# Appendix E: Forecast Error and Compact Rules

The individual rules in the URGWOM ruleset are discussed separately and are grouped based on policy group. The two sections in this appendix are associated with Forecast Error and Rio Grande Compact Policy Groups respectively. The rules fire in reverse order within URGWOM, so the discussion of the policy groups begins with the Forecast Error Policy Group and then moves to the Rio Grande Compact rules.

The discussion for each rule includes a description of the rule including an explanation of the Rule Logic, a list of slots in the model associated with the rule, a log of when and how the rule was last modified, a view of the actual RiverWare rule code, a list of Execution Constraints, and a list of functions referenced in the rule.  These items are stored within the ruleset itself and this appendix is generated automatically from that ruleset so that updates to rule documentation can be easily added whenever a rule is updated.

## Control Display Icon1 RPL Object IconForecastErrors

The rules in this policy group are used to compute a percent forecast error for each month with reference to estimated inflows to El Vado Reservoir. If the rules are turned on, the computed forecast error is then used to incorporate uncertainty in forecasted flows within a simulation. The computed forecast error, not to exceed input maximums, is referenced in other rules including the calculation of a forecasted Otowi flow volume. These rules have been turned off for recent URGWOM applications and is meant to be only turned on for Planning applications, so that there is not perfect knowledge of inflows into the system during model runs.

Rules in the Group:  
ForecastErrorPercent  
CalculatedForecastError  
RewindRandomFile

Policy Group Change Log (newest changes at the top):  
Date: Who. What  
2/13/2019: Jesse Roach. Added Group Description and Group Notes

### Control Display Icon1.1 RPL Object IconRewindRandomFile

Rule Purpose:  
This rule rewinds the random number file. The seed for the random number generator is constant so that simulations can be regenerated if needed. Refer to the discussion of the predefined ResetRanDev function in the RiverWare online help for further discussion of the random number generator.

Rule Logic: Execution Constraint logic is at end of explanation.  
This rule calls the predefined function ResetRanDev. The function is imbedded in a Print statement, so note that it will not execute if the diagnostics are turned off.

This rule fires only at the starting timestep  
Comment:  
Jesse Roach 2/12/2019. I don't think October 31, 1983 needs to be skipped, guessing instead that this rule is just a direct copy of the example use of the rule in the documentation.

Model slots written by rule:  
None

List of key model objects with slots read by the rule or child functions:  
None

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description and notes fields.

Statements

Statements

Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconResetRanDev

### Control Display Icon1.2 RPL Object IconCalculatedForecastError

Rule Purpose:  
This rule is set up to set a randomly generated forecast error within a maximum range. This rule computes a forecast error using a random number generator. The forecast error is then used to compute a percent forecast error.

Rule Logic: Execution Constraint logic is at end of explanation.  
If the current timestep is the first day of a month and the forecast error for the end of the current month is a NaN as set to the ForecastError time series slot in the ForecastData data object, the rule fires. The ComputeForecastError function in the ComputedForecastError utility group is used to compute the forecast error using different methods depending on the month. Up to May, the error is computed as a function of input coefficients, the estimated inflow to El Vado Reservoir, the previous forecast error, and a random number. The forecast error in June is set to half the forecast error for May, and the forecast error in July is set to a quarter of the forecast error for June. During the remaining months, it is set to zero. Within the rule, the results from the ComputeForecastError function are then checked against input maximum forecast errors for each month for El Vado Reservoir with consideration for the sign of the error. If the maximum is exceeded, the forecast error is reset to the maximum with consideration for the sign.

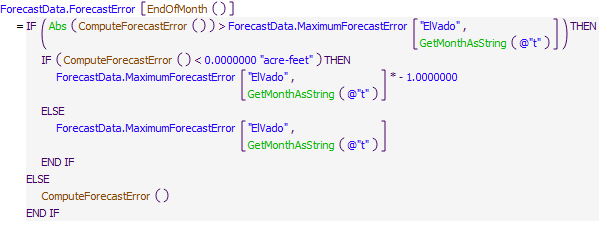
This rule fires at the start of each month if there is no end-of-month Forecast Error already assigned  
Comment:  
Jesse Roach: 2/12/19:  
The rule takes advantage of the fact that according to RPL documentation, functions with no arguments are actually evaluated only once per rule and return this same result on each function call during the execution of that block. Otherwise this rule would need a WITH function to make sure the forecast error was the same in all calls to the ComputeForecastError function.  
Previous comment "Random numbers are currently from a file generated by an old CRSS function." Unsure if that is still true.

