# Appendix O

# Octave code for water budget calculations

**% gen_alderRain22.m**

```
%
% Andrew James Wiebe, 5 Aug 2020
%
% Code for GNU Octave (Eaton et al., 2011).
%
% Objective: Generate synthetic 46-yr time series based on a parametric method that uses
%           mixed exponential models for the probability distributions of both spacing and amounts.
% Overview: Use mixed exponential functions to generate rainfall time series. Check sets of seven (six time-series plus
%          one observed) time series for acceptable spatial correlation (based on observed Pearson
%          Product-Moment and Spearman Rank Correlation coefficients). Save the synthetic time series from sets
%          of six that have acceptable spatial correlation, and a summary file containing a list of assignments of
%          rainfall time series to five virtual stations.
%
% Method:
% Step 1: Generate six random time series based on mixed exponential models for both spacing and daily amounts.
% Step 2: Use the Iman and Conover (1982) method (Tarpanelli et al., 2012) (the "IC method") to
%         modify correlation coefficients among the seven time series (the six random time series plus
%         the observed Roseville rainfall) to be similar to the observed Pearson and Spearman coefficients.
%         Reject any sets of time series with correlation coefficients falling outside the observed Spearman
%         Rank Correlation coefficient range.
% Step 3: Save the data for sets of five or six time series that have acceptable spatial correlation
%         coefficients. A summary file called stats_SummaryX.txt (where X is the total number of valid time series
%         generated) is saved; this file contains min_PET/Ptot and max_PET/Ptot ratios (for reference), and total
%         precipitation sums for each year (1973 to 2018) for each time series
%         (information for one time series per row). Finally, a list of all sets of valid time series
%         is saved in "assignments_VirtualStations22vX.txt" (where X = version). Each row of this file
%         lists a valid set of five or six time series. A zero is an empty placeholder if there are only
%         five valid time series in the set.
%
```

```
% Notes:
% 1. Time series that correlate similarly with Roseville compared to the SOWC field data
%    are adjusted in blocks of 3 (or 4 for the first set) years at a time, because the observed
%    field dataset was 3 years in length.
% 2. The outer "for i" loop generates 5 or 6 random time series during each successful iteration.
% 3. If this file is used more than once to generate additional time series, the "c" and "assignRow"
%    and "version" parameters should be incremented to the next numbers after the respective numbers
%    already processed. Also, the user needs to splice together (separately) the stats_Summary*.txt files
%    and the assignments_virtualStations*.txt files before running rechargeTotal22.m.
% 4. A folder called "Rnd_rainfall22" needs to be created in the folder containing this file prior
%    to running this script.
%
% References:
% Eaton, J.W., Bateman, D., Hauberg, S., 2011. GNU Octave. Edition 3 for Octave version 3.6.4. Manual for
%    high-level interactive language for numerical computations. https://www.gnu.org/software/octave/download.html.
% Iman, R.L., Conover, W.J., 1982. A distribution-free approach to inducing rank correlation among input
%    variables. Commun. Stat. Simul Comput. 11(3), 311-334. https://doi.org/10.1080/03610918208812265.
% Tarpanelli, A., Franchini, M., Brocca, L., Camici, S., Melone, F., Moramarco, T., 2012. A simple approach
%    for stochastic generation of spatial rainfall patterns. J. Hydrol 472-473, 63-76.
%    https://doi.org/10.1016/j.jhydrol.2012.09.010.

clear all;

% apply the observed range of Spearman daily spatial correlation coefficients (among WS2, WS3, ..., WS7, Roseville)
max_correl = 0.8;
min_correl = 0.5;

% 3-year block final index values for 1976, 1979, ..., 2015, 2018 (first block is 4 years)
breaks = [1461, 2556, 3652, 4748, 5844, 6939, 8035, 9131, 10227, 11322, 12418, 13514, 14610, 15705, 16801]';

%%% Load Roseville precipitation dataset for 1973 to 2018
% rville_precip = dlmread("E:\\Octave-364win7\\topic3\\Roseville_daily_rain_snow_totPrecip1973-2018.txt", " ,/:\t", 3, 0);
load Roseville_daily_rain_snow_totPrecip1973-2018.mat; % FORMAT: [year month day rainfall_mm snowfall_mm totalPrecipitation_mm]
```

```
% rville_ETo = dlmread("E:\\Octave-364win7\\topic3\\Roseville_ETo_ws1_6ms.txt", " ,/:\t", 3, 0); % Load ETo estimates
load Roseville_ETo_ws1_6ms.mat; % FORMAT: [year ETo_mm]
rville_ETo = rville_ETo([24:end],:); % select only the data for years 1973 to 2018

load sowc_minus1_Rville_24hr_correl_pearson_rev3.mat; % observed Pearson and Spearman correlation (7x7) matrices for
                                                      % WS2,WS3,WS4,WS5,WS6,WS7,Roseville

assign = zeros(3200,6);

n = 6000;
ny = 46;

statsRnd = zeros(n, 48); %FORMAT: statsRnd = [min_PET/Ptot, max_PET/Ptot, sum1973, sum1974, ..., sum2018]

select = 1;

ndays = length(rville_precip);
p_fit_gaps = [0.11600, 0.55419, 3.46333];
p_fit_depths = [0.50310, 2.09470, 13.26897];

version = 1; % ********** update by one each time this is run *********
c = 1; % ********** update with next value before re-running ***************
assignRow = 1; % ********** update with next value before re-running ***************

for i = 1:1000

   if mod(i,100) == 0
     i
   end

   select = 1;

   rain_yr = zeros(6,ny);
```

```
%%% --- Step 1: generate a set of six rainfall time series
rnd_rain = zeros(ndays,7);

for stn = 1:6
   rnd_rain(:,stn) = rndMixedExpRain(ndays, p_fit_gaps, p_fit_depths);
end

%%% --- Step 2: Calculate spatial correlation coefficients and determine number of valid time series ---
rnd_rain(:,7) = rville_precip(:,4);
rnd_rain2 = zeros(length(rnd_rain), 7);

%%% --- Compare in 3 or 4 year blocks using the indices in "breaks"
minSpearSim = 1;
maxSpearSim = 0;
minPearSim = 1;
maxPearSim = 0;

for j = 1:size(breaks,1)
   if j == 1
      [rnd_rain2([1:breaks(j,1)],:), pearSim, spearSim] = methodIC5(rnd_rain([1:breaks(j,1)],:), Rpear);
      minSpearSim = min( min(spearSim(spearSim > 0), minSpearSim));
      minPearSim = min( min(pearSim(pearSim > 0), minPearSim));
      maxSpearSim = max( max(spearSim(spearSim < 0.99), maxSpearSim));
      maxPearSim = max( max(pearSim(pearSim < 0.99), maxPearSim));
   else
      [rnd_rain2([breaks(j-1,1)+1:breaks(j,1)],:), pearSim, spearSim] = methodIC5(rnd_rain([breaks(j-1,1)+1:breaks(j,1)],:), Rpear);
      minSpearSim = min( min(spearSim(spearSim > 0), minSpearSim));
