rain-days-montly-scatterplot

January 25, 2022

1 Find rain days in a specified region, given 8 upper atm variables and respective Gridmet data.

This notebook calculates average rain on a 128x128 grid, it then averages the respective 8 upper atm variables.

- this version loads netcdf files directly via xarray
- "Gridmet data" can now be replaced with any model's output

1.0.1 TODO items

- better graphs (i.e. include lat/lon scale)

[2]:
```
from IPython import display
import numpy as np

#from skimage.metrics import structural_similarity as ssim
import copy
```

[3]:
```
# load netcdf files directly via xarray
import xarray as xr
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy
```

2 Configuration

[4]:
```
# Ordering: nRow ordering: obs, raw, RegCM4, WRF, MPAS, CNN, SDSM, KDDM, MBCn,␣
 ↪LOCA
mpi_sigfiles=['/glade/work/mcginnis/DCA/data/gen/final/gridmet/hist/prec.hist.
 ↪gridMET.obs.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/mpi/hist/prec.hist.MPI-ESM-LR.raw.day.
 ↪1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/mpi/rcp85/prec.rcp85.MPI-ESM-LR.raw.day.
 ↪2075-2100.NAM-22i.SGP.x098.y36.nc',
```

```
'/glade/work/mcginnis/DCA/data/gen/final/regcm4/hist/prec.hist.MPI-ESM-LR.RegCM4.
 ↪day.1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/regcm4/rcp85/prec.rcp85.MPI-ESM-LR.
 ↪RegCM4.day.2075-2100.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/wrf/hist/prec.hist.MPI-ESM-LR.WRF.day.
 ↪1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/wrf/rcp85/prec.rcp85.MPI-ESM-LR.WRF.day.
 ↪2075-2100.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/mpas/hist/prec.hist.MPI-ESM-LR.MPAS.day.
 ↪1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/mpas/rcp85/prec.rcp85.MPI-ESM-LR.MPAS.
 ↪day.2075-2100.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/cnn/hist/prec.hist.MPI-ESM-LR.CNN.day.
 ↪1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/cnn/rcp85/prec.rcp85.MPI-ESM-LR.CNN.day.
 ↪2075-2100.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/sdsm/hist/prec.hist.MPI-ESM-LR.SDSM.day.
 ↪1976-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/sdsm/rcp85/prec.rcp85.MPI-ESM-LR.SDSM.day.
 ↪2070-2099.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/kddm/hist/prec.hist.MPI-ESM-LR.KDDM.day.
 ↪1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/kddm/rcp85/prec.rcp85.MPI-ESM-LR.KDDM.
 ↪day.2075-2100.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/mbcn/hist/prec.hist.MPI-ESM-LR.MBCn.day.
 ↪1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/mbcn/rcp85/prec.rcp85.MPI-ESM-LR.MBCn.
 ↪day.2075-2100.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/loca/hist/prec.hist.MPI-ESM-LR.LOCA.day.
 ↪1980-2005.NAM-22i.SGP.x098.y36.nc',
'/glade/work/mcginnis/DCA/data/gen/final/loca/rcp85/prec.rcp85.MPI-ESM-LR.LOCA.
 ↪day.2075-2100.NAM-22i.SGP.x098.y36.nc']
```

3 Compact Configuration

[5]:
```
sigdir = '/glade/work/mcginnis/DCA/data/gen/final'
def model2absfilepath(mn, exp, x, y):
    if mn == 'obs' or mn == 'gridMET':
        mn = 'gridMET'
        filename = 'prec.{}.gridMET.{}.day.1980-2005.NAM-22i.SGP.x0{}.y{}.nc'.
 ↪format(exp, 'obs', x, y)
    elif mn == 'raw' or mn == 'mpi':
        mn = 'mpi'
        if exp == 'hist':
```

```
            filename = 'prec.{}.MPI-ESM-LR.{}.day.1980-2005.NAM-22i.SGP.x0{}.y{}.
↪nc'.format(exp, 'raw', x, y)
        elif exp == 'rcp85':
            filename = 'prec.{}.MPI-ESM-LR.{}.day.2075-2100.NAM-22i.SGP.x0{}.y{}.
↪nc'.format(exp, 'raw', x, y)
        else: print("Unknown experiment!")
    elif mn == 'SDSM':  # starts at 1976
        if exp == 'hist':
            filename = 'prec.{}.MPI-ESM-LR.{}.day.1976-2005.NAM-22i.SGP.x0{}.y{}.
↪nc'.format(exp, mn, x, y)
        elif exp == 'rcp85':
            filename = 'prec.{}.MPI-ESM-LR.{}.day.2070-2099.NAM-22i.SGP.x0{}.y{}.
↪nc'.format(exp, mn, x, y)
        else: print("Unknown experiment!")

    elif exp == 'hist':
        filename = 'prec.{}.MPI-ESM-LR.{}.day.1980-2005.NAM-22i.SGP.x0{}.y{}.nc'.
↪format(exp, mn, x, y)
    elif exp == 'rcp85':
        filename = 'prec.{}.MPI-ESM-LR.{}.day.2075-2100.NAM-22i.SGP.x0{}.y{}.nc'.
↪format(exp, mn, x, y)
    else: print("Unknown experiment!")
    return sigdir + '/' + mn.lower() + '/' + exp + '/' + filename
```

[6]:
```
# test single file
model2absfilepath('RegCM4', 'rcp85', 98, 36)
```

[6]:
```
'/glade/work/mcginnis/DCA/data/gen/final/regcm4/rcp85/prec.rcp85.MPI-ESM-
LR.RegCM4.day.2075-2100.NAM-22i.SGP.x098.y36.nc'
```

[7]:
```
# create list of abs filepaths
[model2absfilepath(ff, 'hist', 98, 36) for ff in␣
↪['obs','raw','RegCM4','WRF','MPAS','CNN','SDSM','KDDM','MBCn','LOCA']]
```

[7]:
```
['/glade/work/mcginnis/DCA/data/gen/final/gridmet/hist/prec.hist.gridMET.obs.day
.1980-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/mpi/hist/prec.hist.MPI-ESM-
LR.raw.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/regcm4/hist/prec.hist.MPI-ESM-
LR.RegCM4.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/wrf/hist/prec.hist.MPI-ESM-
LR.WRF.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/mpas/hist/prec.hist.MPI-ESM-
LR.MPAS.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/cnn/hist/prec.hist.MPI-ESM-
LR.CNN.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
```

```
 '/glade/work/mcginnis/DCA/data/gen/final/sdsm/hist/prec.hist.MPI-ESM-
LR.SDSM.day.1976-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/kddm/hist/prec.hist.MPI-ESM-
LR.KDDM.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/mbcn/hist/prec.hist.MPI-ESM-
LR.MBCn.day.1980-2005.NAM-22i.SGP.x098.y36.nc',
 '/glade/work/mcginnis/DCA/data/gen/final/loca/hist/prec.hist.MPI-ESM-
LR.LOCA.day.1980-2005.NAM-22i.SGP.x098.y36.nc']
```