Model slots written by rule:  
1. ForecastData.ForecastError

List of key model objects with slots read by the rule or child functions: 1. ForecastData.MaximumForecastError  
2. ForecastData.ForecastError (value in previous month)  
3. ForecastData.ForecastCoefficients

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description and notes fields.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconComputeForecastError
* RPL Object IconEndOfMonth
* RPL Object IconIsFirstTimestepOfMonth
* RPL Object IconAbs
* RPL Object IconGetMonthAsString

### Control Display Icon1.3 RPL Object IconForecastErrorPercent

Rule Purpose:  
This rule computes a percent forecast error from the forecast error calculated with the CalculatedForecastError Rule. The percent forecast error is used later when forecasting river flows at Otowi or inflows to Abiquiu Reservoir.

Rule Logic: Execution Constraint logic is at end of explanation.  
The percent forecast error is computed differently depending on the month. Through July, the percent forecast error is the lesser of the forecast error divided by the estimated inflow to El Vado Reservoir and the input maximum percent forecast error with consideration for the sign. If the estimated inflow to El Vado Reservoir is zero, the percent forecast error is set to 0.10. During other months, if it is the first year of the simulation, the percent forecast error is set to 0.04, and if it is not the first year of the simulation, the percent forecast error is set to the lesser of 0.1 or the computed percent forecast error for July of the current year.

This rule fires at the start of each month if there is no end-of-month Forecast Error already assigned  
Comment:  
Jesse Roach 2/12/2019: If this rule is used again, add a WITH for EstimateElVadoInflow, and perhaps redo logic to only calculate the percentage once rather than twice.

Model slots written by rule:  
1. ForecastData.PercentForecastError

List of key model objects with slots read by the rule or child functions:  
1. ForecastData.MaxPercentForecastError  
2. ForecastData.ForecastError  
3. ForecastData.PercentForecastError (for previous months)

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description and notes fields.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconEstimateElVadoInflow
* RPL Object IconEndOfMonth
* RPL Object IconIsFirstTimestepOfMonth
* RPL Object IconMin
* RPL Object IconMax
* RPL Object IconGetMonth
* RPL Object IconGetYear

## Control Display Icon2 RPL Object IconRio Grande Compact Accounting

### Control Display Icon2.1 RPL Object IconComputeUsableStorage

Rule Purpose:  
This rule computes the usable storage to be referenced by the SetCompactArticleVIISwitch rule when identifying whether the stipulations of Article VII of the Compact are in effect. Usable storage is computed as the total storage at Elephant Butte and Caballo Reservoirs minus any credit water for New Mexico and Colorado and minus San Juan-Chama Project water including water in the Albuquerque, Santa Fe City, Reclamation, and Combined accounts. Note that the storage in the Compact credit accounts is only subtracted if the account storage is positive and based on the credit amount as of December 31st of the previous year (i.e. any tracked Compact debt as negative account storage is not considered in the calculation and any year-to-date evaporative losses to the Compact accounts is not subtracted). Also, the usable storage is immediately adjusted for any relinquished credit (i.e. the transfer of water from the NMCredit account to Rio Grande storage as a result of relinquished credits). The result is recorded to a series slot that can be reviewed from any simulation.

Rule Logic: Execution Constraint logic is at end of explanation.  
The value for the UsableStorage time series slot in the RioGrandeCompact data object for the current timestep is set to the value for the Storage at Elephant Butte Reservoir at the end of the previous timestep minus the storage in the NMCredit and COCredit storage accounts at Elephant Butte Reservoir on December 31st of the previous year (or the initial timestep if that is more recent), if the corresponding Compact credit account storage on that date is greater than 0.0 acre-ft as assured with the predefined Max function. Also, any relinquished credit in the RelinquishedNMCredits series slot in the RelinquishedCreditsEmergencyDroughtWater data object is added if the current timestep is after the relinquishment date as identified with the user-defined RelinquishmentDate function. Any San Juan-Chama Project storage at the previous timestep is subtracted as computed with reference to the Albuquerque, SantaFeCity, Combined, and Reclamation storage accounts at the previous timestep. The value for the storage at Caballo Reservoir at the previous timestep is also added.

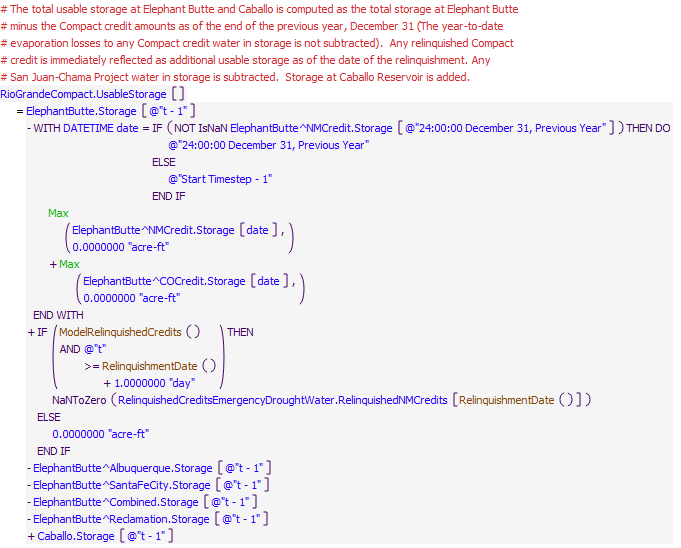
The rule fires if the UsableStorage value is not defined

