Raven Users Symposium 2025
May 29th, 2025

Raven Tricks and Tools:

Water Management in Raven

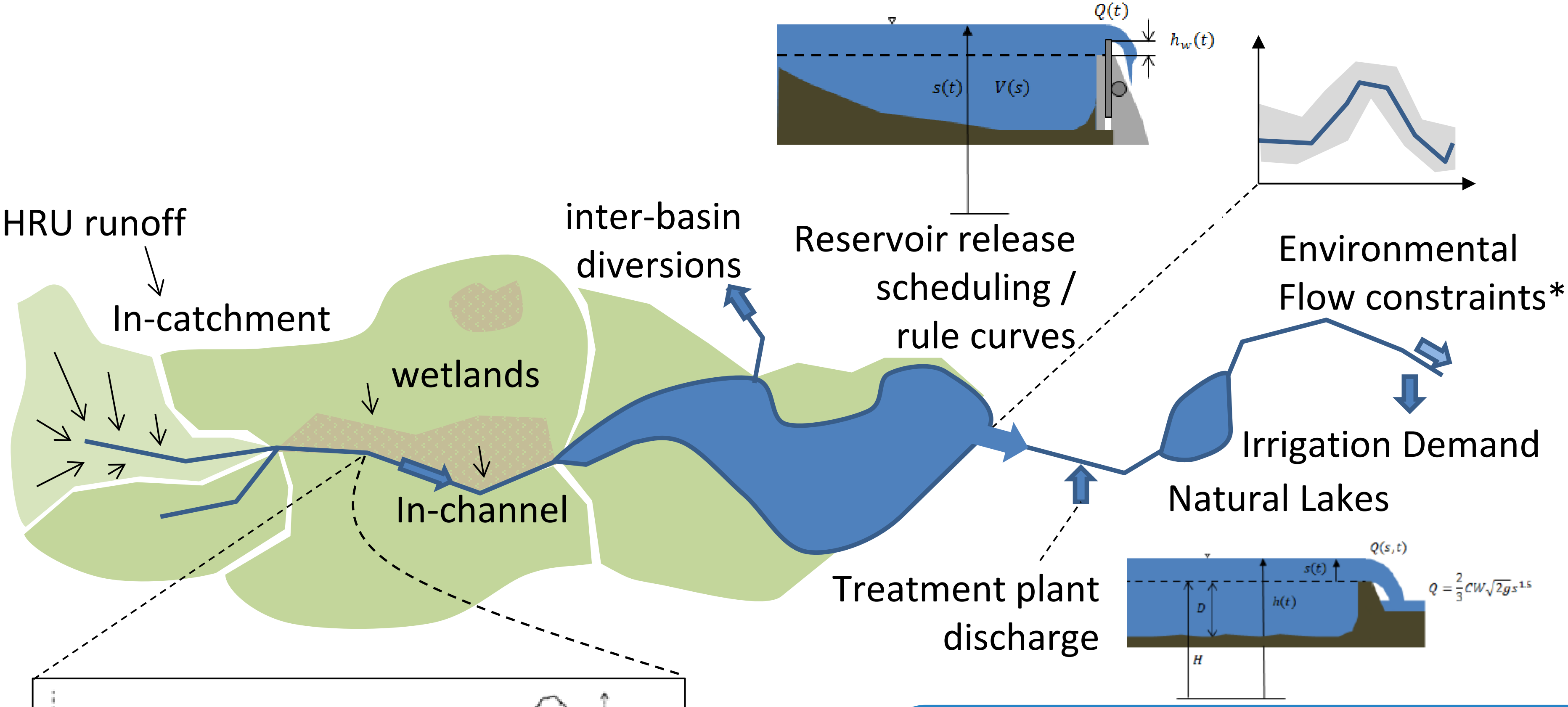
JAMES R. CRAIG, PH.D., P.ENG.