3.0.1 Load model output data

[8]:
```
##################
## LOAD GRIDMET ##
##################
# 1980-1989
istart = 365
#ndays = 3653  # 1980-1989
ndays = 13515 # 1980-2016

dv1 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪prec128_gridmetA_1979-2016.nc')['prec'][istart:istart+ndays]
# dv2 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪tmax128_gridmetA_1979-2016.nc')['tmax'][istart:istart+ndays]
# dv3 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪tmin128_gridmetA_1979-2016.nc')['tmin'][istart:istart+ndays]
# dv4 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪uas128_gridmetA_1979-2016.nc')['uas'][istart:istart+ndays]
# dv5 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪vas128_gridmetA_1979-2016.nc')['vas'][istart:istart+ndays]
# dv6 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪huss128_gridmetA_1979-2016.nc')['huss'][istart:istart+ndays]
# dv7 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪rsds128_gridmetA_1979-2016.nc')['rsds'][istart:istart+ndays]
# dv8 = xr.open_dataset('/glade/work/dkorytin/srgan_data/
 ↪miss128_gridmetB_1979-2016.nc')['miss'][istart:istart+ndays]
```

3.0.2 Load MPI UATM data

[9]:
```
############################
## LOAD ERAI data: daily ##
############################
def load_uatm_erai():
    global mdv1,mdv2,mdv3,mdv4,mdv5,mdv6,mdv7,mdv8,mnvars

    # load eraionmpi
    #mndays = 11688      # 1979-2010
```

```
    mstart = 365
    mndays = 13515        # 1980-2016
    mnvars = 8*1
    mdv1 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/U850.ERAI.
↪MPIGRID.1979-2018.nc')['U'][mstart:mstart+mndays*1]
    mdv2 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/V850.ERAI.
↪MPIGRID.1979-2018.nc')['V'][mstart:mstart+mndays*1]
    mdv3 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/Q850.ERAI.
↪MPIGRID.1979-2018.nc')['Q'][mstart:mstart+mndays*1]
    mdv4 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/T700.ERAI.
↪MPIGRID.1979-2018.nc')['T'][mstart:mstart+mndays*1]
    mdv5 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/Z700.ERAI.
↪MPIGRID.1979-2018.nc')['Z'][mstart:mstart+mndays*1]
    mdv6 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/Z500.ERAI.
↪MPIGRID.1979-2018.nc')['Z'][mstart:mstart+mndays*1]
    mdv7 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/U250.ERAI.
↪MPIGRID.1979-2018.nc')['U'][mstart:mstart+mndays*1]
    mdv8 = xr.open_dataset('/glade/scratch/dkorytin/erai-on-mpigrid/V250.ERAI.
↪MPIGRID.1979-2018.nc')['V'][mstart:mstart+mndays*1]

    print("Days loaded", len(mdv7))
```

[10]:
```
####################
## LOAD MPI      ##
####################
def load_uatm_mpi_hist():
    global mdv1,mdv2,mdv3,mdv4,mdv5,mdv6,mdv7,mdv8,mnvars

    mnvars = 8

    d='/glade/p/ral/risc/rmccrary/CMIP5_CORDEX/NAmerica/MPI-ESM-LR/native/
↪historical/'

    mdv1 = xr.
↪open_dataset(d+'U_MPI-ESM-LR_historical_r1i1p1_NAmerica_p850_19500101-20051231_dayavg_mpigrid
↪nc')['U'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv2 = xr.
↪open_dataset(d+'V_MPI-ESM-LR_historical_r1i1p1_NAmerica_p850_19500101-20051231_dayavg_mpigrid
↪nc')['V'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv3 = xr.
↪open_dataset(d+'Q_MPI-ESM-LR_historical_r1i1p1_NAmerica_p850_19500101-20051231_dayavg_mpigrid
↪nc')['Q'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
```

```
    mdv4 = xr.
↪open_dataset(d+'T_MPI-ESM-LR_historical_r1i1p1_NAmerica_p700_19500101-20051231_dayavg_mpigrid
↪nc')['T'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv5 = xr.
↪open_dataset(d+'Z_MPI-ESM-LR_historical_r1i1p1_NAmerica_p700_19500101-20051231_dayavg_mpigrid
↪nc')['Z'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv6 = xr.
↪open_dataset(d+'Z_MPI-ESM-LR_historical_r1i1p1_NAmerica_p500_19500101-20051231_dayavg_mpigrid
↪nc')['Z'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv7 = xr.
↪open_dataset(d+'U_MPI-ESM-LR_historical_r1i1p1_NAmerica_p250_19500101-20051231_dayavg_mpigrid
↪nc')['U'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv8 = xr.
↪open_dataset(d+'V_MPI-ESM-LR_historical_r1i1p1_NAmerica_p250_19500101-20051231_dayavg_mpigrid
↪nc')['V'].sel(time=slice('1976-01-01T00:00:00', '2006-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))

def load_uatm_mpi_future():
    global mdv1,mdv2,mdv3,mdv4,mdv5,mdv6,mdv7,mdv8,mnvars

    d='/glade/p/ral/risc/rmccrary/CMIP5_CORDEX/NAmerica/MPI-ESM-LR/native/rcp85/'

    mnvars = 8
    mdv1 = xr.
↪open_dataset(d+'U_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p850_20060101-21001231_dayavg_mpigrid.
↪nc')['U'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv2 = xr.
↪open_dataset(d+'V_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p850_20060101-21001231_dayavg_mpigrid.
↪nc')['V'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv3 = xr.
↪open_dataset(d+'Q_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p850_20060101-21001231_dayavg_mpigrid.
↪nc')['Q'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv4 = xr.
↪open_dataset(d+'T_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p700_20060101-21001231_dayavg_mpigrid.
↪nc')['T'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
```

```
    mdv5 = xr.
↪open_dataset(d+'Z_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p700_20060101-21001231_dayavg_mpigrid.
↪nc')['Z'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv6 = xr.
↪open_dataset(d+'Z_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p500_20060101-21001231_dayavg_mpigrid.
↪nc')['Z'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv7 = xr.
↪open_dataset(d+'U_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p250_20060101-21001231_dayavg_mpigrid.
↪nc')['U'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))
    mdv8 = xr.
↪open_dataset(d+'V_MPI-ESM-LR_rcp85_r1i1p1_NAmerica_p250_20060101-21001231_dayavg_mpigrid.
↪nc')['V'].sel(time=slice('2006-01-01T00:00:00', '2101-01-01T00:00:00'),␣
↪lat=slice(23,56), lon=slice(-113,-80))

#load_uatm_mpi_future()
```

4 Averaging code

[11]:
```
###################
## UATM averages ##
###################

def calc_uatm_average(mpr_min, mpr_max, rainsignal, month, ymin, ymax):
    # global inputs: dv[1-8], mdv[1-8], mnvars
    mnvars = 8

    # initialize accumulators
    indices = []
    distribution = []
    raw_ires = len(mdv1[0])
    raininput = np.zeros((mnvars, raw_ires, raw_ires))
    raincount = 0

    for ii in range(0,len(rainsignal)):
        mpr = rainsignal[ii]

        # May = 5
        if mpr["time.month"] == month and mpr["time.year"] >= ymin and mpr["time.
↪year"] < ymax:



            if mpr >= mpr_min and mpr < mpr_max:
```

```
                distribution.append(mpr)
                indices.append(ii)

                #mpr = dv1[ii].sel(lat=slice(32.125,38.125), lon=slice(-101.
↪875,-93.875)).mean()

                # find matching input sample
                #ot= dv1[ii-1]['time']
                ot = str(np.array(mpr['time']))

                mdv1.sel(time=ot,method='nearest')
                isample = [mdv1.sel(time=ot,method='nearest'),mdv2.
↪sel(time=ot,method='nearest'),mdv3.sel(time=ot,method='nearest'),mdv4.
↪sel(time=ot,method='nearest'),mdv5.sel(time=ot,method='nearest'),mdv6.
↪sel(time=ot,method='nearest'),mdv7.sel(time=ot,method='nearest'),mdv8.
↪sel(time=ot,method='nearest')]
                #ot= dv1[ii]['time']
                #assert ot["time.day"]==isample[0]['time.day'], "Days are not␣
↪equal"
                isample = np.array(isample)

                raininput += isample
                raincount += 1

    raininput /= raincount
    #print("days:", raincount)

    return raininput, raincount, distribution, indices


def calc_prec_average(mpr_min, mpr_max, rainsignal, month, ymin, ymax):

    # initialize accumulators
    indices = []
    distribution = []
    raininput = 0
    raincount = 0

    for ii in range(0,len(rainsignal)):
        mpr = rainsignal[ii]

        # May = 5
        if mpr["time.month"] == month and mpr["time.year"] >= ymin and mpr["time.
↪year"] < ymax:

            if mpr >= mpr_min and mpr < mpr_max:
                distribution.append(mpr)
```

```
                indices.append(ii)

                #mpr = dv1[ii].sel(lat=slice(32.125,38.125), lon=slice(-101.
↪875,-93.875)).mean()

                raininput += mpr
                raincount += 1

    raininput /= raincount
    #print("days:", raincount)

    return raininput, raincount, distribution, indices
```