      minPearSim = min( min(pearSim(pearSim > 0), minPearSim));
      maxSpearSim = max( max(spearSim(spearSim < 0.99), maxSpearSim));
      maxPearSim = max( max(pearSim(pearSim < 0.99), maxPearSim));
   end
end

flags = ones(7);
for j = 1:6
```

```
    for k = j+1:7
        if not(and(spearSim(j,k) >= min_correl, spearSim(j,k) <= max_correl))
            flags(j,k) = 0;
            flags(k,j) = 0;
        end
    end
end

temp = sum(flags);

%%% Check correlation coefficient
if sum(temp(1,[1:6]) > 5) >= 5 % If there are at least five valid time series out of six
    printf("correlation OK\n");

    list = zeros(1,6);
    valid = zeros(length(rnd_rain2),6);

    for j = 1:6

        if sum(flags(:,j)) >= 6 %if all correlation coefficients for this time series are within the observed range

            %%% --- Calculate PET/P ratio for reference (no longer a constraint) ---
            for yr = 1973:2018
                %%%sum the simulated precip for the year "yr"
                idx3 = rville_precip(:,1) == yr;

                rain_yr(j,yr - 1972) = sum(rnd_rain2(idx3,j));

                P_tot = sum(rnd_rain2(idx3,j)) + sum(rville_precip(idx3,5)); % random rain (within correlation envelope) + Roseville snow

                PET = rville_ETo(yr - 1972,2);
                ratio(yr - 1972,1) = PET / P_tot;

            end
```

```
            maxRatio = max(ratio);
            minRatio = min(ratio);

            list(1,j) = 1; % all ratios accepted in this version

        end
    end

    %%% --- Step 3: Save lists of valid time series ---
    for j = 1:6
        if list(1,j) == 1
            statsRnd(c,1) = minRatio;
            statsRnd(c,2) = maxRatio;
            statsRnd(c,[3:48]) = rain_yr(j,:);

            filename = sprintf('Rnd_rainfall22\\rainseries%d.txt',c);
            rainseries = rnd_rain2(:,j);
            save(filename,'rainseries');

            assign(assignRow,j) = c;
            c = c + 1;
        end
    end

    assignRow
    assignRow = assignRow + 1;

else
    if and(maxSpearSim > max_correl, minSpearSim < min_correl)
        printf("%g valid time series. Correlation NOT OK: correl both too high and too low.\n", sum(temp(1,[1:6]) > 5));
    elseif maxSpearSim > max_correl
        printf("%g valid time series. Correlation NOT OK: correl too high.\n", sum(temp(1,[1:6]) > 5));
    else
        printf("%g valid time series. Correlation NOT OK: correl too low.\n", sum(temp(1,[1:6]) > 5));
    end
```

```
    end

end

numAssign = c - 1;
filename = sprintf("Rnd_rainfall22\\stats_Summary%d.txt",numAssign);
save(filename, "statsRnd");

filename = sprintf("assignments_VirtualStations22v%d.txt", version);
save(filename, "assign");

% ------------------------------------------------------------------------------------------------------------------------
```

**% rndMixedExpRain.m**

```
%
% Andrew J. Wiebe, 21 Jul 2020
%
% Code for GNU Octave (Eaton et al., 2011).
%
% Objective: generate a daily rainfall timeseries of length equal to ndays based on
%           mixed exponential models for the probability (relative frequency) distributions
%           of both the intervals between rainy days and the daily rainfall amounts.
%
% Input variables:
%   "ndays" is total number of days simulated
%   "spacingparams" is a vector of three values describing a mixed exponential model for the
%      relative frequency distribution of the intervals between days with rainfall.
%   "rainparams" is a vector of three values describing a mixed exponential model for the
%      relative frequency distribution of the intervals between days with rainfall.
%
% Output variable:
%   "rain" is a vector of zero and nonzero rainfall amounts corresponding to a sequence of ndays in length
%
% Notes:
%   The mixed exponential model is based on Li et al. (2013) equations 5a and 5b:
%      f(x) = (p/beta1) * exp(-x/beta1) + ((1-p)/beta2)*exp(-x/beta2),
%   where p represents the mixing fraction, and beta1 and beta2 are scale parameters for two different
%   exponential distributions. Constraints: 0 <= p <= 1, beta1 > 0, beta2 > 0.
%   The equation
%      rt = -bt * ln(vt)
%   was used to simulate the interval or rainfall amounts. In this case, t = time (day index in "rain" vector),
%   bt = either beta1 or beta2 (chosen based on the mixing parameter p), and vt is a uniform random number
%   between 0 and 1.
%
%   The depth generation lines ("depth =") use the natural logarithm, i.e., "log" function in Octave.
%
% References:
%   Eaton, J.W., Bateman, D., Hauberg, S., 2011. GNU Octave. Edition 3 for Octave version 3.6.4. Manual for
```

```
%    high-level interactive language for numerical computations. https://www.gnu.org/software/octave/download.html.
%   Li, Z., Brissette, F., Chen, J., 2013. Finding the most appropriate precipitation probability
%      distribution for stochastic weather generation and hydrological modelling in Nordic watersheds.
%      Hydrol. Process. 27, 3718-3729. https://doi.org/10.1002/hyp.9499.
%
function rain = rndMixedExpRain(ndays, spacingparams, rainparams)

rain=zeros(ndays,1); % initialize output vector with zeros

vt = rand(); % draw a uniformly distributed random number
temp = rand();
if temp < spacingparams(1,1)
   bt = spacingparams(1,2);
else
   bt = spacingparams(1,3);
end

intvl = ceil(-bt*log(vt));

rday = 1 + intvl; % choose first rainy day

vt = rand(); % draw a uniformly distributed random number
temp = rand();
if temp <= rainparams(1,1)
   bt = rainparams(1,2);
else
   bt = rainparams(1,3);
end

depth = -bt*log(vt); % uses natural logarithm, or "log" function in Octave
rain(rday,1) = depth; % assign a rainfall amount to the first rainy day

while rday<=ndays

            vt = rand(); % draw a uniformly distributed random number
```

```
        temp = rand();
        if temp < spacingparams(1,1)
          bt = spacingparams(1,2);
        else
          bt = spacingparams(1,3);
        end

        intvl = max(1,ceil(-bt*log(vt)));

        rday = rday + intvl; % advance the index of the current rainy day

        vt = rand(); % draw a uniformly distributed random number
        temp = rand();
        if temp <= rainparams(1,1)
          bt = rainparams(1,2);
        else
          bt = rainparams(1,3);
        end

        depth = -bt*log(vt);
        rain(rday,1) = depth; % assign a rainfall amount

end

rain = rain(1:ndays,1);

return
% ------------------------------------------------------------------------------------------------------------------
```

**% methodIC5.m**

```
% Andrew J. Wiebe, 24 Jul 2019
%
% After Tarpanelli et al. (2012)
%
% Code for GNU Octave (Eaton et al., 2011).
