setup_pest_interface

April 29, 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
        import redis
```

```
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"
```

This seemingly simple function call will spatially rediscretize the original freyberg model by cutting each row and column by 3's

```
In [3]: #redis_fac = 3
        #mr = redis.redis_freyberg(fac=redis_fac,b_d=b_d)
        #b_d = mr.model_ws
```

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

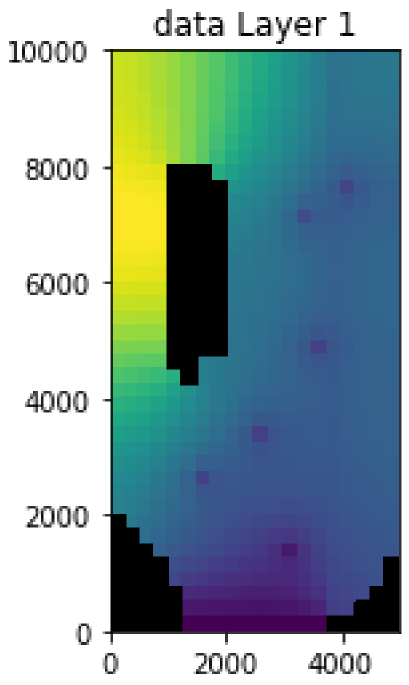
```
In [4]: m = flopy.modflow.Modflow.load(nam_file,model_ws=b_d,check=False,forgive=False)
```

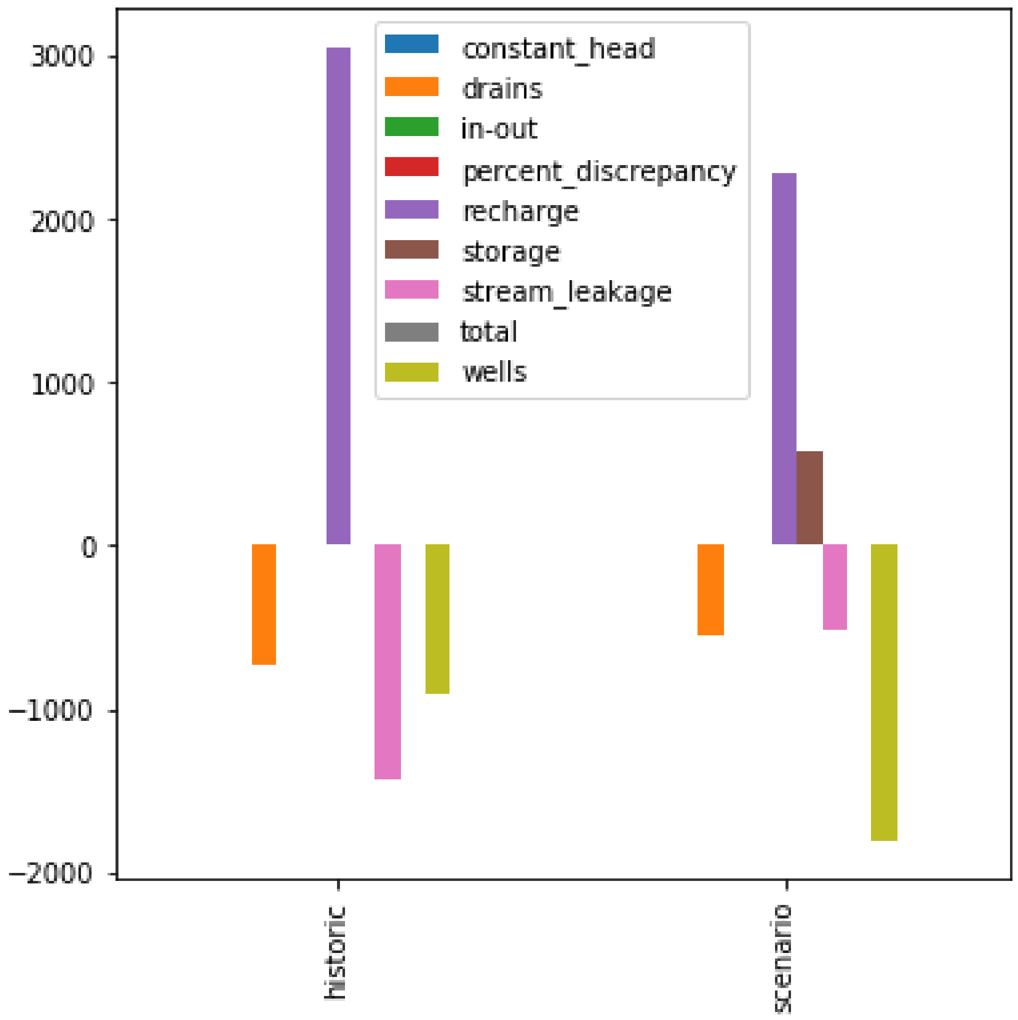
```
In [5]: m.exe_name = "mfnwt"
        m.change_model_ws("temp",reset_external=True)
        m.write_input()
        prep_deps.prep_template(t_d="temp")
        pyemu.os_utils.run("{0} {1}".format("mfnwt",m.name+".nam"),cwd=m.model_ws)
```

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

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

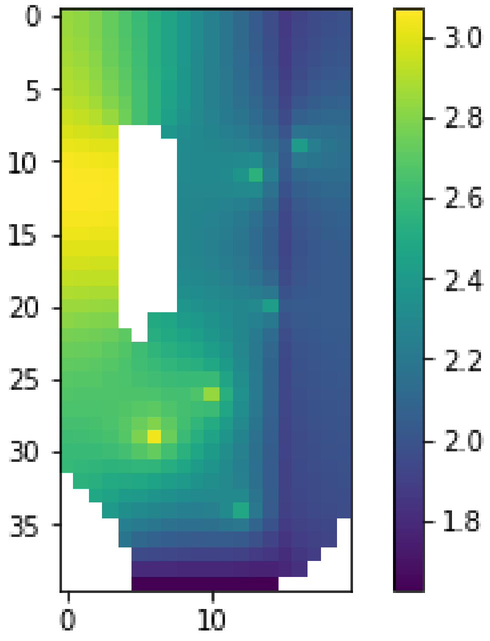




We can see the effect of the "scenario" in the second stress period with less recharge and more abstraction.

Plot depth to water

```
In [7]: dtw = m.dis.top.array - hds.get_data()[0,:,:]
        dtw = np.ma.masked_where(m.bas6.ibound[0].array==0,dtw)
        c = plt.imshow(dtw)
        plt.colorbar(c)
        plt.show()
```

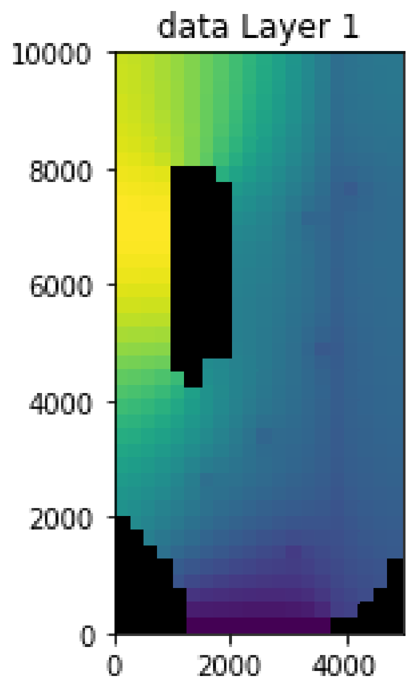


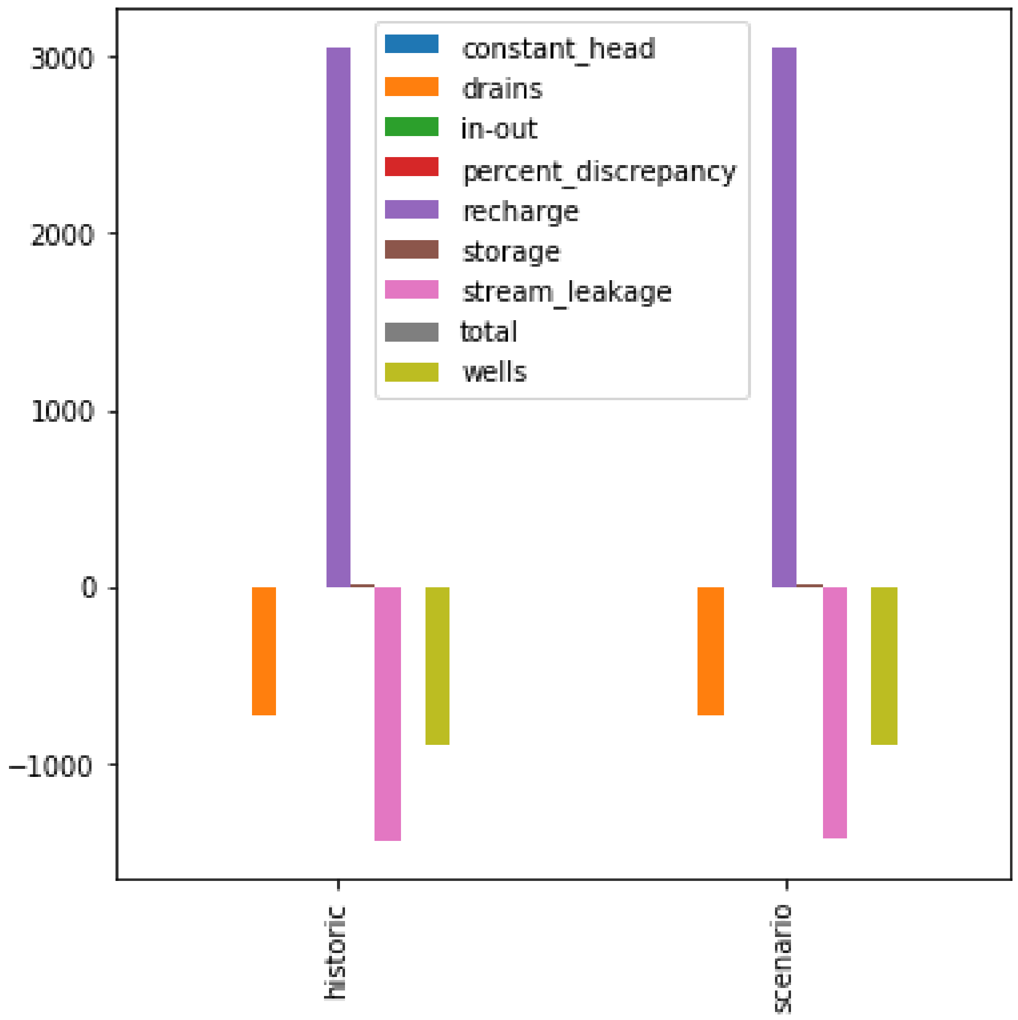
We are going to do is implement the scenario with parameters so we can more easy account for the stochastic nature of the forcing conditions during the scenario stress period and also make implemention of future scenarios work in this stochastic framework:

```
In [8]: # 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[8]: [Text(0, 0, 'historic'), Text(0, 0, 'scenario')]
```


data Layer 1
10000
8000
6000
4000
2000
0
0
2000
4000



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 [9]: 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 [10]: spatial_list_props = [["wel.flux",2],["drn.cond",0]]
         temporal_list_props = [["wel.flux",0],["wel.flux",1]]
```

```
In [11]: 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 [12]: sfr_obs_dict = {"hw":np.arange(1,int(m.nrow/2))}
         sfr_obs_dict["tw"] = np.arange(int(m.nrow/2),m.nrow)
         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

This will be slow because the pure python kriging...but, hey, its free!

