recharge_comparison__Maui_75m

November 16, 2015

1 *Potential Recharge:* Comparison between SWB and HWB

1.1 Preliminaries

1.1.1 Bring in Python modules

```
In [22]: %matplotlib inline

         import os
         import gdal
         import sys
         import matplotlib.pyplot as plt
         import matplotlib.colors as colors
         from netCDF4 import Dataset as netcdf_dataset
         import numpy as np
         import numpy.ma as ma
```

1.1.2 Set variables that determine the model output being compared

```
In [23]: SWB_param_name             = "potential_recharge"
         SWB_filename               = 'potential_recharge_2001_2002__705_by_1054.nc'
         SWB_grid_dimensions        = "1054_by_705"

         Simulation_start_year      = "2001"
         Simulation_stop_year       = "2002"
         Simulation_length_in_years = 2.0

         HWB_param_name             = "Maui__75m__25_sim_2001_2002_Rech"

         Plot_title_param           = "Potential Recharge"
         Plot_title_units           = ", in Inches"
```

1.1.3 Define basic plot types

```
In [24]: %run plot_and_table_functions.ipynb
```

1.1.4 Open and read model output files

```
In [25]: # Define the pathname to the SWB output file.
         fname_swb = SWB_filename
         dataset_swb = netcdf_dataset( os.path.join('..', fname_swb ) )
         if dataset_swb is None:
             print 'Could not open netCDF grid file'
             sys.exit(1)
```

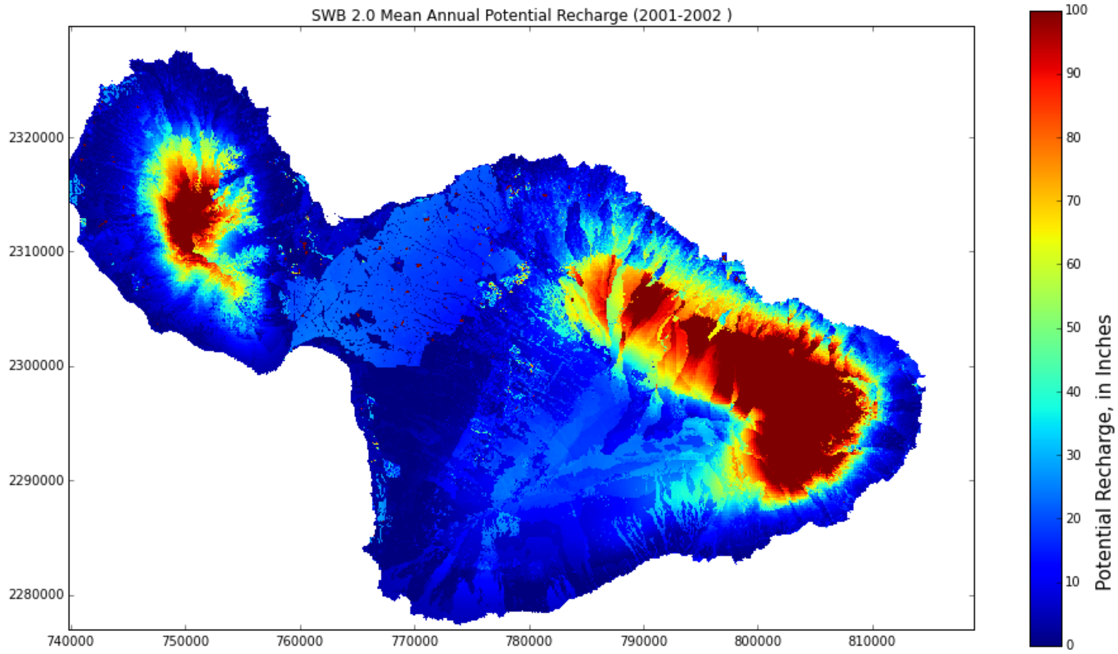
```
# Define pathname to the HWB output file
fname_hwb = '../../../Maui_Common_Data/reference_HWB_output/' + HWB_param_name + '.asc'
dataset_hwb = gdal.Open( fname_hwb )
if dataset_hwb is None:
    print 'Could not open data grid'
    sys.exit(1)

# Define the pathname to the landuse Arc ASCII file output by SWB
fname_lu = os.path.join('..', 'Landuse_land_cover__as_read_into_SWB.asc' )
dataset_lu = gdal.Open( fname_lu )
if dataset_lu is None:
    print 'Could not open landuse grid'
    sys.exit(1)

# Define the pathname to the SWB landuse lookup table
fname_lu_table = '../../../Maui_Common_Data/std_input/LU_lookup_Engott_v3_5.txt'
lu_table = np.genfromtxt( fname_lu_table, names=True, delimiter='\t', dtype=None)


# Last, define the pathname to the Hawaii Aquifer Code grid
fname_aquifer_cd = '../../../Maui_Common_Data/reference_HWB_output/Maui__75m__25_sim_2001_2002_
dataset_aquifer = gdal.Open( fname_aquifer_cd )
if dataset_aquifer is None:
    print 'Could not open data grid'
    sys.exit(1)
```

