, 'historic'), Text(0, 0, 'scenario')]
```



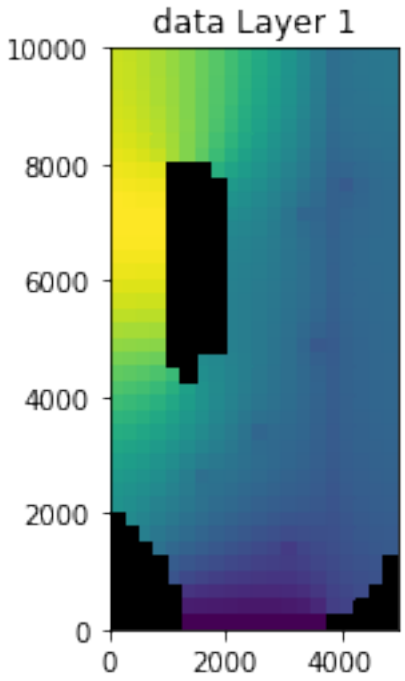


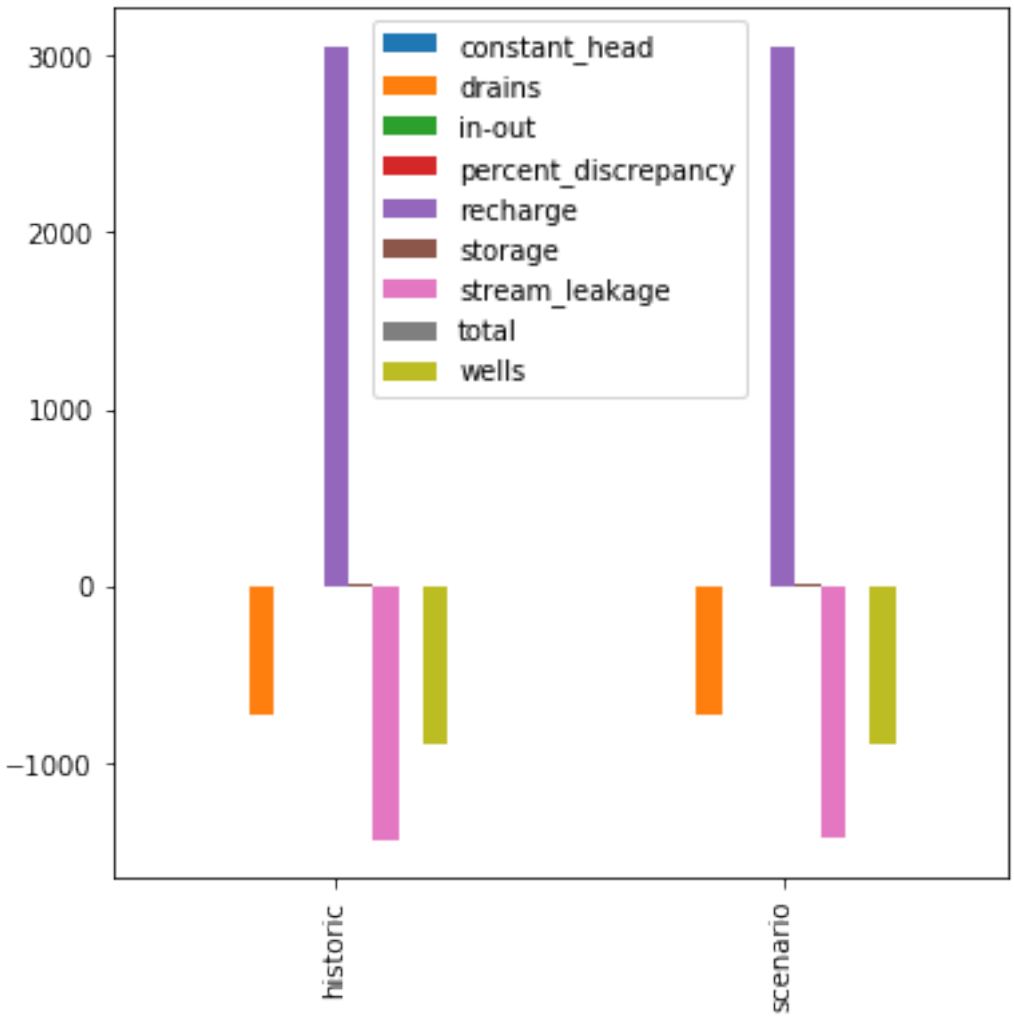
We can see the effect of the "scenario" in the second stress period with less recharge and more abstraction. However, what we are going to do is implement this scenario with parameters so we can more easy account for the stochastic nature of the forcing conditions during the scenario stress period and also make implemention of future scenarios work in this stochastic framework:

```
In [6]: # reset scenario period recharge
        m.rch.rech[1] = m.rch.rech[0]
        # reset scenario period abstraction
        m.wel.stress_period_data[1] = m.wel.stress_period_data[0]
        m.write_input()
        pyemu.os_utils.run("{0} {1}".format("mfnwt",m.name+".nam"),cwd=m.model_ws)
        hds = flopy.utils.HeadFile(os.path.join(m.model_ws,m.name+".hds"),model=m)
        axes = hds.plot(mflay=0)

        lst = flopy.utils.MfListBudget(os.path.join(m.model_ws,m.name+".list"))
        df = lst.get_dataframes(diff=True)[0]
```

```
ax = df.plot(kind="bar",figsize=(6,6))
ax.set_xticklabels(["historic","scenario"])
```

```
Out[6]: [Text(0, 0, 'historic'), Text(0, 0, 'scenario')]
```





Now we see that the scenario and historic periods have the same water balance

1.0.2 setup data structures related to what we want to parameterize and what we want to observe

```
In [7]: props = []
        paks = ["upw.hk","upw.vka","upw.ss","upw.sy","bas6.strt"]
        for k in range(m.nlay):
            props.extend([[p,k] for p in paks])
        props.append(["rch.rech",0])
        props.append(["rch.rech",1])
```

```
In [8]: spatial_list_props = [["wel.flux",2],["drn.cond",0]]
        temporal_list_props = [["wel.flux",0],["wel.flux",1]]
```

```
In [9]: hds_kperk = [[0,k] for k in range(m.nlay)]
        hds_kperk.extend([[1,k] for k in range(m.nlay)])
```

Here we setup monitoring of the SFR ASCII outputs. we will accumulate the first 20 reaches and last 20 reaches together to form forecasts of sw-gw exchange in the headwaters (`hw`) and tailwaters (`tw`). Then we will also add each reach individually for monitoring as well

```
In [10]: sfr_obs_dict = {"hw":np.arange(1,21)}
         sfr_obs_dict["tw"] = np.arange(20,40)
         for i in range(m.nrow):
             sfr_obs_dict[i] = i+1
```

1.0.3 here we go...

This class has grown into a monster...it does (among other things): - sets up combinations of multiplier parameters for array inputs, including uniform, zones, pilot points, grids, and KL expansion types - sets up combinations of multiplier parmaeters for list inputs - handles several of the shitty modflow exceptions to the array and list style inputs - sets up large numbers of observations based on arrays or time series - writes .tpl, .ins, .pst, etc - writes a python forward run script (WAT?!) - writes a prior parameter covaraince matrix using geostatistical correlations - draws from the prior parameter covariance matrix to generate a prior parameter ensemble

```
In [11]: pst_helper = pyemu.helpers.PstFromFlopyModel(nam_file,new_model_ws="template",org_mode
                                                       const_props=props,spatial_list_props=spati
                                                        temporal_list_props=temporal_list_props,
                                                       grid_props=props,pp_props=props,sfr_pars=
                                                        sfr_obs=sfr_obs_dict,build_prior=False,m
                                                       pp_space=3)
         prep_deps.prep_template(t_d=pst_helper.new_model_ws)
```

```
2019-04-28 15:06:02.616574 starting: loading flopy model

Creating new model with name: freyberg
----------------------------------------------------

Parsing the namefile --> temp/freyberg.nam

----------------------------------------------------
External unit dictionary:
OrderedDict([(2, filename:temp/freyberg.list, filetype:LIST), (11, filename:temp/freyberg.dis,
----------------------------------------------------

ModflowBas6 free format:True

loading dis package file...
   Loading dis package with:
      3 layers, 40 rows, 20 columns, and 2 stress periods
   loading laycbd...
   loading delr...
   loading delc...
   loading top...
   loading botm...
```

```
      for 3 layers and 0 confining beds
   loading stress period data...
       for 2 stress periods
adding Package:  DIS
   DIS  package load...success
   LIST package load...skipped
loading bas6 package file...
adding Package:  BAS6
   BAS6 package load...success
loading upw package file...
   loading ipakcb, HDRY, NPUPW, IPHDRY...
   loading LAYTYP...
   loading LAYAVG...
   loading CHANI...
   loading LAYVKA...
   loading LAYWET...
   loading hk layer   1...
   loading vka layer   1...
   loading ss layer   1...
   loading sy layer   1...
   loading hk layer   2...
   loading vka layer   2...
   loading ss layer   2...
   loading sy layer   2...
   loading hk layer   3...
   loading vka layer   3...
   loading ss layer   3...
   loading sy layer   3...
Adding freyberg.cbc (unit=50) to the output list.
adding Package:  UPW
   UPW  package load...success
loading rch package file...
   loading rech stress period   1...
   loading rech stress period   2...
adding Package:  RCH
   RCH  package load...success
loading nwt package file...
adding Package:  NWT
   NWT  package load...success
loading oc package file...
Adding freyberg.hds (unit=51) to the output list.
adding Package:  OC
   OC   package load...success
loading lmt package file...
adding Package:  LMT6
   LMT6 package load...success
loading wel package file...
   loading <class 'flopy.modflow.mfwel.ModflowWel'> for kper     1
```

```
   loading <class 'flopy.modflow.mfwel.ModflowWel'> for kper     2
adding Package:  WEL
   WEL  package load...success
loading sfr2 package file...
Adding freyberg.sfr.out (unit=60) to the output list.
adding Package:  SFR
   SFR  package load...success
loading drn package file...
   loading <class 'flopy.modflow.mfdrn.ModflowDrn'> for kper     1
   loading <class 'flopy.modflow.mfdrn.ModflowDrn'> for kper     2
adding Package:  DRN
   DRN  package load...success
   DATA(BINARY) file load...skipped
      freyberg.cbc
   DATA(BINARY) file load...skipped
      freyberg.hds
   DATA file load...skipped
      freyberg.sfr.out
Warning: external file unit 0 does not exist in ext_unit_dict.