Model slots written by rule:  
1. RioGrandeCompact.UsableStorage

List of key model objects with slots read by the rule or child functions:  
1. ElephantButte (Level Power Reservoir Object)  
2. Caballo (Storage Reservoir Object)  
3. RelinquishedCreditsEmergencyDroughtWater (Data Object)

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description and notes fields.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconRelinquishmentDate
* RPL Object IconModelRelinquishedCredits
* RPL Object IconMax

### Control Display Icon2.2 RPL Object IconSetCompactArticleVIISwitch

Rule Purpose:  
This rule sets a flag to indicate whether Article VII of the Compact is in effect for the current timestep. Article VII is in effect if the usable storage at Elephant Butte and Caballo Reservoir is less than 400,000 acre-ft. Article VII is particularly significant because native water cannot be stored in any of the reservoirs upstream of Elephant Butte currently modeled in URGWOM if Article VII is in effect.

Rule Logic: Execution Constraint logic is at end of explanation.  
If the value for the UsableStorage time series slot in the RioGrandeCompact data object for the previous timestep is less than the minimum storage input to the MinUsableStorageToAvoidArticleVIIRestrictions table slot in the RioGrandeCompact data object (400,000 acre-ft), the value for the switch is set to 1.0. Otherwise, the value is set to zero. The result is recorded to the ArticleVIISwitch time series slot in the RioGrandeCompact data object for the current timestep. The input minimum storage is referenced with the user-defined CompactMinStorage function. An identical assignment statement is included to be applied at the Start Timestep for specifically setting the value for the Initial Timestep if a value was not input.

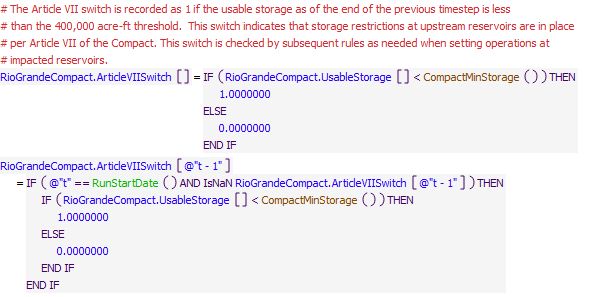
This rule fires if the value in the RioGrandeCompact.ArticleVIISwitch time series slot is a NaN for the current timestep.

Model slots written by rule:  
1. RioGrandeCompact.ArticleVIISwitch

List of key model objects with slots read by the rule or child functions:  
1. RioGrandeCompact (Data Object)

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description and notes fields.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconCompactMinStorage
* RPL Object IconRunStartDate

### Control Display Icon2.3 RPL Object IconSetAvgNambeFallsDataForRGCompactCalcs

Rule Purpose:  
This rule uses synthetic values for the San Juan-Chama use above Otowi and the Return Flow Credit Pojoaque Unit as needed for Compact calculations if values were not directly input. Warning messages are displayed in diagnostics reminding the model user that it is preferable for data to be estimated and imported, so that averages would not be required.

Rule Logic: Execution Constraint logic is at end of explanation.  
This rule sets the UseAboveOtowiPeriodic and ReturnFlowCreditPojoaqueUnitPeriodic times series slots on the RioGrandeCompact data object equal to the values in the UseAboveOtowi and ReturnFlowCreditPojoaqueUnit periodic slots, respectively, on the RioGrandeCompact data object, if these time series slots haven't already been input. The assignments are made from the start timestep through end of the run.  
This rule also prints warning messages to diagnostics if these two time series slots haven't already been input, warning that they were "NOT directly input, so average monthly uniform flows will be used."  
This rule also prints a warning message to diagnostics if the Gage Inflow at Otowi hasn't been input back to the beginning of the year, warning that "Not all data is available to sum back to beginning of year for Rio Grande Compact numbers".

This rule fires on the first timestep of the run if the UseAboveOtowi and Return Flow CreditPojoaqueUnit slots contain NaNs

Comment:  
2/12/2019 Jesse Roach: This rule is really an initialization rule, but left here because it relates to the RG Compact.  
This rule uses synthetic values for the San Juan-Chama use above Otowi and the Return Flow Credit Pojoaque Unit as needed for Compact calculations if values were not directly input. It is preferred that data be estimated and imported or Nambe computations be included in future versions of the model, so that averages would not be required.