1.2 Plots and Comparisons

1.2.1 Plot of Soil-Water-Balance Model output

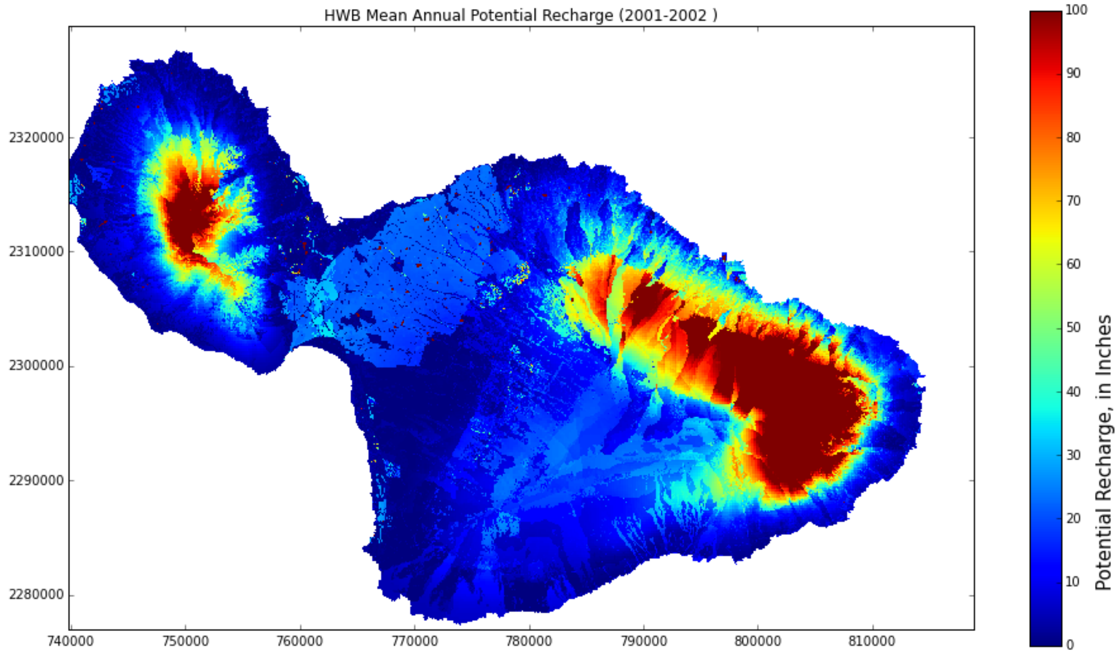
```
In [26]: SWB_values = dataset_swb.variables[ SWB_param_name ][:, :, :]
         # the netCDF values come in with not-a-number (NaN) values in the inactive cells;
         # need to deal with these NaNs before doing any processing
         SWB_values = ma.masked_where( np.isnan(SWB_values), SWB_values )

         # sum daily model outputs over the time dimension (axis "0")
         SWB_values = SWB_values.sum(axis=0) / Simulation_length_in_years

         x = dataset_swb.variables['x'][:]
         y = dataset_swb.variables['y'][:]

         make_plot(x=x, y=y, var=SWB_values,
                   title="SWB 2.0 Mean Annual " + Plot_title_param + " ("
                   + Simulation_start_year + "-" + Simulation_stop_year + " )",
                   barlabel=Plot_title_param + Plot_title_units,
                   minz=0., maxz=100., n=11 )
```

```
c:\Anaconda3\envs\python27\lib\site-packages\matplotlib\collections.py:590: FutureWarning: elementwise c
  if self._edgecolors == str('face'):
```