   The following 10 packages were successfully loaded.
      freyberg.dis
      freyberg.bas
      freyberg.upw
      freyberg.rch
      freyberg.nwt
      freyberg.oc
      freyberg.lmt6
      freyberg.wel
      freyberg.sfr
      freyberg.drn
   The following 1 packages were not loaded.
      freyberg.list
2019-04-28 15:06:02.649077 finished: loading flopy model took: 0:00:00.032503
2019-04-28 15:06:02.649196 starting: updating model attributes
2019-04-28 15:06:02.649300 finished: updating model attributes took: 0:00:00.000104
2019-04-28 15:06:02.649486 WARNING: removing existing 'new_model_ws

creating model workspace...
   template

changing model workspace...
   template
2019-04-28 15:06:03.890388 starting: writing new modflow input files

Writing packages:
   Package:  DIS
Util2d:delr: resetting 'how' to external
```

```
Util2d:delc: resetting 'how' to external
Util2d:model_top: resetting 'how' to external
Util2d:botm_layer_0: resetting 'how' to external
Util2d:botm_layer_1: resetting 'how' to external
Util2d:botm_layer_2: resetting 'how' to external
   Package:  BAS6
Util2d:ibound_layer_0: resetting 'how' to external
Util2d:ibound_layer_1: resetting 'how' to external
Util2d:ibound_layer_2: resetting 'how' to external
Util2d:strt_layer_0: resetting 'how' to external
Util2d:strt_layer_1: resetting 'how' to external
Util2d:strt_layer_2: resetting 'how' to external
   Package:  UPW
Util2d:hk: resetting 'how' to external
Util2d:vka: resetting 'how' to external
Util2d:ss: resetting 'how' to external
Util2d:sy: resetting 'how' to external
Util2d:hk: resetting 'how' to external
Util2d:vka: resetting 'how' to external
Util2d:ss: resetting 'how' to external
Util2d:sy: resetting 'how' to external
Util2d:hk: resetting 'how' to external
Util2d:vka: resetting 'how' to external
Util2d:ss: resetting 'how' to external
Util2d:sy: resetting 'how' to external
   Package:  RCH
Util2d:rech_1: resetting 'how' to external
Util2d:rech_2: resetting 'how' to external
   Package:  NWT
   Package:  OC
   Package:  LMT6
   Package:  WEL
   Package:  SFR
   Package:  DRN

2019-04-28 15:06:04.082586 finished: writing new modflow input files took: 0:00:00.192198
2019-04-28 15:06:04.083598 forward_run line:pyemu.os_utils.run('mfnwt freyberg.nam 1>freyberg.r
2019-04-28 15:06:04.084040 starting: setting up 'template/arr_org' dir
2019-04-28 15:06:04.084547 finished: setting up 'template/arr_org' dir took: 0:00:00.000507
2019-04-28 15:06:04.084949 starting: setting up 'template/arr_mlt' dir
2019-04-28 15:06:04.085524 finished: setting up 'template/arr_mlt' dir took: 0:00:00.000575
2019-04-28 15:06:04.086087 starting: setting up 'template/list_org' dir
2019-04-28 15:06:04.087419 finished: setting up 'template/list_org' dir took: 0:00:00.001332
2019-04-28 15:06:04.088044 starting: setting up 'template/list_mlt' dir
2019-04-28 15:06:04.088635 finished: setting up 'template/list_mlt' dir took: 0:00:00.000591
2019-04-28 15:06:04.089087 starting: processing temporal_list_props
2019-04-28 15:06:04.131416 finished: processing temporal_list_props took: 0:00:00.042329
2019-04-28 15:06:04.131921 starting: processing spatial_list_props
```

```
2019-04-28 15:06:04.226378 finished: processing spatial_list_props took: 0:00:00.094457
2019-04-28 15:06:04.281630 forward_run line:pyemu.helpers.apply_list_pars()

2019-04-28 15:06:04.650013 starting: writing grid tpl:hk3.dat_gr.tpl
2019-04-28 15:06:04.661348 finished: writing grid tpl:hk3.dat_gr.tpl took: 0:00:00.011335
2019-04-28 15:06:04.664106 starting: writing grid tpl:vka3.dat_gr.tpl
2019-04-28 15:06:04.674055 finished: writing grid tpl:vka3.dat_gr.tpl took: 0:00:00.009949
2019-04-28 15:06:04.676920 starting: writing grid tpl:ss3.dat_gr.tpl
2019-04-28 15:06:04.688173 finished: writing grid tpl:ss3.dat_gr.tpl took: 0:00:00.011253
2019-04-28 15:06:04.690933 starting: writing grid tpl:sy3.dat_gr.tpl
2019-04-28 15:06:04.700420 finished: writing grid tpl:sy3.dat_gr.tpl took: 0:00:00.009487
2019-04-28 15:06:04.704214 starting: writing grid tpl:strt3.dat_gr.tpl
2019-04-28 15:06:04.713462 finished: writing grid tpl:strt3.dat_gr.tpl took: 0:00:00.009248
2019-04-28 15:06:04.716283 starting: writing grid tpl:hk4.dat_gr.tpl
2019-04-28 15:06:04.726965 finished: writing grid tpl:hk4.dat_gr.tpl took: 0:00:00.010682
2019-04-28 15:06:04.729693 starting: writing grid tpl:vka4.dat_gr.tpl
2019-04-28 15:06:04.750266 finished: writing grid tpl:vka4.dat_gr.tpl took: 0:00:00.020573
2019-04-28 15:06:04.755107 starting: writing grid tpl:ss4.dat_gr.tpl
2019-04-28 15:06:04.773019 finished: writing grid tpl:ss4.dat_gr.tpl took: 0:00:00.017912
2019-04-28 15:06:04.777627 starting: writing grid tpl:sy4.dat_gr.tpl
2019-04-28 15:06:04.793723 finished: writing grid tpl:sy4.dat_gr.tpl took: 0:00:00.016096
2019-04-28 15:06:04.797746 starting: writing grid tpl:strt4.dat_gr.tpl
2019-04-28 15:06:04.811680 finished: writing grid tpl:strt4.dat_gr.tpl took: 0:00:00.013934
2019-04-28 15:06:04.815053 starting: writing grid tpl:hk5.dat_gr.tpl
2019-04-28 15:06:04.830633 finished: writing grid tpl:hk5.dat_gr.tpl took: 0:00:00.015580
2019-04-28 15:06:04.834145 starting: writing grid tpl:vka5.dat_gr.tpl
2019-04-28 15:06:04.846211 finished: writing grid tpl:vka5.dat_gr.tpl took: 0:00:00.012066
2019-04-28 15:06:04.849326 starting: writing grid tpl:ss5.dat_gr.tpl
2019-04-28 15:06:04.860784 finished: writing grid tpl:ss5.dat_gr.tpl took: 0:00:00.011458
2019-04-28 15:06:04.863853 starting: writing grid tpl:sy5.dat_gr.tpl
2019-04-28 15:06:04.873695 finished: writing grid tpl:sy5.dat_gr.tpl took: 0:00:00.009842
2019-04-28 15:06:04.876696 starting: writing grid tpl:strt5.dat_gr.tpl
2019-04-28 15:06:04.886374 finished: writing grid tpl:strt5.dat_gr.tpl took: 0:00:00.009678
2019-04-28 15:06:04.889341 starting: writing grid tpl:rech2.dat_gr.tpl
2019-04-28 15:06:04.898884 finished: writing grid tpl:rech2.dat_gr.tpl took: 0:00:00.009543
2019-04-28 15:06:04.901874 starting: writing grid tpl:rech3.dat_gr.tpl
2019-04-28 15:06:04.911849 finished: writing grid tpl:rech3.dat_gr.tpl took: 0:00:00.009975
2019-04-28 15:06:04.914639 starting: writing const tpl:hk6.dat_cn.tpl
2019-04-28 15:06:04.920836 finished: writing const tpl:hk6.dat_cn.tpl took: 0:00:00.006197
2019-04-28 15:06:04.923659 starting: writing const tpl:vka6.dat_cn.tpl
2019-04-28 15:06:04.929948 finished: writing const tpl:vka6.dat_cn.tpl took: 0:00:00.006289
2019-04-28 15:06:04.932921 starting: writing const tpl:ss6.dat_cn.tpl
2019-04-28 15:06:04.939009 finished: writing const tpl:ss6.dat_cn.tpl took: 0:00:00.006088
2019-04-28 15:06:04.941672 starting: writing const tpl:sy6.dat_cn.tpl
2019-04-28 15:06:04.947392 finished: writing const tpl:sy6.dat_cn.tpl took: 0:00:00.005720
2019-04-28 15:06:04.950027 starting: writing const tpl:strt6.dat_cn.tpl
2019-04-28 15:06:04.956445 finished: writing const tpl:strt6.dat_cn.tpl took: 0:00:00.006418
2019-04-28 15:06:04.959239 starting: writing const tpl:hk7.dat_cn.tpl
```

```
2019-04-28 15:06:04.965374 finished: writing const tpl:hk7.dat_cn.tpl took: 0:00:00.006135
2019-04-28 15:06:04.968066 starting: writing const tpl:vka7.dat_cn.tpl
2019-04-28 15:06:04.973990 finished: writing const tpl:vka7.dat_cn.tpl took: 0:00:00.005924
2019-04-28 15:06:04.976649 starting: writing const tpl:ss7.dat_cn.tpl
2019-04-28 15:06:04.983479 finished: writing const tpl:ss7.dat_cn.tpl took: 0:00:00.006830
2019-04-28 15:06:04.986481 starting: writing const tpl:sy7.dat_cn.tpl
2019-04-28 15:06:04.992849 finished: writing const tpl:sy7.dat_cn.tpl took: 0:00:00.006368
2019-04-28 15:06:04.995579 starting: writing const tpl:strt7.dat_cn.tpl
2019-04-28 15:06:05.001924 finished: writing const tpl:strt7.dat_cn.tpl took: 0:00:00.006345
2019-04-28 15:06:05.004917 starting: writing const tpl:hk8.dat_cn.tpl
2019-04-28 15:06:05.013227 finished: writing const tpl:hk8.dat_cn.tpl took: 0:00:00.008310
2019-04-28 15:06:05.017570 starting: writing const tpl:vka8.dat_cn.tpl
2019-04-28 15:06:05.024260 finished: writing const tpl:vka8.dat_cn.tpl took: 0:00:00.006690
2019-04-28 15:06:05.027521 starting: writing const tpl:ss8.dat_cn.tpl
2019-04-28 15:06:05.034476 finished: writing const tpl:ss8.dat_cn.tpl took: 0:00:00.006955
2019-04-28 15:06:05.037147 starting: writing const tpl:sy8.dat_cn.tpl
2019-04-28 15:06:05.043693 finished: writing const tpl:sy8.dat_cn.tpl took: 0:00:00.006546
2019-04-28 15:06:05.047842 starting: writing const tpl:strt8.dat_cn.tpl
2019-04-28 15:06:05.054240 finished: writing const tpl:strt8.dat_cn.tpl took: 0:00:00.006398
2019-04-28 15:06:05.057048 starting: writing const tpl:rech4.dat_cn.tpl
2019-04-28 15:06:05.066127 finished: writing const tpl:rech4.dat_cn.tpl took: 0:00:00.009079
2019-04-28 15:06:05.070810 starting: writing const tpl:rech5.dat_cn.tpl
2019-04-28 15:06:05.079153 finished: writing const tpl:rech5.dat_cn.tpl took: 0:00:00.008343
2019-04-28 15:06:05.105075 starting: setting up pilot point process
2019-04-28 15:06:05.105572 WARNING: pp_geostruct is None, using ExpVario with contribution=1 ar
2019-04-28 15:06:05.109378 pp_dict: {0: ['hk0', 'vka0', 'ss0', 'sy0', 'strt0', 'rech0', 'rech1
2019-04-28 15:06:05.109933 starting: calling setup_pilot_point_grid()
2019-04-28 15:06:06.177974 1139 pilot point parameters created
2019-04-28 15:06:06.178908 pilot point 'pargp':hk0,vka0,ss0,sy0,strt0,rech0,rech1,ss1,hk1,strt
2019-04-28 15:06:06.179211 finished: calling setup_pilot_point_grid() took: 0:00:01.069278
2019-04-28 15:06:06.182059 starting: calculating factors for p=hk0, k=0
2019-04-28 15:06:06.183130 saving krige variance file:template/pp_k0_general_zn.fac
2019-04-28 15:06:06.183485 saving krige factors file:template/pp_k0_general_zn.fac
starting interp point loop for 800 points
took 2.91235 seconds
2019-04-28 15:06:09.160129 finished: calculating factors for p=hk0, k=0 took: 0:00:02.978070
2019-04-28 15:06:09.161420 starting: calculating factors for p=vka0, k=0
2019-04-28 15:06:09.162183 finished: calculating factors for p=vka0, k=0 took: 0:00:00.000763
2019-04-28 15:06:09.162767 starting: calculating factors for p=ss0, k=0
2019-04-28 15:06:09.163499 finished: calculating factors for p=ss0, k=0 took: 0:00:00.000732
2019-04-28 15:06:09.164111 starting: calculating factors for p=sy0, k=0
2019-04-28 15:06:09.165102 finished: calculating factors for p=sy0, k=0 took: 0:00:00.000991
2019-04-28 15:06:09.166363 starting: calculating factors for p=strt0, k=0
2019-04-28 15:06:09.167463 finished: calculating factors for p=strt0, k=0 took: 0:00:00.001100
2019-04-28 15:06:09.168305 starting: calculating factors for p=rech0, k=0
2019-04-28 15:06:09.169049 finished: calculating factors for p=rech0, k=0 took: 0:00:00.000744
2019-04-28 15:06:09.169938 starting: calculating factors for p=rech1, k=0
2019-04-28 15:06:09.170863 finished: calculating factors for p=rech1, k=0 took: 0:00:00.000925
```