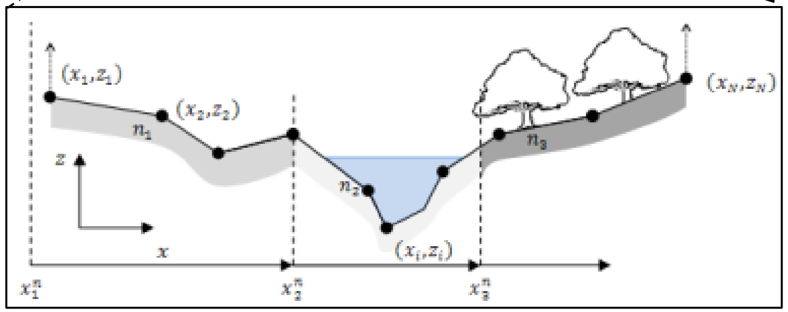
UNIVERSITY OF WATERLOO

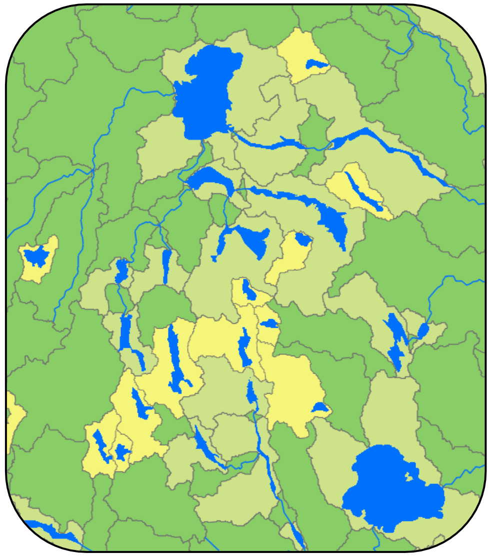
Overview

- The basics of the Water management functionality in Raven
- Syntax essentials
- An example tutorial

Routing and Water Management Support







Uses vector-based routing networks (10000+ basins)

PREVIOUSLY MISSING: Ability to support:

1) *Fully generalized* operation/water allocation rules
2) *Optimal* satisfaction of management goals
3) Inter-basin constraints (e.g., environmental flow needs)

Water Management Models

Water management models have to be able to:

- Codify management goals
- Codify priorities when management goals conflict
- Codify operational rules

This has to be VERY general to support the wide range of feasible operating strategies

Common to handle using linear programming (LP) optimization.

- Not necessarily because we need an “optimal” solution, but because LP is great for formulating problems expressed as hundreds of competing goals and constraints

Water Management Optimization in Raven

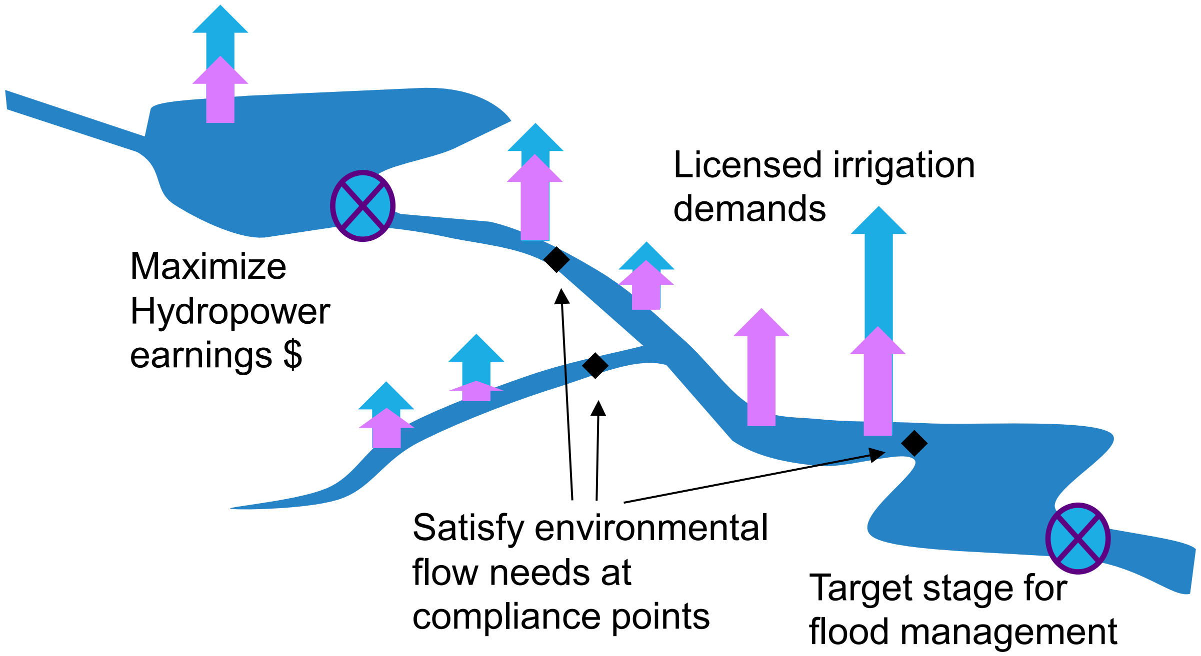
- Solves non-linear optimization problem in each time step
- Minimize unmet water demands and goal penalties