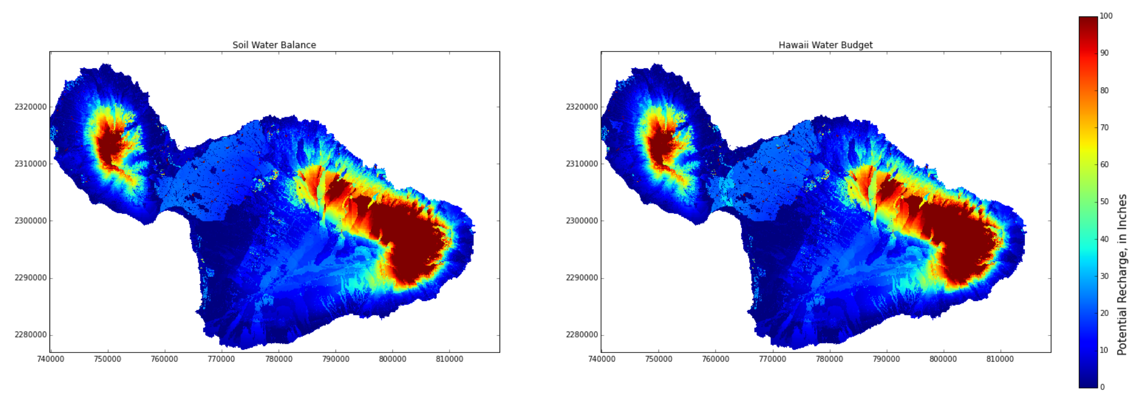
1.2.2 Plot of Hawaii-Water-Budget Model output

In [27]:
```
HWB_values = dataset_hwb.ReadAsArray()
# the HWB output comes in as an Arc ASCII grid, with "-9999" values denoting the
# areas of inactive model domain; as with the previous grid, must deal with these
# before doing any further analysis. numpy "masked array" does what we need.
HWB_values = ma.masked_where( HWB_values < 0, HWB_values )

make_plot(x=x, y=y, var=HWB_values,
          title="HWB Mean Annual " + Plot_title_param + " ("
          + Simulation_start_year + "-" + Simulation_stop_year + " )",
          barlabel=Plot_title_param + Plot_title_units,
          minz=0., maxz=100., n=11 )
```



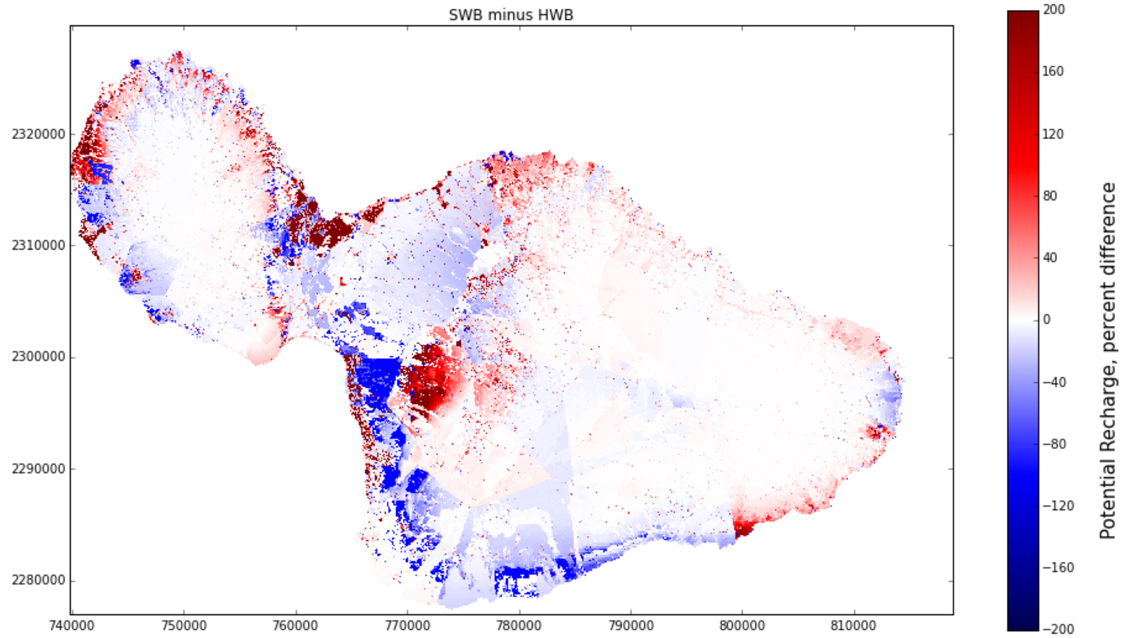
1.2.3 Comparison Plots

```
In [28]: make_side_by_side_plots(x=x, y=y, var1=SWB_values, var2=HWB_values, minz=0., maxz=100., n=11,
                                  title1='Soil Water Balance', title2='Hawaii Water Budget',
                                  figsize_x=27., figsize_y=9.,
                                  barlabel=Plot_title_param + Plot_title_units )
```