```
In [13]: pst_helper = pyemu.helpers.PstFromFlopyModel(nam_file,new_model_ws="template",org_mode
                                                       const_props=props,spatial_list_props=spat
                                                        temporal_list_props=temporal_list_props,
                                                       grid_props=props,pp_props=props,sfr_pars=
                                                        sfr_obs=sfr_obs_dict,build_prior=False,mo
                                                       pp_space=4)
         prep_deps.prep_template(t_d=pst_helper.new_model_ws)
```

```
2019-04-29 17:32:13.999377 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
2019-04-29 17:32:14.032795 finished: loading flopy model took: 0:00:00.033418
2019-04-29 17:32:14.032913 starting: updating model attributes
2019-04-29 17:32:14.033057 finished: updating model attributes took: 0:00:00.000144
2019-04-29 17:32:14.033252 WARNING: removing existing 'new_model_ws

creating model workspace...
   template

changing model workspace...
   template
2019-04-29 17:32:15.359425 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

2019-04-29 17:32:15.480968 finished: writing new modflow input files took: 0:00:00.121543
2019-04-29 17:32:15.481638 forward_run line:pyemu.os_utils.run('mfnwt freyberg.nam 1>freyberg.
2019-04-29 17:32:15.481785 starting: setting up 'template/arr_org' dir
2019-04-29 17:32:15.482395 finished: setting up 'template/arr_org' dir took: 0:00:00.000610
2019-04-29 17:32:15.482605 starting: setting up 'template/arr_mlt' dir
2019-04-29 17:32:15.483111 finished: setting up 'template/arr_mlt' dir took: 0:00:00.000506
2019-04-29 17:32:15.483346 starting: setting up 'template/list_org' dir
2019-04-29 17:32:15.483728 finished: setting up 'template/list_org' dir took: 0:00:00.000382
2019-04-29 17:32:15.484054 starting: setting up 'template/list_mlt' dir
2019-04-29 17:32:15.484605 finished: setting up 'template/list_mlt' dir took: 0:00:00.000551
2019-04-29 17:32:15.484822 starting: processing temporal_list_props
2019-04-29 17:32:15.508214 finished: processing temporal_list_props took: 0:00:00.023392
```

```
2019-04-29 17:32:15.508602 starting: processing spatial_list_props
2019-04-29 17:32:15.587620 finished: processing spatial_list_props took: 0:00:00.079018
2019-04-29 17:32:15.642908 forward_run line:pyemu.helpers.apply_list_pars()

2019-04-29 17:32:16.020905 starting: writing grid tpl:hk3.dat_gr.tpl
2019-04-29 17:32:16.029497 finished: writing grid tpl:hk3.dat_gr.tpl took: 0:00:00.008592
2019-04-29 17:32:16.032446 starting: writing grid tpl:vka3.dat_gr.tpl
2019-04-29 17:32:16.041798 finished: writing grid tpl:vka3.dat_gr.tpl took: 0:00:00.009352
2019-04-29 17:32:16.044589 starting: writing grid tpl:ss3.dat_gr.tpl
2019-04-29 17:32:16.053663 finished: writing grid tpl:ss3.dat_gr.tpl took: 0:00:00.009074
2019-04-29 17:32:16.056398 starting: writing grid tpl:sy3.dat_gr.tpl
2019-04-29 17:32:16.065635 finished: writing grid tpl:sy3.dat_gr.tpl took: 0:00:00.009237
2019-04-29 17:32:16.068534 starting: writing grid tpl:strt3.dat_gr.tpl
2019-04-29 17:32:16.077954 finished: writing grid tpl:strt3.dat_gr.tpl took: 0:00:00.009420
2019-04-29 17:32:16.080863 starting: writing grid tpl:hk4.dat_gr.tpl
2019-04-29 17:32:16.090327 finished: writing grid tpl:hk4.dat_gr.tpl took: 0:00:00.009464
2019-04-29 17:32:16.093042 starting: writing grid tpl:vka4.dat_gr.tpl
2019-04-29 17:32:16.102365 finished: writing grid tpl:vka4.dat_gr.tpl took: 0:00:00.009323
2019-04-29 17:32:16.105271 starting: writing grid tpl:ss4.dat_gr.tpl
2019-04-29 17:32:16.114140 finished: writing grid tpl:ss4.dat_gr.tpl took: 0:00:00.008869
2019-04-29 17:32:16.116675 starting: writing grid tpl:sy4.dat_gr.tpl
2019-04-29 17:32:16.126102 finished: writing grid tpl:sy4.dat_gr.tpl took: 0:00:00.009427
2019-04-29 17:32:16.128888 starting: writing grid tpl:strt4.dat_gr.tpl
2019-04-29 17:32:16.138216 finished: writing grid tpl:strt4.dat_gr.tpl took: 0:00:00.009328
2019-04-29 17:32:16.140976 starting: writing grid tpl:hk5.dat_gr.tpl
2019-04-29 17:32:16.150103 finished: writing grid tpl:hk5.dat_gr.tpl took: 0:00:00.009127
2019-04-29 17:32:16.152934 starting: writing grid tpl:vka5.dat_gr.tpl
2019-04-29 17:32:16.162218 finished: writing grid tpl:vka5.dat_gr.tpl took: 0:00:00.009284
2019-04-29 17:32:16.164909 starting: writing grid tpl:ss5.dat_gr.tpl
2019-04-29 17:32:16.174285 finished: writing grid tpl:ss5.dat_gr.tpl took: 0:00:00.009376
2019-04-29 17:32:16.177320 starting: writing grid tpl:sy5.dat_gr.tpl
2019-04-29 17:32:16.187295 finished: writing grid tpl:sy5.dat_gr.tpl took: 0:00:00.009975
2019-04-29 17:32:16.190154 starting: writing grid tpl:strt5.dat_gr.tpl
2019-04-29 17:32:16.199501 finished: writing grid tpl:strt5.dat_gr.tpl took: 0:00:00.009347
2019-04-29 17:32:16.202474 starting: writing grid tpl:rech2.dat_gr.tpl
2019-04-29 17:32:16.211676 finished: writing grid tpl:rech2.dat_gr.tpl took: 0:00:00.009202
2019-04-29 17:32:16.214366 starting: writing grid tpl:rech3.dat_gr.tpl
2019-04-29 17:32:16.223764 finished: writing grid tpl:rech3.dat_gr.tpl took: 0:00:00.009398
2019-04-29 17:32:16.226433 starting: writing const tpl:hk6.dat_cn.tpl
2019-04-29 17:32:16.232959 finished: writing const tpl:hk6.dat_cn.tpl took: 0:00:00.006526
2019-04-29 17:32:16.235496 starting: writing const tpl:vka6.dat_cn.tpl
2019-04-29 17:32:16.241788 finished: writing const tpl:vka6.dat_cn.tpl took: 0:00:00.006292
2019-04-29 17:32:16.244912 starting: writing const tpl:ss6.dat_cn.tpl
2019-04-29 17:32:16.251282 finished: writing const tpl:ss6.dat_cn.tpl took: 0:00:00.006370
2019-04-29 17:32:16.254228 starting: writing const tpl:sy6.dat_cn.tpl
2019-04-29 17:32:16.260015 finished: writing const tpl:sy6.dat_cn.tpl took: 0:00:00.005787
2019-04-29 17:32:16.262754 starting: writing const tpl:strt6.dat_cn.tpl
2019-04-29 17:32:16.268601 finished: writing const tpl:strt6.dat_cn.tpl took: 0:00:00.005847
```

```
2019-04-29 17:32:16.271784 starting: writing const tpl:hk7.dat_cn.tpl
2019-04-29 17:32:16.278422 finished: writing const tpl:hk7.dat_cn.tpl took: 0:00:00.006638
2019-04-29 17:32:16.281132 starting: writing const tpl:vka7.dat_cn.tpl
2019-04-29 17:32:16.286985 finished: writing const tpl:vka7.dat_cn.tpl took: 0:00:00.005853
2019-04-29 17:32:16.289662 starting: writing const tpl:ss7.dat_cn.tpl
2019-04-29 17:32:16.295423 finished: writing const tpl:ss7.dat_cn.tpl took: 0:00:00.005761
2019-04-29 17:32:16.298547 starting: writing const tpl:sy7.dat_cn.tpl
2019-04-29 17:32:16.304944 finished: writing const tpl:sy7.dat_cn.tpl took: 0:00:00.006397
2019-04-29 17:32:16.307749 starting: writing const tpl:strt7.dat_cn.tpl
2019-04-29 17:32:16.313637 finished: writing const tpl:strt7.dat_cn.tpl took: 0:00:00.005888
2019-04-29 17:32:16.316407 starting: writing const tpl:hk8.dat_cn.tpl
2019-04-29 17:32:16.323187 finished: writing const tpl:hk8.dat_cn.tpl took: 0:00:00.006780
2019-04-29 17:32:16.326063 starting: writing const tpl:vka8.dat_cn.tpl
2019-04-29 17:32:16.332009 finished: writing const tpl:vka8.dat_cn.tpl took: 0:00:00.005946
2019-04-29 17:32:16.334706 starting: writing const tpl:ss8.dat_cn.tpl
2019-04-29 17:32:16.340403 finished: writing const tpl:ss8.dat_cn.tpl took: 0:00:00.005697
2019-04-29 17:32:16.343071 starting: writing const tpl:sy8.dat_cn.tpl
2019-04-29 17:32:16.348942 finished: writing const tpl:sy8.dat_cn.tpl took: 0:00:00.005871
2019-04-29 17:32:16.351653 starting: writing const tpl:strt8.dat_cn.tpl
2019-04-29 17:32:16.357672 finished: writing const tpl:strt8.dat_cn.tpl took: 0:00:00.006019
2019-04-29 17:32:16.360410 starting: writing const tpl:rech4.dat_cn.tpl
2019-04-29 17:32:16.366324 finished: writing const tpl:rech4.dat_cn.tpl took: 0:00:00.005914
2019-04-29 17:32:16.368990 starting: writing const tpl:rech5.dat_cn.tpl
2019-04-29 17:32:16.374814 finished: writing const tpl:rech5.dat_cn.tpl took: 0:00:00.005824
2019-04-29 17:32:16.395020 starting: setting up pilot point process
2019-04-29 17:32:16.395384 WARNING: pp_geostruct is None, using ExpVario with contribution=1 an
2019-04-29 17:32:16.398057 pp_dict: {0: ['hk0', 'vka0', 'ss0', 'sy0', 'strt0', 'rech0', 'rech1
2019-04-29 17:32:16.398114 starting: calling setup_pilot_point_grid()
2019-04-29 17:32:16.920656 544 pilot point parameters created
2019-04-29 17:32:16.921232 pilot point 'pargp':hk0,vka0,ss0,sy0,strt0,rech0,rech1,sy1,hk1,ss1,s
2019-04-29 17:32:16.921673 finished: calling setup_pilot_point_grid() took: 0:00:00.523559
2019-04-29 17:32:16.923990 starting: calculating factors for p=hk0, k=0
2019-04-29 17:32:16.924918 saving krige variance file:template/pp_k0_general_zn.fac
2019-04-29 17:32:16.924977 saving krige factors file:template/pp_k0_general_zn.fac
starting interp point loop for 800 points
took 2.42657 seconds
2019-04-29 17:32:19.405493 finished: calculating factors for p=hk0, k=0 took: 0:00:02.481503
2019-04-29 17:32:19.406666 starting: calculating factors for p=vka0, k=0
2019-04-29 17:32:19.407653 finished: calculating factors for p=vka0, k=0 took: 0:00:00.000987
2019-04-29 17:32:19.408236 starting: calculating factors for p=ss0, k=0
2019-04-29 17:32:19.408935 finished: calculating factors for p=ss0, k=0 took: 0:00:00.000699
2019-04-29 17:32:19.409920 starting: calculating factors for p=sy0, k=0
2019-04-29 17:32:19.411088 finished: calculating factors for p=sy0, k=0 took: 0:00:00.001168
2019-04-29 17:32:19.411655 starting: calculating factors for p=strt0, k=0
2019-04-29 17:32:19.412322 finished: calculating factors for p=strt0, k=0 took: 0:00:00.000667
2019-04-29 17:32:19.412874 starting: calculating factors for p=rech0, k=0
2019-04-29 17:32:19.413550 finished: calculating factors for p=rech0, k=0 took: 0:00:00.000676
2019-04-29 17:32:19.414915 starting: calculating factors for p=rech1, k=0
```