5 Plot routines

[12]:
```
from os.path import basename
from scipy.stats import spearmanr

def linear_regression(x, y):
    coefs = np.polynomial.polynomial.polyfit(x, y, 1)
    ffit = np.poly1d(coefs)
    m = ffit[0]
    b = ffit[1]
    eq = 'y = {}x + {}'.format(round(m, 3), round(b, 3))
    pc = np.corrcoef(x, y)[0, 1]

    return pc, eq, m, b
```

[79]:
```
def plot_scatter_row(X, Y, ax, title, minxx, maxx):
    names = ['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']
    markers = ['P','o','X','s','d','^','v','p','*']

    # plot best fit line
    pc, eq, m, b = linear_regression(X,Y)
    xmaxreg = max(X)
    ymaxreg = b + m*xmaxreg
    if ymaxreg > max(Y):
        xmaxreg = (max(Y)-b) / m
    #ax.plot([0,xmaxreg], [b, b + m*xmaxreg], c = 'r', label=eq)

    #ax.scatter(X, Y, color="tab:blue", s=1, label='prec (mm/day)')
    #ax.scatter(X, Y, color="tab:blue", label='prec (mm/day)')
```

```
    for ii, txt in enumerate(names):
        ax.scatter(X[ii], Y[ii], marker=markers[ii], s=80, color="tab:blue",␣
 ↪label='prec (mm/day)')



    # draw 1:1 line
    #XX = np.linspace(0, xmaxreg, num=50)
    XX = np.linspace(minxx, maxx, num=50)
    #ax.plot(XX, XX, 'x')
    ax.plot([minxx,maxx], [minxx, maxx], c = 'r', label='1=1')

    ## legend
    #r2_text = 'R² = {}'.format(round(pc**2, 3))
    #r2_text += '\nPC = {}'.format(round(pc, 3))
    #sc, delme = spearmanr(X,Y)
    #r2_text += '\nSC = {}'.format(round(sc, 3))
    #rmse = ((np.array(X)-np.array(Y))**2).mean()**0.5
    #r2_text += '\nRMSE = {}'.format(round(rmse, 3))
    #ax.legend(loc="lower right", title=r2_text)

    # axis labels
    ax.set_xlabel('historical prec (mm/day)')
    ax.set_ylabel('rcp85 prec (mm/day)')

    # title
    ax.set_title(title.upper())

    #plt.show()


def plot_scatter_row_3x(XX, YY, title, minxx, maxx):
    names = ['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']
    markers = ['P','o','X','s','d','^','v','p','*']


    #minxx = min(min(np.array(XX).flatten()), min(np.array(YY).flatten()))


    #ax.scatter(X, Y, color="tab:blue", s=1, label='prec (mm/day)')
#     plt.scatter(XX[0], YY[0], marker=markers, color="tab:orange", label='Dry␣
 ↪(mm/day)')
#     plt.scatter(XX[1], YY[1], color="tab:green", label='Moist (mm/day)')
#     plt.scatter(XX[2], YY[2], color="tab:blue", label='Wet (mm/day)')
#     plt.scatter(XX[3], YY[3], color="tab:pink", label='All (mm/day)')
    for ii, txt in enumerate(names):
```

```
        plt.scatter(XX[0][ii], YY[0][ii], marker=markers[ii], color="tab:
↪orange", label='Dry (mm/day)')
        plt.scatter(XX[1][ii], YY[1][ii], marker=markers[ii], color="tab:green",␣
↪label='Moist (mm/day)')
        plt.scatter(XX[2][ii], YY[2][ii], marker=markers[ii], color="tab:blue",␣
↪label='Wet (mm/day)')
        plt.scatter(XX[3][ii], YY[3][ii], marker=markers[ii], color="tab:pink",␣
↪label='All (mm/day)')

    # draw 1:1 line
    #XX = np.linspace(0, xmaxreg, num=50)
    XX = np.linspace(0, maxx, num=50)
    #ax.plot(XX, XX, 'x')
    plt.plot([minxx,maxx], [minxx, maxx], c = 'r', label='1=1')

#     # legend
#     r2_text = 'R² = {}'.format(round(pc**2, 3))
#     r2_text += '\nPC = {}'.format(round(pc, 3))
#     #print(spearmanr(rs1,rs2))
#     sc, delme = spearmanr(X,Y)
#     r2_text += '\nSC = {}'.format(round(sc, 3))
#     rmse = ((np.array(X)-np.array(Y))**2).mean()**0.5
#     r2_text += '\nRMSE = {}'.format(round(rmse, 3))
#     plt.legend(loc="lower right", title=r2_text)
    #plt.legend(loc="lower right")
    plt.legend(['Dry','Moist','Wet','All'], loc="upper left")

    # axis labels
    #plt.set_xlabel('historical prec (mm/day)')
    #plt.set_ylabel('rcp85 prec (mm/day)')
    plt.xlabel('historical prec (mm/day)')
    plt.ylabel('rcp85 prec (mm/day)')

    # title
    #plt.set_title(title.upper())
    plt.title(title.upper())
```