$$\sum \omega_i (D_i^* - d_i) + \sum \omega_j P_j \rightarrow \min$$

  - Subject to
    - Satisfaction of routing mass balance
    - User-specified constraints

- User-specified management goals to handle arbitrarily complex allocation rules and management decisions
- Posed as *iterative* linear programming problem

- Comparable functionality to programs such as RiverWare, WEAP, MODSIM, or OASIS but
  - Free and open-source
  - Integrated into a fully functional hydrology model



Precipitaion
Temperature
Radiation
Wind
Speed/etc.

In each time step,

Iteratively solves
linear programming
problem

Raven
Hydrological
Model

Runoff
Reservoir Precip./Evap.

Raven
Management
Optimization
Scheme

Management Goals
Management Constraints
Control Variable Definitions

Optimal Demand Delivery
Return Flows
Diversion flows
Reservoir Outflows

Time Series or Annual Patterns
(Forecasts/Rule Curves/Demand)
Water license info
Lookup Tables

Streamflows
Temperatures
Res. Stages

Recast Mass Balance/Routing problem

In-reach routing:

Runoff

Proscribed inflows

Diversions (*non-linear*)

$$Q_p^{n+1} = U_0^p \sum_{i=0}^{N_p^{in}} Q_i^{n+1} + \sum_{j=1}^{N} U_j^p \sum_{i=0}^{N_p^{in}} Q_i^{n-j+1} + R_p^n - D_p^n + I_p^n \pm F_p^n$$

Reach outflow

Delivered Demand

Time lagged inflow
(diffusive wave routing)

Instantaneous inflow

In-reservoir/lake routing:

Inflow

Outflow

$$\begin{aligned}
\frac{A^n}{\Delta t}(h^{n+1} - h^n) &= \frac{1}{2}(Q_{in}^n + Q_{in}^{n+1}) - Q_{out} \\
&\quad + P \cdot A^n - E \cdot A^n - \frac{1}{2}\left(S(h^n) + S(h^{n+1})\right) \\
Q_{out} &= \frac{1}{2}\left(Q(h^n) + Q(h^{n+1})\right)
\end{aligned}$$

Volume change

Precip/ET/Seepage

Reservoir outflow

non-linear

Raven Upgrades

Linear programming matrix is assembled from very general statements:

- Management goals (should ideally be satisfied)
  - Prioritization of goals is determined by magnitude of penalty associated with not meeting goal
  - All demand time series automatically converted to goals
  - All environmental flow minimum regulations automatically converted to goals
- Management constraints (must be satisfied) (not too many)
- Mass balance constraints (must be satisfied)

- Goals/Constraints can be turned on or off using conditionals, representing different operating regimes

- All reservoir management time series (e.g., :TargetStage) should instead be treated using management statements
  - Allows priorities to be codified as penalties

Installation

Management model requires a bit more work to compile

- Uses the open-source lp_solve library
- In practice (on Windows), just requires lpsolve.dll library file to be in same directory as Raven.exe
- Requires re-compilation of lp_solve AND Raven on MacOS/linux