```
2019-04-29 17:32:19.415770 finished: calculating factors for p=rech1, k=0 took: 0:00:00.000855
2019-04-29 17:32:19.416409 starting: calculating factors for p=sy1, k=1
2019-04-29 17:32:19.417919 saving krige variance file:template/pp_k1_general_zn.fac
2019-04-29 17:32:19.418065 saving krige factors file:template/pp_k1_general_zn.fac
starting interp point loop for 800 points
took 2.376878 seconds
2019-04-29 17:32:21.848974 finished: calculating factors for p=sy1, k=1 took: 0:00:02.432565
2019-04-29 17:32:21.850256 starting: calculating factors for p=hk1, k=1
2019-04-29 17:32:21.851338 finished: calculating factors for p=hk1, k=1 took: 0:00:00.001082
2019-04-29 17:32:21.851894 starting: calculating factors for p=ss1, k=1
2019-04-29 17:32:21.852547 finished: calculating factors for p=ss1, k=1 took: 0:00:00.000653
2019-04-29 17:32:21.853085 starting: calculating factors for p=strt1, k=1
2019-04-29 17:32:21.853726 finished: calculating factors for p=strt1, k=1 took: 0:00:00.000641
2019-04-29 17:32:21.855271 starting: calculating factors for p=vka1, k=1
2019-04-29 17:32:21.856093 finished: calculating factors for p=vka1, k=1 took: 0:00:00.000822
2019-04-29 17:32:21.856682 starting: calculating factors for p=vka2, k=2
2019-04-29 17:32:21.857398 saving krige variance file:template/pp_k2_general_zn.fac
2019-04-29 17:32:21.857512 saving krige factors file:template/pp_k2_general_zn.fac
starting interp point loop for 800 points
took 2.364682 seconds
2019-04-29 17:32:24.276916 finished: calculating factors for p=vka2, k=2 took: 0:00:02.420234
2019-04-29 17:32:24.277984 starting: calculating factors for p=strt2, k=2
2019-04-29 17:32:24.278898 finished: calculating factors for p=strt2, k=2 took: 0:00:00.000914
2019-04-29 17:32:24.280449 starting: calculating factors for p=ss2, k=2
2019-04-29 17:32:24.281415 finished: calculating factors for p=ss2, k=2 took: 0:00:00.000966
2019-04-29 17:32:24.282200 starting: calculating factors for p=sy2, k=2
2019-04-29 17:32:24.283040 finished: calculating factors for p=sy2, k=2 took: 0:00:00.000840
2019-04-29 17:32:24.283878 starting: calculating factors for p=hk2, k=2
2019-04-29 17:32:24.285090 finished: calculating factors for p=hk2, k=2 took: 0:00:00.001212
2019-04-29 17:32:24.285495 starting: processing pp_prefix:vka2
2019-04-29 17:32:24.298986 starting: processing pp_prefix:strt1
2019-04-29 17:32:24.307435 starting: processing pp_prefix:ss2
2019-04-29 17:32:24.315563 starting: processing pp_prefix:hk0
2019-04-29 17:32:24.323462 starting: processing pp_prefix:vka0
2019-04-29 17:32:24.331263 starting: processing pp_prefix:strt2
2019-04-29 17:32:24.338926 starting: processing pp_prefix:rech1
2019-04-29 17:32:24.348388 starting: processing pp_prefix:vka1
2019-04-29 17:32:24.356590 starting: processing pp_prefix:rech0
2019-04-29 17:32:24.364409 starting: processing pp_prefix:strt0
2019-04-29 17:32:24.375412 starting: processing pp_prefix:sy0
2019-04-29 17:32:24.383840 starting: processing pp_prefix:ss1
2019-04-29 17:32:24.392501 starting: processing pp_prefix:ss0
2019-04-29 17:32:24.400856 starting: processing pp_prefix:sy2
2019-04-29 17:32:24.409389 starting: processing pp_prefix:hk1
2019-04-29 17:32:24.417550 starting: processing pp_prefix:hk2
2019-04-29 17:32:24.426433 starting: processing pp_prefix:sy1
2019-04-29 17:32:24.515423 finished: setting up pilot point process took: 0:00:08.120403
2019-04-29 17:32:24.515607 starting: setting up grid process
```

```
2019-04-29 17:32:24.515676 WARNING: grid_geostruct is None, using ExpVario with contribution=1
2019-04-29 17:32:24.515791 finished: setting up grid process took: 0:00:00.000184
2019-04-29 17:32:24.519466 starting: save test mlt array arr_mlt/hk0.dat_pp
2019-04-29 17:32:24.521658 finished: save test mlt array arr_mlt/hk0.dat_pp took: 0:00:00.00219
2019-04-29 17:32:24.522469 starting: save test mlt array arr_mlt/vka0.dat_pp
2019-04-29 17:32:24.524289 finished: save test mlt array arr_mlt/vka0.dat_pp took: 0:00:00.0018
2019-04-29 17:32:24.525257 starting: save test mlt array arr_mlt/ss0.dat_pp
2019-04-29 17:32:24.532418 finished: save test mlt array arr_mlt/ss0.dat_pp took: 0:00:00.00716
2019-04-29 17:32:24.533480 starting: save test mlt array arr_mlt/sy0.dat_pp
2019-04-29 17:32:24.535586 finished: save test mlt array arr_mlt/sy0.dat_pp took: 0:00:00.00210
2019-04-29 17:32:24.536574 starting: save test mlt array arr_mlt/strt0.dat_pp
2019-04-29 17:32:24.538755 finished: save test mlt array arr_mlt/strt0.dat_pp took: 0:00:00.002
2019-04-29 17:32:24.539774 starting: save test mlt array arr_mlt/hk1.dat_pp
2019-04-29 17:32:24.542238 finished: save test mlt array arr_mlt/hk1.dat_pp took: 0:00:00.00246
2019-04-29 17:32:24.543257 starting: save test mlt array arr_mlt/vka1.dat_pp
2019-04-29 17:32:24.545827 finished: save test mlt array arr_mlt/vka1.dat_pp took: 0:00:00.0025
2019-04-29 17:32:24.546573 starting: save test mlt array arr_mlt/ss1.dat_pp
2019-04-29 17:32:24.548928 finished: save test mlt array arr_mlt/ss1.dat_pp took: 0:00:00.00235
2019-04-29 17:32:24.549790 starting: save test mlt array arr_mlt/sy1.dat_pp
2019-04-29 17:32:24.551808 finished: save test mlt array arr_mlt/sy1.dat_pp took: 0:00:00.00201
2019-04-29 17:32:24.552515 starting: save test mlt array arr_mlt/strt1.dat_pp
2019-04-29 17:32:24.554719 finished: save test mlt array arr_mlt/strt1.dat_pp took: 0:00:00.002
2019-04-29 17:32:24.555634 starting: save test mlt array arr_mlt/hk2.dat_pp
2019-04-29 17:32:24.557666 finished: save test mlt array arr_mlt/hk2.dat_pp took: 0:00:00.00203
2019-04-29 17:32:24.558553 starting: save test mlt array arr_mlt/vka2.dat_pp
2019-04-29 17:32:24.560558 finished: save test mlt array arr_mlt/vka2.dat_pp took: 0:00:00.0020
2019-04-29 17:32:24.561397 starting: save test mlt array arr_mlt/ss2.dat_pp
2019-04-29 17:32:24.563370 finished: save test mlt array arr_mlt/ss2.dat_pp took: 0:00:00.00197
2019-04-29 17:32:24.564142 starting: save test mlt array arr_mlt/sy2.dat_pp
2019-04-29 17:32:24.566336 finished: save test mlt array arr_mlt/sy2.dat_pp took: 0:00:00.00219
2019-04-29 17:32:24.567313 starting: save test mlt array arr_mlt/strt2.dat_pp
2019-04-29 17:32:24.569456 finished: save test mlt array arr_mlt/strt2.dat_pp took: 0:00:00.002
2019-04-29 17:32:24.570319 starting: save test mlt array arr_mlt/rech0.dat_pp
2019-04-29 17:32:24.572343 finished: save test mlt array arr_mlt/rech0.dat_pp took: 0:00:00.002
2019-04-29 17:32:24.573209 starting: save test mlt array arr_mlt/rech1.dat_pp
2019-04-29 17:32:24.575237 finished: save test mlt array arr_mlt/rech1.dat_pp took: 0:00:00.002
2019-04-29 17:32:24.576117 starting: save test mlt array arr_mlt/hk3.dat_gr
2019-04-29 17:32:24.578436 finished: save test mlt array arr_mlt/hk3.dat_gr took: 0:00:00.00231
2019-04-29 17:32:24.580071 starting: save test mlt array arr_mlt/vka3.dat_gr
2019-04-29 17:32:24.582665 finished: save test mlt array arr_mlt/vka3.dat_gr took: 0:00:00.0025
2019-04-29 17:32:24.583702 starting: save test mlt array arr_mlt/ss3.dat_gr
2019-04-29 17:32:24.586672 finished: save test mlt array arr_mlt/ss3.dat_gr took: 0:00:00.00297
2019-04-29 17:32:24.587983 starting: save test mlt array arr_mlt/sy3.dat_gr
2019-04-29 17:32:24.591167 finished: save test mlt array arr_mlt/sy3.dat_gr took: 0:00:00.00318
2019-04-29 17:32:24.592505 starting: save test mlt array arr_mlt/strt3.dat_gr
2019-04-29 17:32:24.595183 finished: save test mlt array arr_mlt/strt3.dat_gr took: 0:00:00.002
2019-04-29 17:32:24.596754 starting: save test mlt array arr_mlt/hk4.dat_gr
2019-04-29 17:32:24.599808 finished: save test mlt array arr_mlt/hk4.dat_gr took: 0:00:00.00305
```

