

CHECKING RESULTS in RiverWare

You've run a model and looked at the results. You want to verify that an object mass-balances. At first glance, you think it looks wrong. Let's take an example:

We have a reservoir with Inflow, Outflow, and Storage slots. Initial storage is given by user input, and all inflow and outflows are given; storage for all other days are computed. The reservoir mass-balances if and only if

$$\text{Inflow} + \text{Previous Storage} - \text{Outflow} = \text{Storage}.$$

Table 1: Reservoir Slots Displayed in Acre-Feet/Day

Day	Inflow	Outflow	Storage	Apparent Error
0			0.00	
1	7.00	0.00	6.99	-.01
2	0.00	6.99	0.00	
3	69.96	49.05	20.91	
4	0.00	20.91	0.00	
5	10.00	0.94	9.00	-.06

What's going on? It looks like days 1 and 5 don't mass-balance. You might suspect **rounding** errors, which can occur whenever you use floating point on a digital computer, but this seldom explains the differences, especially large differences. RiverWare does not guarantee exactly the same results each time you run a simulation, in part due to natural side effects of compiler optimization. Such differences tend to be in the least significant digit, usually in the range of +/- 1.0E-8 or smaller, and that's not what we see here (Table 1).

The values in day 1 suggest that the problem might simply be a matter of **display precision**. So you expand the display precision to 12 and look again (Table 2).

Sure enough, Outflow is not exactly zero, so the result makes sense, and we can see that when precision is 2, the rounding up of the result accounts for the apparent error with day 1 ... but now day 3 looks wrong. We expect a storage of 20.9087891009 acre-feet/day that day. Perhaps the correct value was computed, but was within **convergence** of the old value, so it wasn't changed. Our convergence factor is .01 percent, or .0001 of the old value. Let's do a little arithmetic:

$$20.906999990000 * .0001 = .002090699999$$

Table 2: Reservoir Slots Displayed in Acre-Feet/Day with Precision 12

Day	Inflow	Outflow	Storage	Apparent Error
0			0.000000000000	
1	6.995990010090	0.004905111100	6.991084898990	
2	0.000000000000	6.991084898990	0.000000000000	
3	69.959900100900	49.051111000000	20.906999990000	-0.0017891109
4	0.000000000000	20.906999990000	0.000000000000	
5	10.000000000000	0.940000000000	9.000000000000	-0.06

which is larger than the difference... yup, it was a convergence matter. We try to verify this by turning on the diagnostics for this slot and seeing the convergence. We notice that the diagnostics give all values in ...

internal units. Ah, Ha! That might explain the other apparent errors. Volumes are stored in *cubic meters*, and flow in *cms*. We display the values in internal units (conversion factor is 1/70.04561994344840) (Table 3):

Table 3: Reservoir Slots Displayed in Internal Units

Day	Inflow (cms)	Outflow (cms)	Storage (m3)
0			0.000000000000
1	0.099877622837	0.000070027378	8623.376247656906
2	0.000000000000	0.099807595459	-0.000000000002
3	0.998776228369	0.700273779283	25790.611601071716
4	0.000000000000	0.298502449086	0.000000000007
5	0.142764101568	0.013419825547	11175.34544818054

and do the necessary arithmetic for day 5:

$$0.142764101568 - 0.013419825547 = 0.129344276021$$

in cms. To convert to storage, we multiply by the number of seconds in a day, 86400, to get the net gain in storage:

$$0.129344276021 * 86400 = 11175.3454482184$$

Convergence happens when the new value is within .0001 of the old value, i.e., when it is within .0001 * 11175.34544818054 = 1.11753454482184

of the old value. The difference between the correct (new) value and the old is 11175.3454482184 - 11175.34544818054 = .00000003786

and

$$.00000003786 < 1.117534544818054$$

so the convergence criterion is met, and the old value is not changed. Notice that the convergence criterion is met with a much larger difference than we have here, due to fact that the stored values

are fairly large (relative to the input values that are used to compute the result). Thus, where .0001 may be a fine convergence factor for some slots, other slots might need much smaller convergence factors.

If we were using a model with **monthly timesteps**, we must remember that the number of seconds in the month varies, so we must vary the computations done when converting from flow to volume.

When reconciling account values with simulation objects' values, remember that convergence is used in both simulation and accounting, and the **convergence factors might differ**.

Even if the convergence factors are the same, corresponding values in the two systems might **converge from opposite directions**, giving a larger difference than expected. Thus, a simulation object slot (e.g., object.Storage) might converge with

$$\begin{aligned} &\text{old value}_{\text{sim}} < \text{new value}_{\text{sim}} \\ \text{diff}_{\text{sim}} &= | \text{new value}_{\text{sim}} - \text{old value}_{\text{sim}} | \end{aligned}$$

while the corresponding slot on the object's account (e.g., object^account.Storage) might converge with

$$\begin{aligned} &\text{old value}_{\text{acct}} > \text{new value}_{\text{acct}} \\ \text{diff}_{\text{acct}} &= | \text{new value}_{\text{acct}} - \text{old value}_{\text{acct}} | \end{aligned}$$

and this means that the simulation value and the account value differ by

$$\text{diff}_{\text{sim}} + \text{diff}_{\text{acct}},$$

which may exceed the convergence criteria of both simulation and accounting. For example, let's look at some values for a reservoir and its account (Table 4):

Table 4: Reconciling Account Values with Simulation Object Values

	Account slot	Simulation Object slot
Old value (stored)	10.00090000	9.99910000
New value (not stored)	10.00000000	10.00000000
Difference in old & new values	0.00090000	0.00090000
Convergence limit +/- @.01%	0.00100009	0.00099991
Difference between smaller of low end of ranges and larger of high end of ranges	10.00190009 - 9.99810009 = 0.00380000	

Table 4: Reconciling Account Values with Simulation Object Values

	Account slot	Simulation Object slot
Convergence range @.01%	9.99989991 – 10.00190009	9.99810009 – 10.00009991
Difference between smaller of low end of ranges and larger of high end of ranges	10.00190009 – 9.99810009 = 0.00380000	

As you can see, the account value can differ from the simulation value in the above example by as much as 0.0038 while the account and simulation object mass-balance with the convergence criterion of .01% (.0001 of old value).

The accounting system configuration **GUI differs** from the corresponding simulation GUI for setting convergence. In the simulation world, if you select X PERCENT, your convergence criterion is

$$| \text{new value} - \text{old value} | < (X / 100) * \text{old value}$$

whereas in the accounting world, you cannot select a PERCENT, but you can enter the raw multiplier used, and the convergence criterion is

$$| \text{new value} - \text{old value} | < X * \text{old value}.$$

SUMMARY

You must take into account the following issues when checking the mass-balancing of objects:

- **Rounding errors** can explain very small differences, or larger differences when calculations involve very small numbers that are then used to calculate large numbers.
- Values are often **displayed** with **precision** that causes rounding.
- **Convergence** criteria may cause results not to be exact.
- Arithmetic is done in **internal units**. Diagnostics show convergence in internal units.
- Arithmetic with **monthly** models must take into account the differing month lengths.
- Convergence criteria are specified for each slot, both in the accounting system and in the simulation system and **convergence factors might differ** (the defaults match, however). Because the **GUIs differ** for the accounting and simulation systems, it is worth checking your convergence criteria if you have used anything other than the defaults.
- Reconciling accounts with their underlying objects must take into account that values on the simulation object might **converge from different directions** than the values on the account.