%
% Objective: Follow the steps in Tarpanelli et al (2012) to use the Iman
% and Conover (1982) method to generate random rainfall time series with a
% desired level of Pearson (and Spearman) correlation.
%
% Input parameters:
%    X = m by n matrix (n random time series of length m).
%    C_star = desired Pearson correlation matrix, calculated from (external) base matrix (e.g., field data).
%
% Output parameters:
%    X_star = set of n time series of length m, adjusted to exhibit a Pearson spatial correlation
%             matrix similar to C_star.
%    pear = Pearson spatial correlation coefficient matrix for X_star.
%    spear = Spearman spatial correlation coefficient matrix for X_star.
%
% The correlation matrix R of the columns of X is calculated within this function for convenience.
%
% References:
%   Eaton, J.W., Bateman, D., Hauberg, S., 2011. GNU Octave. Edition 3 for Octave version 3.6.4. Manual for
%      high-level interactive language for numerical computations. https://www.gnu.org/software/octave/download.html.
%   Iman, R.L., Conover, W.J., 1982. A distribution-free approach to inducing rank correlation among input
%      variables. Commun. Stat. Simul Comput. 11(3), 311-334. https://doi.org/10.1080/03610918208812265.
%   Tarpanelli, A., Franchini, M., Brocca, L., Camici, S., Melone, F., Moramarco, T., 2012. A simple approach
%      for stochastic generation of spatial rainfall patterns. J. Hydrol 472-473, 63-76.
%      https://doi.org/10.1016/j.jhydrol.2012.09.010.

function [X_star, pear, spear] = methodIC5(X, C_star);

%%% calculate pearson coefficient matrix for random input matrix X
```

```
n = length(C_star);
R = zeros(n,n);

for i = 1:n
   for j = i:n
      R(i,j) = corr(X(:,i), X(:,j));
      R(j,i) = R(i,j);
   end
end

Q = chol(R,'lower');

P = chol(C_star, 'lower');

X1 = X * ((P * inv(Q))');

%%% Now rearrange columns of X to have the same ranks as the columns of X1?
[X1_sort, idx1] = sort(X1,1, 'descend'); % sort the columns of X1

X_sort = sort(X,1,'descend');

for col = 1:n
   for row = 1:length(X)
      X_star(idx1(row,col),col) = X_sort(row,col);
   end
end

%%% check overall cross-correlation for random time series
for i = 1:n
   for j = i:n
      pear(i,j) = corr(X_star(:,i), X_star(:,j));
      pear(j,i) = pear(i,j);
      spear(i,j) = spearman(X_star(:,i), X_star(:,j));
      spear(j,i) = spear(i,j);
   end
```

```
end
% ------------------------------------------------------------------------------------------------------------
```

**% gen_alderRain22mix.m**

```
%
% Andrew James Wiebe, 5 Aug 2020
%
% Code for GNU Octave (Eaton et al., 2011).
%
% Objective: Generate acceptably spatially correlated combinations of synthetic 46-yr rainfall time series.
% Overview: Load sets of six rainfall time-series, adjust time series based on the Iman and Conover (1982) method (Tarpanelli et al., 2012) and check for
%           acceptable spatial correlation. Save a summary file listing new assignments of rainfall time series to five virtual stations.
%
% Method:
% Step 1: This version ("...22mix") uses previously generated time series and draws six random ones.
%       The time series were generated (gen_alderRain22.m) based on mixed exponential models for both spacing and daily amounts.
% Step 2: Use the Iman and Conover (1982) method (Tarpanelli et al., 2012) (the "IC method") to modify correlation coefficients among the seven time series
%       (the six random time series plus the observed Roseville rainfall) to be similar to the observed Pearson and Spearman coefficients. Reject any sets of time
%       series with correlation coefficients falling outside the observed Spearman rank correlation coefficient range.
% Step 3: Save the list of five or six time series that have acceptable spatial correlation coefficients. The list of time series is saved in
%       "assignments_VirtualStations22vXmixY.txt" (where X = version from gen_alderRain22.m, and Y is the number of times (adjusted manually) that
%       the current .m file has been run), where each row of the assignments file contains a valid set of five or six time series label numbers. A zero is an empty
%       placeholder if there are only five valid time series in the set.
%
% Notes:
% 1. Time series that correlate similarly with Roseville compared to the SOWC field data
%   are adjusted in blocks of 3 (or 4 for the first set) years at a time, because the observed
%   field dataset was 3 years in length.
% 2. The outer "for i" loop generates 5 or 6 random time series during each successful iteration.
% 3. The "assignRow" parameter should be one greater than the number of valid sets of time series already
%   generated using gen_alderRain22.m (number of rows of time series in "assignments_VirtualStations22vX.txt").
% 3. If this file is used more than once to generate additional time series, "assignRow"
%   parameter should be incremented to be one greater than the total number of valid sets of time series
%   already identified. Also, the user needs to splice together the "assignments_virtualStations*.txt" files
%   before running rechargeTotal22.m.
% 4. It is assumed that a folder called "Rnd_rainfall22" has been created in the folder containing this file prior
%   to running this script, and that it contains the files generated by gen_alderRain22.m.
%
```

```
% References:
% Eaton, J.W., Bateman, D., Hauberg, S., 2011. GNU Octave. Edition 3 for Octave version 3.6.4. Manual for
%    high-level interactive language for numerical computations. https://www.gnu.org/software/octave/download.html.
% Iman, R.L., Conover, W.J., 1982. A distribution-free approach to inducing rank correlation among input
%    variables. Commun. Stat. Simul Comput. 11(3), 311-334. https://doi.org/10.1080/03610918208812265.
% Tarpanelli, A., Franchini, M., Brocca, L., Camici, S., Melone, F., Moramarco, T., 2012. A simple approach
%    for stochastic generation of spatial rainfall patterns. J. Hydrol 472-473, 63-76.