```
2019-04-29 17:32:24.601282 starting: save test mlt array arr_mlt/vka4.dat_gr
2019-04-29 17:32:24.604920 finished: save test mlt array arr_mlt/vka4.dat_gr took: 0:00:00.0036
2019-04-29 17:32:24.606198 starting: save test mlt array arr_mlt/ss4.dat_gr
2019-04-29 17:32:24.609611 finished: save test mlt array arr_mlt/ss4.dat_gr took: 0:00:00.00341
2019-04-29 17:32:24.611252 starting: save test mlt array arr_mlt/sy4.dat_gr
2019-04-29 17:32:24.614670 finished: save test mlt array arr_mlt/sy4.dat_gr took: 0:00:00.00341
2019-04-29 17:32:24.616143 starting: save test mlt array arr_mlt/strt4.dat_gr
2019-04-29 17:32:24.619585 finished: save test mlt array arr_mlt/strt4.dat_gr took: 0:00:00.003
2019-04-29 17:32:24.620715 starting: save test mlt array arr_mlt/hk5.dat_gr
2019-04-29 17:32:24.623819 finished: save test mlt array arr_mlt/hk5.dat_gr took: 0:00:00.00310
2019-04-29 17:32:24.625297 starting: save test mlt array arr_mlt/vka5.dat_gr
2019-04-29 17:32:24.628914 finished: save test mlt array arr_mlt/vka5.dat_gr took: 0:00:00.0036
2019-04-29 17:32:24.630716 starting: save test mlt array arr_mlt/ss5.dat_gr
2019-04-29 17:32:24.634246 finished: save test mlt array arr_mlt/ss5.dat_gr took: 0:00:00.00353
2019-04-29 17:32:24.635880 starting: save test mlt array arr_mlt/sy5.dat_gr
2019-04-29 17:32:24.639430 finished: save test mlt array arr_mlt/sy5.dat_gr took: 0:00:00.00355
2019-04-29 17:32:24.640855 starting: save test mlt array arr_mlt/strt5.dat_gr
2019-04-29 17:32:24.644033 finished: save test mlt array arr_mlt/strt5.dat_gr took: 0:00:00.003
2019-04-29 17:32:24.645357 starting: save test mlt array arr_mlt/rech2.dat_gr
2019-04-29 17:32:24.648917 finished: save test mlt array arr_mlt/rech2.dat_gr took: 0:00:00.003
2019-04-29 17:32:24.650409 starting: save test mlt array arr_mlt/rech3.dat_gr
2019-04-29 17:32:24.654239 finished: save test mlt array arr_mlt/rech3.dat_gr took: 0:00:00.003
2019-04-29 17:32:24.655604 starting: save test mlt array arr_mlt/hk6.dat_cn
2019-04-29 17:32:24.659147 finished: save test mlt array arr_mlt/hk6.dat_cn took: 0:00:00.00354
2019-04-29 17:32:24.660644 starting: save test mlt array arr_mlt/vka6.dat_cn
2019-04-29 17:32:24.663721 finished: save test mlt array arr_mlt/vka6.dat_cn took: 0:00:00.0030
2019-04-29 17:32:24.665103 starting: save test mlt array arr_mlt/ss6.dat_cn
2019-04-29 17:32:24.668952 finished: save test mlt array arr_mlt/ss6.dat_cn took: 0:00:00.00384
2019-04-29 17:32:24.670365 starting: save test mlt array arr_mlt/sy6.dat_cn
2019-04-29 17:32:24.673791 finished: save test mlt array arr_mlt/sy6.dat_cn took: 0:00:00.00342
2019-04-29 17:32:24.675246 starting: save test mlt array arr_mlt/strt6.dat_cn
2019-04-29 17:32:24.679991 finished: save test mlt array arr_mlt/strt6.dat_cn took: 0:00:00.004
2019-04-29 17:32:24.681430 starting: save test mlt array arr_mlt/hk7.dat_cn
2019-04-29 17:32:24.684439 finished: save test mlt array arr_mlt/hk7.dat_cn took: 0:00:00.00300
2019-04-29 17:32:24.685757 starting: save test mlt array arr_mlt/vka7.dat_cn
2019-04-29 17:32:24.689033 finished: save test mlt array arr_mlt/vka7.dat_cn took: 0:00:00.0032
2019-04-29 17:32:24.690292 starting: save test mlt array arr_mlt/ss7.dat_cn
2019-04-29 17:32:24.693843 finished: save test mlt array arr_mlt/ss7.dat_cn took: 0:00:00.00355
2019-04-29 17:32:24.695231 starting: save test mlt array arr_mlt/sy7.dat_cn
2019-04-29 17:32:24.698437 finished: save test mlt array arr_mlt/sy7.dat_cn took: 0:00:00.00320
2019-04-29 17:32:24.699836 starting: save test mlt array arr_mlt/strt7.dat_cn
2019-04-29 17:32:24.705158 finished: save test mlt array arr_mlt/strt7.dat_cn took: 0:00:00.005
2019-04-29 17:32:24.706790 starting: save test mlt array arr_mlt/hk8.dat_cn
2019-04-29 17:32:24.709903 finished: save test mlt array arr_mlt/hk8.dat_cn took: 0:00:00.00311
2019-04-29 17:32:24.711231 starting: save test mlt array arr_mlt/vka8.dat_cn
2019-04-29 17:32:24.716023 finished: save test mlt array arr_mlt/vka8.dat_cn took: 0:00:00.0047
2019-04-29 17:32:24.717599 starting: save test mlt array arr_mlt/ss8.dat_cn
2019-04-29 17:32:24.721262 finished: save test mlt array arr_mlt/ss8.dat_cn took: 0:00:00.00366
```

```
2019-04-29 17:32:24.722571 starting: save test mlt array arr_mlt/sy8.dat_cn
2019-04-29 17:32:24.726435 finished: save test mlt array arr_mlt/sy8.dat_cn took: 0:00:00.0038
2019-04-29 17:32:24.727955 starting: save test mlt array arr_mlt/strt8.dat_cn
2019-04-29 17:32:24.731228 finished: save test mlt array arr_mlt/strt8.dat_cn took: 0:00:00.003
2019-04-29 17:32:24.732712 starting: save test mlt array arr_mlt/rech4.dat_cn
2019-04-29 17:32:24.737770 finished: save test mlt array arr_mlt/rech4.dat_cn took: 0:00:00.005
2019-04-29 17:32:24.739222 starting: save test mlt array arr_mlt/rech5.dat_cn
2019-04-29 17:32:24.742650 finished: save test mlt array arr_mlt/rech5.dat_cn took: 0:00:00.003
2019-04-29 17:32:25.297795 forward_run line:pyemu.helpers.apply_array_pars()

all zeros for runoff...skipping...
all zeros for hcond1...skipping...
all zeros for pptsw...skipping...
2019-04-29 17:32:25.432644 starting: processing obs type mflist water budget obs
2019-04-29 17:32:25.525892 forward_run line:pyemu.gw_utils.apply_mflist_budget_obs('freyberg.li
2019-04-29 17:32:25.526071 finished: processing obs type mflist water budget obs took: 0:00:00
2019-04-29 17:32:25.526665 starting: processing obs type hyd file
2019-04-29 17:32:25.526845 finished: processing obs type hyd file took: 0:00:00.000180
2019-04-29 17:32:25.526921 starting: processing obs type external obs-sim smp files
2019-04-29 17:32:25.528113 finished: processing obs type external obs-sim smp files took: 0:00
2019-04-29 17:32:25.528455 starting: processing obs type hob
2019-04-29 17:32:25.528797 finished: processing obs type hob took: 0:00:00.000342
2019-04-29 17:32:25.528911 starting: processing obs type hds
[[0, 0], [0, 1], [0, 2], [1, 0], [1, 1], [1, 2]]
2019-04-29 17:32:25.940607 finished: processing obs type hds took: 0:00:00.411696
2019-04-29 17:32:25.941348 starting: processing obs type sfr
writing 'sfr_obs.config' to template/sfr_obs.config
2019-04-29 17:32:26.256657 finished: processing obs type sfr took: 0:00:00.315309
2019-04-29 17:32:26.257303 changing dir in to template
2019-04-29 17:32:26.258326 starting: instantiating control file from i/o files
2019-04-29 17:32:26.258575 tpl files: drn.csv.tpl,wel.csv.tpl,hk3.dat_gr.tpl,vka3.dat_gr.tpl,ss
2019-04-29 17:32:26.258707 ins files: freyberg.hds.dat.ins,vol.dat.ins,freyberg.sfr.out.process
2019-04-29 17:32:26.586204 finished: instantiating control file from i/o files took: 0:00:00.32
2019-04-29 17:32:26.804965 starting: writing forward_run.py
2019-04-29 17:32:26.806516 finished: writing forward_run.py took: 0:00:00.001551
2019-04-29 17:32:26.806988 writing pst template/freyberg.pst
2019-04-29 17:32:28.245557 starting: running pestchek on freyberg.pst
2019-04-29 17:32:28.893877 pestcheck:PESTCHEK Version 13.0. Watermark Numerical Computing.
2019-04-29 17:32:28.894319 pestcheck:
2019-04-29 17:32:28.894380 pestcheck:Errors ----->
2019-04-29 17:32:28.894419 pestcheck:No errors encountered.
2019-04-29 17:32:28.894557 pestcheck:
2019-04-29 17:32:28.894612 pestcheck:Warnings ----->
2019-04-29 17:32:28.894651 pestcheck:NUMLAM is supplied as negative. This will be reset to posi
2019-04-29 17:32:28.894969 pestcheck:PEST or BEOPEST is used PARLAM will automatically be set t
2019-04-29 17:32:28.895028 pestcheck:NOPTMAX provided as zero. No optimisation iterations will
2019-04-29 17:32:28.895066 pestcheck:objective function and residuals will be recorded for init
2019-04-29 17:32:28.895289 pestcheck:estimates only.
```

```
2019-04-29 17:32:28.895342 pestcheck:MAXSING in the singular value decomposition section is gre
2019-04-29 17:32:28.895380 pestcheck:number of adjustable parameters.
2019-04-29 17:32:28.895636 finished: running pestchek on freyberg.pst took: 0:00:00.650079
2019-04-29 17:32:28.895702 starting: saving intermediate _setup_<> dfs into template
2019-04-29 17:32:28.988014 finished: saving intermediate _setup_<> dfs into template took: 0:00
2019-04-29 17:32:28.988181 all done
```

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

```
In [14]: pst = pst_helper.pst
         pst.npar,pst.nobs
```

```
Out[14]: (12605, 4434)
```

Oh snap!

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