Model slots written by rule:  
1. RioGrandeCompact.UseAboveOtowi  
2. RioGrandeCompact.ReturnFlowCreditPojoaqueUnit

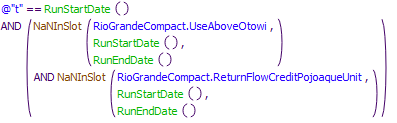
List of key model objects with slots read by the rule or child functions:  
1. Otowi.Gage Inflow  
2. RioGrandeCompact.UseAboveOtowiPeriodic  
3. RioGrandeCompact.ReturnFlowCreditPojoaqueUnitPeriodic

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description and notes fields.  
11/10/17: Updated during timestep generalization of MRG rules

Statements



Execution Constraint



Referenced Functions

* RPL Object IconNaNInSlot
* RPL Object IconFirstTimestepOfYear
* RPL Object IconUseAverageMessage
* RPL Object IconRunStartDate
* RPL Object IconRunEndDate

### Control Display Icon2.4 RPL Object IconZeroCreditsForElephantButteSpill

Rule Purpose:  
This rule executes if Elephant Butte is spilling as indicated by RioGrandeCompact.EBSpill series slot. If a spill is occurring, accounting supplies are set to reset the New Mexico and Colorado Compact credits to zero.

Rule Logic: Execution Constraint logic is at end of explanation.  
This rule has four assignments. In the first assignment, if not already computed, the NMCreditElephantButteToRioGrandeElephantButte.Supply on the previous timestep is set equal to the previous storage in the NMCredit account on Elephant Butte if the storage in this account was larger than or equal to 0 acre-feet. Otherwise, the supply is set equal to 0 cfs.  
In the second assignment, if not already computed, the RioGrandeElephantButteToNMCreditElephantButte.Supply on the previous timestep is set equal to the negative of the previous storage in the NMCredit account on Elephant Butte if the storage in this account was smaller than 0 acre-feet. Otherwise, the supply is set equal to 0 cfs.  
In the third assignment, if not already computed, the COCreditElephantButteToRioGrandeElephantButte.Supply on the previous timestep is set equal to the previous storage in the COCredit account on Elephant Butte if the storage in this account was larger than or equal to 0 acre-feet. Otherwise, the supply is set equal to 0 cfs.  
In the fourth assignment, if not already computed, the RioGrandeElephantButteToCOCreditElephantButte.Supply opn the previous timestep is set equal to the negative of the previous storage in the COCredit account on Elephant Butte if the storage in this account was smaller than 0 acre-feet. Otherwise, the supply is set equal to 0 cfs.

This rule fires if the Elephant Butte Spill Switch is set to 1 AND the timestep is at or after the beginning timestep for rulebased simulation as input by the model user to the RulebasedSimulationStartDate scalar slot in the ModelRunTypeTriggers data object and as identified with the user-defined GetStartDate function.

Model supplies written by rule:  
1. NMCreditElephantButteToRioGrandeElephantButte.Supply  
2. RioGrandeElephantButteToNMCreditElephantButte.Supply  
3. COCreditElephantButteToRioGrandeElephantButte.Supply  
4. RioGrandeElephantButteToCOCreditElephantButte.Supply

List of key model objects with slots read by the rule or child functions:  
1. ElephantButte  
2. RioGrandeCompact

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Updated description and notes fields.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconVolumeToFlow
* RPL Object IconGetStartDate
* RPL Object IconPreviousAccountStorage

### Control Display Icon2.5 RPL Object IconCompactDebitAccountingOnElVado

Rule Purpose:  
If Compact Debit storage on El Vado is enabled and there is a NM Debit on Dec 31st, the volume of the debit will be transferred out of all the Rio Grande accounts on El Vado (excluding P&P), pro-rata, and limited to the volume available in these accounts, into the CompactDebit account on El Vado, limited to a user-input maximum total transfer for the year.

Rule Logic: Execution Constraint logic is at end of explanation.  
If the previous timestep was December 31st, if the "ElephantButte^NMCredit.Storage" was less than 0 acre-feet on this date, if the sum of the storages in the "MRGCDDrought", "SupplementalESA", "MRGCDoutofArticleVII" and "RioGrande" accounts on El Vado is larger than 0 acre-feet, and if "RioGrandeCompact.EnableCompactDebitStorageOnElVado" is set to 1, then the negative of the "ElephantButte^NMCredit.Storage", minus what has already been transferred into the CompactDebit accounts on ElVado and Abiquiu year-to-date, is transferred from those four accounts on El Vado into the CompactDebit account on El Vado, pro-rata according to the storage in each of those four accounts, limited to the storage in the four accounts, and limited to the value in the scalar slot RioGrandeCompact.MaxCompactDebitStorageElVado. Otherwise, these supplies are set to 0 cfs.