```
2019-04-28 15:06:09.172073 starting: calculating factors for p=ss1, k=1
2019-04-28 15:06:09.173031 saving krige variance file:template/pp_k1_general_zn.fac
2019-04-28 15:06:09.173393 saving krige factors file:template/pp_k1_general_zn.fac
starting interp point loop for 800 points
took 3.015954 seconds
2019-04-28 15:06:12.253421 finished: calculating factors for p=ss1, k=1 took: 0:00:03.081348
2019-04-28 15:06:12.254633 starting: calculating factors for p=hk1, k=1
2019-04-28 15:06:12.255928 finished: calculating factors for p=hk1, k=1 took: 0:00:00.001295
2019-04-28 15:06:12.256581 starting: calculating factors for p=strt1, k=1
2019-04-28 15:06:12.257314 finished: calculating factors for p=strt1, k=1 took: 0:00:00.000733
2019-04-28 15:06:12.257919 starting: calculating factors for p=vka1, k=1
2019-04-28 15:06:12.259034 finished: calculating factors for p=vka1, k=1 took: 0:00:00.001115
2019-04-28 15:06:12.260042 starting: calculating factors for p=sy1, k=1
2019-04-28 15:06:12.261018 finished: calculating factors for p=sy1, k=1 took: 0:00:00.000976
2019-04-28 15:06:12.261801 starting: calculating factors for p=sy2, k=2
2019-04-28 15:06:12.262577 saving krige variance file:template/pp_k2_general_zn.fac
2019-04-28 15:06:12.262637 saving krige factors file:template/pp_k2_general_zn.fac
starting interp point loop for 800 points
took 2.549692 seconds
2019-04-28 15:06:14.872890 finished: calculating factors for p=sy2, k=2 took: 0:00:02.611089
2019-04-28 15:06:14.874037 starting: calculating factors for p=vka2, k=2
2019-04-28 15:06:14.874848 finished: calculating factors for p=vka2, k=2 took: 0:00:00.000811
2019-04-28 15:06:14.875451 starting: calculating factors for p=strt2, k=2
2019-04-28 15:06:14.876194 finished: calculating factors for p=strt2, k=2 took: 0:00:00.000743
2019-04-28 15:06:14.877202 starting: calculating factors for p=hk2, k=2
2019-04-28 15:06:14.878377 finished: calculating factors for p=hk2, k=2 took: 0:00:00.001175
2019-04-28 15:06:14.879217 starting: calculating factors for p=ss2, k=2
2019-04-28 15:06:14.880065 finished: calculating factors for p=ss2, k=2 took: 0:00:00.000848
2019-04-28 15:06:14.880156 starting: processing pp_prefix:hk2
2019-04-28 15:06:14.892404 starting: processing pp_prefix:hk1
2019-04-28 15:06:14.901822 starting: processing pp_prefix:vka0
2019-04-28 15:06:14.911015 starting: processing pp_prefix:strt2
2019-04-28 15:06:14.920039 starting: processing pp_prefix:rech1
2019-04-28 15:06:14.928597 starting: processing pp_prefix:ss0
2019-04-28 15:06:14.937013 starting: processing pp_prefix:hk0
2019-04-28 15:06:14.946220 starting: processing pp_prefix:ss2
2019-04-28 15:06:14.954495 starting: processing pp_prefix:sy2
2019-04-28 15:06:14.963187 starting: processing pp_prefix:strt0
2019-04-28 15:06:14.972207 starting: processing pp_prefix:rech0
2019-04-28 15:06:14.980722 starting: processing pp_prefix:sy0
2019-04-28 15:06:14.989324 starting: processing pp_prefix:ss1
2019-04-28 15:06:14.997631 starting: processing pp_prefix:vka1
2019-04-28 15:06:15.006271 starting: processing pp_prefix:sy1
2019-04-28 15:06:15.014776 starting: processing pp_prefix:strt1
2019-04-28 15:06:15.023527 starting: processing pp_prefix:vka2
2019-04-28 15:06:15.113409 finished: setting up pilot point process took: 0:00:10.008334
2019-04-28 15:06:15.113965 starting: setting up grid process
2019-04-28 15:06:15.114044 WARNING: grid_geostruct is None, using ExpVario with contribution=1
```

```
2019-04-28 15:06:15.114163 finished: setting up grid process took: 0:00:00.000198
2019-04-28 15:06:15.117164 starting: save test mlt array arr_mlt/hk0.dat_pp
2019-04-28 15:06:15.119319 finished: save test mlt array arr_mlt/hk0.dat_pp took: 0:00:00.0021
2019-04-28 15:06:15.120191 starting: save test mlt array arr_mlt/vka0.dat_pp
2019-04-28 15:06:15.122512 finished: save test mlt array arr_mlt/vka0.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.123439 starting: save test mlt array arr_mlt/ss0.dat_pp
2019-04-28 15:06:15.129140 finished: save test mlt array arr_mlt/ss0.dat_pp took: 0:00:00.0057
2019-04-28 15:06:15.130251 starting: save test mlt array arr_mlt/sy0.dat_pp
2019-04-28 15:06:15.132375 finished: save test mlt array arr_mlt/sy0.dat_pp took: 0:00:00.0021
2019-04-28 15:06:15.133424 starting: save test mlt array arr_mlt/strt0.dat_pp
2019-04-28 15:06:15.135549 finished: save test mlt array arr_mlt/strt0.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.136436 starting: save test mlt array arr_mlt/hk1.dat_pp
2019-04-28 15:06:15.138591 finished: save test mlt array arr_mlt/hk1.dat_pp took: 0:00:00.0021
2019-04-28 15:06:15.139597 starting: save test mlt array arr_mlt/vka1.dat_pp
2019-04-28 15:06:15.141850 finished: save test mlt array arr_mlt/vka1.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.142781 starting: save test mlt array arr_mlt/ss1.dat_pp
2019-04-28 15:06:15.144881 finished: save test mlt array arr_mlt/ss1.dat_pp took: 0:00:00.0021
2019-04-28 15:06:15.145601 starting: save test mlt array arr_mlt/sy1.dat_pp
2019-04-28 15:06:15.147968 finished: save test mlt array arr_mlt/sy1.dat_pp took: 0:00:00.0023
2019-04-28 15:06:15.148878 starting: save test mlt array arr_mlt/strt1.dat_pp
2019-04-28 15:06:15.150963 finished: save test mlt array arr_mlt/strt1.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.151859 starting: save test mlt array arr_mlt/hk2.dat_pp
2019-04-28 15:06:15.153955 finished: save test mlt array arr_mlt/hk2.dat_pp took: 0:00:00.0020
2019-04-28 15:06:15.154983 starting: save test mlt array arr_mlt/vka2.dat_pp
2019-04-28 15:06:15.157240 finished: save test mlt array arr_mlt/vka2.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.158165 starting: save test mlt array arr_mlt/ss2.dat_pp
2019-04-28 15:06:15.160230 finished: save test mlt array arr_mlt/ss2.dat_pp took: 0:00:00.0020
2019-04-28 15:06:15.161043 starting: save test mlt array arr_mlt/sy2.dat_pp
2019-04-28 15:06:15.163337 finished: save test mlt array arr_mlt/sy2.dat_pp took: 0:00:00.0022
2019-04-28 15:06:15.164357 starting: save test mlt array arr_mlt/strt2.dat_pp
2019-04-28 15:06:15.166709 finished: save test mlt array arr_mlt/strt2.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.167645 starting: save test mlt array arr_mlt/rech0.dat_pp
2019-04-28 15:06:15.169762 finished: save test mlt array arr_mlt/rech0.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.170640 starting: save test mlt array arr_mlt/rech1.dat_pp
2019-04-28 15:06:15.173067 finished: save test mlt array arr_mlt/rech1.dat_pp took: 0:00:00.002
2019-04-28 15:06:15.174246 starting: save test mlt array arr_mlt/hk3.dat_gr
2019-04-28 15:06:15.176909 finished: save test mlt array arr_mlt/hk3.dat_gr took: 0:00:00.0026
2019-04-28 15:06:15.178068 starting: save test mlt array arr_mlt/vka3.dat_gr
2019-04-28 15:06:15.181052 finished: save test mlt array arr_mlt/vka3.dat_gr took: 0:00:00.0029
2019-04-28 15:06:15.182142 starting: save test mlt array arr_mlt/ss3.dat_gr
2019-04-28 15:06:15.184345 finished: save test mlt array arr_mlt/ss3.dat_gr took: 0:00:00.0022
2019-04-28 15:06:15.185404 starting: save test mlt array arr_mlt/sy3.dat_gr
2019-04-28 15:06:15.188476 finished: save test mlt array arr_mlt/sy3.dat_gr took: 0:00:00.0030
2019-04-28 15:06:15.189977 starting: save test mlt array arr_mlt/strt3.dat_gr
2019-04-28 15:06:15.193428 finished: save test mlt array arr_mlt/strt3.dat_gr took: 0:00:00.003
2019-04-28 15:06:15.194777 starting: save test mlt array arr_mlt/hk4.dat_gr
2019-04-28 15:06:15.198174 finished: save test mlt array arr_mlt/hk4.dat_gr took: 0:00:00.0033
2019-04-28 15:06:15.199776 starting: save test mlt array arr_mlt/vka4.dat_gr
```

```
2019-04-28 15:06:15.203060 finished: save test mlt array arr_mlt/vka4.dat_gr took: 0:00:00.003
2019-04-28 15:06:15.204463 starting: save test mlt array arr_mlt/ss4.dat_gr
2019-04-28 15:06:15.207913 finished: save test mlt array arr_mlt/ss4.dat_gr took: 0:00:00.0034
2019-04-28 15:06:15.209410 starting: save test mlt array arr_mlt/sy4.dat_gr
2019-04-28 15:06:15.212890 finished: save test mlt array arr_mlt/sy4.dat_gr took: 0:00:00.0034
2019-04-28 15:06:15.214415 starting: save test mlt array arr_mlt/strt4.dat_gr
2019-04-28 15:06:15.217250 finished: save test mlt array arr_mlt/strt4.dat_gr took: 0:00:00.002
2019-04-28 15:06:15.218320 starting: save test mlt array arr_mlt/hk5.dat_gr
2019-04-28 15:06:15.221627 finished: save test mlt array arr_mlt/hk5.dat_gr took: 0:00:00.0033
2019-04-28 15:06:15.223171 starting: save test mlt array arr_mlt/vka5.dat_gr
2019-04-28 15:06:15.226621 finished: save test mlt array arr_mlt/vka5.dat_gr took: 0:00:00.003
2019-04-28 15:06:15.228046 starting: save test mlt array arr_mlt/ss5.dat_gr
2019-04-28 15:06:15.231645 finished: save test mlt array arr_mlt/ss5.dat_gr took: 0:00:00.0035
2019-04-28 15:06:15.233150 starting: save test mlt array arr_mlt/sy5.dat_gr
2019-04-28 15:06:15.236591 finished: save test mlt array arr_mlt/sy5.dat_gr took: 0:00:00.0034
2019-04-28 15:06:15.237789 starting: save test mlt array arr_mlt/strt5.dat_gr
2019-04-28 15:06:15.241215 finished: save test mlt array arr_mlt/strt5.dat_gr took: 0:00:00.003
2019-04-28 15:06:15.242875 starting: save test mlt array arr_mlt/rech2.dat_gr
2019-04-28 15:06:15.246092 finished: save test mlt array arr_mlt/rech2.dat_gr took: 0:00:00.003
2019-04-28 15:06:15.247351 starting: save test mlt array arr_mlt/rech3.dat_gr
2019-04-28 15:06:15.250399 finished: save test mlt array arr_mlt/rech3.dat_gr took: 0:00:00.003
2019-04-28 15:06:15.251470 starting: save test mlt array arr_mlt/hk6.dat_cn
2019-04-28 15:06:15.254285 finished: save test mlt array arr_mlt/hk6.dat_cn took: 0:00:00.0028
2019-04-28 15:06:15.255748 starting: save test mlt array arr_mlt/vka6.dat_cn
2019-04-28 15:06:15.258830 finished: save test mlt array arr_mlt/vka6.dat_cn took: 0:00:00.0030
2019-04-28 15:06:15.260303 starting: save test mlt array arr_mlt/ss6.dat_cn
2019-04-28 15:06:15.263289 finished: save test mlt array arr_mlt/ss6.dat_cn took: 0:00:00.0029
2019-04-28 15:06:15.264585 starting: save test mlt array arr_mlt/sy6.dat_cn
2019-04-28 15:06:15.267757 finished: save test mlt array arr_mlt/sy6.dat_cn took: 0:00:00.0031
2019-04-28 15:06:15.269274 starting: save test mlt array arr_mlt/strt6.dat_cn
2019-04-28 15:06:15.272254 finished: save test mlt array arr_mlt/strt6.dat_cn took: 0:00:00.002
2019-04-28 15:06:15.273299 starting: save test mlt array arr_mlt/hk7.dat_cn
2019-04-28 15:06:15.275860 finished: save test mlt array arr_mlt/hk7.dat_cn took: 0:00:00.0025
2019-04-28 15:06:15.277274 starting: save test mlt array arr_mlt/vka7.dat_cn
2019-04-28 15:06:15.280852 finished: save test mlt array arr_mlt/vka7.dat_cn took: 0:00:00.0035
2019-04-28 15:06:15.282208 starting: save test mlt array arr_mlt/ss7.dat_cn
2019-04-28 15:06:15.285809 finished: save test mlt array arr_mlt/ss7.dat_cn took: 0:00:00.0036
2019-04-28 15:06:15.287163 starting: save test mlt array arr_mlt/sy7.dat_cn
2019-04-28 15:06:15.290468 finished: save test mlt array arr_mlt/sy7.dat_cn took: 0:00:00.0033
2019-04-28 15:06:15.291984 starting: save test mlt array arr_mlt/strt7.dat_cn
2019-04-28 15:06:15.295931 finished: save test mlt array arr_mlt/strt7.dat_cn took: 0:00:00.003
2019-04-28 15:06:15.297475 starting: save test mlt array arr_mlt/hk8.dat_cn
2019-04-28 15:06:15.299734 finished: save test mlt array arr_mlt/hk8.dat_cn took: 0:00:00.0022
2019-04-28 15:06:15.301101 starting: save test mlt array arr_mlt/vka8.dat_cn
2019-04-28 15:06:15.304620 finished: save test mlt array arr_mlt/vka8.dat_cn took: 0:00:00.0035
2019-04-28 15:06:15.306049 starting: save test mlt array arr_mlt/ss8.dat_cn
2019-04-28 15:06:15.308957 finished: save test mlt array arr_mlt/ss8.dat_cn took: 0:00:00.0029
2019-04-28 15:06:15.310006 starting: save test mlt array arr_mlt/sy8.dat_cn
```