6 Run over all models

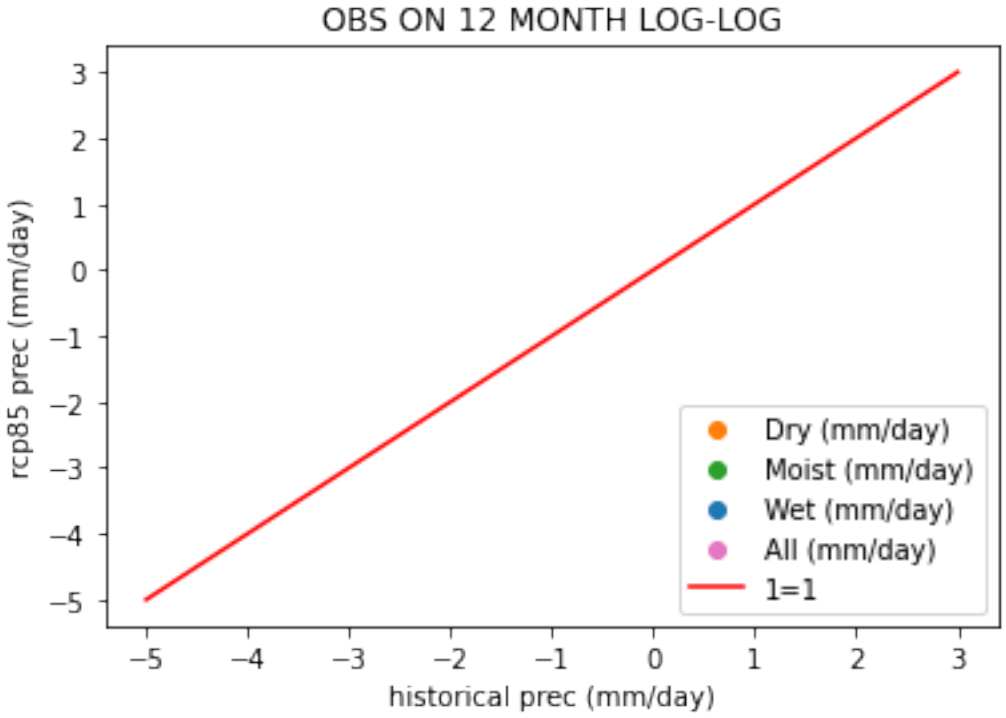
6.0.1 Month vs Month

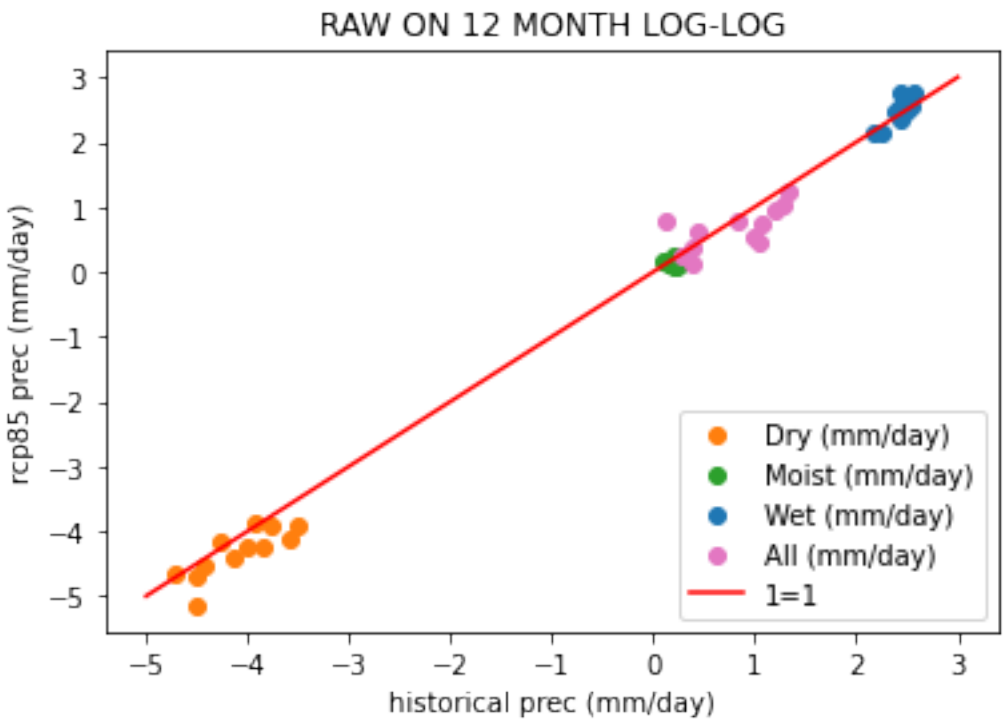
[29]:
```
# Plot ALL models
#[model2absfilepath(ff, 'hist', 98, 36) for ff in␣
↪['obs','raw','RegCM4','WRF','MPAS','CNN','SDSM','KDDM','MBCn','LOCA']]

names = ['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']
```

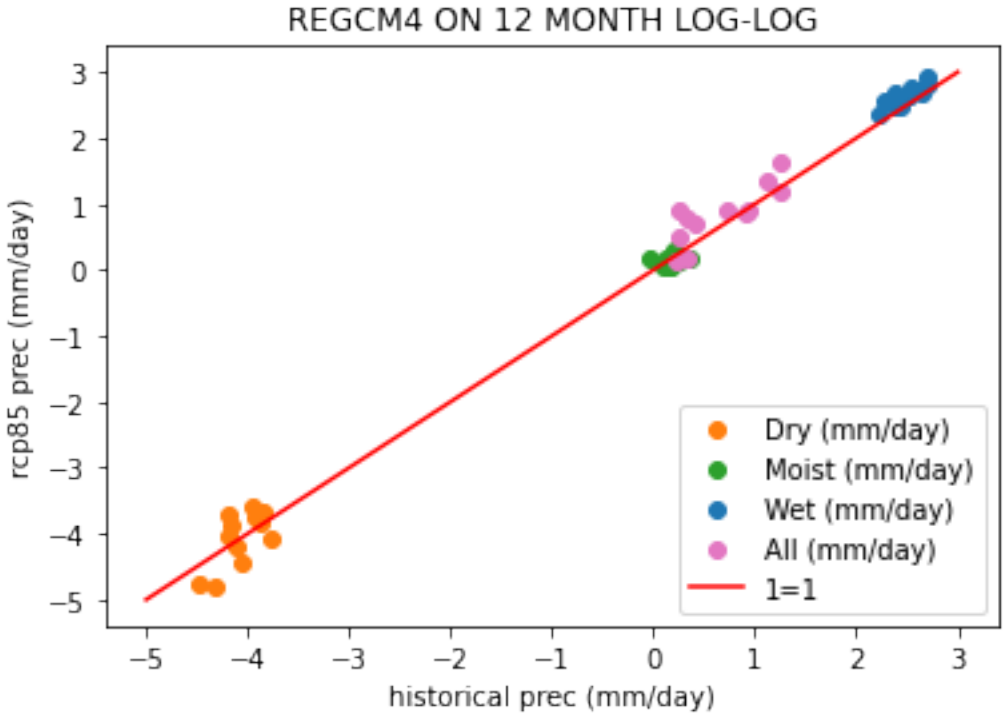
```
sigfiles_hist = [model2absfilepath(ff, 'hist', 98, 36) for ff in␣
 ↪['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']]
sigfiles_rcp85 = [model2absfilepath(ff, 'rcp85', 98, 36) for ff in␣
 ↪['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']]

# interate every model
AXX=[]; AYY=[]      # XX and YY for all models
for mii in range(len(sigfiles_hist)):
    sigfn1 = sigfiles_hist[mii]
    rainsignal1 = xr.open_dataset(sigfn1)['prec']

    # future doesn't have obs
    if mii > 0:
        sigfn2 = sigfiles_rcp85[mii]
        rainsignal2 = xr.open_dataset(sigfn2)['prec']


    # get averages for each of the twelve months
    XX=[]; YY=[]
    for mpr_min, mpr_max in [(0,0.254), (0.254, 3.0), (3.0, 9999.0), (0, 9999.
 ↪0)]:
        X=[]; Y=[]
        for ii in range(1,13):
            avginput1, count1, distribution1, indices1 =␣
 ↪calc_prec_average(mpr_min, mpr_max, rainsignal1, ii, -9999, 9999)

            # obs doesn't have future
            if mii == 0:
                avginput2 = 0
            else:
                avginput2, count2, distribution2, indices2 =␣
 ↪calc_prec_average(mpr_min, mpr_max, rainsignal2, ii, -9999, 9999)

            X.append(float(avginput1))
            Y.append(float(avginput2))
        XX.append(X)
        YY.append(Y)
    AXX.append(XX)
    AYY.append(YY)

    # plot row
    mname = names[mii]
#     fig, axarr = plt.subplots(1, 3, figsize = (20,20/4))
#     plot_scatter_row(XX[0], YY[0], axarr[0], mname+': dry', .031)
#     plot_scatter_row(XX[1], YY[1], axarr[1], mname+': moist', 1.5)
#     plot_scatter_row(XX[2], YY[2], axarr[2], mname+': wet', 22)
#     plt.show()
```