The rule first fires for all timesteps at or after the beginning timestep for rulebased simulation as input by the model user to the RulebasedSimulationStartDate scalar slot in the ModelRunTypeTriggers data object and as identified with the user-defined GetStartDate function.

Model supplies written by rule:  
1. RioGrandeElVadoToCompactDebitElVado.Supply  
2. MRGCDDroughtElVadoToCompactDebitElVado.Supply  
3. SupplementalESAElVadoToCompactDebitElVado.Supply  
4. MRGCDoutofArticleVIIElVadoToCompactDebitElVado.Supply

List of key model objects with slots read by the rule or child functions:  
1. Abiquiu  
2. ElVado  
3. ElephantButte  
4. RioGrandeCompact

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
6/23/2023: Nick Mander - Hydros, edited rule and description, to limit compact debit storage in El Vado  
9/19/2022: Nick Mander - Hydros, added a reference to the new "RioGrandeAbiquiuToCompactDebitAbiquiu.Supply"  
12/02/2020: Nick Mander - Hydros, corrected assignment/computation for MRGCDOutofArticleVIIElVadoToCompactDebitElVado.supply (fromSupplementalESAElVadoToCompactDebitElVado.supply) incorrectly modified in a previous ruleset update (09-28-20).  
2/12/19: Jesse Roach. Updated description and notes fields.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconFirstTimestepOfYear
* RPL Object IconMin
* RPL Object IconMax
* RPL Object IconSum
* RPL Object IconSumFlowsToVolumeSkipNaN
* RPL Object IconHasFlag
* RPL Object IconVolumeToFlow
* RPL Object IconHasRuleFiredSuccessfully
* RPL Object IconGetStartDate

### Control Display Icon2.6 RPL Object IconCompactDebitAccountingOnAbiquiu

Rule Purpose:  
If Compact Debit storage on Abiquiu is enabled and there is a NM Debit on Dec 31st, the volume of the debit will be transferred out of the Rio Grande account on Abiquiu into the CompactDebit account on Abiquiu, limited to the volume available in the RioGrande account and a user-input maximum total transfer for the year.

Rule Logic: Execution Constraint logic is at end of explanation.  
If the previous timestep was December 31st, if the "ElephantButte^NMCredit.Storage" was less than 0 acre-feet on this date, if the storage in the "RioGrande" account on Abiquiu is larger than 0 acre-feet, and if "RioGrandeCompact.EnableCompactDebitStorageOnAbiquiu" is set to 1, then the negative of the "ElephantButte^NMCredit.Storage", minus what has already been transferred into the CompactDebit accounts on ElVado and Abiquiu year-to-date, is transferred from the RioGrande account on Abiquiu into the CompactDebit account on Abiquiu, limited to the storage in the RioGrande account and limited to the value in the scalar slot RioGrandeCompact.MaxCompactDebitStorageAbiquiu. Otherwise, the supply is set to 0 cfs.

The rule first fires for all timesteps at or after the beginning timestep for rulebased simulation as input by the model user to the RulebasedSimulationStartDate scalar slot in the ModelRunTypeTriggers data object and as identified with the user-defined GetStartDate function.

Model supplies written by rule:  
1. RioGrandeAbiquiuToCompactDebitAbiquiu.Supply

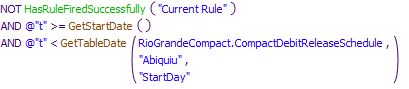
List of key model objects with slots read by the rule or child functions:  
1. Abiquiu  
2. ElVado  
3. ElephantButte  
4. RioGrandeCompact

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
6/23/2023: Nick Mander - Hydros, edited rule and description, to limit compact debit storage in Abiquiu  
9/19/2022: Nick Mander - Hydros, created rule.

Statements



Execution Constraint



Referenced Functions

* RPL Object IconGetTableDate
* RPL Object IconFirstTimestepOfYear
* RPL Object IconMin
* RPL Object IconMax
* RPL Object IconSum
* RPL Object IconSumFlowsToVolumeSkipNaN
* RPL Object IconHasFlag
* RPL Object IconVolumeToFlow
* RPL Object IconHasRuleFiredSuccessfully
* RPL Object IconGetStartDate

### Control Display Icon2.7 RPL Object IconRecordEOYNMandCOCreditAdjustments

Rule Purpose:  
Record calculated NMCredit and COCredit adjustments to separate slots in the RioGrande.Compact data object.