To activate water management use:

```
:ApplyManagementOptimization
```

In .rvi file

.rvm (Raven Management File)

```
# Global Constants
:NamedConstant ACREFT_TO_M3 1233.48

:LookbackDuration 7
:DebugLevel 0

:OverrideStageDischargeCurve 12 # Managed Reservoir 1
:OverrideStageDischargeCurve 16 # Managed Reservoir 2

# Water Demand Penalties and Groups
#-----------------------------------------------------------------------
:RedirectToFile WaterManagement/Demands.rvm

# Management Constraints and Goals
#-----------------------------------------------------------------------

# Management Tables/Timeseries
:RedirectToFile WaterManagement/Timeseries_GlenmoreConstraints.rvt
:RedirectToFile WaterManagement/ReservoirTargetStage.rvt
:RedirectToFile WaterManagement/ReservoirMaxQ.rvt

:RedirectToFile WaterManagement/ReservoirLookupTables.rvm

# Management Ops/Logic
:RedirectToFile WaterManagement/Operations_Reservoir1.rvm
:RedirectToFile WaterManagement/Operations_Reservoir2.rvm
```

Number of days stored in history

Similar `:RedirectToFile` conventions to handle child files.

- Can now have nested :RedirectToFiles, e.g., grandchildren

Demand time series and user specified time series can be read from here rather than rvt file

`:OverrideStageDischargeCurve` indicates this is a managed reservoir and outflow will be a function of management

Demands.rvm

```
# :WaterDemand [SBID] [demand_ID] [demand_name]
:WaterDemand 240 2401 ChetsSoybeans
  :DemandTimeSeries
    :AnnualCycle 0.01 0.01 0.01 0.01 0.04 0.1 0.1 0.1 0.1 0.03 0.03 0.01
  :EndDemandTimeSeries
  :ReturnTimeSeries
    :AnnualCycle 0.0 0.0 0.0 0.0 0.012 0.06 0.06 0.05 0.05 0.012 0.012 0.0
  :EndReturnTimeSeries
  :Penalty 32
:EndWaterDemand

:WaterDemand 240 2401 ChetsSoybeans
  :DemandTimeSeries
    :AnnualCycle 0.01 0.01 0.01 0.01 0.04 0.1 0.1 0.1 0.1 0.03 0.03 0.01
  :EndDemandTimeSeries
  :ReturnFraction 0.6
  :Penalty 32
:EndWaterDemand

:WaterDemand 232 2329 CanalDiversion
  :DemandTimeSeries
    :AnnualCycle 10 10 10 10 10 10 10 10 10 10 10 10
  :EndDemandTimeSeries
  :ReturnFraction 1.0
  :ReturnDestination 238
  :Penalty 58
:EndWaterDemand
```

Demands are withdrawn from downstream end of subbasin

Time series or expressions

Diversions can be handled as water demands with 100% return flow to elsewhere in network

Each demand now requires unique demand ID

Management Goals

```
:ManagementGoal
  :OperatingRegime R1
    :Expression !Q1 = 18
    :Condition !h2[-1] > 1324.5
    :Condition DAY_OF_YEAR IS_BETWEEN 200 263
  :EndOperatingRegime
  :OperatingRegime R2
    :Expression !Q1 = 23
    :Condition !Q2[-1] + !Q3[-1] < 22
  :EndOperatingRegime
  :OperatingRegime default
    :Expression !Q1 = 15
  :EndOperatinRegime
  :Penalty 34     #determines priority
:EndManagementGoal
```

The heart of the management functionality

General expressions tied to operating regimes triggered by conditions

- First operating regime whose conditions are met applies
- Conditions cannot be expressed in terms of raw state variables (evaluated prior to solution)

Penalty weights determine priority of goal relative to other goals

User-specified Management Goals: Example Syntax

```
:NamedConstant           cMyMultiplier        = 1.5

:DefineDecisionVariable vFarmerBobsDelivered = !D121 + !D122 + !D134

# change in flow rate < 4m3/s/d if in zone A
:ManagementGoal MothLakeFlowRamping_zoneA
  :Expression !q130 < 40
  :Conditional !h130 BETWEEN 1224 1233
  :Penalty    20
:EndManagementGoal

# target flows to meet power goals
:ManagementGoal MeetSummerPowerTarget
  :Expression  !Q120 + !Q181 = @ts(Q130_power_target,0) * 1.2 * cMyMultiplier * vPowerConvert130
  :Conditional MONTH BETWEEN 6 8
  :Conditional !Q120 GREATER_THAN 200
  :Penalty     12
  :Priority    2
:EndManagementGoal
```