%    https://doi.org/10.1016/j.jhydrol.2012.09.010.

clear all;

statsRnd = dlmread("E:\\Octave-364win7\\topic3\\Rnd_rainfall22\\stats_Summary19761compiled.txt", " ,/:\t", 5, 0);

% apply the observed range of Spearman daily spatial correlation coefficients (among WS2, WS3, ..., WS7, Roseville)
max_correl = 0.8;
min_correl = 0.5;

% 3-year block final index values for 1976, 1979, ..., 2015, 2018 (first block is 4 years)
breaks = [1461, 2556, 3652, 4748, 5844, 6939, 8035, 9131, 10227, 11322, 12418, 13514, 14610, 15705, 16801]';

%%% Load Roseville precipitation dataset for 1973 to 2018
% rville_precip = dlmread("E:\\Octave-364win7\\topic3\\Roseville_daily_rain_snow_totPrecip1973-2018.txt", " ,/:\t", 3, 0);
load Roseville_daily_rain_snow_totPrecip1973-2018.mat; % FORMAT: [year month day rainfall_mm snowfall_mm totalPrecipitation_mm]

% rville_ETo = dlmread("E:\\Octave-364win7\\topic3\\Roseville_ETo_ws1_6ms.txt", " ,/:\t", 3, 0); % Load ETo estimates
load Roseville_ETo_ws1_6ms.mat; % FORMAT: [year ETo_mm]
rville_ETo = rville_ETo([24:end],:); % select only the data for years 1973 to 2018

load sowc_minus1_Rville_24hr_correl_pearson_rev3.mat; % observed Pearson and Spearman correlation (7x7) matrices for
WS2,WS3,WS4,WS5,WS6,WS7,Roseville

assign = zeros(10000,6);

n = 10000;
ny = 46;
```

```
% statsRnd = zeros(n, 48); %statsRnd = [min_PET/Ptot, max_PET/Ptot, sum1973, sum1974, ..., sum2018]

select = 1;

ndays = length(rville_precip);

n_ts = 19761; % ********** this is based on the number of rainfall time series generated by gen_alderRain22.m and stays the same for mix *********

version = 5; % ********** this should be the same as the last version used in gen_alderRain22.m and stays the same for mix *********
c = 1; % ********** this can be 1 for mix ***************
assignRow = 5514; % ********** start with the largest number of sets of time series from gen_alderRain22.m and
%%%                                        update with next value before re-running ***************

for i = 1:1000

   if mod(i,100) == 0
      i
   end

   select = 1;

   rain_yr = zeros(6,ny);

   %%% --- Step 1: choose six rainfall time series of those already generated
   rnd_rain = zeros(ndays,7);
   series_list = zeros(1,6);


   for stn = 1:6
      rndSeriesNo = floor(rand() * n_ts) + 1;
      filename = sprintf('Rnd_rainfall22\\rainseries%d.txt',rndSeriesNo);
      load(filename);
      rnd_rain(:,stn) = rainseries;
     series_list(1,stn) = rndSeriesNo;
```

```
end

%%% --- Step 2: Calculate spatial correlation coefficients and determine number of valid time series ---
rnd_rain(:,7) = rville_precip(:,4);
rnd_rain2 = zeros(length(rnd_rain), 7);

%%% --- Compare in 3 or 4 year blocks using the indices in "breaks"
minSpearSim = 1;
maxSpearSim = 0;
minPearSim = 1;
maxPearSim = 0;

for j = 1:size(breaks,1)
   if j == 1
      [rnd_rain2([1:breaks(j,1)],:), pearSim, spearSim] = methodIC6(rnd_rain([1:breaks(j,1)],:), Rpear);
      if not(sum(sum(pearSim)) == 0)
         minSpearSim = min( min(spearSim(spearSim > 0), minSpearSim));
         minPearSim = min( min(pearSim(pearSim > 0), minPearSim));
         maxSpearSim = max( max(spearSim(spearSim < 0.99), maxSpearSim));
         maxPearSim = max( max(pearSim(pearSim < 0.99), maxPearSim));
      else
         minPearSim = 0;
         maxPearSim = 0;
         minSpearSim = 0;
         maxSpearSim = 0;
      end
   else
      [rnd_rain2([breaks(j-1,1)+1:breaks(j,1)],:), pearSim, spearSim] = methodIC6(rnd_rain([breaks(j-1,1)+1:breaks(j,1)],:), Rpear);

     if not(sum(sum(pearSim)) == 0)
        minSpearSim = min( min(spearSim(spearSim > 0), minSpearSim));
        minPearSim = min( min(pearSim(pearSim > 0), minPearSim));
        maxSpearSim = max( max(spearSim(spearSim < 0.99), maxSpearSim));
        maxPearSim = max( max(pearSim(pearSim < 0.99), maxPearSim));
      else
```

```
            minPearSim = 0;
            maxPearSim = 0;
            minSpearSim = 0;
            maxSpearSim = 0;
        end
    end
end

flags = ones(7);
for j = 1:6
    for k = j+1:7
        if not(and(spearSim(j,k) >= min_correl, spearSim(j,k) <= max_correl))
            flags(j,k) = 0;
            flags(k,j) = 0;
        end
    end
end

temp = sum(flags);

%%% Check correlation coefficient
if sum(temp(1,[1:6]) > 5) >= 5 % If there are at least five valid time series out of six
    printf("correlation OK\n");

    list = zeros(1,6);
    valid = zeros(length(rnd_rain2),6);

    for j = 1:6

        if sum(flags(:,j)) >= 6 %if all correlation coefficients for this time series are within the observed range

            % --- Calculate PET/P ratio for reference (no longer a constraint) ---
            for yr = 1973:2018
                %%%sum the simulated precip for the year "yr"
                idx3 = rville_precip(:,1) == yr;
```

```
            rain_yr(j,yr - 1972) = sum(rnd_rain2(idx3,j));

            P_tot = sum(rnd_rain2(idx3,j)) + sum(rville_precip(idx3,5)); % random rain (within correlation envelope) + Roseville snow

            PET = rville_ETo(yr - 1972,2);
             ratio(yr - 1972,1) = PET / P_tot;

         end

         maxRatio = max(ratio);
         minRatio = min(ratio);

         list(1,j) = 1; % all ratios accepted in this version

      end
   end

   %%% --- Step 3: Save lists of valid time series ---
   for j = 1:6
      if list(1,j) == 1
         assign(assignRow,j) = series_list(1,j);
         c = c + 1;
      end
   end

   assignRow
   assignRow = assignRow + 1;

else
   if and(maxSpearSim > max_correl, minSpearSim < min_correl)
      printf("%g valid time series. Correlation NOT OK: correl both too high and too low.\n", sum(temp(1,[1:6]) > 5));
   elseif maxSpearSim > max_correl
      printf("%g valid time series. Correlation NOT OK: correl too high.\n", sum(temp(1,[1:6]) > 5));
   else
```

```
            printf("%g valid time series. Correlation NOT OK: correl too low.\n", sum(temp(1,[1:6]) > 5));
        end
    end
end

filename = sprintf("assignments_VirtualStations22v%dmix%d.txt", version,mixversion);
save(filename, "assign");
% ------------------------------------------------------------------------------------------------------------------------
```

**% methodIC6.m**

```
%
% Andrew J. Wiebe, 5 Aug 2020
%
% Code for GNU Octave (Eaton et al., 2011).
