# 6. Yearly Aggregation Post Processor

## 6.1 Introduction

The Yearly Aggregation Post Processor (**YAPP**) manipulates or processes data resulting from a RiverWare™ model run.

This data must be in the RiverWare™ data format (i.e. rdf) which can be generated by the DMI Selector, or Output Management options located under the RiverWare™ utilities menu.

Processing a RiverWare™ data format (.rdf) file as input from a monthly timestep model, **YAPP** aggregates the data to yearly according to user-specified methods. The **YAPP** then writes the yearly data back to an (.rdf) file.

The **YAPP** is specific, taking only monthly data and subsequently aggregating it to yearly. Therefore, it will not work with other timesteps.

The **YAPP** is a command line tool and is available for the Solaris and the Windows platforms. In Windows it must be run from a command prompt, typically available from **Start ➥ Programs ➥ Accessories ➥ Command Prompt**.

## 6.2 Limitations

The **YAPP** does not support the aggregation of table slots. Therefore, the control file will not accept "Object.Slot.Column: method" as a valid entry.

On the other hand, the **YAPP** can handle aggregate object names, such as: "CoRivMohaveToHavasu:RRHavasuNWR.Outflow: Sum".

The **YAPP** will only aggregate monthly data to yearly. Other timesteps are not supported by the **YAPP**.

The control file syntax is similar to that for the dmi control file used in RiverWare™. Nevertheless, in the dmi, slots are set by priority where a specific declaration of a slot will take precedence over a more general declaration made with an object type, or a wildcard. In contrast, the **YAPP** fixes the method at the first identification of the slot.

## 6.3 Where to Find This Program

The **YAPP** is not packaged with the RiverWare™ release. It is available on the download page of the CADSWES web site. If you FTP to CADSWES, it can also be found in the pub/releases/yearlyagg directory.

If you have trouble finding the **YAPP** or have other questions, please contact RiverWare support at riverware-support@colorado.edu.

### 6.4 Usage Overview

The executable is called with command line arguments in the following patterns:

- yearlyagg -i file1 -o file2 -c file3 [-l file4] [-y number]
- yearlyagg -h
- yearlyagg -v

The syntax shown in the second and third bullets print help information and the version number, respectively.

**In the first bullet:**

-i file1 represents the input file name,

-o file 2 represents the output file name,

-c file3 represents the control file name,

[-l file4] represents an optional log file name,

and [-y number] represents an optional month (0 to 12) that the user wants the year to end with.

These arguments are explained in detail in the next section.

### 6.5 Command Line Argument

The user controls the behavior of the **YAPP** utilizing arguments in the command line, which in turn call to the **YAPP**'s executable. The following sections explain the purpose and requirements for the arguments used in calling the yearlyagg executable.

#### 6.5.1 Input File

The input file argument is -i file1 as shown **HERE (Section 6.4)**. This is a required argument to the **YAPP**.

If the file is in the same directory as the executable, -i file1 represents the input file argument. However, if the file is in another directory, -i file1 represents a valid path, plus the input file argument.

The input file must be in a RiverWare™ data format (i.e. .rdf) that contains output from a monthly timestep model within RiverWare™. As data inputs into the **YAPP**, the monthly values can then be aggregated to yearly values by the program.

#### 6.5.2 Output File

The output file argument is -o file2 as shown **HERE (Section 6.4)**. This is a required argument to the **YAPP**.

If the file is in the same directory as the executable, -o file2 represents the output file argument. However, if the file is in another directory, -o file2 represents a valid path, plus the output file argument.

The output file from the **YAPP** will be in a RiverWare™ data format (i.e. .rdf) The resulting data will be aggregated to a yearly timestep.

In the output file, the slot header information for each slot will have a line inserted that shows the method by which the data was aggregated to yearly.

### 6.5.3 Control File

The control file argument is -c file3 as shown **HERE (Section 6.4)**. This is a required argument to the **YAPP**.

If the file is in the same directory as the executable, -c file3 represents the control file argument. However, if the file is in another directory, -c file3 represents a valid path, plus the control file argument.

The control file is created by the user. It is a series of lines to indicate which slots to process, and the method for annualizing the data for each slot.

The input file can contain information for other slots that are not specified in the control file. The **YAPP** will simply ignore this additional slot data.

The control file uses one line in the following syntax to specify the aggregation for a slot:

- Object.Slot: method
  - Object is the object name as it appears in RiverWare™,
  - Slot is the slot name i,
  - Method is one of the following:
    - **Sum** - adds all of the monthly values for the year. Result is Nan if any input values are Nan.
    - **Average** - is the average of the monthly values for the year. Result is Nan if any input values are Nan.
    - **Max** - is the maximum of the monthly values for the year. Result is Nan if any input values are Nan.
    - **Min** - is the minimum of the monthly values for the year. Result is Nan if any input values are Nan.
    - **End** - takes the last monthly value in the year. Result is Nan if any input values are Nan.
    - **Begin** -takes the first monthly value in the year. Result is Nan if any input values are Nan.
    - **Salinity** - does a flow-weighted yearly average of salinity for a slot containing salt concentrations. Result is Nan if any input values are Nan.
    - **SumNan -** adds all of the nonNan monthly values for the year and assigns that result.
    - **AverageNan** - sums all of the nonNan monthly values for the year and divides by (12) twelve.
    - **MaxNan** - ignores Nan values and generates the maximum of the nonNan values.
    - **MinNan** - ignores Nan values and generates the minimum of the nonNan values.
    - **EndNan** - reports the last value of the year, even if Nan values are present in the input data.
    - **BeginNan** - reports the first value of the year, even if Nan values are present in the input data.

Wildcards can be used for both the Object and Slot name in the control file. The first example below specifies a slot (ex. Inflow) on any object. The second specification would be for all slots on a particular object.

- *.Slot: method
- Object.*: method

The object type name can also be used in the control file for the Object name to specify slots for all objects of that type:

- LevelPowerReservoir.Slot: method

The same slot for an object can potentially be identified more than once through specific declarations and/or wildcards. The **YAPP**'s behavior is to set the method for a slot the first time it is identified, and never change the method at subsequent identifications.

### 6.5.4 Log File

The log file argument is -1 file3 as shown **HERE (Section 6.4)**. This is an optional argument to the **YAPP**.

If the file is in the same directory as the executable, -1 file3 represents the log file argument. However, if the file is in another directory, -1 file3 represents a valid path, plus the log file argument.

The **YAPP** will write entries to the log file in order to report what is happening as the program executes. This log file can be useful to check if everything completed successfully, or to track down a problem.

If the user does not include a log file argument to the **YAPP**, by default, the log information is written to a file named yearlyagg.log in the same directory as the executable.

### 6.5.5 Year End Specification

The year end specification is -y number as shown **HERE (Section 6.4)**. This is an optional argument where the user can specify the month which the year should end on for aggregating the monthly data.

The number has to be in the range 1 to 12 to specify the month for year end. For example, to aggregate data for a water year beginning on October 1, this argument would be -y 9 to show that September is the last month of the aggregation year. If the user does not include a year end argument in the call to yearlyagg, by default, the **YAPP** will select 12 to aggregate data according to a calendar year ending in December.