Note no operating regime here – defaults to ‘always on’ operating regime

Expressions support some functions, mathematical operations

Example: Conditional Irrigation Demand

```
:ManagementConstraint DemandControl
  :OperatingRegime DrySoil
    :Expression !D.RobertsFarm = 0.3 #m3/s
    :Condition @HRU_var( SOIL_SAT[0] , 253 ) >0.25
    :Condition MONTH IS_BETWEEN 4 9
  :EndOperatingRegime
  :OperatingRegime WetSoil
    :Expression !D.RobertsFarm= 0 #m3/s
  :EndOperatingRegime
:EndManagementConstraint
```

(In English)

Sets demand to 0.3 m^3/s

If soil saturation in field 253 is >25%

Between April and Sept

Otherwise set to zero

Example: Water Licenses

```
:LoopThrough DEMAND_GROUP OakValleyDemands
  :LoopVector $Alloc_AcFt$ 12 22 6 9.3
  :DemandResetDate $ID$ Jan-1
  :ManagementConstraint AllocExpiry_$ID$
    :Expression !d.$ID$ = 0
    :Condition  !C.$ID$ > @convert($Alloc_AcFt$,ACREFT_TO_M3)
  :EndManagementConstraint
:EndLoopThrough
```

Loops through 4 demands in demand group

Creates vector of wildcard values (annual license quantities, in ac-ft)

ID is automatic wildcard for demand groups

`!d.123` is the current delivery of demand with index 123

`!C.123` is the cumulative delivery to index 123

This command sets water demand delivery to zero if the license has been used up, which re-boots every January 1
The loop applies this to four different demands

Another Real Example…

```
:DeclareDecisionVariable IJCRuleCurve

:ManagementConstraint IJC1938
# Once past peak, and has lowered below 1743.32 ft, that is max until sept 1
    :OperatingRegime Summer
        :Condition DAY_OF_YEAR IS_BETWEEN 150 244
        :Condition !hl[-1] < 1743.32 * FT_TO_M
        :Expression IJCRuleCurve = 1743.32 * FT_TO_M
    :EndOperatingRegime
# Once spring freshet declared, but before summer freshet, use lowering table
    :OperatingRegime Spring
        :Condition DAY_OF_YEAR IS_BETWEEN 105 244
        :Expression IJCRuleCurve = 1929QueensBayLevel - RequiredLowering
    :EndOperatingRegime
# For rest of year, use Rule1938MaxZ
    :OperatingRegime Winter
        :Condition DAY_OF_YEAR IS_BETWEEN 244 105
        :Expression IJCRuleCurve = @ts(Rule1938MaxZ,0) * FT_TO_M
    :EndOperatingRegime
:EndManagementConstraint
```

Expressions

Expressions are linear expressions. Linear means they must be in the following form:

$$A \cdot x_1 + B \cdot x_2 + C \cdot x_3 \ldots + Z \leq 0$$

or $\geq$, $=$

where x_i are model state variables, and A..Z are constants with respect to the state variables. In Raven, the state variables include:

`!Qxxx`	Basin/reservoir discharge from basin ID xxx
`!hxxx`	Reservoir/lake stage in basin xxx
`!Ixxx`	Reservoir inflow in basin xxx
`!Dyyy`	Demand delivery for demand yyy

Expressions

Raven supports the following components of expressions:

`2.34 (e.g.)`	Numeric constant
`vMyVar`	User-specified named constants OR workflow variables
`CFS_TO_CMS`	Some built in named constants for unit conversion
`!Q123 or !Q.RexRiver`	State variables
`!Q123[-7]`	State variables at previous timesteps (e.g., 7 days ago)
`@max(a,b)`	Maximum of two quantities
`@min(a,b)`	Minimum of two quantities
`@pow(x,a)`	Power function – x^a
`@lookup(table,x)`	Lookup table with name ‘`table`’ and input variable x
`@ts(timeser,n)`	Time series with name ‘`timeser`’ at time index n relative to current time
`@SB_var(var,SBID)`	The spatial mean of state variable var (e.g., SNOW) in subbasin SBID
`@HRU_var(var,ID)`	The spatial mean of state variable var in HRU with specified ID

Syntax support

Conditionals
- Operating regimes are defined by conditions expressed in terms of flow / stage / soil moisture / SWE / time of year / explicit dates / arbitrary time series / cumulative delivery / etc.