```
2019-04-28 15:06:15.312636 finished: save test mlt array arr_mlt/sy8.dat_cn took: 0:00:00.00263
2019-04-28 15:06:15.314010 starting: save test mlt array arr_mlt/strt8.dat_cn
2019-04-28 15:06:15.317938 finished: save test mlt array arr_mlt/strt8.dat_cn took: 0:00:00.003
2019-04-28 15:06:15.319519 starting: save test mlt array arr_mlt/rech4.dat_cn
2019-04-28 15:06:15.322733 finished: save test mlt array arr_mlt/rech4.dat_cn took: 0:00:00.003
2019-04-28 15:06:15.324099 starting: save test mlt array arr_mlt/rech5.dat_cn
2019-04-28 15:06:15.327356 finished: save test mlt array arr_mlt/rech5.dat_cn took: 0:00:00.003
2019-04-28 15:06:15.827200 forward_run line:pyemu.helpers.apply_array_pars()

all zeros for runoff...skipping...
all zeros for hcond1...skipping...
all zeros for pptsw...skipping...
2019-04-28 15:06:15.956803 starting: processing obs type mflist water budget obs
2019-04-28 15:06:16.019082 forward_run line:pyemu.gw_utils.apply_mflist_budget_obs('freyberg.li
2019-04-28 15:06:16.019259 finished: processing obs type mflist water budget obs took: 0:00:00
2019-04-28 15:06:16.019953 starting: processing obs type hyd file
2019-04-28 15:06:16.020535 finished: processing obs type hyd file took: 0:00:00.000582
2019-04-28 15:06:16.020947 starting: processing obs type external obs-sim smp files
2019-04-28 15:06:16.021119 finished: processing obs type external obs-sim smp files took: 0:00
2019-04-28 15:06:16.021177 starting: processing obs type hob
2019-04-28 15:06:16.021272 finished: processing obs type hob took: 0:00:00.000095
2019-04-28 15:06:16.021343 starting: processing obs type hds
[[0, 0], [0, 1], [0, 2], [1, 0], [1, 1], [1, 2]]
2019-04-28 15:06:16.489611 finished: processing obs type hds took: 0:00:00.468268
2019-04-28 15:06:16.490195 starting: processing obs type sfr
writing 'sfr_obs.config' to template/sfr_obs.config
2019-04-28 15:06:16.915354 finished: processing obs type sfr took: 0:00:00.425159
2019-04-28 15:06:16.916029 changing dir in to template
2019-04-28 15:06:16.917185 starting: instantiating control file from i/o files
2019-04-28 15:06:16.917564 tpl files: drn.csv.tpl,wel.csv.tpl,hk3.dat_gr.tpl,vka3.dat_gr.tpl,ss
2019-04-28 15:06:16.917907 ins files: freyberg.hds.dat.ins,vol.dat.ins,freyberg.sfr.out.process
error using inschek for instruction file freyberg.hds.dat.ins:File b'./freyberg.hds.dat.obf' do
observations in this instruction file will havegeneric values.
error using inschek for instruction file vol.dat.ins:File b'./vol.dat.obf' does not exist
observations in this instruction file will havegeneric values.
error using inschek for instruction file freyberg.sfr.out.processed.ins:File b'./freyberg.sfr.o
observations in this instruction file will havegeneric values.
error using inschek for instruction file flux.dat.ins:File b'./flux.dat.obf' does not exist
observations in this instruction file will havegeneric values.
2019-04-28 15:06:17.059094 finished: instantiating control file from i/o files took: 0:00:00.14
2019-04-28 15:06:17.342304 starting: writing forward_run.py
2019-04-28 15:06:17.343018 finished: writing forward_run.py took: 0:00:00.000714
2019-04-28 15:06:17.343114 writing pst template/freyberg.pst
2019-04-28 15:06:19.080104 starting: running pestchek on freyberg.pst
2019-04-28 15:06:19.087062 finished: running pestchek on freyberg.pst took: 0:00:00.006958
2019-04-28 15:06:19.087458 starting: saving intermediate _setup_<> dfs into template
2019-04-28 15:06:19.196166 finished: saving intermediate _setup_<> dfs into template took: 0:00
2019-04-28 15:06:19.196779 all done
```

The `pst_helper` instance contains the `pyemu.Pst` instance:

```
In [12]: pst = pst_helper.pst
         pst.npar,pst.nobs
```

```
Out[12]: (13200, 4434)
```

Oh snap!

We need to set some realistic parameter bounds and account for expected (but stochastic) scenario conditions:

```
In [13]: par = pst.parameter_data
         # properties
         tag_dict = {"hk":[0.1,10.0],"vka":[0.1,10],"strt":[0.95,1.05]}
         for t,[l,u] in tag_dict.items():
             t_pars = par.loc[par.parnme.apply(lambda x: t in x ),"parnme"]
             par.loc[t_pars,"parubnd"] = u
             par.loc[t_pars,"parlbnd"] = l

         # recharge - just change the uniform recharge mult
         scen_rch = ["rech5_cn"]
         hist_rch = ["rech4_cn"]
         par.loc[par.pargp.apply(lambda x: x in scen_rch),"parubnd"] = 0.8
         par.loc[par.pargp.apply(lambda x: x in scen_rch),"parlbnd"] = 0.1
         par.loc[par.pargp.apply(lambda x: x in scen_rch),"parval1"] = 0.4
         par.loc[par.pargp.apply(lambda x: x in hist_rch),"parubnd"] = 1.2
         par.loc[par.pargp.apply(lambda x: x in hist_rch),"parlbnd"] = 0.8
         par.loc[par.pargp.apply(lambda x: x in hist_rch),"parval1"] = 1.0

         # well abstraction
         par.loc["welflux_001","parval1"] = 1.5
         par.loc["welflux_001","parlbnd"] = 1.0
         par.loc["welflux_001","parubnd"] = 2.0
         par.loc["welflux_000","parval1"] = 1.0
         par.loc["welflux_000","parlbnd"] = 0.5
         par.loc["welflux_000","parubnd"] = 1.5
```

```
In [14]: # table can also be written to a .tex file
         pst.write_par_summary_table(filename="none").sort_index()
```

```
Out[14]:                    type transform  count      initial value  \
         drncond_k00  drncond_k00       log     10                  0
         flow                flow       log      1                  0
         grhk3              grhk3       log    705                  0
         grhk4              grhk4       log    705                  0
         grhk5              grhk5       log    705                  0
         grrech2          grrech2       log    705                  0
```

grrech3	grrech3	log	705	0
grss3	grss3	log	705	0
grss4	grss4	log	705	0
grss5	grss5	log	705	0
grstrt3	grstrt3	log	705	0
grstrt4	grstrt4	log	705	0
grstrt5	grstrt5	log	705	0
grsy3	grsy3	log	705	0
grsy4	grsy4	log	705	0
grsy5	grsy5	log	705	0
grvka3	grvka3	log	705	0
grvka4	grvka4	log	705	0
grvka5	grvka5	log	705	0
hk6_cn	hk6_cn	log	1	0
hk7_cn	hk7_cn	log	1	0
hk8_cn	hk8_cn	log	1	0
pp_hk0	pp_hk0	log	67	0
pp_hk1	pp_hk1	log	67	0
pp_hk2	pp_hk2	log	67	0
pp_rech0	pp_rech0	log	67	0
pp_rech1	pp_rech1	log	67	0
pp_ss0	pp_ss0	log	67	0
pp_ss1	pp_ss1	log	67	0
pp_ss2	pp_ss2	log	67	0
pp_strt0	pp_strt0	log	67	0
pp_strt1	pp_strt1	log	67	0
pp_strt2	pp_strt2	log	67	0
pp_sy0	pp_sy0	log	67	0
pp_sy1	pp_sy1	log	67	0
pp_sy2	pp_sy2	log	67	0
pp_vka0	pp_vka0	log	67	0
pp_vka1	pp_vka1	log	67	0
pp_vka2	pp_vka2	log	67	0
rech4_cn	rech4_cn	log	1	0
rech5_cn	rech5_cn	log	1	-0.39794
ss6_cn	ss6_cn	log	1	0
ss7_cn	ss7_cn	log	1	0
ss8_cn	ss8_cn	log	1	0
strk	strk	log	40	0
strt6_cn	strt6_cn	log	1	0
strt7_cn	strt7_cn	log	1	0
strt8_cn	strt8_cn	log	1	0
sy6_cn	sy6_cn	log	1	0
sy7_cn	sy7_cn	log	1	0
sy8_cn	sy8_cn	log	1	0
vka6_cn	vka6_cn	log	1	0
vka7_cn	vka7_cn	log	1	0
vka8_cn	vka8_cn	log	1	0