Rule Logic: Execution Constraint logic is at end of explanation.  
On the first timestep of the year, this rule records the credit adjustments at the end of the previous year.

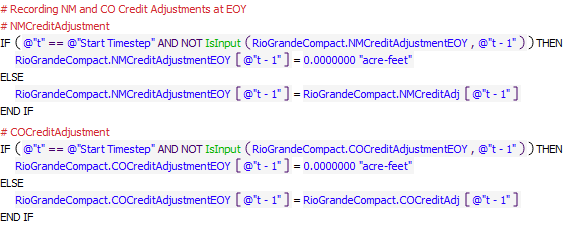
This rule fires at the first timestep of the year.

Model slots written by rule:  
1. RioGrandeCompact.NMCreditAdjustmentEOY  
2. RioGrandeCompact.COCreditAdjustmentEOY

List of key model objects with slots read by the rule or child functions:  
1. RioGrandeCompact

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description field and updated notes field.  
10/05/2018: Marc Sidlow. New rule to record NMCredit andr COCredit storages in RioGrande.Compact data object.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconIsFirstTimestepOfYear
* RPL Object IconIsInput

### Control Display Icon2.8 RPL Object IconRecordEOYNMandCOCreditStorages

Rule Purpose:  
Record NMCredit and COCredit storage in EB at the end of the year to separate slots in the RioGrandeCompact data object.

Rule Logic: Execution Constraint logic is at end of explanation.  
If there is no data in RioGrandCompact.NMCreditDebit or RioGrandCompact.COCreditDebit, set them equal to the previous timestep storage of the given credit water in Elephant Butte.

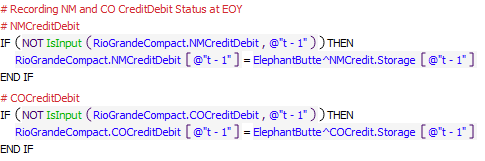
This rule fires at first timestep of the year.  
Comment:  
Nearly identical to RecordEOYNMandCOCreditStoragesPreAdjustment. Seems that one of the two rules and associated slots could be removed.

Model slots written by rule:  
1. RioGrandeCompact.NMCreditDebit  
2. RioGrandeCompact.COCreditDebit

List of key model objects with slots read by the rule or child functions:  
1. RioGrandeCompact  
2. ElephantButte

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description field and updated notes field.  
10/05/2018: Marc Sidlow. New rule to record NMCredit andr COCredit storages in RioGrande.Compact data object.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconIsFirstTimestepOfYear
* RPL Object IconIsInput

### Control Display Icon2.9 RPL Object IconRecordEOYNMandCOCreditStorages\_Accumulated

Rule Purpose:  
Record accumulated New Mexico and Colorado Rio Grande Compact credit adjustments with and without evaporation to separate slots in the RioGrande.Compact data object.

Rule Logic: Execution Constraint logic is at end of explanation.  
On the first timestep of the year, record the previous timestep storage of the given credit water in Elephant Butte. On the first timestep of subsequent years, sum all previous credit adjustments for the sum without evaporation. For the sum with evaporation, sum all previous years with evaporation, and add the adjustment and evaporation for the previous year.

This rule fires at the first timestep of the year.

Model slots written by rule:  
1. RioGrandeCompact.NMCreditDebit\_Accumulated  
2. RioGrandeCompact.COCreditDebit\_Accumulated  
3. RioGrandeCompact.NMCreditDebit\_Accumulated\_WithEvaporation  
4. RioGrandeCompact.COCreditDebit\_Accumulated\_WithEvaporation

List of key model objects with slots read by the rule or child functions:  
5. RioGrandeCompact  
6. ElephatnButte

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description field and updated notes field.  
10/05/2018: Marc Sidlow. New rule to record NMCredit andr COCredit storages in RioGrande.Compact data object.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconIsFirstTimestepOfYear
* RPL Object IconNMCreditGainLoss
* RPL Object IconNMCreditAdjustment
* RPL Object IconCOCreditAdjustment
* RPL Object IconCOAccumulatedWithEvap
* RPL Object IconCOCreditGainLoss
* RPL Object IconNMAccumulatedWithEvap
* RPL Object IconSumSlotSkipNaN
* RPL Object IconIsInput
* RPL Object IconRoundVolume

### Control Display Icon2.10 RPL Object IconSetInitialCreditStorage

Rule Purpose:  
Reset New Mexico and Colorado credit storage to zero if they are negative.

Rule Logic: Execution Constraint logic is at end of explanation.  
Two WITH statements define the accumulated credit volume for New Mexico and Colorado. Two if statements then set Credit Storage to end the previous year equal to the larger of 0 acre-feet and the accumulated storage.