%
% After Tarpanelli et al. (2012, JofH)
%
% Objective: Follow the steps in Tarpanelli et al. (2012) to use the Iman and Conover (1982)
%           method to generate random rainfall time series with a desired level of
%           Pearson and Spearman correlation.
%
% Notes:
% The correlation matrix R for the columns of X is calculated within this function for convenience.
% Returns three matrices of zeros if Pearson correlation matrix for X is not positive definite according to a determinant check.
% The determinant check does not catch all non-positive definite matrices. If an error occurs related to the chol() function,
%   save assignment files manually, update numbers (version, c, assignRow) as necessary in gen_alderRain22.m or gen_alderRain22mix.m,
%   and restart the gen_alderRain22*.m file.
%
% Input parameters:
%   X = m by n matrix (n random time series of length m).
%   C_star = desired Pearson correlation matrix, calculated from (external) base matrix (e.g., field data).
%
% Output parameters:
%   X_star = set of n time series of length m, adjusted to exhibit a Pearson spatial correlation
%            matrix similar to C_star.
%   pear = Pearson spatial correlation coefficient matrix for X_star.
%   spear = Spearman spatial correlation coefficient matrix for X_star.
%
% References:
% Eaton, J.W., Bateman, D., Hauberg, S., 2011. GNU Octave. Edition 3 for Octave version 3.6.4. Manual for high-level interactive language for numerical
%     computations. https://www.gnu.org/software/octave/download.html.
% Iman, R.L., Conover, W.J., 1982. A distribution-free approach to inducing rank correlation among input variables. Commun. Stat. Simul Comput. 11(3),
%     311-334. https://doi.org/10.1080/03610918208812265.
% Mathuranathan, 2013. Check positive definite matrix in Matlab. GaussianWaves: Signal Processing Simplified.
```

```
%      Blog post. 27 May 2013. https://www.gaussianwaves.com/2013/05/check-positive-definite-matrix-in-matlab/.
% Tarpanelli, A., Franchini, M., Brocca, L., Camici, S., Melone, F., Moramarco, T., 2012. A simple approach for stochastic generation of spatial rainfall patterns.
%       J. Hydrol 472-473, 63-76. https://doi.org/10.1016/j.jhydrol.2012.09.010.

function [X_star, pear, spear] = methodIC6(X, C_star);

posDef = 1; % if the correlation matrix for X is positive definite, posDef = 1, otherwise posDef = 0

%%% calculate pearson coefficient matrix for random input matrix X
n = length(C_star);
R = zeros(n,n);

for i = 1:n
   for j = i:n
      R(i,j) = corr(X(:,i), X(:,j));
      R(j,i) = R(i,j);
   end
end

i = 2;
done = 0;
while and(i < n, !done)
   %%% Check if positive definite
   %%% Idea from: Mathuranathan (2013), https://www.gaussianwaves.com/2013/05/check-positive-definite-matrix-in-matlab/
   subMatrix = R(1:i,1:i);
   if det(subMatrix) <= 0
      posDef = 0;
      done = 1;
   end
   i = i + 1;
end

if posDef == 1

   Q = chol(R,'lower');
```

```
    P = chol(C_star, 'lower');

    X1 = X * ((P * inv(Q))');
    %%% Now rearrange columns of X to have the same ranks as the columns of X1
    [X1_sort, idx1] = sort(X1,1, 'descend'); % sort the columns of X1

    X_sort = sort(X,1,'descend');

    for col = 1:n
       for row = 1:length(X)
          X_star(idx1(row,col),col) = X_sort(row,col);
       end
    end

    %%% Check overall cross-correlation for random time series
    for i = 1:n
       for j = i:n
          pear(i,j) = corr(X_star(:,i), X_star(:,j));
          pear(j,i) = pear(i,j);
          spear(i,j) = spearman(X_star(:,i), X_star(:,j));
          spear(j,i) = spear(i,j);
       end
    end
else
    X_star = zeros(length(X),n);
    pear = zeros(n,n);
    spear = zeros(n,n);
end
% ------------------------------------------------------------------------------------------------------------------------
```

**% rechargeTotal22.m**

```
%
% Andrew J. Wiebe, 04 Sep 2020
%
% Code for GNU Octave (Eaton et al., 2011).
%
% Overall Objective: Read in random rainfall time series, generate random AET/P estimates,
%                    and calculate the recharge for the Alder Creek watershed
%                    for each realization based on a stochastic annual vadose zone water budget.
%
% Method:
% Use annual streamflow and BFI values derived from the Water Survey of Canada gauge within the watershed (WSC, 2019).
% Assume uniform snowfall across the watershed and use the annual values from the Environment Canada
%      weather station at Roseville (Government of Canada, 2019)
% Calculate the ratio of AET / P from the Budyko curve (Budyko, 1961; Gentine et al., 2012), with or without random perturbation within
%      observed annual US MOPEX (Duan et al., 2006) variation.
%
% This script calculates:
%           1) the Thiessen polygon, weighted average total precipitation from three virtual stations
%              (after adding Roseville snowfall to the annual rain sums at the virtual stations),
%           2) an estimate of the AET/P ratio based on the Budyko curve and observed variability in
%              45 US MOPEX watersheds (Duan et al., 2006) where 0.8 <= PET/P <= 0.9 (for long term average PET and P), and
%           3) an estimate of total recharge to the watershed over n = 46 years.
%
% References:
% Barlow, P.M., Cunningham, W.L., Zhai, T., and Gray, M., 2015. U.S. Geological Survey Groundwater Toolbox,
%    a graphical and mapping interface for analysis of hydrologic data (version 1.0) – User guide for
%    estimation of base flow, runoff, and groundwater recharge from streamflow data. U.S. Geological Survey
%    Techniques and Methods, book 3, chap. B10, 27 p. https://doi.org/10.3133/tm3B10.