```
welflux          welflux       log      2      0 to 0.176091
welflux_k02  welflux_k02       log      6                  0

                    upper bound        lower bound    standard deviation
drncond_k00                   1                 -1                   0.5
flow                    0.09691          -0.124939             0.0554622
grhk3                         1                 -1                   0.5
grhk4                         1                 -1                   0.5
grhk5                         1                 -1                   0.5
grrech2               0.0413927         -0.0457575             0.0217875
grrech3               0.0413927         -0.0457575             0.0217875
grss3                         1                 -1                   0.5
grss4                         1                 -1                   0.5
grss5                         1                 -1                   0.5
grstrt3               0.0211893         -0.0222764             0.0108664
grstrt4               0.0211893         -0.0222764             0.0108664
grstrt5               0.0211893         -0.0222764             0.0108664
grsy3                  0.243038           -0.60206              0.211275
grsy4                  0.243038           -0.60206              0.211275
grsy5                  0.243038           -0.60206              0.211275
grvka3                        1                 -1                   0.5
grvka4                        1                 -1                   0.5
grvka5                        1                 -1                   0.5
hk6_cn                        1                 -1                   0.5
hk7_cn                        1                 -1                   0.5
hk8_cn                        1                 -1                   0.5
pp_hk0                        1                 -1                   0.5
pp_hk1                        1                 -1                   0.5
pp_hk2                        1                 -1                   0.5
pp_rech0              0.0413927         -0.0457575             0.0217875
pp_rech1              0.0413927         -0.0457575             0.0217875
pp_ss0                        1                 -1                   0.5
pp_ss1                        1                 -1                   0.5
pp_ss2                        1                 -1                   0.5
pp_strt0              0.0211893         -0.0222764             0.0108664
pp_strt1              0.0211893         -0.0222764             0.0108664
pp_strt2              0.0211893         -0.0222764             0.0108664
pp_sy0                 0.243038           -0.60206              0.211275
pp_sy1                 0.243038           -0.60206              0.211275
pp_sy2                 0.243038           -0.60206              0.211275
pp_vka0                       1                 -1                   0.5
pp_vka1                       1                 -1                   0.5
pp_vka2                       1                 -1                   0.5
rech4_cn              0.0791812           -0.09691             0.0440228
rech5_cn               -0.09691                 -1              0.225772
ss6_cn                        1                 -1                   0.5
ss7_cn                        1                 -1                   0.5
ss8_cn                        1                 -1                   0.5
```

```
strk                          2                -2                    1
strt6_cn              0.0211893        -0.0222764            0.0108664
strt7_cn              0.0211893        -0.0222764            0.0108664
strt8_cn              0.0211893        -0.0222764            0.0108664
sy6_cn                 0.243038          -0.60206             0.211275
sy7_cn                 0.243038          -0.60206             0.211275
sy8_cn                 0.243038          -0.60206             0.211275
vka6_cn                       1                -1                  0.5
vka7_cn                       1                -1                  0.5
vka8_cn                       1                -1                  0.5
welflux      0.176091 to 0.30103  -0.30103 to     0  0.0752575 to 0.11928
welflux_k02                   1                -1                  0.5
```

```
In [15]: pst.write_obs_summary_table(filename="none")
```

Out[15]:

	group	value	non-zero weight	zero weight	weight	\
flaqx	flaqx	-1013.42 to 32.171	84	0	1	
flout	flout	10069 to 226396	84	0	1	
hds	hds	32.5065 to 39.6612	4230	0	1	
obgnme	obgnme	1E+10	36	0	1	

	standard deviation	percent error
flaqx	1	0.0986757 to 833.333
flout	1	0.000441704 to 0.00993147
hds	1	2.52136 to 3.07631
obgnme	1	1E-08

Lets run the process once (noptmax=0) to make sure its all plumbed up

```
In [16]: pst.control_data.noptmax = 0
         pst.write(os.path.join(pst_helper.new_model_ws,"freyberg.pst"))
         pyemu.os_utils.run("pestpp-ies freyberg.pst",cwd=pst_helper.new_model_ws)
```

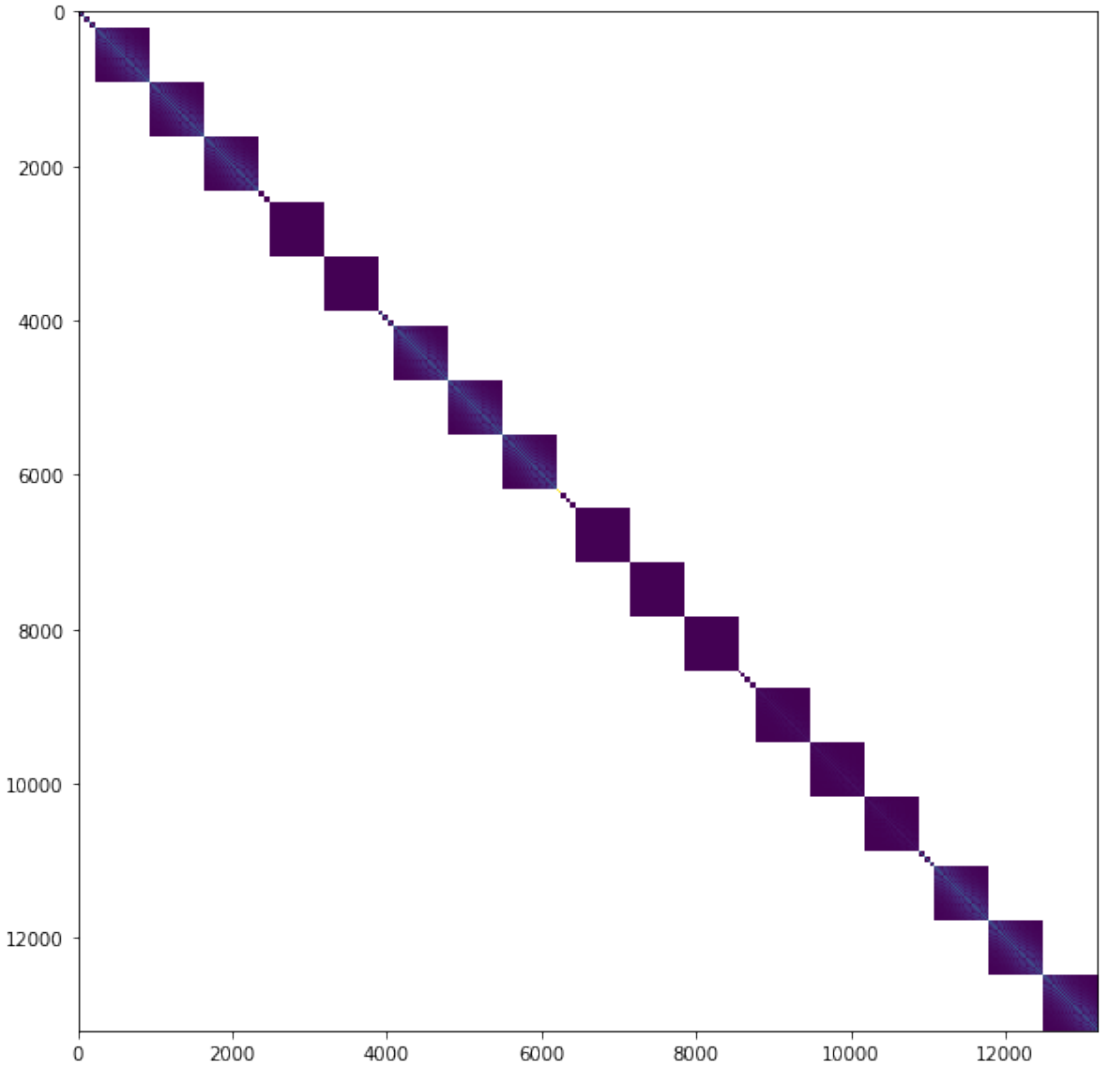
Now we need to generate the prior parameter covariance matrix and stochastic realizations. We will use the geostatistical covariance information in the pst_helper instance for this:

```
In [17]: cov = pst_helper.build_prior(fmt="binary",filename=os.path.join(pst_helper.new_model_w
         cov = np.ma.masked_where(cov.x==0,cov.x)
         fig = plt.figure(figsize=(10,10))
         ax = plt.subplot(111)
         ax.imshow(cov)
```

```
2019-04-28 15:06:28.098271 starting: building prior covariance matrix
2019-04-28 15:06:28.194162 WARNING: geospatial prior not implemented for SFR pars
```

```
/Users/jeremyw/miniconda3/lib/python3.5/site-packages/pandas/core/indexing.py:362: SettingWithC
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  self.obj[key] = _infer_fill_value(value)
/Users/jeremyw/miniconda3/lib/python3.5/site-packages/pandas/core/indexing.py:543: SettingWithC
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  self.obj[item] = s


2019-04-28 15:06:33.611545 saving prior covariance matrix to file template/prior_cov.jcb
[[0.11111111 0.10053749 0.09097008 ... 0.         0.         0.        ]
 [0.10053749 0.11111111 0.10053749 ... 0.         0.         0.        ]
 [0.09097008 0.10053749 0.11111111 ... 0.         0.         0.        ]
 ...
 [0.         0.         0.         ... 0.11111111 0.0673923  0.04871123]
 [0.         0.         0.         ... 0.0673923  0.11111111 0.05088182]
 [0.         0.         0.         ... 0.04871123 0.05088182 0.11111111]]
<class 'numpy.ndarray'>
2019-04-28 15:06:37.151700 finished: building prior covariance matrix took: 0:00:09.053429


Out[17]: <matplotlib.image.AxesImage at 0x1824ade278>
```



Boom! 10K+ parameter covariance matrix

```
In [18]: pe = pst_helper.draw(200)

2019-04-28 15:06:51.941861 starting: drawing realizations
building diagonal cov
processing  name:struct1,nugget:0.0,structures:
name:var1,contribution:1.0,a:750.0,anisotropy:1.0,bearing:0.0

working on pargroups ['pp_hk0']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
```

```
working on pargroups ['pp_vka0']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_ss0']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_sy0']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_strt0']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_rech0']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_rech1']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_ss1']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_hk1']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
```

```
working on pargroups ['pp_strt1']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_vka1']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_sy1']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_sy2']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_vka2']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_strt2']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_hk2']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['pp_ss2']
build cov matrix
done
getting diag var cov 67
scaling full cov by diag var cov
making full cov draws with home-grown goodness
```

```
processing  name:struct1,nugget:0.0,structures:
name:var1,contribution:1.0,a:180.0,anisotropy:1.0,bearing:0.0

working on pargroups ['welflux']
build cov matrix
done
getting diag var cov 2
scaling full cov by diag var cov
making full cov draws with home-grown goodness
processing  name:struct1,nugget:0.0,structures:
name:var1,contribution:1.0,a:2500.0,anisotropy:1.0,bearing:0.0

working on pargroups ['grhk3']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grvka3']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grss3']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grsy3']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grstrt3']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grhk4']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
```

```
working on pargroups ['grvka4']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grss4']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grsy4']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grstrt4']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grhk5']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grvka5']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grss5']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grsy5']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
```

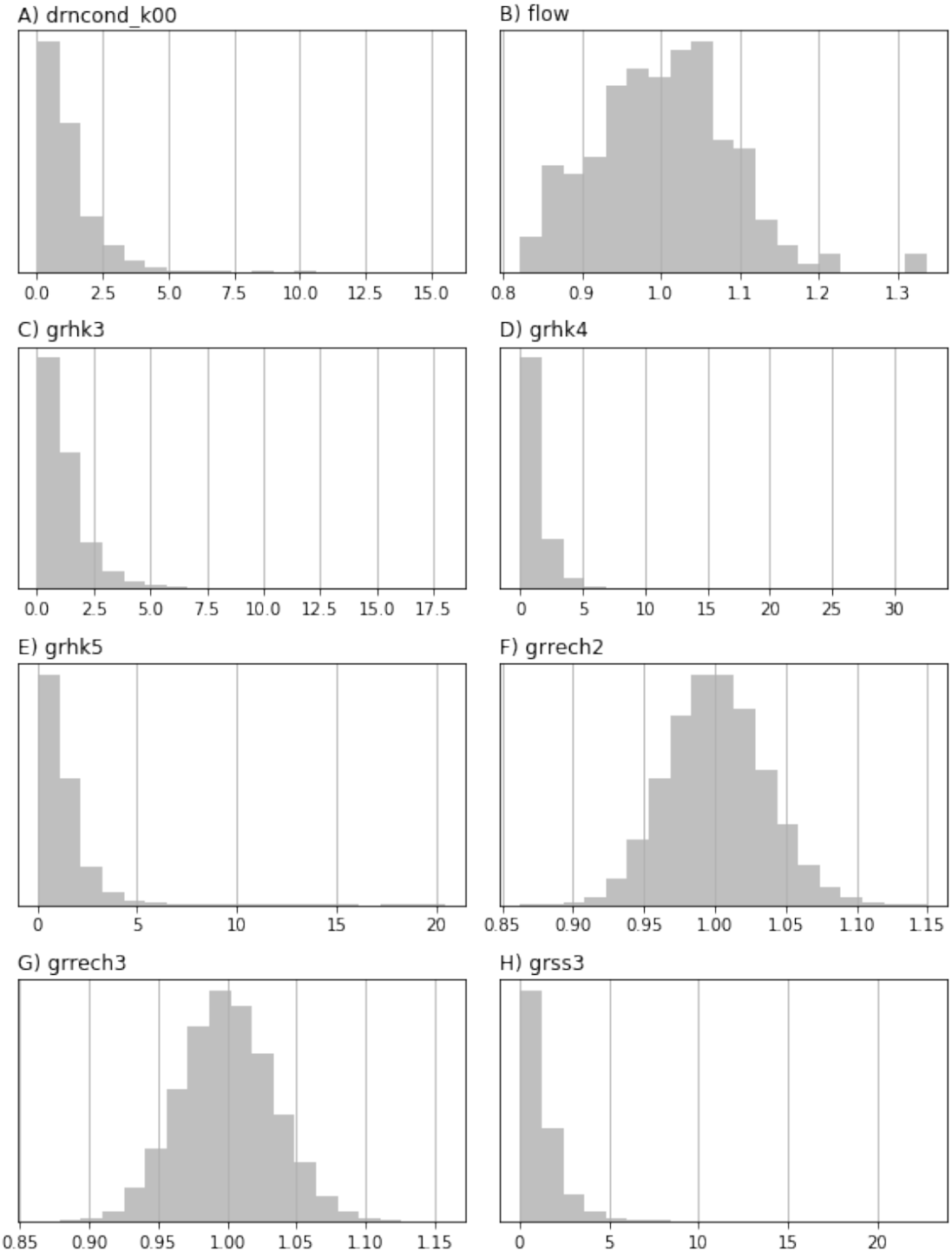
```
working on pargroups ['grstrt5']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grrech2']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grrech3']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
processing  name:struct1,nugget:0.0,structures:
name:var1,contribution:1.0,a:2500.0,anisotropy:1.0,bearing:0.0