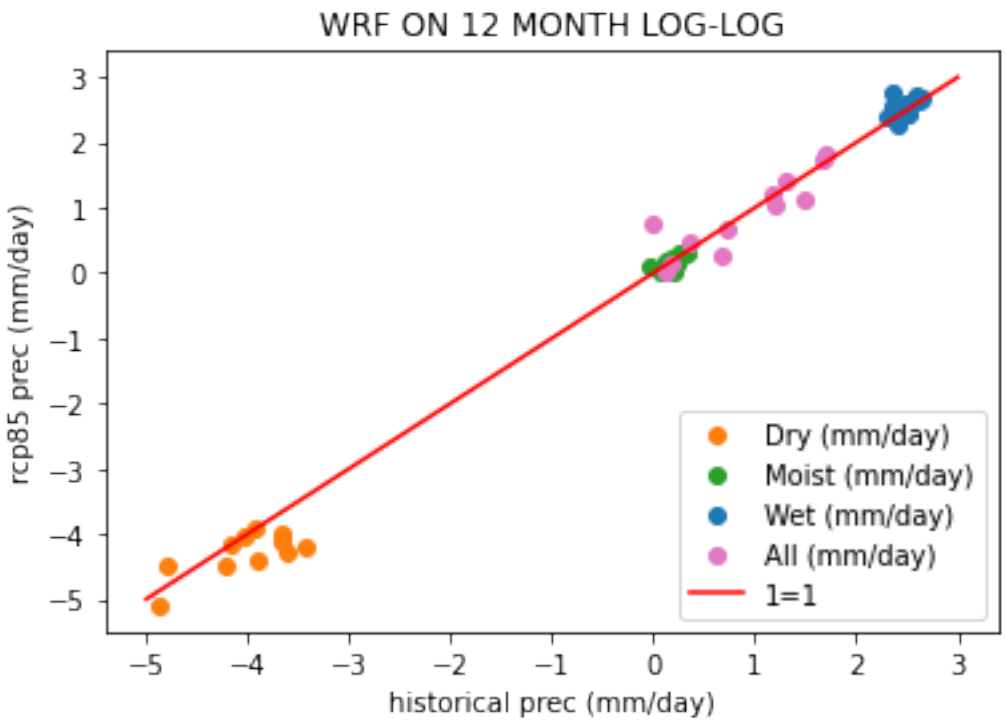
```
    # dry/moist/wet on same plot, log-log
    plot_scatter_row_3x(np.log(XX), np.log(YY), mname+' on 12 month log-log',␣
↪-5, 3)
    plt.show()
```

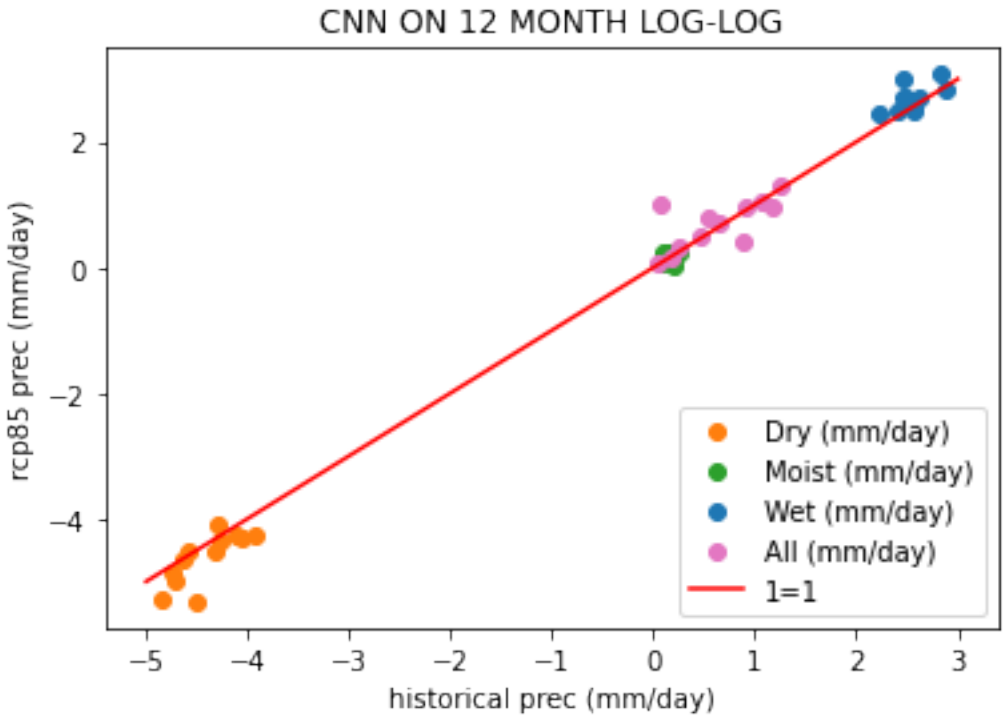
```
/glade/scratch/dkorytin/ipykernel_4001/2894838844.py:49: RuntimeWarning: divide
by zero encountered in log
  plot_scatter_row_3x(np.log(XX), np.log(YY), mname+' on 12 month log-log', -5,
3)
```