% Budyko, M., 1961. The heat balance of the Earth's surface. Natl. Weather Serv., U.S. Dep. of Commer.,
%    Washington, D.C., USA.
% Eaton, J.W., Bateman, D., Hauberg, S., 2011. GNU Octave. Edition 3 for Octave version 3.6.4. Manual for
%    high-level interactive language for numerical computations.
%    https://www.gnu.org/software/octave/download.html.
% Gentine, P., D'Odorico, P., Lintner, B.R., Sivandran, G., Salvucci, G., 2012. Interdependence of climate,
```

```
%    soil, and vegetation as constrained by the Budyko curve. Geophys. Res. Lett. 39, L19404.
%    https://doi.org/10.1029/2012GL053492.
% Government of Canada, 2019. Historical data: Rainfall, snowfall, and air temperature data for the Roseville,
%    ON, weather station data [computer files].
%    http://climate.weather.gc.ca/historical_data/search_historic_data_e.html.
% Matrix Solutions Inc. (Matrix), S.S. Papadopulos and Associates Inc. (SSPA), 2014b. Region of Waterloo Tier Three Water Budget
%    and Local Area Risk Assessment. Final Report, Sep. 2014. Prepared for: Region of Waterloo.
%    https://www.sourcewater.ca/en/source-protection-areas/resources/Documents/Grand/RMOW-September-2014-WQRA_chpt-1-10.pdf.
% Roberson, W., 2017. remove rows with all zeros. MATLAB Answers.
%    https://www.mathworks.com/matlabcentral/answers/40390-remove-rows-with-all-zeros.
% Water Survey of Canada (WSC), 2019. Daily volumetric flow rates for the New Dundee gauging
%    station (02GA030) for 1965 to 2018. https://wateroffice.ec.gc.ca/mainmenu/historical_data_index_e.html.
%
clear all; clc;

window = 400; % window size for checking percentage change in metrics with a set of realizations
converge = 0.1; % convergence threshold in percent (0.1% = 0.001)
convergence = 0; % this variable will change to 1 if convergence occurs

aet_option = 2; % 1 -- budyko curve directly; 2 -- budykoRNDnormal (normally distributed about Budyko curve)

%%% FORMAT: statsRnd = [min_PET/Ptot, max_PET/Ptot, sum1973, sum1974, ..., sum2018]
statsRnd = dlmread("E:\\Octave-364win7\\topic3\\Rnd_rainfall22\\stats_Summary19761compiled.txt", " ,/:\t", 5, 0);
statsRnd = statsRnd(any(statsRnd,2) != 0,:); % remove zero timestamp rows (Roberson, 2017)

%%% assign contains a list of rows in statsRnd that relate to each of the five virtual rain gauges
assign = dlmread("E:\\Octave-364win7\\topic3\\assignments_VirtualStations22v5mix17compiled.txt", " ,/:\t", 5, 0);
assign = assign(any(assign,2) != 0,:); % remove zero timestamp rows (Roberson, 2017)

%%% use ETo calculations with average wind speed = 1.6 m/s, from the SOWC stations
rville_ETo = dlmread("E:\\Octave-364win7\\topic3\\Roseville_ETo_ws1_6ms.txt", " ,/:\t", 3, 0); % FORMAT: [year ETo_mm]
pet_annual = rville_ETo([24:end],2);  % select only the data for years 1973 to 2018
pet_avg = sum(pet_annual) / length(pet_annual);

load rville_precip; % FORMAT: [year month day rainfall_mm snowfall_mm totalPrecipitation_mm]
```

```
for yr = 1973:2018
   idx2 = rville_precip(:,1) == yr;
   rville_snow(yr - 1972,1) = sum(rville_precip(idx2,5));
end

bfi = 0.56; % average based on analysis with PART in USGS Groundwater Toolbox (Barlow et al., 2015)
bfi_results = dlmread("E:\\Octave-364win7\\topic3\\Alder_PART_BFI_results.txt", " ,/:\t", 3, 0); FORMAT: bfi_results = [year bfi]
bfi_annual = bfi_results(:,2); % select only the annual values
Q_tot_avg = 216; % 216 mm is based on streamflow, VZ, and SZ water budgets using average precip estimate of 900 mm from Govt of Canada (2019),
%                        Budyko curve AET/P estimate, and Qpumping estimate from Matrix and SSPA (2014b); see Appendix Q
Qscalingfactor = 216 / 140.5;
Q_sw_fraction = (1- bfi) * Q_tot_avg;
Q_tot_annual_results = dlmread("E:\\Octave-364win7\\topic3\\WSC_Qannual.txt", " ,/:\t", 2, 0);
Q_tot_annual = Q_tot_annual_results(:,2);

n = 46;

nrealz = length(assign);

%%% Exclude AET from the saturated zone via a correction factor derived from applying PET to areas mapped as bog/swamp/wetland/marsh/open water
vzAET_factor = 0.91; %0.89; % 0.93; % see Appendix P

recharge = zeros(nrealz,8); %FORMAT: recharge = [row1 row2 row3 row4 row5 p_ws_avg, aet_p, recharge]

percent_avgrch_change = zeros(nrealz,1);

convStats = zeros(nrealz, 4); % FORMAT = [avg_recharge_avg_i, percent_diff_wrt_avg_i, percent_diff_mass_loadings_change(i,1), factor_change(i,1)];

q_K26_well = 6756 * 365; % pumping volume in m3

for i = 1:nrealz

   p_list = zeros(n, 5);
```

```
%%% Read each row; pick the first five valid time series of the six
temp = assign(i,:)';
temp = temp(any(temp,2) != 0,:); % remove zero timestamp rows (Roberson, 2017)

p_list(:,1) = statsRnd(temp(1,1), [3:48])' + rville_snow;
% p_list(:,2) = statsRnd(temp(2,1), [3:48])' + rville_snow;
p_list(:,3) = statsRnd(temp(3,1), [3:48])' + rville_snow;
% p_list(:,4) = statsRnd(temp(4,1), [3:48])' + rville_snow;
p_list(:,5) = statsRnd(temp(5,1), [3:48])' + rville_snow;

p_ws_avg_annual = calcWSavgPrecip357annual(p_list(:,[1,3,5])); % 3 Thiessen polygons; will now be a list of 46 amounts
p_ws_avg = sum(p_ws_avg_annual) / n;

if aet_option == 1
   aet_p = vzAET_factor * budyko(pet_avg, p_ws_avg);
   aet_p_annual = ones(n,1) * aet_p;
elseif aet_option == 2
   for j = 1:n
      aet_p_annual(j,1) = vzAET_factor * budykoRNDnormalMOPEX(pet_annual(j,1), p_ws_avg_annual(j,1));
   end

   %%% save the annual AET values for the current realization to a file
   %filename = sprintf("Rnd_aet_p22\\aet_p_annual_%d.txt", i);
   %save(filename, "aet_p_annual");