Working Variables
- User-specified variables updated during simulation but not calculated via optimization

Time series specified as irregular annual patterns

Lookup tables

History Variables

Loops
- For assigning rules to large sets of demand locations, reservoirs, etc., without repetition

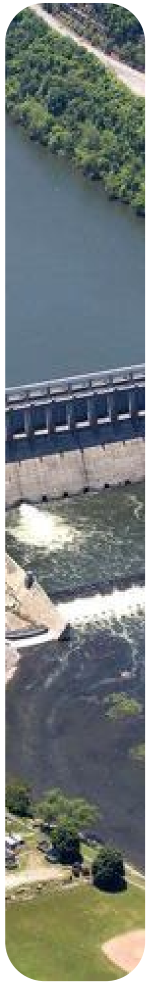
Non-linear variables**

Demand Groups

This was the hardest part of implementation: correctly interpreting very very general goal/constraint expressions

**In version 4.1 / current Github version

Upgrades



Reservoirs / Diversions

Old

- **Rule-based operation**
- Explicit: Control structure operation tied to system state
- Implicit: Constrained rule curves

New

- **Goal-based operation**
- Can (e.g.) optimally satisfy all downstream water demands subject to flow regulations
- Can have rules tied to rainfall or energy price forecasts
- Can mix rule curves with explicit control structure operations
- Can have complex conditional diversion rules
- …

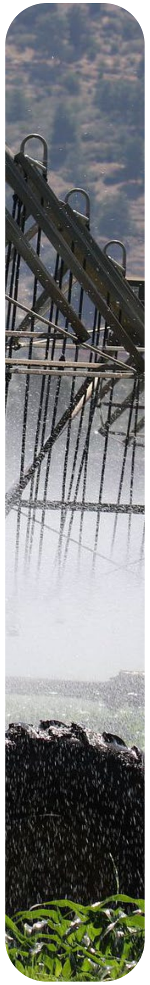
Irrigation/Water demand

Old

- **Specified demands,** locally adjusted by local flow minima
- Specified time series of return flows

New

- **Responsive constrained demands**
- Annual licenses based on cumulative use
- Demand reacts to flow constraints far downstream
- Irrigation demand responds to (e.g.) soil moisture, weather forecasts, storage reservoir capacity
- Irrigation withdrawals applied back to land as ‘precip’ or stored
- …



Output Files

When running in optimization mode, one old and three new output files may be of interest:

- Demands.csv – reports demand/delivery/return
- ManagementOptimization.csv – tracks all state variables in system except demand variables
- GoalSatisfaction.csv – reports to what degree goals are unmet (e.g., unmet demand)
- overconstrained_lp_matrix.csv – returns full LP system of equations if no feasible solutions are found (useful for debugging overconstrained problem)
- can be suppressed with `:SuppressOutput` command

rowname	Q4	Q5	Q9	I3	Q8	Q10	Q2	I6	Q7	I1	Q3	Q6	Q1	h3	h6	h1	dh3	dh6	dh1	B3	B6	B1	LibbyVarQ	LibbyVarQ2	1929QueensBayLevel	RequiredLowering	IJCRuleCurve
Objective F	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BlankGoal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_5	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_9	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_8	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_10	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_6	0	0	0	0	0	-0.3	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_7	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reach_MB_1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
reser_MB_3	0	0	0	-0.5	0	0	0	0	0	0	0.5	0	0	0	0	0	895	0	0	0	0	0	0	0	0	0	0
dh_def_3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	-1	0	0	0	0	0	0	0	0	0	0
reserv_Q_A3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
reserv_Q_B3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
reserv_Q_D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
reserv_Q_E3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
reserv_Q_F3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
reser_MB_6	0	0	0	0	0	0	0	-0.5	0	0	0	0.5	0	0	0	0	0	334	0	0	0	0	0	0	0	0	0
dh_def_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	-1	0	0	0	0	0	0	0	0	0
reserv_Q_A6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
reserv_Q_B6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
reserv_Q_D6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
reserv_Q_E6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
reserv_Q_F6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
reser_MB_1	0	0	0	0	0	0	0	0	0	-0.5	0	0	0.5	0	0	0	0	0	###	0	0	0	0	0	0	0	0
dh_def_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	-1	0	0	0	0	0	0	0	0
reserv_Q_A1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
reserv_Q_B1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
reserv_Q_D1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
reserv_Q_E1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
reserv_Q_F1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
DuncanMaxOutflow	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DuncanMinOutflow	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DuncanMaxQchange_I	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DuncanMaxQchange_I	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DuncanFSL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
DuncanLSL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
LowerDuncanMinFlow	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LowerDuncanMaxFlow	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DuncanTargetUpper	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
DuncanTargetLower	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
LibbyFloodControlPool	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyFullPool	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyMinPool	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyTargetRefill	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyMinFlow	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyPowerCapacity	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BonnersFlooding	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyRampUp	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyRampDown	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyTargetUpper	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyTargetLower	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LibbyVarQ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
LibbyVarQ2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1929QueensBayLevel	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
RequiredLowering	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
IJC1938	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
GrohmanNarrows	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
KootenayMinLevel	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
KootenayMinFlow	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Kootenay1938	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	-1