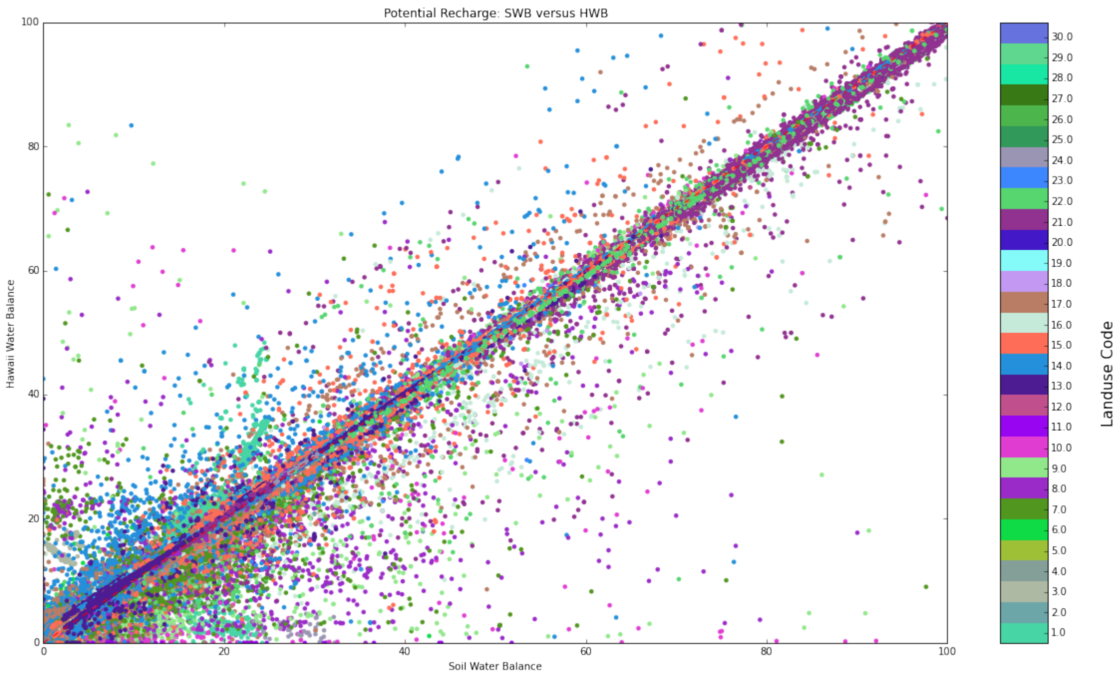
1.2.4 Plot of differences: SWB output minus HWB output

```
In [29]: mean_grid = ( SWB_values + HWB_values ) / 2.0
         pct_diff = (SWB_values - HWB_values) / HWB_values * 100.
         make_diffplot(x=x, y=y, var=pct_diff, n=11,
                   minz=-200., maxz=200.,
                   title="SWB minus HWB",
                   barlabel=Plot_title_param + ", percent difference" )
```