```
In [15]: 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 [16]: # table can also be written to a .tex file
         pst.write_par_summary_table(filename="none").sort_index()
```

```
Out[16]:                    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     32                  0
         pp_hk1            pp_hk1       log     32                  0
         pp_hk2            pp_hk2       log     32                  0
         pp_rech0        pp_rech0       log     32                  0
         pp_rech1        pp_rech1       log     32                  0
         pp_ss0            pp_ss0       log     32                  0
         pp_ss1            pp_ss1       log     32                  0
         pp_ss2            pp_ss2       log     32                  0
         pp_strt0        pp_strt0       log     32                  0
         pp_strt1        pp_strt1       log     32                  0
         pp_strt2        pp_strt2       log     32                  0
         pp_sy0            pp_sy0       log     32                  0
         pp_sy1            pp_sy1       log     32                  0
         pp_sy2            pp_sy2       log     32                  0
         pp_vka0          pp_vka0       log     32                  0
         pp_vka1          pp_vka1       log     32                  0
         pp_vka2          pp_vka2       log     32                  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 [17]: pst.write_obs_summary_table(filename="none")
```

```
Out[17]:                    group                          value  non-zero weight  \
         flaqx              flaqx            -977.239 to 32.171               84
         flout              flout              10069 to 226396               84
         flx_constan  flx_constan                            0                2
         flx_drains    flx_drains          -723.325 to -723.028               2
         flx_in-out    flx_in-out          0.012695 to 0.046143               2
         flx_percent  flx_percent                            0                2
         flx_recharg  flx_recharg                       3045.6                2
         flx_storage  flx_storage              5.7734 to 8.01049               2
         flx_stream_  flx_stream_             -1430.27 to -1428.3              2
         flx_total      flx_total       0.0126953 to 0.0461426               2
         flx_wells      flx_wells                         -900                2
         hds                  hds            32.5065 to 39.6612             4230
         vol_constan  vol_constan                            0                2
         vol_drains    vol_drains  -2.90404E+06 to -2.64014E+06               2
         vol_in-out    vol_in-out                     45 to    63             2
         vol_percent  vol_percent                            0                2
         vol_recharg  vol_recharg    1.11164E+07 to 1.22281E+07               2
         vol_storage  vol_storage            29238.3 to 31345.6               2
         vol_stream_  vol_stream_  -5.74182E+06 to -5.22049E+06               2
         vol_total      vol_total                     45 to    63             2
         vol_wells      vol_wells     -3.6135E+06 to -3.285E+06               2

                      zero weight weight standard deviation              percent error
         flaqx                  0      1                  1          0.102329 to 833.333
         flout                  0      1                  1    0.000441704 to 0.00993147
```

```
flx_constan          0      1                  1                          NA
flx_drains           0      1                  1        0.13825 to 0.138307
flx_in-out           0      1                  1          2167.18 to 7877.12
flx_percent          0      1                  1                          NA
flx_recharg          0      1                  1                   0.0328343
flx_storage          0      1                  1          12.4836 to 17.3208
flx_stream_          0      1                  1      0.0699167 to 0.0700133
flx_total            0      1                  1           2167.2 to 7876.92
flx_wells            0      1                  1                    0.111111
hds                  0      1                  1          2.52136 to 3.07631
vol_constan          0      1                  1                          NA
vol_drains           0      1                  1  3.44348E-05 to 3.78768E-05
vol_in-out           0      1                  1           1.5873 to 2.22222
vol_percent          0      1                  1                          NA
vol_recharg          0      1                  1   8.1779E-06 to 8.99569E-06
vol_storage          0      1                  1    0.00319024 to 0.00342017
vol_stream_          0      1                  1  1.74161E-05 to 1.91553E-05
vol_total            0      1                  1           1.5873 to 2.22222
vol_wells            0      1                  1   2.7674E-05 to 3.04414E-05
```

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

```
In [18]: 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 [19]: if pst_helper.pst.npar < 15000:
             cov = pst_helper.build_prior(fmt="binary",filename=os.path.join(pst_helper.new_moc
             cov = np.ma.masked_where(cov.x==0,cov.x)
             fig = plt.figure(figsize=(10,10))
             ax = plt.subplot(111)
             ax.imshow(cov)
```

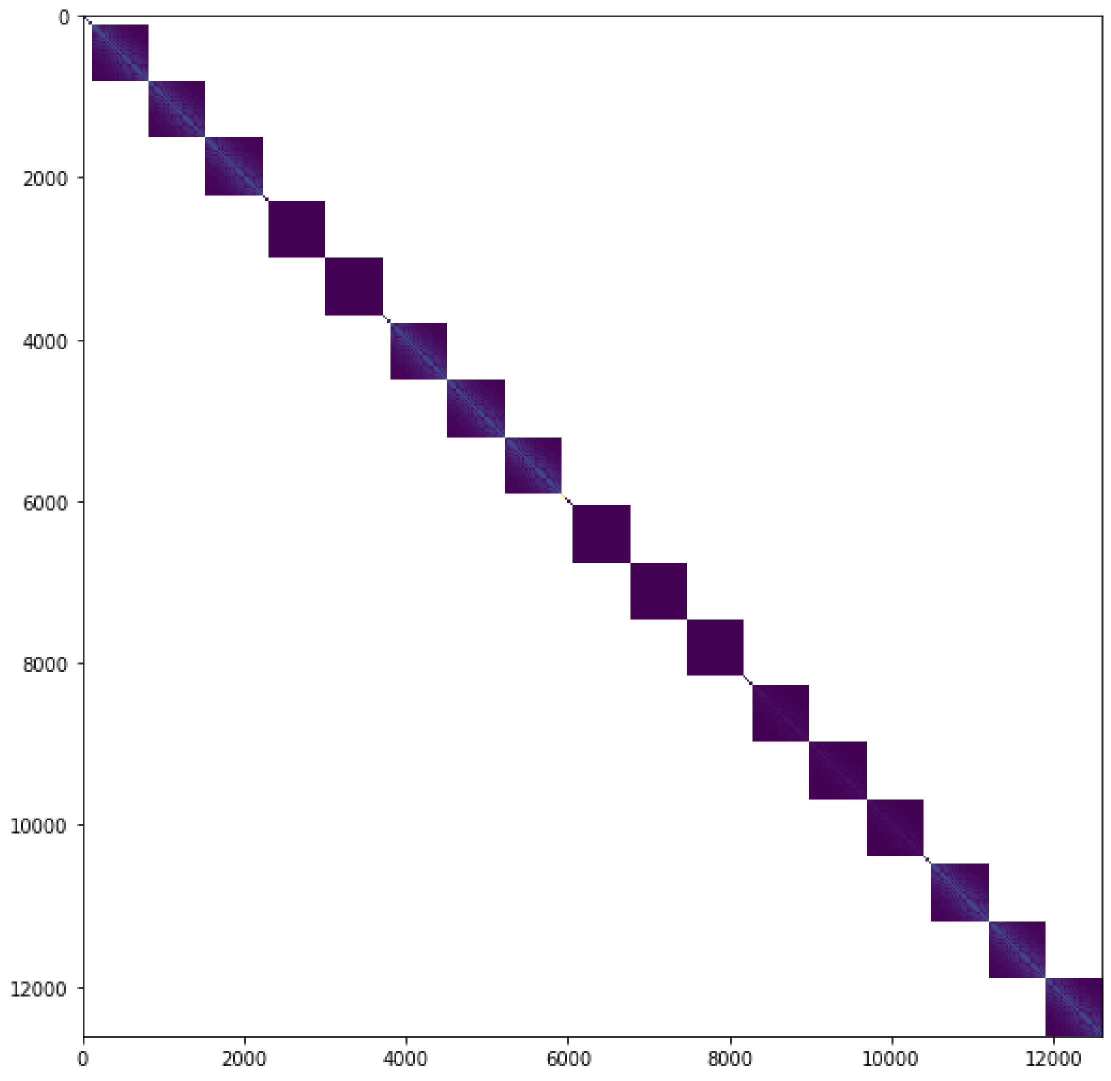
```
2019-04-29 17:32:37.712536 starting: building prior covariance matrix
2019-04-29 17:32:37.806209 WARNING: geospatial prior not implemented for SFR pars


/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


2019-04-29 17:32:42.678735 saving prior covariance matrix to file template/prior_cov.jcb
2019-04-29 17:32:45.476041 finished: building prior covariance matrix took: 0:00:07.763505
```



```
In [20]: pe = pst_helper.draw(200)
```

```
2019-04-29 17:32:56.794390 starting: drawing realizations
building diagonal cov
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:1000.0,anisotropy:1.0,bearing:0.0

working on pargroups ['pp_hk0']
build cov matrix
done
getting diag var cov 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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.htm
  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.htm
  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
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
```

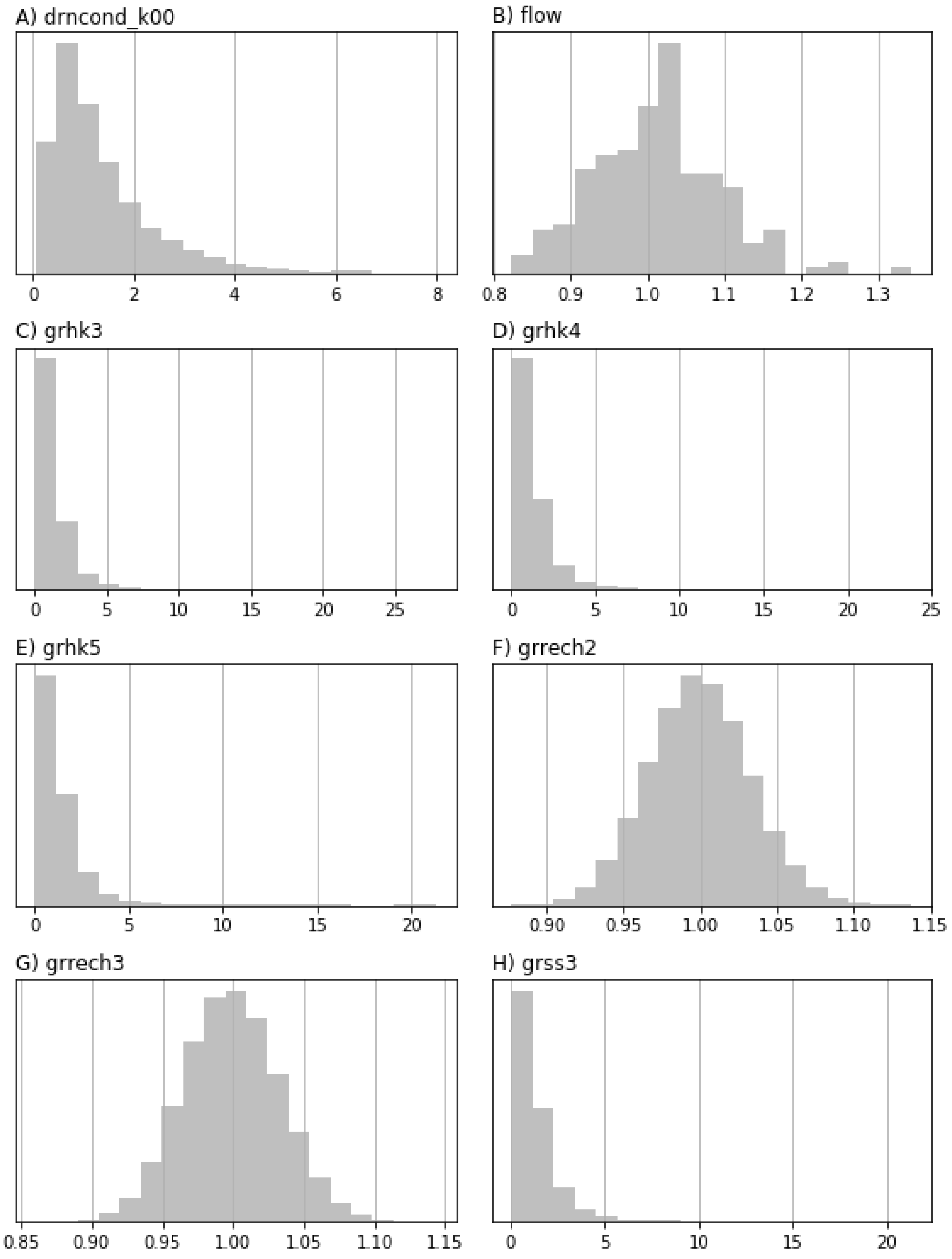
```
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grsy4']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grstrt4']
build cov matrix
done
getting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