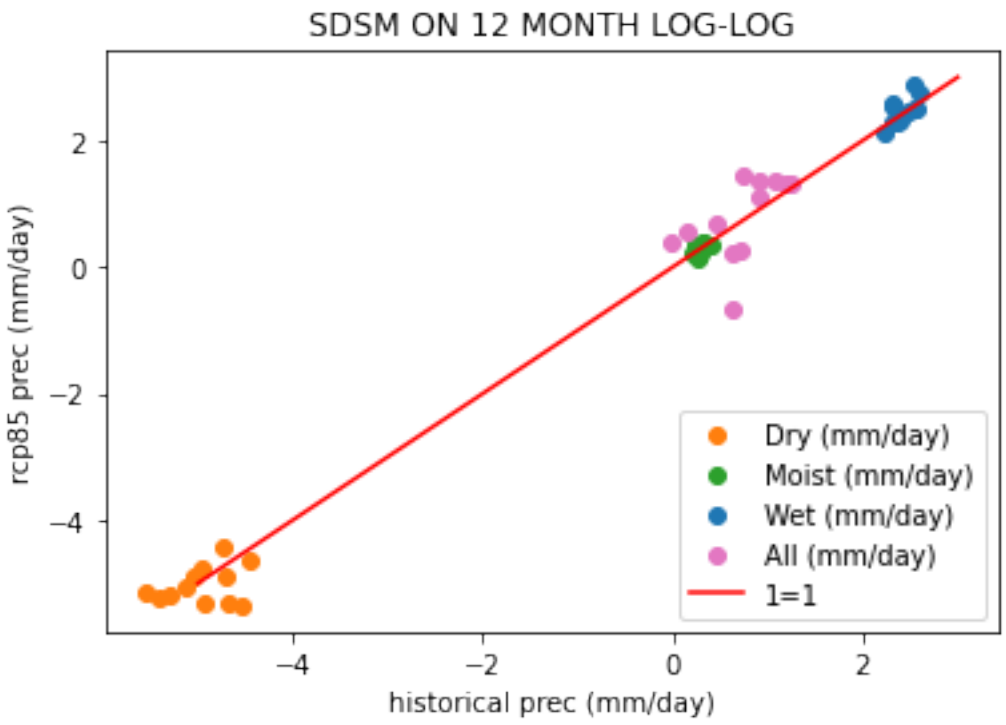


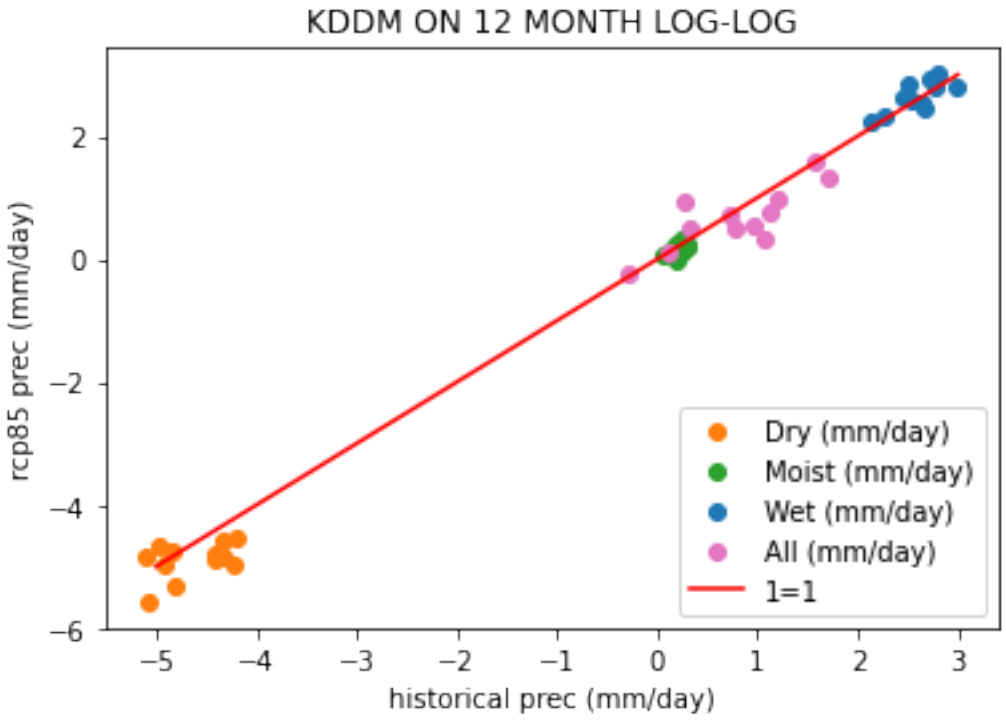

RAW ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1

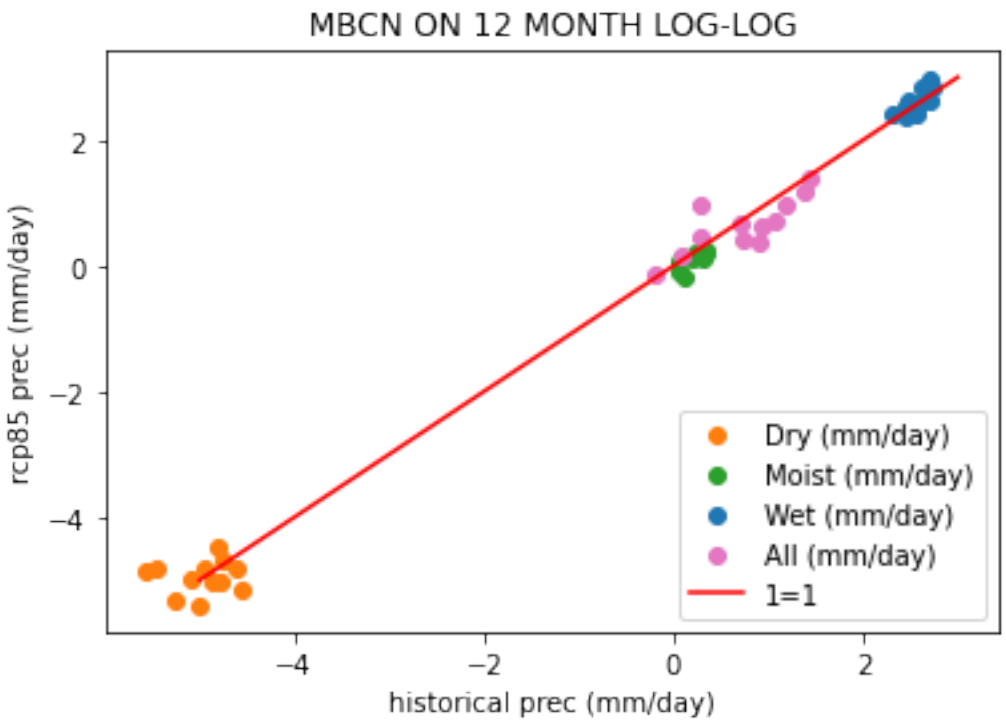

REGCM4 ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1

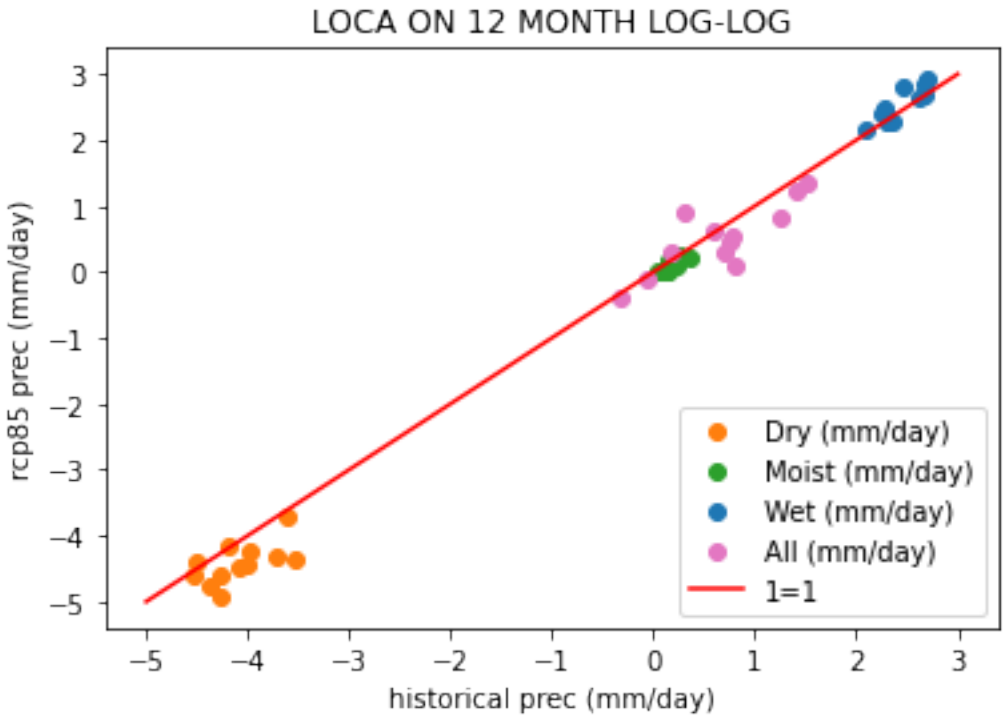

WRF ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1


CNN ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1

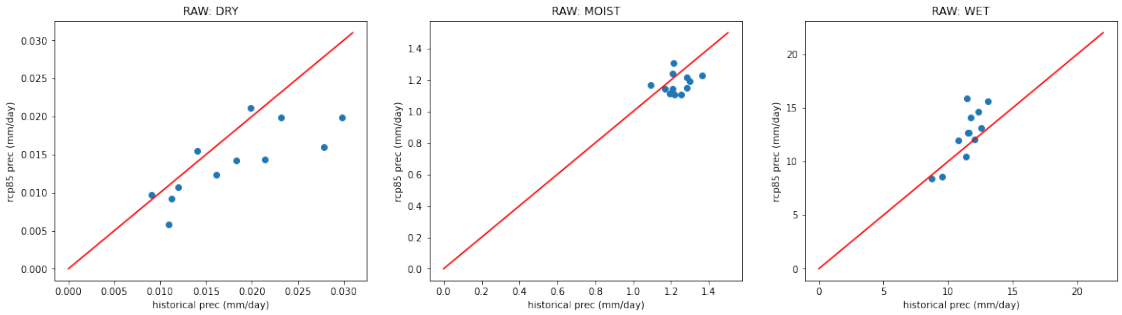

SDSM ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1


KDDM ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1


MBCN ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1


LOCA ON 12 MONTH LOG-LOG
rcp85 prec (mm/day)
historical prec (mm/day)
Dry (mm/day)
Moist (mm/day)
Wet (mm/day)
All (mm/day)
1=1

[31]:
```
# Simpler example using plot_scatter_row()
fig, axarr = plt.subplots(1, 3, figsize = (20,20/4))

# look at model at index 1
mname = names[1]
XX = AXX[1]
YY = AYY[1]

plot_scatter_row(XX[0], YY[0], axarr[0], mname+': dry', 0, .031)
plot_scatter_row(XX[1], YY[1], axarr[1], mname+': moist', 0, 1.5)
plot_scatter_row(XX[2], YY[2], axarr[2], mname+': wet', 0, 22)
#plot_scatter_row(XX[3], YY[3], axarr[3], mname+': all', 0, 22)
plt.show()
```



6.0.2 HIST vs. FUTURE

[47]:
```
# HIST vs FUTURE

#[model2absfilepath(ff, 'hist', 98, 36) for ff in␣
 ↪['obs','raw','RegCM4','WRF','MPAS','CNN','SDSM','KDDM','MBCn','LOCA']]

sigfiles_hist = [model2absfilepath(ff, 'hist', 98, 36) for ff in␣
 ↪['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']]
sigfiles_rcp85 = [model2absfilepath(ff, 'rcp85', 98, 36) for ff in␣
 ↪['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']]

# interate every model
CXX=[]; CYY=[]
for mii in range(len(sigfiles_hist)):
    sigfn1 = sigfiles_hist[mii]
    rainsignal1 = xr.open_dataset(sigfn1)['prec']

    # future doesn't have obs
    if mii > 0:
        sigfn2 = sigfiles_rcp85[mii]
```

```
        rainsignal2 = xr.open_dataset(sigfn2)['prec']


    # get averages for each of the twelve months
    X=[]; Y=[]
    for mpr_min, mpr_max in [(0,0.254), (0.254, 3.0), (3.0, 9999.0), (0, 9999.
↪0)]:
        avginput1, count1, distribution1, indices1 = calc_prec_average(mpr_min,␣
↪mpr_max, rainsignal1, 5, -9999, 9999)

        # obs doesn't have future, use hist obs for future obs
        if mii == 0:
            #avginput2 = 0
            avginput2, count2, distribution2, indices2 =␣
↪calc_prec_average(mpr_min, mpr_max, rainsignal1, 5, -9999, 9999)
        else:
            avginput2, count2, distribution2, indices2 =␣
↪calc_prec_average(mpr_min, mpr_max, rainsignal2, 5, -9999, 9999)

        X.append(float(avginput1))
        Y.append(float(avginput2))
    CXX.append(X)
    CYY.append(Y)
    print(X, Y)
```

```
[0.012228437699377537, 1.255117416381836, 13.313586235046387,
3.8706345558166504] [0.012228437699377537, 1.255117416381836,
13.313586235046387, 3.8706345558166504]
[0.029797572642564774, 1.2972118854522705, 10.812373161315918,
3.758000612258911] [0.01993877999484539, 1.194233775138855, 11.996299743652344,
3.5159120559692383]
[0.019958218559622765, 1.2237801551818848, 14.880117416381836,
3.532254695892334] [0.0240038949996233, 1.3314791917800903, 18.49971580505371,
5.100991249084473]
[0.025427045300602913, 1.385020136833191, 13.355270385742188, 5.519029140472412]
[0.018242888152599335, 1.3382331132888794, 14.992368698120117,
6.1182684898376465]
[0.01956319436430931, 1.1639140844345093, 13.66250991821289, 3.4940967559814453]
[0.014185821637511253, 1.2548773288726807, 15.089552879333496,
3.6323812007904053]
[0.010634824633598328, 1.3579655885696411, 9.94668960571289, 2.9183058738708496]
[0.0047411490231752396, 1.4592456817626953, 12.71196460723877,
3.9304232597351074]
[0.012278249487280846, 1.2112849950790405, 15.967031478881836,
4.862980842590332] [0.008136054500937462, 1.269968032836914, 18.575885772705078,
4.802936553955078]
[0.007694782689213753, 1.2700670957565308, 14.606186866760254,
```

```
4.162026882171631] [0.006710864137858152, 1.2183786630630493,
15.900899887084961, 4.111354351043701]
[0.024187644943594933, 1.1902779340744019, 13.54931926727295, 4.597790241241455]
[0.013271589763462543, 1.2461310625076294, 14.01877498626709,
3.9148566722869873]
```