1.2.5 Plot of SWB and HWB model outputs

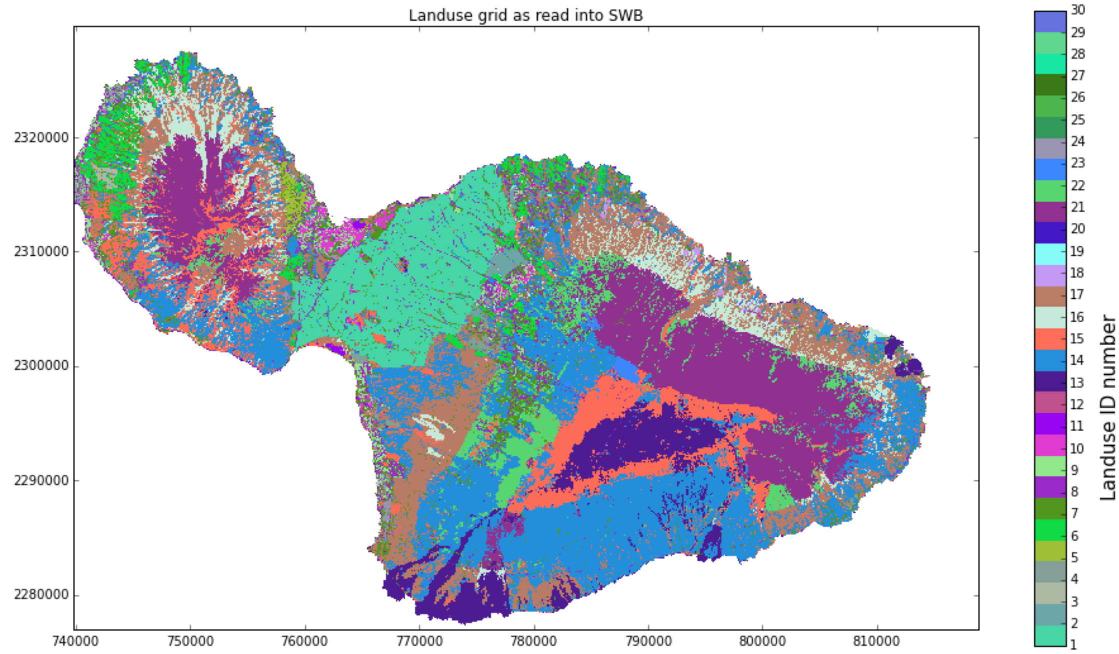
In [30]:
```
lu = dataset_lu.ReadAsArray()
lu = ma.masked_where( lu <= 0, lu )


lu_descriptions=lu_table['Description']
lu_lookup_values=lu_table['LU_code']

n_lu = np.nanmax(lu.flatten()) - np.nanmin(lu.flatten()) + 1

#cmap=plt.get_cmap('gist_ncar', n_lu )
cmap = colors.ListedColormap ( np.random.rand ( n_lu, 3))

make_scatter(x=SWB_values, y=HWB_values, color=lu, cmap=cmap, title=Plot_title_param
             + ": SWB versus HWB", xlab="Soil Water Balance", barlabel="Landuse Code",
             ylab="Hawaii Water Balance", xmax=100., ymax=100. )
```



Comments: The color coding above corresponds to the landuse code from the LU_code field in the landuse lookup table. Linear strings of points that do not fall on the 1:1 slope line appear to be caused by mismatches in grid cell coverage and resolution in the various input files.

1.2.6 Plot of landuse as read in by SWB

```
In [31]: make_plot(x=x, y=y, var=lu,
                   title="Landuse grid as read into SWB",
                   barlabel="Landuse ID number",
                   cmap=cmap,
                   each=True)
```



1.2.7 Table of model differences by landuse code

```
In [32]: lu_descriptions=lu_table['Description']
         lu_lookup_values=lu_table['LU_code']
         make_comparison_table( x=SWB_values, y=HWB_values, factor=lu,
                                description=lu_descriptions,
                                lookup_vals=lu_lookup_values)
```

```
c:\Anaconda3\envs\python27\lib\site-packages\numpy\lib\nanfunctions.py:675: RuntimeWarning: Mean of empt
  warnings.warn("Mean of empty slice", RuntimeWarning)
```