working on pargroups ['drncond_k00']


/Users/jeremyw/miniconda3/lib/python3.5/site-packages/pandas/core/indexing.py:362: SettingWith
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  self.obj[key] = _infer_fill_value(value)
/Users/jeremyw/miniconda3/lib/python3.5/site-packages/pandas/core/indexing.py:543: SettingWith
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  self.obj[item] = s


build cov matrix
done
getting diag var cov 10
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['welflux_k02']
build cov matrix
done
getting diag var cov 6
scaling full cov by diag var cov
```

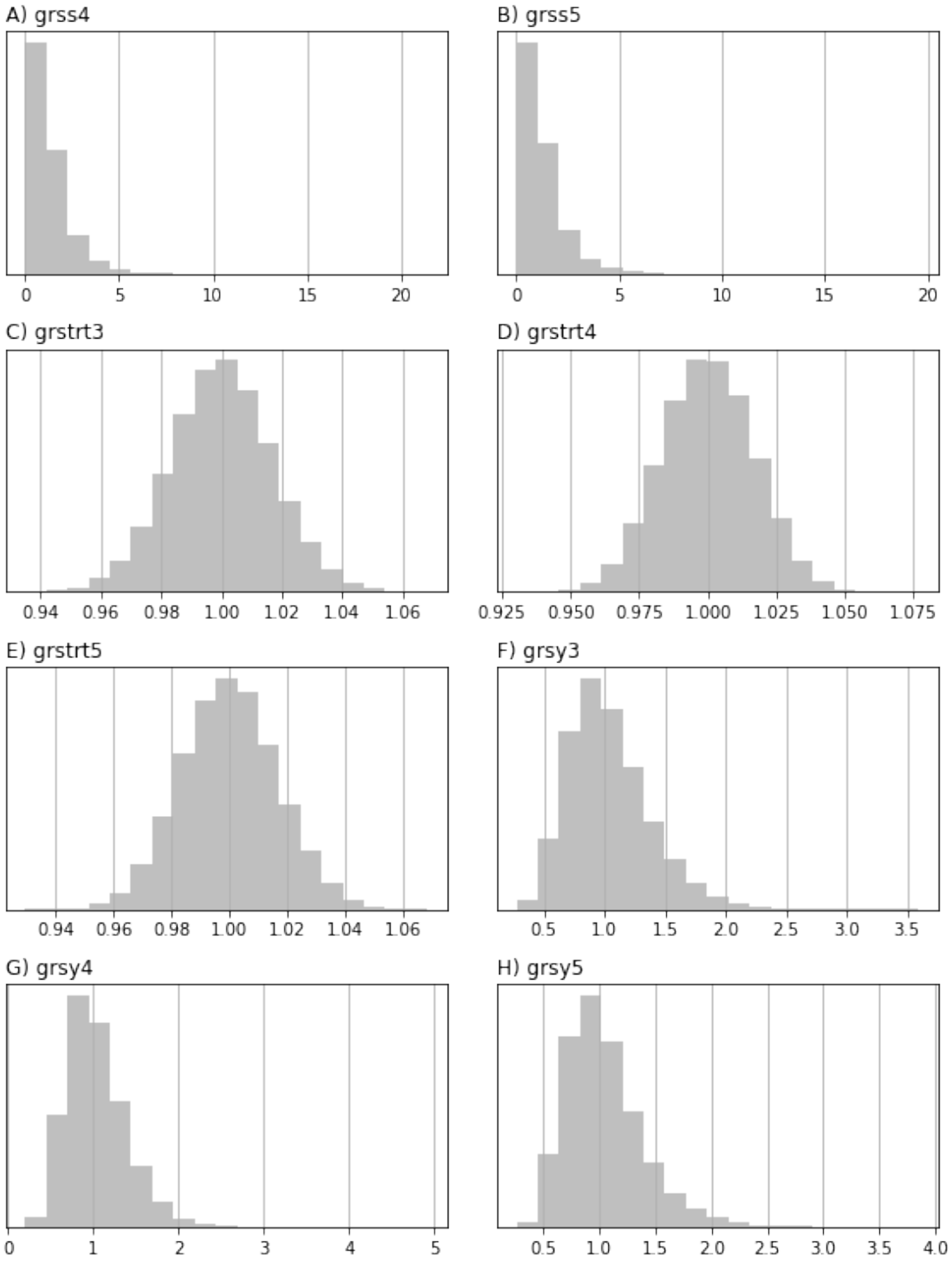
```
making full cov draws with home-grown goodness
adding remaining parameters to diagonal
2019-04-28 15:06:58.911101 finished: drawing realizations took: 0:00:06.969240
```

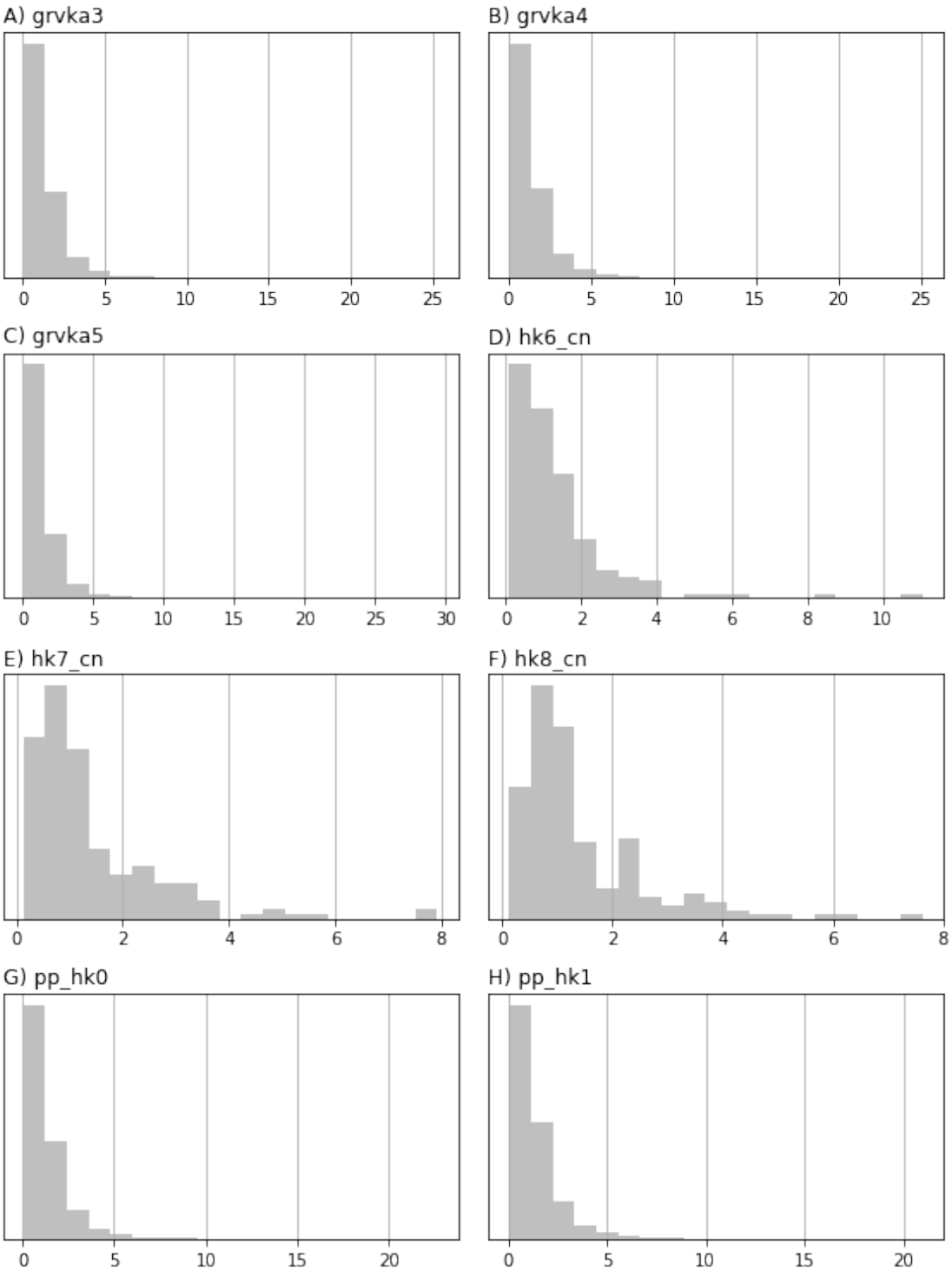
You can see that parameters are treated in parameter group (pargp) blocks for this ensemble generation. Let's plot one parameter:

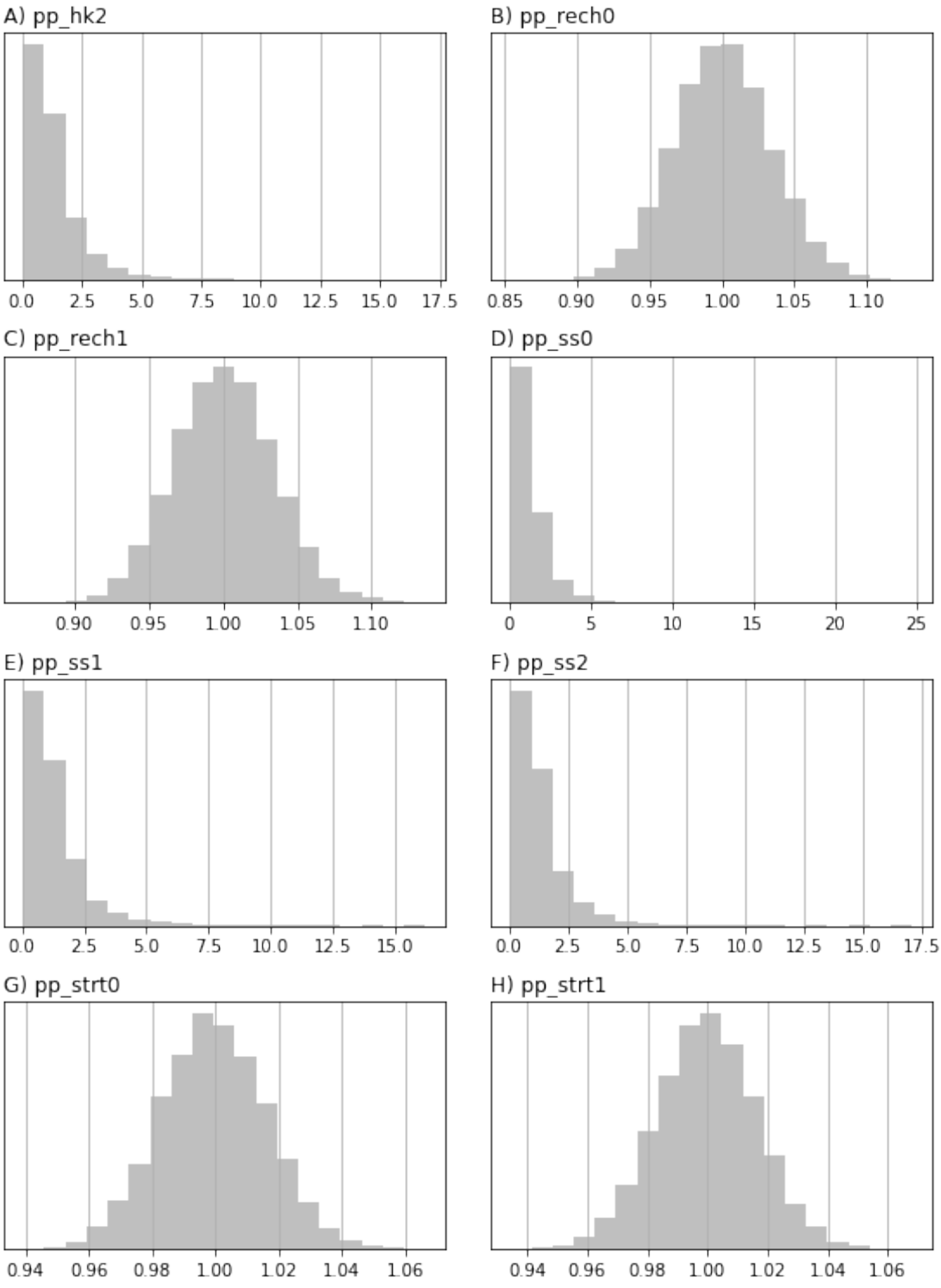
```
In [19]: par = pst_helper.pst.parameter_data
         pyemu.plot_utils.ensemble_helper(pe,plot_cols=par.groupby("pargp").groups,bins=20)
```

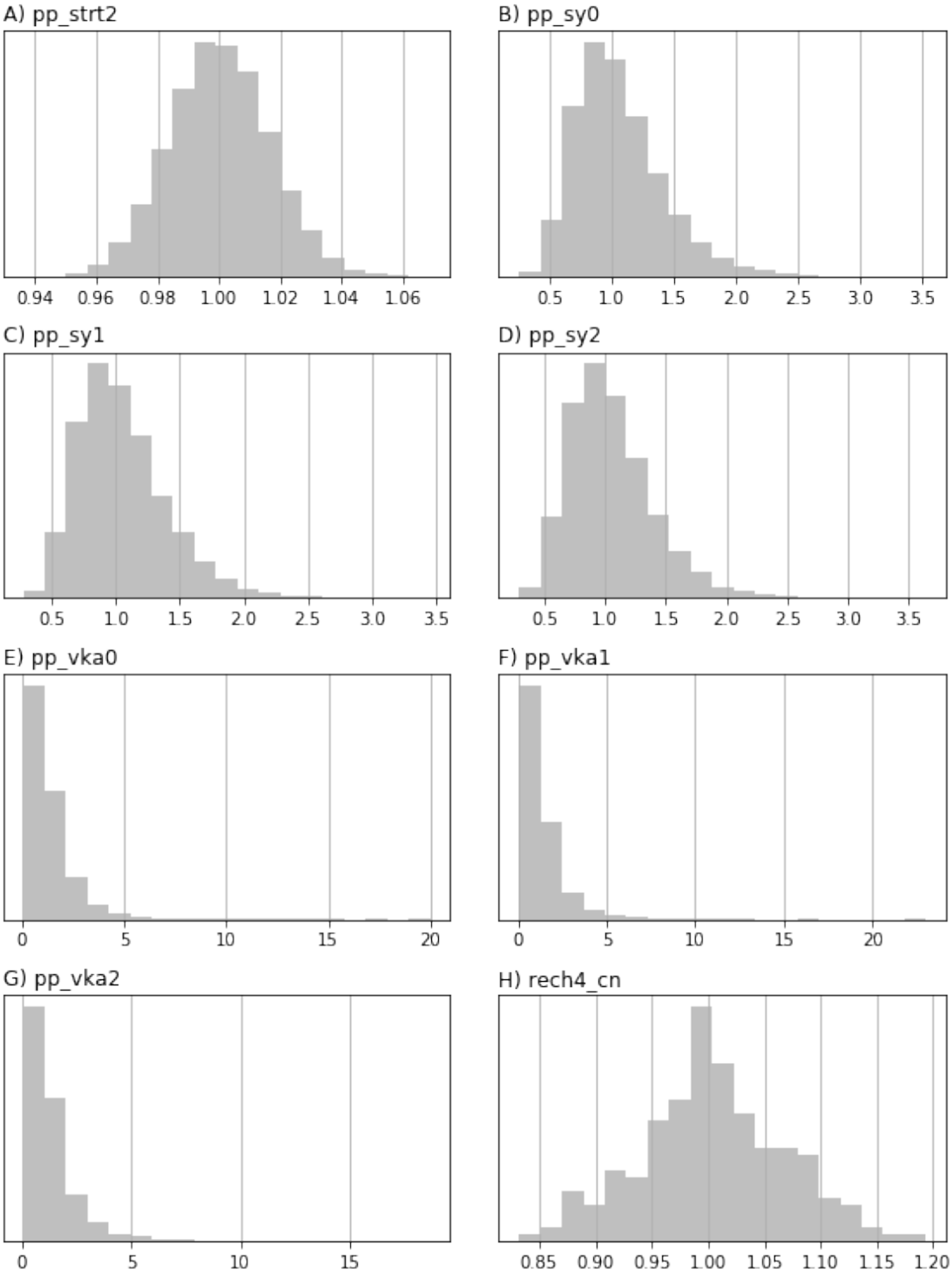
```
<Figure size 576x756 with 0 Axes>
```

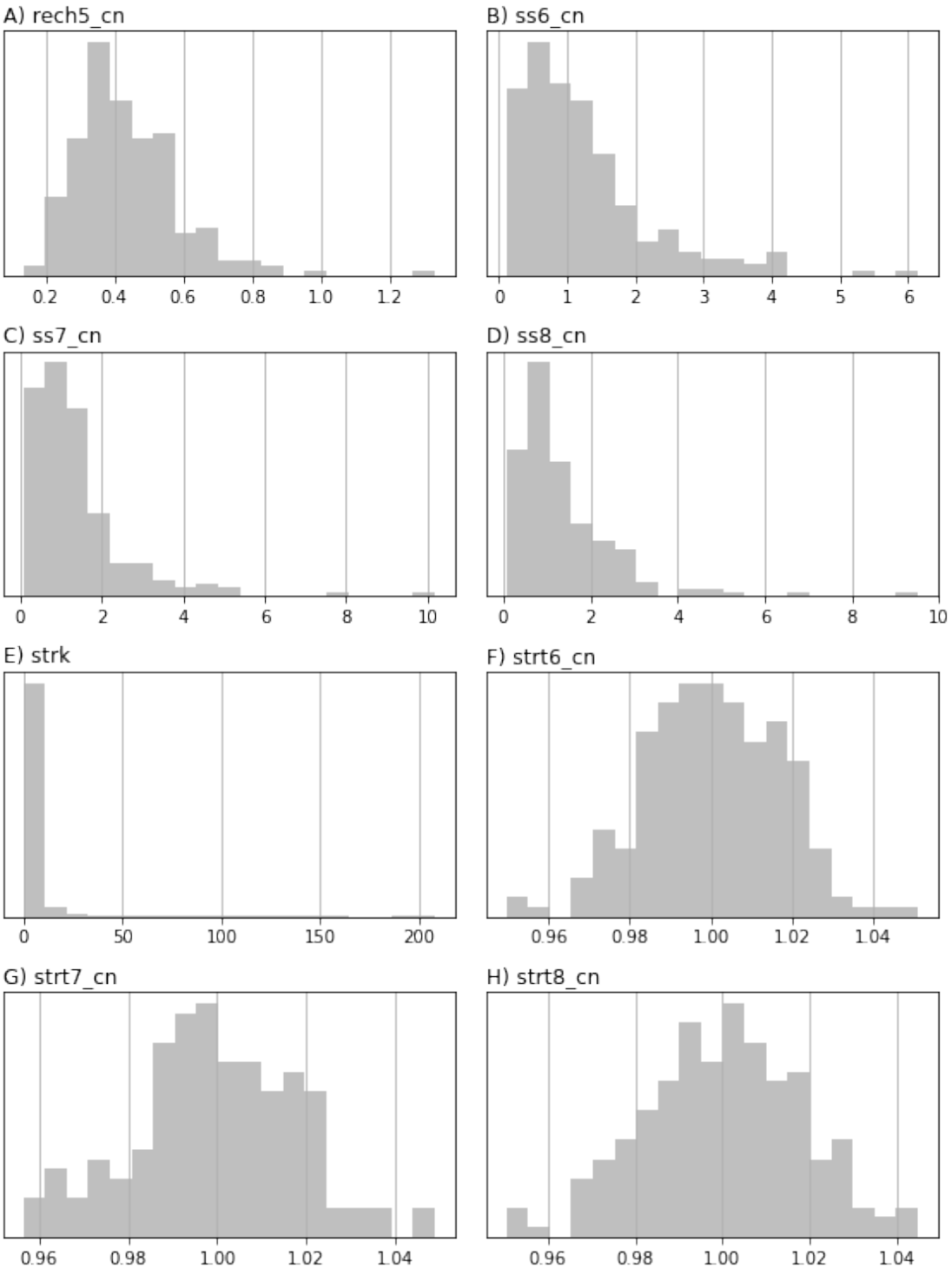

A) drncond_k00
0.0 2.5 5.0 7.5 10.0 12.5 15.0
B) flow
0.8 0.9 1.0 1.1 1.2 1.3
C) grhk3
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5
D) grhk4
0 5 10 15 20 25 30
E) grhk5
0 5 10 15 20
F) grrech2
0.85 0.90 0.95 1.00 1.05 1.10 1.15
G) grrech3