   aet_p = sum(aet_p_annual) / n;

end

recharge(i,[1:5]) = temp([1:5],1)';
recharge(i,6) = p_ws_avg;
recharge(i,7) = aet_p;

%%% use scaling factor to convert WSC total streamflow to ~watershed total streamflow
%%% use WSC BFI values
```

```
recharge_star_annual = p_ws_avg_annual - (p_ws_avg_annual .* aet_p_annual) - Qscalingfactor * (1 - bfi_annual) .* Q_tot_annual;
recharge(i,8) = sum(recharge_star_annual);

if i >= 2 * window  % quantify change in metrics (avg change in value with additional 'window' realizations)
   recharge_avg_list_i = recharge([1:i],8) / n;
   recharge_avg_list_prev = recharge([1:(i-window)],8) / n;

   quan0025_avg_recharge_i = quantile(recharge_avg_list_i,[0.025]);
   quan0975_avg_recharge_i = quantile(recharge_avg_list_i,[0.975]);
   quan0025_avg_recharge_prev = quantile(recharge_avg_list_prev,[0.025]);
   quan0975_avg_recharge_prev = quantile(recharge_avg_list_prev,[0.975]);

   avg_recharge_avg_i = sum(recharge_avg_list_i) / i;
   avg_recharge_avg_prev = sum(recharge_avg_list_prev) / (i-window);

   %%% metric = change when additional 'window' averages considered
   percent_avgrch_change(i,1) = 100*(avg_recharge_avg_prev - avg_recharge_avg_i) / avg_recharge_avg_prev;

   %%% metric = cap zone area change
   area1_K26 = q_K26_well / (quan0025_avg_recharge_i / 1000); % area in m2
   area2_K26 = q_K26_well / (quan0975_avg_recharge_i / 1000); % area in m2
   areaK26avg = q_K26_well / (avg_recharge_avg_i / 1000); % area in m2
   areaDiff_K26 = max(abs(area1_K26 - areaK26avg), abs(areaK26avg - area2_K26)) / 1e6;   % difference in sq. km
   areaK26_avg = areaK26avg / 1e6;
   percent_diff_wrt_avg_i = 100 * areaDiff_K26 / areaK26_avg;

   area1_K26 = q_K26_well / (quan0025_avg_recharge_prev / 1000); % area in m2
   area2_K26 = q_K26_well / (quan0975_avg_recharge_prev / 1000); % area in m2
   areaK26avg = q_K26_well / (avg_recharge_avg_prev / 1000); % area in m2
   areaDiff_K26 = max(abs(area1_K26 - areaK26avg), abs(areaK26avg - area2_K26)) / 1e6;   % difference in sq. km
   areaK26_avg = areaK26avg / 1e6;
   percent_diff_wrt_avg_prev = 100 * areaDiff_K26 / areaK26_avg;

   percent_capzoneareachange_change(i,1) = 100*(percent_diff_wrt_avg_prev - percent_diff_wrt_avg_i)/percent_diff_wrt_avg_prev;
```

```
    %%% metric = mass loadings change
    percent_diff_mass_loadings_i = 100 * max(abs(quan0975_avg_recharge_i - avg_recharge_avg_i), ...
       abs(quan0025_avg_recharge_i - avg_recharge_avg_i)) / avg_recharge_avg_i;
    percent_diff_mass_loadings_prev = 100 * max(abs(quan0975_avg_recharge_prev - avg_recharge_avg_prev), ...
       abs(quan0025_avg_recharge_prev - avg_recharge_avg_prev)) / avg_recharge_avg_prev;
    percent_diff_mass_loadings_change(i,1) = 100 * (percent_diff_mass_loadings_prev - percent_diff_mass_loadings_i)/percent_diff_mass_loadings_prev;

    %%% metric = max cumulative recharge / min cumulative recharge
    factor_i = quantile(recharge([1:i],8),[0.975]) / quantile(recharge([1:i],8),[0.025]);
    factor_prev = quantile(recharge([1:(i-window)],8),[0.975]) / quantile(recharge([1:(i-window)],8),[0.025]);
    factor_change(i,1) = 100*(factor_prev - factor_i)/factor_prev;

    convStats(i,:) = [avg_recharge_avg_i, percent_diff_wrt_avg_i, percent_diff_mass_loadings_i, factor_i];

  end

  if or(mod(i,500) == 0, i == nrealz)
    i
  end

  if and(i >= 2 * window, i > 5000, not(convergence))
    %%% Determine whether average change in each metric is less than convergence criterion
    conv1 = sum(abs(percent_avgrch_change([i-window+1]:i,1)))/window < converge;
    conv2 = sum(abs(percent_capzoneareachange_change([i-window+1]:i,1)))/window < converge; % not OK after 10328 realizations
    conv3 = sum(abs(percent_diff_mass_loadings_change([i-window+1]:i,1)))/window < converge; % not OK after 10328 realizations
    conv4 = sum(abs(factor_change([i-window+1]:i,1)))/window < converge;

    if and(conv1,conv2,conv3,conv4)
      printf("All metrics < %f%% after %d iterations.\n",converge,i);
      convergence = i;
    end
  end

end
```

```
%%% --- RESULTS ---------------------------------------------------------------

vzAET_factor100 = 100 * vzAET_factor;

if aet_option == 1
   filename = sprintf("rechargeTotal22_357_results_no_scatter_aetvzfactor0_%d.txt", vzAET_factor100);
   diary(filename);
elseif aet_option == 2
   filename = sprintf("rechargeTotal22_357_results_scatter_aetvzfactor0_%d.txt", vzAET_factor100);
   diary(filename);
end

nrealz
aet_option
convcheck = [conv1 conv2 conv3 conv4]
if and(conv1,conv2,conv3,conv4)
   printf("All metrics < %f%% after %d iterations.\n",converge, convergence);
   convergence = i;
end

bfi
Q_tot_avg

p_ws_avg_overall = sum(recharge(:,6)) / nrealz
stdev_p_ws_avg = std(recharge(:,6));

aet_p_avg = sum(recharge(:,7)) / nrealz
stdev_aet_p = std(recharge(:,7))

recharge_avg = recharge(:,8) / n;
avg_recharge_avg = sum(recharge_avg) / nrealz
stdev = std(recharge_avg)

min_avg_recharge = min(recharge_avg)
```

```
quan0025_avg_recharge = quantile(recharge_avg,[0.025])
quan0975_avg_recharge = quantile(recharge_avg,[0.975])
max_avg_recharge = max(recharge_avg)

%%% --- Mass Loadings ---

printf("\n");
worst_case_percent_diff_mass_loadings_wrt_avg = 100 * max(abs(quan0975_avg_recharge - avg_recharge_avg), abs(quan0025_avg_recharge -
avg_recharge_avg)) / avg_recharge_avg
printf("\n");

%%% --- Well Capture Zone ---

area1_K26 = q_K26_well / (quan0025_avg_recharge / 1000); % area in m2
area2_K26 = q_K26_well / (quan0975_avg_recharge / 1000); % area in m2