	Code	Description	Count	SWB Mean	HWB Mean	\
0	0	Background	407945	0.266	9.428	
1	1	Sugarcane	28792	19.727	21.978	
2	2	Pineapple	1263	18.816	18.756	
3	3	Coffee	385	1.840	14.436	
4	4	Diversified Agriculture	1327	8.454	8.828	
5	5	Macadamia	938	0.840	0.841	
6	6	Fallow_grassland	7507	3.568	3.459	
7	7	Developed Open Space	17584	9.328	9.036	
8	8	Developed Low intensity	9796	11.993	10.840	
9	9	Developed Medium intensity	3924	11.919	8.119	
10	10	Developed High intensity	2679	11.009	5.513	
11	11	Water body	326	6.924	0.155	
12	12	Wetland	133	84.487	85.625	
13	13	Sparsely vegetated	20343	15.407	15.598	
14	14	Grassland	79370	11.420	11.572	
15	15	Shrubland	29857	25.873	26.149	
16	16	Native forest	18303	33.719	33.154	
17	17	Alien forest	45624	16.360	16.150	

```
18    18            Tree plantation     489    32.554    31.660
19    19                 Reservoirs     150   552.221   500.258
21    21          Native forest fog   49419    95.927    95.207
22    22           Alien forest fog   14177    51.995    51.570
23    23        Tree plantation fog    1120    35.406    34.991
24    24                Golf course    1273    27.342    26.571
25    25                       Taro      44   462.286   435.817
30    30             HC&S reservoirs     302 1,276.059 1,238.859

    Percent Difference
0              -97.178
1              -10.242
2                0.324
3              -87.255
4               -4.239
5               -0.057
6                3.165
7                3.226
8               10.640
9               46.809
10              99.702
11           4,359.513
12              -1.328
13              -1.221
14              -1.308
15              -1.056
16               1.703
17               1.300
18               2.823
19              10.387
21               0.757
22               0.824
23               1.188
24               2.903
25               6.073
30               3.003
```

In [33]:
```python
aquifer_codes = dataset_aquifer.ReadAsArray()
aquifer_codes = ma.masked_where( aquifer_codes <= 0, aquifer_codes )
aquifer_descriptions=np.unique( aquifer_codes.data )
aquifer_lookup_values=np.unique( aquifer_codes.data )
make_comparison_table( x=SWB_values, y=HWB_values, factor=aquifer_codes,
                       description=np.sort(aquifer_descriptions[aquifer_descriptions > 0]),
                       lookup_vals=np.sort(aquifer_lookup_values[aquifer_lookup_values > 0]))
```

```
        Code  Description  Count  SWB Mean  HWB Mean  Percent Difference
0  60,101.000   60,101.000   7977    18.675    18.497               0.960
1  60,102.000   60,102.000  11635    24.470    23.869               2.515
2  60,103.000   60,103.000   5721    25.098    24.377               2.957
3  60,104.000   60,104.000   4951    21.574    21.443               0.608
4  60,201.000   60,201.000   6394    34.859    35.038              -0.512
5  60,202.000   60,202.000   8192    23.843    24.003              -0.668
6  60,203.000   60,203.000  10780    25.415    25.575              -0.626
7  60,204.000   60,204.000   9561    32.354    32.519              -0.508
```

8	60,205.000	60,205.000	3732	26.004	26.235	-0.882
9	60,206.000	60,206.000	5447	20.334	20.395	-0.303
10	60,301.000	60,301.000	12661	17.975	18.188	-1.169
11	60,302.000	60,302.000	27179	25.367	27.045	-6.205
12	60,303.000	60,303.000	23847	14.688	14.492	1.354
13	60,304.000	60,304.000	42075	7.012	6.813	2.914
14	60,401.000	60,401.000	16198	28.648	27.521	4.096
15	60,402.000	60,402.000	8159	35.386	34.609	2.245
16	60,403.000	60,403.000	11981	56.645	55.410	2.230
17	60,404.000	60,404.000	25433	69.187	68.793	0.573
18	60,501.000	60,501.000	6259	115.665	114.596	0.933
19	60,502.000	60,502.000	14621	59.450	58.793	1.117
20	60,503.000	60,503.000	6632	67.117	66.425	1.042
21	60,504.000	60,504.000	13665	72.261	71.093	1.643
22	60,601.000	60,601.000	9362	26.621	26.559	0.232
23	60,602.000	60,602.000	13891	19.534	19.550	-0.081
24	60,603.000	60,603.000	28763	7.805	8.371	-6.765

```
In [ ]:
```