This rule fires at the first timestep of the year.  
Comment:  
12/2/2019: Jesse Roach. Does this need to occur only if debit accounting is occuring at El Vado? Is the ELSE statement necessary?

Model slots written by rule:  
1. ElephantButte^NMCredit.Storage  
2. ElephantButte^COCredit.Storage

List of key model objects with slots read by the rule or child functions:  
1. RioGrandeCompact

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description field and updated notes field.  
10/05/2018: Marc Sidlow. New rule to set negative NMCredit and/or COCredit storages to zero and compute RioGrande storage to balance accounts.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconIsFirstTimestepOfYear
* RPL Object IconNMCreditAdjustment
* RPL Object IconCOCreditAdjustment
* RPL Object IconRunStartDate
* RPL Object IconIsInput

### Control Display Icon2.11 RPL Object IconSetInitialRioGrandeStorage

Rule Purpose:  
Reset Rio Grande storage in Elephant Butte based on total storage less all other accounts with positive storage.

Rule Logic: Execution Constraint logic is at end of explanation.  
Nested if statements are used to separate three cases, one where both NM And CO credit are negative, and two where one or the other are negative. Rio Grande storage is set equal to the total storage less the sum of all SJC in storage and any positive credit storage.

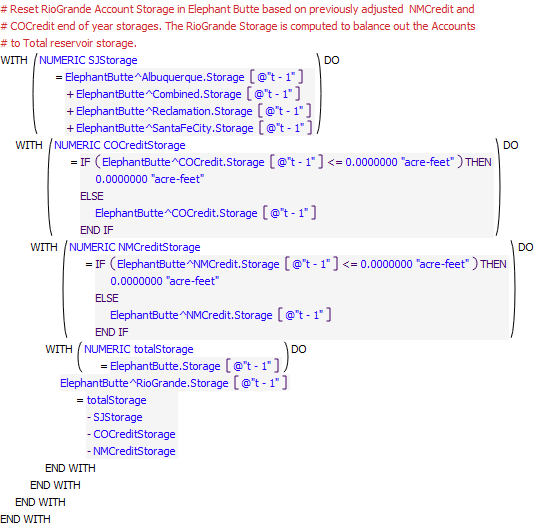
This rule fires at the first timestep of the year.

Model slots written by rule:  
1. ElephantButte^RioGrande.Storage

List of key model objects with slots read by the rule or child functions:  
1. ElephantButte

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description field and updated notes field.  
10/05/2018: Marc Sidlow. New rule to set negative NMCredit and/or COCredit storages to zero and compute RioGrande storage to balance accounts.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconIsFirstTimestepOfYear

### Control Display Icon2.12 RPL Object IconSetRelinquishedCompactCredits

Rule Purpose:  
This rule records an amount of relinquished Compact credits. Relinquished credits allow for subsequent storage of Emergency Drought water at El Vado Reservoir when Article VII is in effect to be used by MRGCD and as supplemental water for ESA operations. If a switch has been set by the model user, Compact credits will be relinquished on an input date for the relinquishment to occur. If a threshold Compact credit is exceeded, Compact credits will be relinquished to reduce the credit to a target lower Compact credit.

Rule Logic: Execution Constraint logic is at end of explanation.  
The value for the RelinquishedNMCredits series slot in the RelinquishedCreditsEmergencyDroughtWater data object is set using two IF THEN ELSE statements. If relinquished credits are to be modeled based on an input value greater than 0.1 in the the TriggerModelRelinquishedCredits scalar slot as checked with the ModelRelinquishedCredits user-defined function and the current timestep matches the date for relinquishments as input to the DateOfRelinquishment column of the RelinquishedCreditsTriggers table slot, a relinquishment volume will be computed; otherwise, the value is set to zero. If the Compact credits as of December 31 of the previous year as checked with the CreditWater user-defined function is less than the value input to the ThresholdForRelinquishment column of the RelinquishedCreditsTriggers table slot, the relinquishment value is set to zero. Otherwise, the value is set to the amount of the Compact credit as of December 31 minus the value input to the CreditAfterRelinquishment column of the RelinquishedCreditsTriggers table slot.

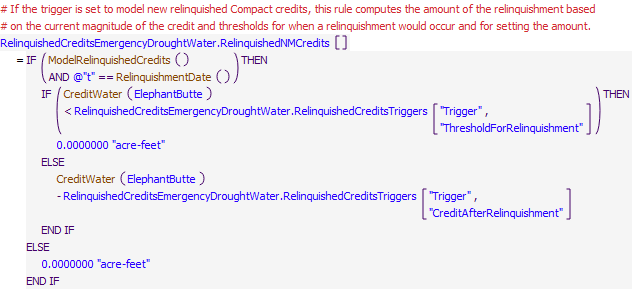
This rule fires if the current timestep is greater than or equal to the beginning timestep for rulebased simulation as input by the model user to the RulebasedSimulationStartDate scalar slot in the ModelRunTypeTriggers data object and as identified with the user-defined GetStartDate function and if the value for the RelinquishedNMCredits series slot in the RelinquishedCreditsEmergencyDroughtWater data object is a NaN for the current timestep.