areaK26avg = q_K26_well / (avg_recharge_avg / 1000); % area in m2
areaDiff_K26 = max(area1_K26 - areaK26avg, areaK26avg - area2_K26) / 1e6   % difference in sq. km

areaK26_avg = areaK26avg / 1e6

printf("\n");
percent_diff_wrt_avg = 100 * areaDiff_K26 / areaK26_avg
printf("\n");

%%% --- Cumulative Recharge ---

dif_cum_tot = quantile(recharge(:,8),[0.975]) - quantile(recharge(:,8),[0.025]); % difference in mm
dif_cum_tot_avg = dif_cum_tot / n  % average difference per year in mm
printf("\n");

factor_cum_quan0975_div_by_cum_quan0025 = quantile(recharge(:,8),[0.975]) / quantile(recharge(:,8),[0.025])
printf("\n");
diary off;
```

```
if aet_option == 1
    filename = sprintf("rechargeTotal22_357_budyko_no_scatter_aetvzfactor0_%d.txt", vzAET_factor100);
    save(filename, "recharge");
elseif aet_option == 2
    filename = sprintf("rechargeTotal22_357_budyko_scatter_aetvzfactor0_%d.txt", vzAET_factor100);
    save(filename, "recharge");
end

filename = sprintf("convergenceStats22_0_%d_aetopt%d.txt", vzAET_factor100, aet_option);
save(filename, "convStats");

% ------------------------------------------------------------------------------------------------------------------------
```

**% calcWSavgPrecip357annual.m**

```
%
% Andrew J. Wiebe, 18 Aug 2020
%
% Code for GNU Octave (Eaton et al., 2011).
%
% Objective: Calculate the average precipitation over the Alder Creek watershed from three
%            virtual stations by weighting with their representative Thiessen polygon areas.
% Input parameters:
%    totP3 = a matrix where each column represents one station and has n years of total precipitation
%            sums (random rainfall plus observed Roseville snowfall). Assume 3 stations (i.e., 3 columns in totP3).
% Output parameter:
%    p = list of average annual watershed total precipitation amounts, weighted by subwatershed area.
%
% Reference:
% Eaton, J.W., Bateman, D., Hauberg, S., 2011. GNU Octave. Edition 3 for Octave version 3.6.4. Manual for high-level interactive language for numerical
%     computations. https://www.gnu.org/software/octave/download.html.

function p = calcWSavgPrecip357annual(totP3)

  area = [25.85297449;...
          45.61051953;...
          6.596530245;...
  ];

  n = length(totP3);

  p = zeros(n,1); % annual average precip estimate, via weighted (by area) average

  for i = 1:n
    p(i,1) = area' * totP3(i,:)' / sum(area);
  end

end
% ------------------------------------------------------------------------------------------------------------------------
```

**% budyko.m**

```
%
% Andrew J. Wiebe, 18 Jan 2019
%
% Calculate a watershed's average ratio of AET / P (i.e., actual evapotranspiration (AET) divided by precipitation)
%    given PET (potential ET) and P (total precipitation, i.e., rain plus snow).
% Uses the Budyko curve equation by Gentine et al. (2012).
%
% Input parameters:
%    pet = the long-term average PET estimate (mm/unit watershed area)
%    p   = the long term average total precipitation for the watershed
%
% Output:
%     aet_p is the estimate of the ratio of AET (avg) to P (avg) based on the Budyko curve.
%
% Reference:
% Gentine, P., D'Odorico, P., Lintner, B.R., Sivandran, G., Salvucci, G., 2012. Interdependence of climate,
%    soil, and vegetation as constrained by the Budyko curve. Geophys. Res. Lett. 39, L19404.
%    https://doi.org/10.1029/2012GL053492.

function aet_p = budyko(pet, p)

   phi = pet/p;

   aet_p = sqrt(phi * tanh(1/phi) * (1 - exp(-phi)));

end

% ------------------------------------------------------------------------------------------------------------------------
```

**% budykoRNDnormalMOPEX.m**

```
%
% Andrew J. Wiebe, 9 Jul 2020
%
% Generate a watershed's annual ratio of AET / P (i.e., actual evapotranspiration (ET) divided by precipitation) given PET (annual potential ET) and P
%     (annual total precipitation, i.e., rain plus snow).
% Add some randomness using the 'normrnd' function, where the mean value is the Budyko curve estimate (Gentine et al., 2012) and the standard deviation
%     is based on the scatter of annual MOPEX points about the Budyko curve for 45 watersheds where 0.8 <= PET/P <= 0.9 (where PET and P were the
%     long-term averages when considering this range; calculated from the US MOPEX dataset – Duan et al., 2006).
%
% Input parameters:
%    pet = potential evapotranspiration for a watershed
%    p = total precipitation for a watershed
%
% Output:
%     aet_p is an estimate of the ratio of AET (avg) to P (avg) based on the Budyko curve and observed variability.
%
% References:
% Gentine, P., D'Odorico, P., Lintner, B.R., Sivandran, G., Salvucci, G., 2012. Interdependence of climate, soil, and vegetation as constrained by the Budyko
%     curve. Geophys. Res. Lett. 39, L19404. https://doi.org/10.1029/2012GL053492.
% Duan, Q., et al., 2006. Model Parameter Estimation Experiment (MOPEX): An overview of science strategy and major results from the second and third
%     workshops. J. Hydrol. 320, 3-17. https://doi.org/10.1016/j.jhydrol.2005.07.031.

function aet_p = budykoRNDnormalMOPEX(pet, p)

   phi = pet/p;
   aet_p_curve = sqrt(phi * tanh(1/phi) * (1 - exp(-phi)));

   sigma = 0.14; % use maximum observed standard deviation from 45 US MOPEX watersheds, where 0.8 <= PET/P <= 0.9 for long-term data

   aet_p = normrnd(aet_p_curve, sigma, 1, 1); % generate one random value based on the normal distribution
end
% ------------------------------------------------------------------------------------------------------------------------
```