0.85 0.90 0.95 1.00 1.05 1.10 1.15
H) grss3
0 5 10 15 20

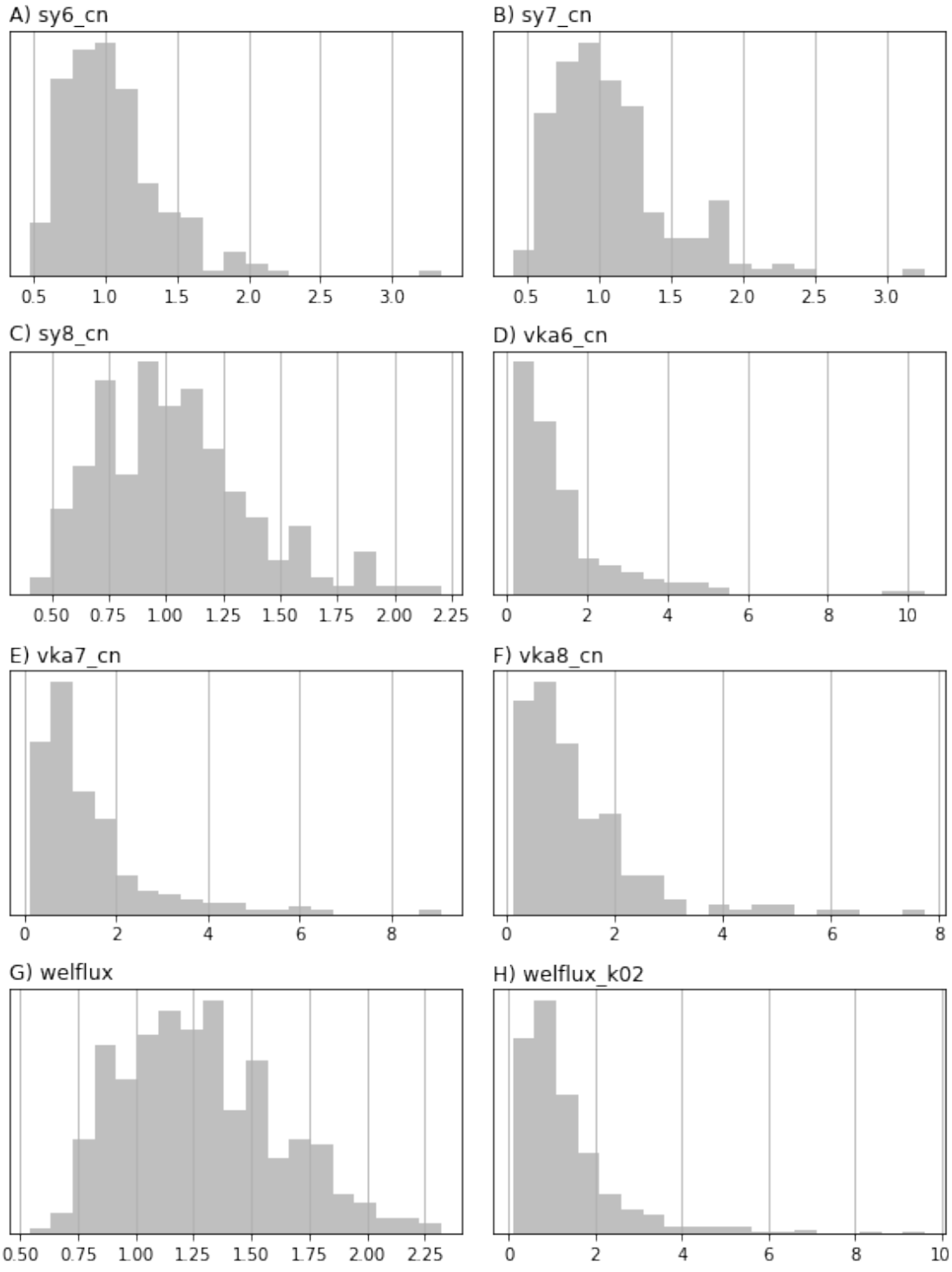

A) grss4
0
5
10
15
20
B) grss5
0
5
10
15
20
C) grstrt3
0.94
0.96
0.98
1.00
1.02
1.04
1.06
D) grstrt4
0.925
0.950
0.975
1.000
1.025
1.050
1.075
E) grstrt5
0.94
0.96
0.98
1.00
1.02
1.04
1.06
F) grsy3
0.5
1.0
1.5
2.0
2.5
3.0
3.5
G) grsy4
0
1
2
3
4
5
H) grsy5
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


A) grvka3
0
5
10
15
20
25
B) grvka4
0
5
10
15
20
25
C) grvka5
0
5
10
15
20
25
30
D) hk6_cn
0
2
4
6
8
10
E) hk7_cn
0
2
4
6
8
F) hk8_cn
0
2
4
6
8
G) pp_hk0
0
5
10
15
20
H) pp_hk1
0
5
10
15
20


A) pp_hk2
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5
B) pp_rech0
0.85 0.90 0.95 1.00 1.05 1.10
C) pp_rech1
0.90 0.95 1.00 1.05 1.10
D) pp_ss0
0 5 10 15 20 25
E) pp_ss1
0.0 2.5 5.0 7.5 10.0 12.5 15.0
F) pp_ss2
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5
G) pp_strt0
0.94 0.96 0.98 1.00 1.02 1.04 1.06
H) pp_strt1
0.94 0.96 0.98 1.00 1.02 1.04 1.06


A) pp_strt2
0.94 0.96 0.98 1.00 1.02 1.04 1.06
B) pp_sy0
0.5 1.0 1.5 2.0 2.5 3.0 3.5
C) pp_sy1
0.5 1.0 1.5 2.0 2.5 3.0 3.5
D) pp_sy2
0.5 1.0 1.5 2.0 2.5 3.0 3.5
E) pp_vka0
0 5 10 15 20
F) pp_vka1
0 5 10 15 20
G) pp_vka2
0 5 10 15
H) rech4_cn
0.85 0.90 0.95 1.00 1.05 1.10 1.15 1.20


A) rech5_cn
0.2
0.4
0.6
0.8
1.0
1.2
B) ss6_cn
0
1
2
3
4
5
6
C) ss7_cn
0
2
4
6
8
10
D) ss8_cn
0
2
4
6
8
10
E) strk
0
50
100
150
200
F) strt6_cn
0.96
0.98
1.00
1.02
1.04
G) strt7_cn
0.96
0.98
1.00
1.02
1.04
H) strt8_cn
0.96
0.98
1.00
1.02
1.04



Now we need to enforce parameter bounds and save this ensemble for later