Model slots written by rule:  
1. RelinquishedCreditsEmergencyDroughtWater.RelinquishedNMCredits

List of key model objects with slots read by the rule or child functions:  
2. RelinquishedCreditsEmergencyDroughtWater  
3. ElephantButte

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description field and updated notes field.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconCreditWater
* RPL Object IconRelinquishmentDate
* RPL Object IconModelRelinquishedCredits
* RPL Object IconGetStartDate

### Control Display Icon2.13 RPL Object IconUpdateEmergencyDroughtStorageAllocations

Rule Purpose:  
This rule includes three assignment statements to track the allocations for storage of Emergency Drought water for MRGCD, ESA, and use by municipalities where the allocations are increased for a proportion of any relinquished Compact credits. Note that the allocations include water still in storage and the allocations do not decrease until the water is released from storage. Also, the allocations for municipalities are tracked but URGWOM is not set up to model the storage or use of this water for municipalities.

Rule Logic: Execution Constraint logic is at end of explanation.  
Values for the MRGCDDroughtAllocation and SupplementalESAAllocation series slots in the RelinquishedCreditsEmergencyDroughtWater data object are set with reference to the user-defined UpdatedAllocationForEmergencyDroughtWaterStorage function. The computation with the function starts with the allocation at the previous timestep. If the current timestep is the Start Timestep and an initial allocation was not input, the initial allocation is set to the initial storage for the corresponding Emergency Drought Water account. The allocation is then updated based on the previous gain/loss for the corresponding account and to reflect any new allocation based on the amount of the relinquished Compact credit at the current timestep (which will often be zero unless a relinquishment occurred at the current timestep) multiplied by the proportion of that relinquishment to be allocated for the corresponding account as input to the ProportionsForNewEmergencyDroughtAllocations table slot. The allocation is then reduced based on any release of water from the allocated storage at the previous timestep as identified by the value for the relevant accounting supply (i.e. the allocation as tracked in URGWOM is not reduced when the water is stored but is reduced when the water is released). A separate similar calculation is completed within the rule for tracking the allocation for municipalities, but with no check against gain/loss or release for the account since there is no account or storage or release to reference for municipalities. The allocation for municipalities is tracked but not used in URGWOM.

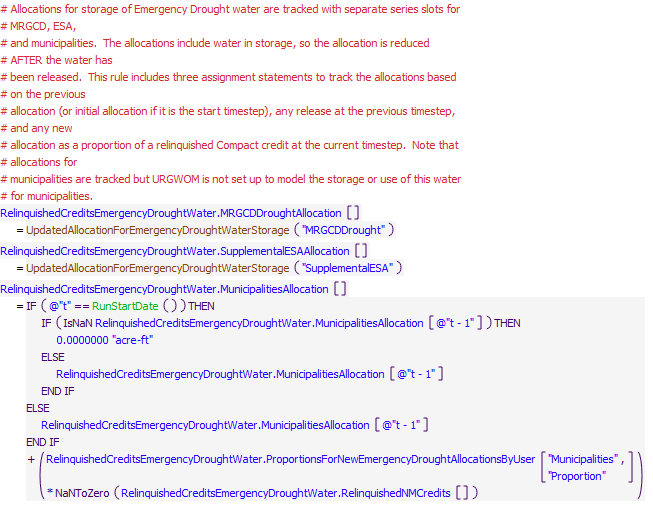
This rule fires if the three slots written (MRGCDDroughtAllocation, SupplementalESAAllocation and MunicipalitiesAllocation) do not already have valu

Model slots written by rule:  
1. RelinquishedCreditsEmergencyDroughtWater.MRGCDDroughtAllocation  
2. RelinquishedCreditsEmergencyDroughtWater.SupplementalESAAllocation  
3. RelinquishedCreditsEmergencyDroughtWater.MunicipalitiesAllocation

List of key model objects with slots read by the rule or child functions:  
1. RelinquishedCreditsEmergencyDroughtWater  
2. ElVado

Log of when and how the rule has been modified (newest changes at the top):  
Date: Who. What  
2/12/19: Jesse Roach. Added description field and updated notes field.

Statements



Execution Constraint

Execution Constraint

Referenced Functions

* RPL Object IconUpdatedAllocationForEmergencyDroughtWaterStorage
* RPL Object IconRunStartDate