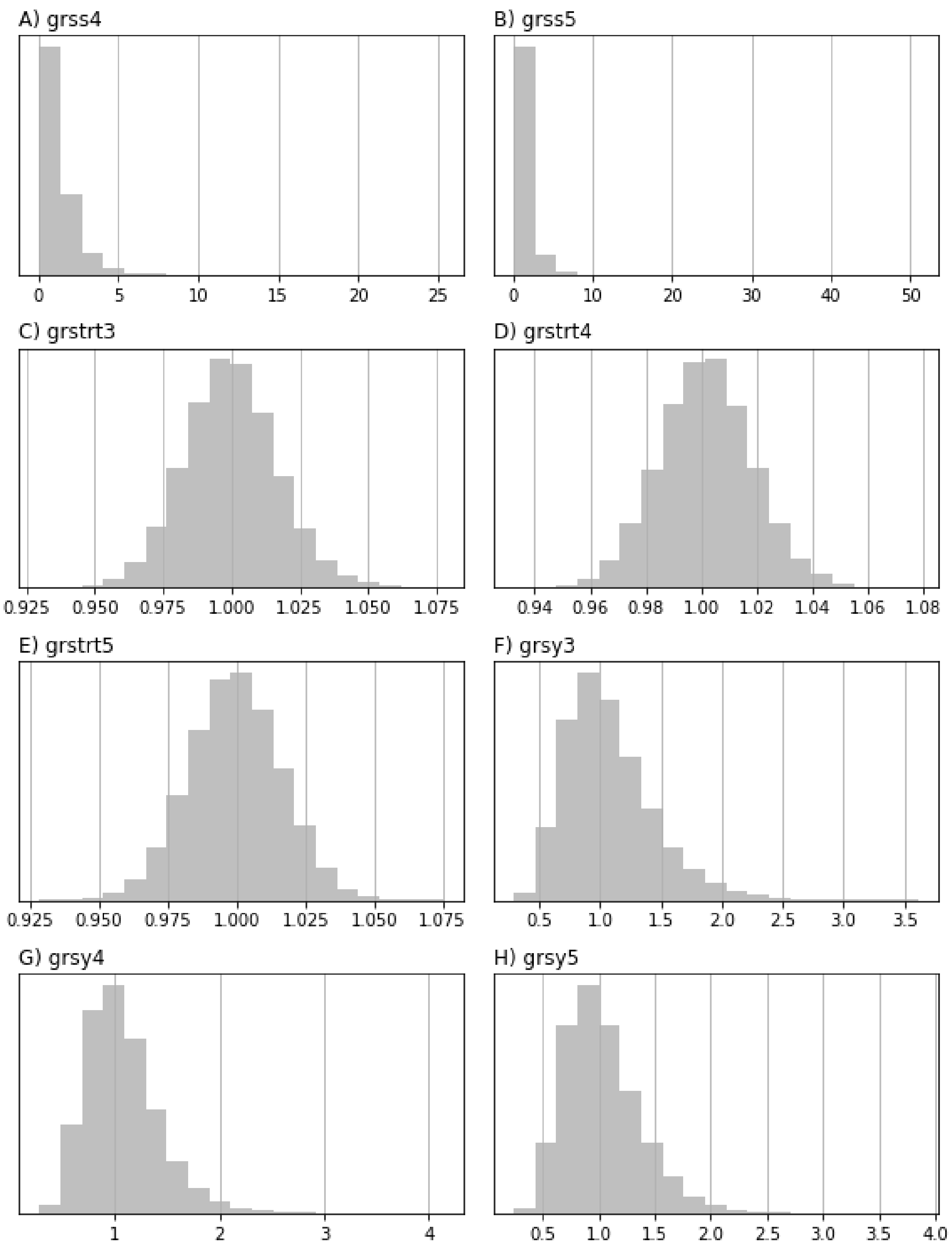
```
scaling full cov by diag var cov
making full cov draws with home-grown goodness
working on pargroups ['grrech3']
build cov matrix
done
getting diag var cov 705
scaling full cov by diag var cov
making full cov draws with home-grown goodness
adding remaining parameters to diagonal
2019-04-29 17:33:02.719051 finished: drawing realizations took: 0:00:05.924661
```

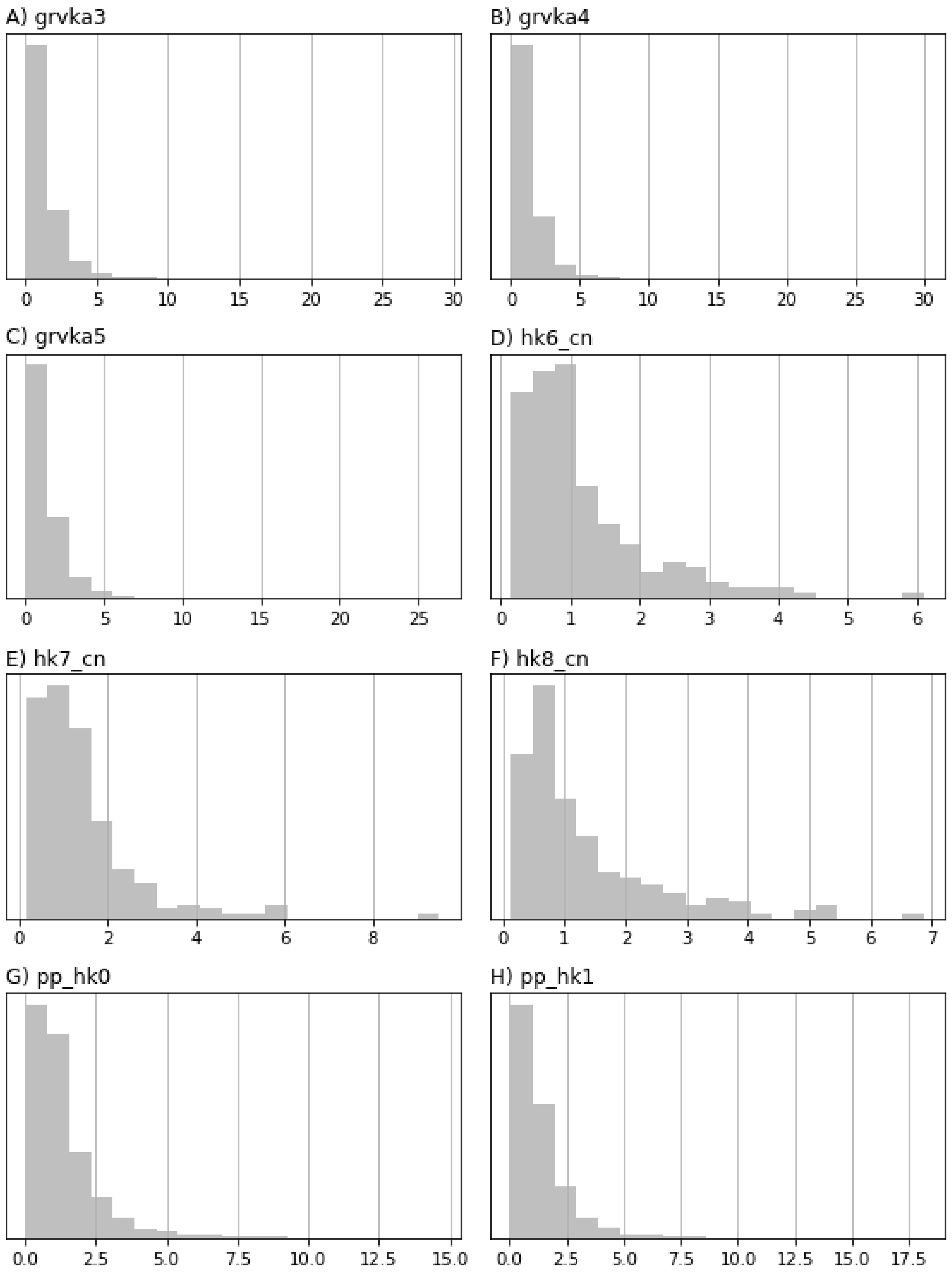
You can see that parameters are treated in parameter group (pargp) blocks for this ensemble generation. Let's plot one parameter:

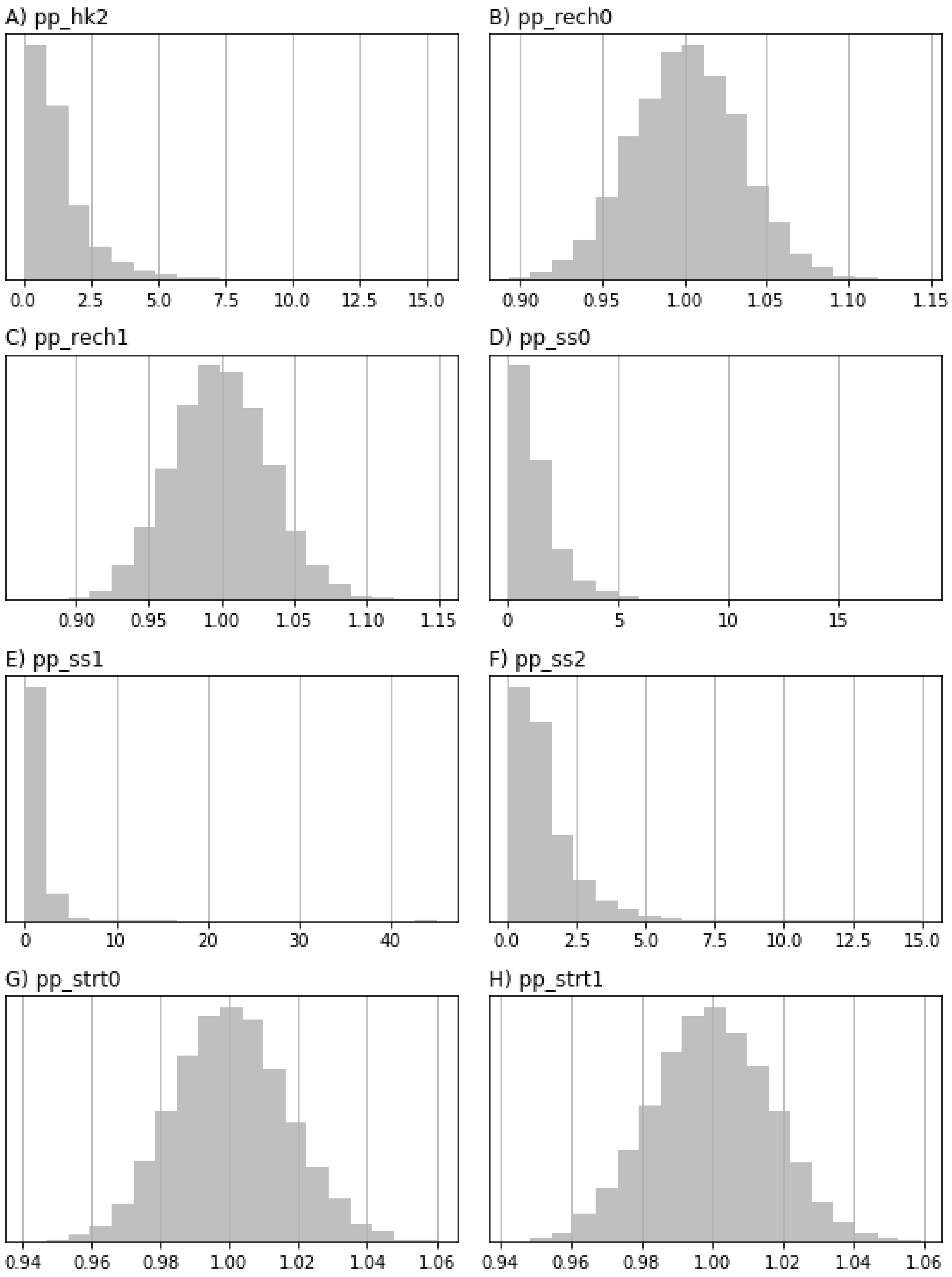
```
In [21]: par = pst_helper.pst.parameter_data
         pyemu.plot_utils.ensemble_helper(pe,plot_cols=par.groupby("pargp").groups,bins=20)
```

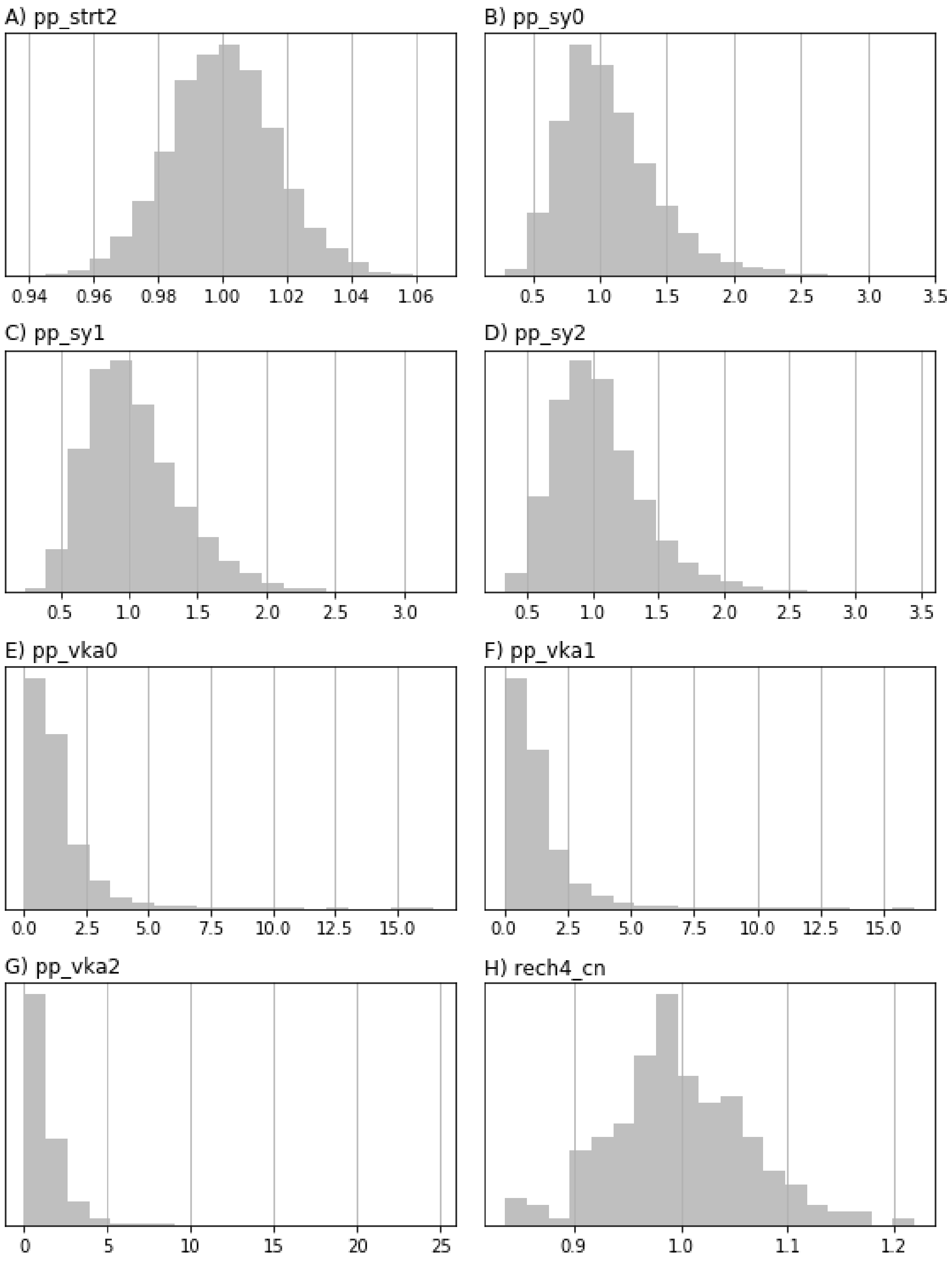
```
<Figure size 576x756 with 0 Axes>
```

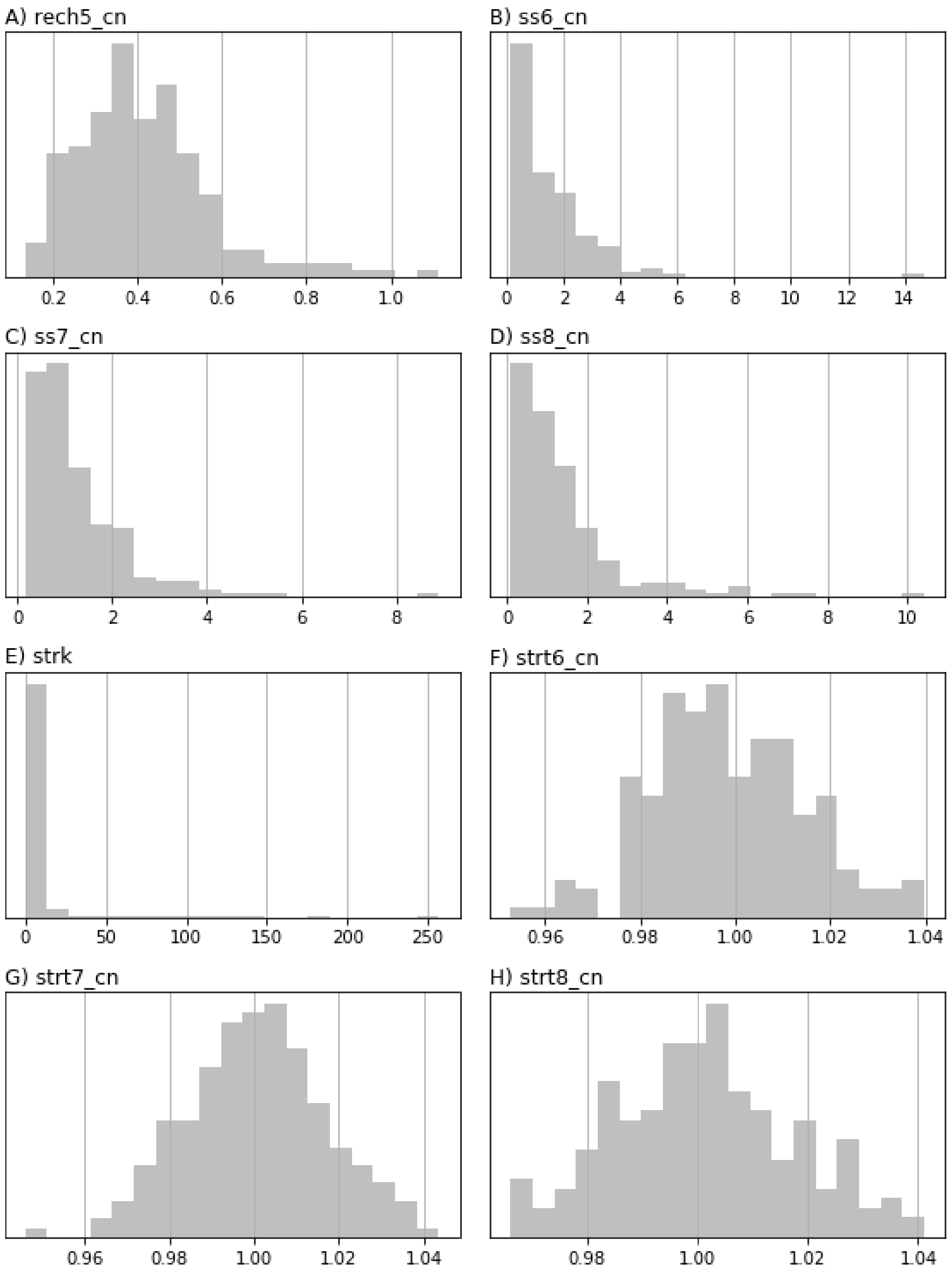

A) drncond_k00
B) flow
C) grhk3
D) grhk4
E) grhk5
F) grrech2
G) grrech3
H) grss3
0
2
4
6
8
0.8
0.9
1.0
1.1
1.2
1.3
0
5
10
15
20
25
0.90
0.95
1.00
1.05
1.10
1.15
0.85