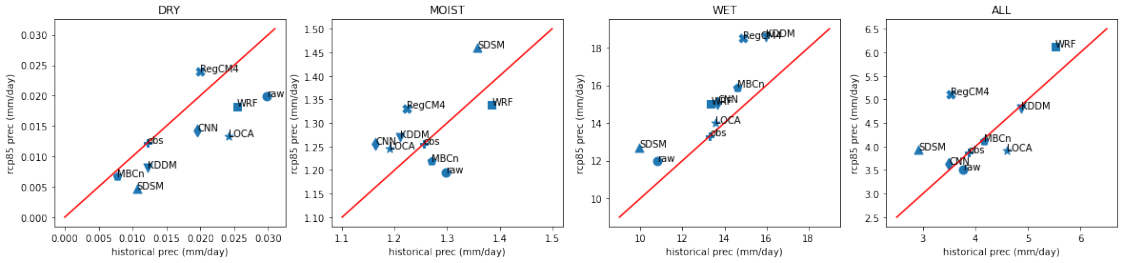
[48]:
```
np.array(CXX)[:,0].shape
```

[48]:
```
(9,)
```

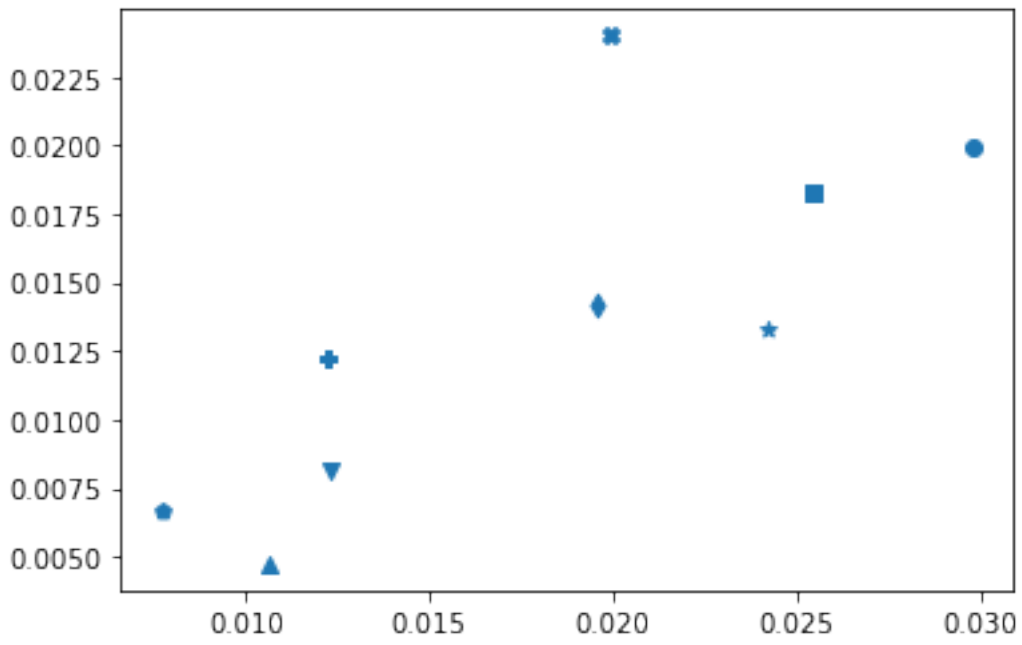
[49]:
```
np.array(CXX)[:,0]
```

[49]:
```
array([0.01222844, 0.02979757, 0.01995822, 0.02542705, 0.01956319,
       0.01063482, 0.01227825, 0.00769478, 0.02418764])
```

[80]:
```
names = ['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']

fig, axarr = plt.subplots(1, 4, figsize = (20,20/5))
mname = 'raw'
plot_scatter_row(np.array(CXX)[:,0], np.array(CYY)[:,0], axarr[0], 'Dry', 0, .
 ↪031)
for ii, txt in enumerate(names):
    axarr[0].annotate(txt, (np.array(CXX)[ii,0], np.array(CYY)[ii,0]))

plot_scatter_row(np.array(CXX)[:,1], np.array(CYY)[:,1], axarr[1], 'Moist', 1.1,␣
 ↪1.5)
for ii, txt in enumerate(names):
    axarr[1].annotate(txt, (np.array(CXX)[ii,1], np.array(CYY)[ii,1]))

plot_scatter_row(np.array(CXX)[:,2], np.array(CYY)[:,2], axarr[2], 'Wet', 9, 19)
for ii, txt in enumerate(names):
    axarr[2].annotate(txt, (np.array(CXX)[ii,2], np.array(CYY)[ii,2]))

plot_scatter_row(np.array(CXX)[:,3], np.array(CYY)[:,3], axarr[3], 'All', 2.5, 6.
 ↪5)
for ii, txt in enumerate(names):
    axarr[3].annotate(txt, (np.array(CXX)[ii,3], np.array(CYY)[ii,3]))


plt.show()
```

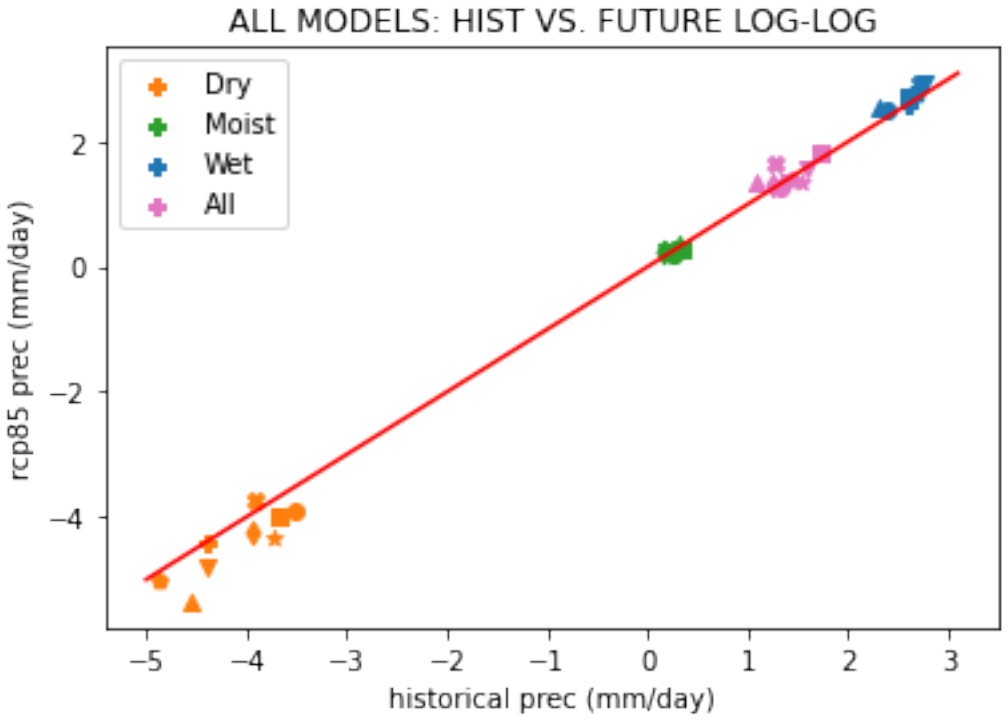


[65]:
```
# test symbols
names = ['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']
markers = ['P','o','X','s','d','^','v','p','*']

for ii, txt in enumerate(names):
    plt.scatter(np.array(CXX)[ii,0], np.array(CYY)[ii,0], marker=markers[ii],␣
 ↪color="tab:blue", label='prec (mm/day)')
# for ii, txt in enumerate(names):
#     axarr[0].annotate(txt, (np.array(CXX)[ii,0], np.array(CYY)[ii,0]))
```



[64]:
```
# dry/moist/wet all on a log-log graph
names = ['obs','raw','RegCM4','WRF','CNN','SDSM','KDDM','MBCn','LOCA']

#fig, axarr = plt.subplots(1, 3, figsize = (20,20/4))
#plot_scatter_row(np.array(XX)[:,0], np.array(YY)[:,0], axarr[0], 'Dry', .031)
plot_scatter_row_3x(np.log(CXX).swapaxes(0,1), np.log(CYY).swapaxes(0,1), 'All␣
 ↪models: hist vs. future log-log', -5, np.log(22))
plt.show()


# # label points
# for ii, txt in enumerate(names):
#     axarr[0].annotate(txt, (np.array(XX)[ii,0], np.array(YY)[ii,0]))
```

```
# for ii, txt in enumerate(names):
#     axarr[1].annotate(txt, (np.array(XX)[ii,1], np.array(YY)[ii,1]))

# for ii, txt in enumerate(names):
#     axarr[2].annotate(txt, (np.array(XX)[ii,2], np.array(YY)[ii,2]))

plt.show()
```



[164]:
```
np.array(CXX).shape
```

[164]:
```
(8, 4)
```

[ ]:
```

```