```
In [20]: pe.enforce()
         pe.to_binary(os.path.join(pst_helper.new_model_ws,"prior.jcb"))
```

1.0.4 set weights for "observations" and identify forecasts

The next major task is to set the weights on the observations. So far, in the `pst_helper` process, we simply identified what outputs from the model we want to observe. We now use a pre-cooked csv file to set nonzero weights only for GW level observation locations used in the original Freyberg model. We will also use the SFR flow out of the last reach (`fo_39_19791230`)

```
In [21]: obs_locs = pd.read_csv(os.path.join("..","base_model_files","obs_loc.csv"))
         #build obs names that correspond to the obsnme values in the control file
         obs_locs.loc[:,"obsnme"] = obs_locs.apply(lambda x: "hds_00_{0:03d}_{1:03d}_000".forma
         obs_locs
```

Out[21]:

	row	col	obsnme
0	3	16	hds_00_002_015_000
1	3	10	hds_00_002_009_000
2	4	9	hds_00_003_008_000
3	10	2	hds_00_009_001_000
4	14	11	hds_00_013_010_000
5	16	17	hds_00_015_016_000
6	22	11	hds_00_021_010_000
7	23	16	hds_00_022_015_000
8	25	5	hds_00_024_004_000
9	27	7	hds_00_026_006_000
10	30	16	hds_00_029_015_000
11	34	8	hds_00_033_007_000
12	35	11	hds_00_034_010_000

Set all weights to zero first, then turn on the weights at only a few locations. These nonzero obs will be given meaningful weights in the prior monte carlo excersize |

```
In [22]: obs = pst_helper.pst.observation_data
         obs.loc[:,"weight"] = 0.0
         obs.loc[obs_locs.obsnme,"weight"] = 1.0
         obs.loc[obs_locs.obsnme,"obgnme"] = "calhead"
         obs.loc["fo_39_19791230","weight"] = 1.0
         obs.loc["fo_39_19791230","obgnme"] = "calflux"
         pst.nnz_obs_names
```

```
Out[22]: ['fo_39_19791230',
          'hds_00_002_009_000',
          'hds_00_002_015_000',
          'hds_00_003_008_000',
          'hds_00_009_001_000',
          'hds_00_013_010_000',
          'hds_00_015_016_000',
          'hds_00_021_010_000',
          'hds_00_022_015_000',
          'hds_00_024_004_000',
          'hds_00_026_006_000',
```

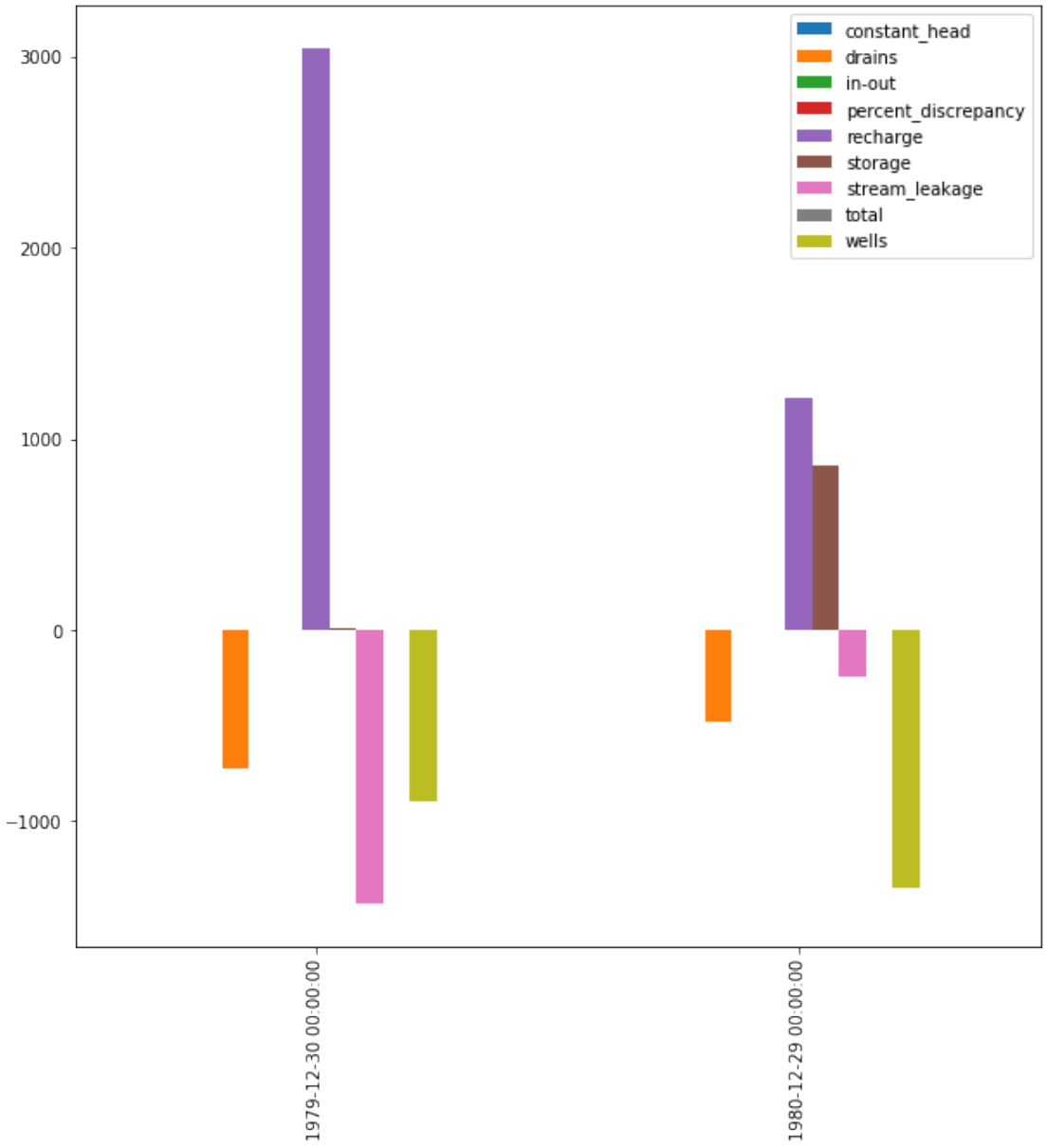
```
 'hds_00_029_015_000',
 'hds_00_033_007_000',
 'hds_00_034_010_000']
```

Now we will define which model outputs are going to be treated as "forecasts" and save the control file

```
In [23]: swgw_forecasts = obs.loc[obs.obsnme.apply(lambda x: "fa" in x and ("hw" in x or "tw" i
         hds_forecasts = obs.loc[obs.obsnme.apply(lambda x: "hds_00_015_002" in x),"obsnme"].to
         forecasts = swgw_forecasts
         forecasts.extend(hds_forecasts)
         pst_helper.pst.pestpp_options["forecasts"] = forecasts
         pst.write(os.path.join(pst_helper.new_model_ws,"freyberg.pst"))
```

Run one last time. `phi` should be near zero since we haven't change the `parval1` values for historic stress period and only the 13 gw level obs have nonzero weights

```
In [24]: pyemu.os_utils.run("pestpp-ies.exe freyberg.pst",cwd=pst_helper.new_model_ws)
         pst = pyemu.Pst(os.path.join(pst_helper.new_model_ws,"freyberg.pst"))
         pst.phi

Out[24]: 9.456182577320024e-19

In [25]: lst = flopy.utils.MfListBudget(os.path.join("template","freyberg.list"))
         df = lst.get_dataframes(diff=True)[0]
         df.plot(kind="bar",figsize=(10,10))

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x182448f2e8>
```



We see the effect of our parameterized scenario - a large drop in recharge and more abstraction.