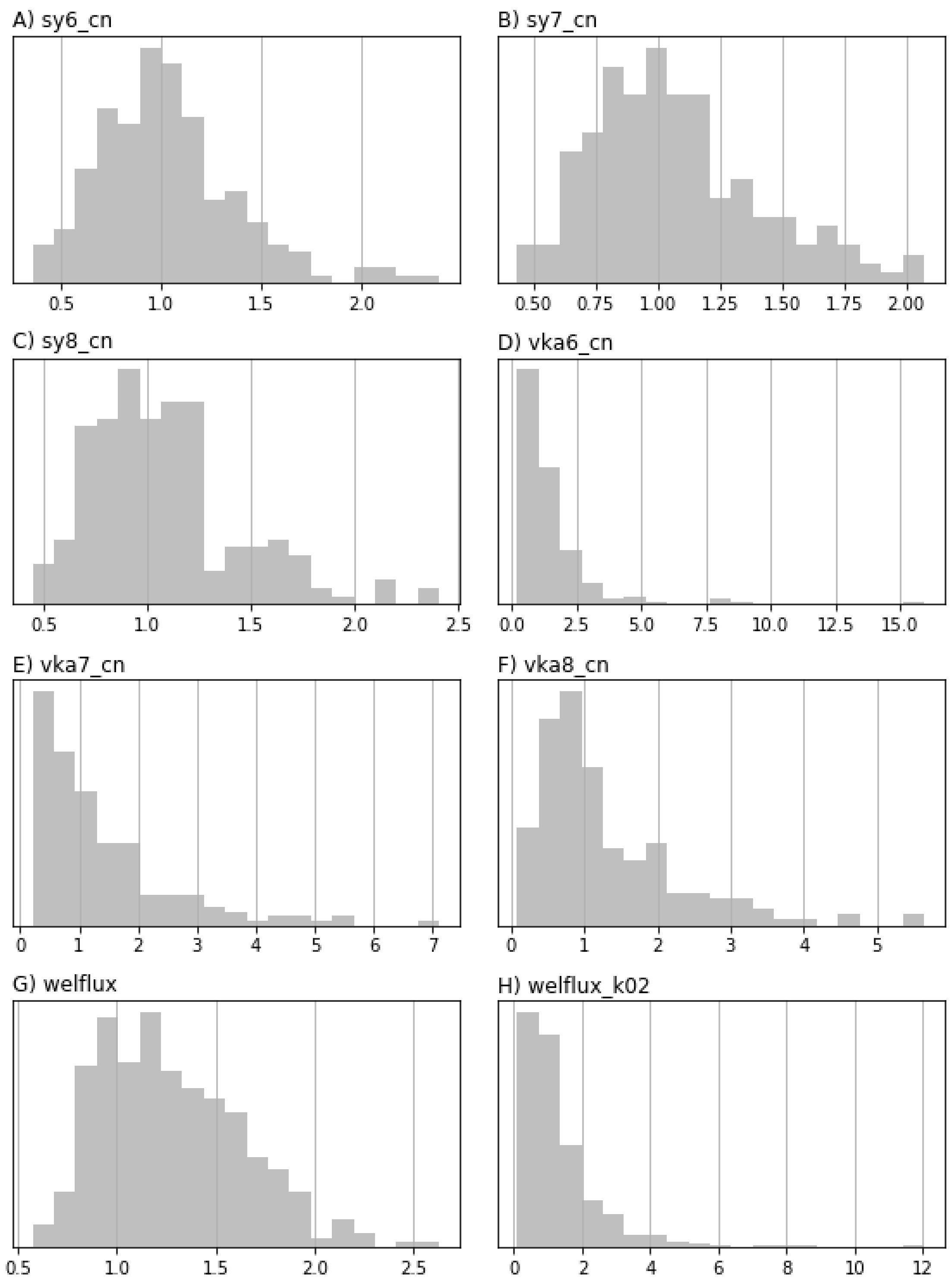

A) grss4
0 5 10 15 20 25
B) grss5
0 10 20 30 40 50
C) grstrt3
0.925 0.950 0.975 1.000 1.025 1.050 1.075
D) grstrt4
0.94 0.96 0.98 1.00 1.02 1.04 1.06 1.08
E) grstrt5
0.925 0.950 0.975 1.000 1.025 1.050 1.075
F) grsy3
0.5 1.0 1.5 2.0 2.5 3.0 3.5
G) grsy4
1 2 3 4
H) grsy5
0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0


A) grvka3
0 5 10 15 20 25 30
B) grvka4
0 5 10 15 20 25 30
C) grvka5
0 5 10 15 20 25
D) hk6_cn
0 1 2 3 4 5 6
E) hk7_cn
0 2 4 6 8
F) hk8_cn
0 1 2 3 4 5 6 7
G) pp_hk0
0.0 2.5 5.0 7.5 10.0 12.5 15.0
H) pp_hk1
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5


A) pp_hk2
0.0
2.5
5.0
7.5
10.0
12.5
15.0
B) pp_rech0
0.90
0.95
1.00
1.05
1.10
1.15
C) pp_rech1
0.90
0.95
1.00
1.05
1.10
1.15
D) pp_ss0
0
5
10
15
E) pp_ss1
0
10
20
30
40
F) pp_ss2
0.0
2.5
5.0
7.5
10.0
12.5
15.0
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
D) pp_sy2
0.5 1.0 1.5 2.0 2.5 3.0 3.5
E) pp_vka0
0.0 2.5 5.0 7.5 10.0 12.5 15.0
F) pp_vka1
0.0 2.5 5.0 7.5 10.0 12.5 15.0
G) pp_vka2
0 5 10 15 20 25
H) rech4_cn
0.9 1.0 1.1 1.2


A) rech5_cn
0.2
0.4
0.6
0.8
1.0
B) ss6_cn
0
2
4
6
8
10
12
14
C) ss7_cn
0
2
4
6
8
D) ss8_cn
0
2
4
6
8
10
E) strk
0
50
100
150
200
250
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
0.98
1.00
1.02
1.04



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

```
In [22]: 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` in the last row in 19791230)

```
In [23]: obs_locs = pd.read_csv(os.path.join("..","base_model_files","obs_loc.csv"))
         if pst_helper.m.nrow != 40:
             obs_locs.loc[:,"row"] = (obs_locs.row * redis_fac) + int(redis_fac / 2.0)
             obs_locs.loc[:,"col"] = (obs_locs.col * redis_fac) + int(redis_fac / 2.0)
         #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[23]:     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 [24]: obs = pst.observation_data
         obs.loc[:,"weight"] = 0.0
         obs.loc[obs_locs.obsnme,"weight"] = 1.0
         obs.loc[obs_locs.obsnme,"obgnme"] = "calhead"
         fo_obs = "fo_{0}_19791230".format(pst_helper.m.nrow-1)
         obs.loc[fo_obs,"weight"] = 1.0
         obs.loc[fo_obs,"obgnme"] = "calflux"
         pst.nnz_obs_names
```

```
Out[24]: ['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 [25]: swgw_forecasts = obs.loc[obs.obsnme.apply(lambda x: "fa" in x and ("hw" in x or "tw" i
         print(swgw_forecasts)
         hds_fore_name = "hds_00_{0:03d}_{1:03d}".format(int(pst_helper.m.nrow/3),int(pst_help
         hds_forecasts = obs.loc[obs.obsnme.apply(lambda x: hds_fore_name in x),"obsnme"].tolis
         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"))
```

```
['fa_hw_19791230', 'fa_hw_19801229', 'fa_tw_19791230', 'fa_tw_19801229']
```

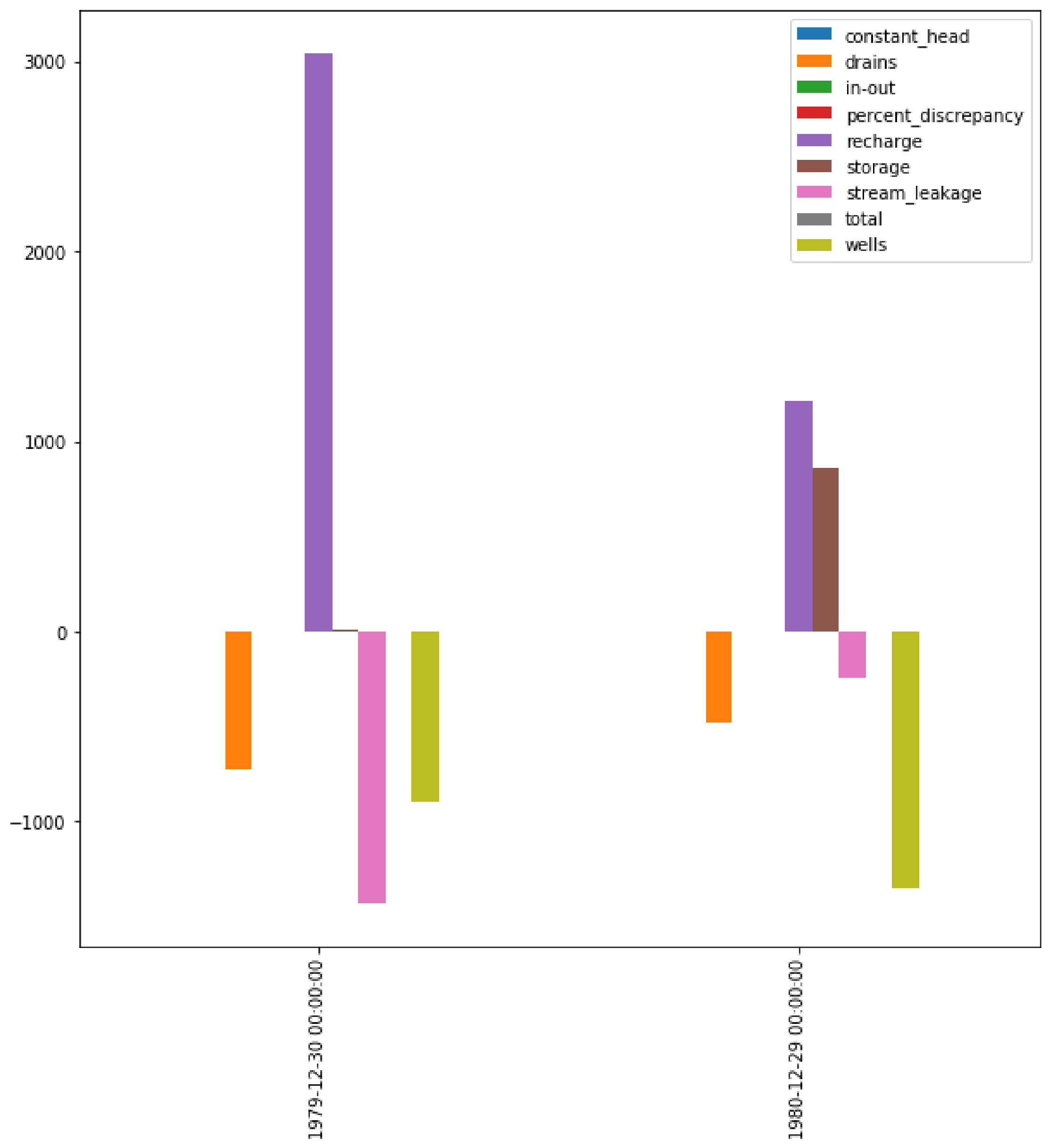
Run one last time. `phi` should be near zero since we haven't change the `parval1` values for historic stress period and only the 13 gw level obs have nonzero weights

```
In [26]: 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[26]: 9.456182577320024e-19
```

```
In [27]: 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[27]: <matplotlib.axes._subplots.AxesSubplot at 0x11ef7e668>
```



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