rowname	SL-	SL+	SL-	SL+	SL-	SL+	SL+	SL-	SL-	SL+	SL-	SL-	SL+	SL+	SL+	SL-	SL-	SL-	SL+	SL-	SL+	SL+	SL+	SL-	compar	RHS
Objective F	##	##	100	100	##	##	100	2	1	1	##	10	##	1	##	100	##	100	100	1	1	##	##	##		0
BlankGoal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0
reach_MB_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	9.00E-06
reach_MB_5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	1.33E-05
reach_MB_9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	1.37E-05
reach_MB_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	9.81E-06
reach_MB_8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	1.36E-05
reach_MB_10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	2.84E-05
reach_MB_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	113
reach_MB_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	2.18E-05
reach_MB_7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	0.47324
reach_MB_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	113.31
reser_MB_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	-53.2798
dh_def_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	704.958
reserv_Q_A3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_B3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_D3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_E3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_F3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reser_MB_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	-3.23522
dh_def_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	546.8
reserv_Q_A6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_B6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_D6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_E6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_F6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reser_MB_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	-66.3136
dh_def_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	523.976
reserv_Q_A1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_B1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_D1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_E1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
reserv_Q_F1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	0
DuncanMaxOutflow	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	283.168
DuncanMinOutflow	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	0.7636
DuncanMaxQchange_I	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	120.712
DuncanMaxQchange_I	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	-105.822
DuncanFSL	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	576.682
DuncanLSL	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	546.872
LowerDuncanMinFlow	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	73
LowerDuncanMaxFlow	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	250
DuncanTargetUpper	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	568.248
DuncanTargetLower	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	>=	564.605
LibbyFloodControlPool	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	750.11
LibbyFullPool	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	<=	749.5
LibbyMinPool	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	>=	697.07
LibbyTargetRefill	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	>=	747.98
LibbyMinFlow	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	>=	113
LibbyPowerCapacity	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	<=	750.4
BonnersFlooding	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	<=	849.504
LibbyRampUp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	<=	169.6
LibbyRampDown	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	>=	84.7
LibbyTargetUpper	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	<=	739.352
LibbyTargetLower	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	>=	735.963
LibbyVarQ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	670.6
LibbyVarQ2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	670.6
1929QueensBayLevel	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	529.858
RequiredLowering	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	0.295
IJC1938	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	==	531.974
GrohmanNarrows	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<=	410.594
KootenayMinLevel	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	>=	530
KootenayMinFlow	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	>=	250
Kootenay1938	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	<=	0

Augmented by Existing Raven Functionality

BMI (Basic Model Interface) –

- Raven can be run as dynamic linked library (DLL) and tied to other models (e.g., crop growth or groundwater)

External scripting –

- Raven can call external scripts each time step (e.g., proprietary management goals in an open-source model)

Synthetic Tracers

- Water can be ‘tagged’ to track (e.g.) the water released from a specific reservoir at a given time as it passes through the system
- Very useful for analysis of choices

Stream Temperature

- Management rules may be linked to simulated stream temperatures

Conclusions

The Raven hydrologic modelling framework has been upgraded to a fully functional water resources management optimization tool

- Published as Raven v4.0 in January
- Used in several applications in Alberta, BC, and Ontario

New GenAI Mascot “Muninn”

Questions?

Raven

raven.uwaterloo.ca

UNIVERSITY OF WATERLOO

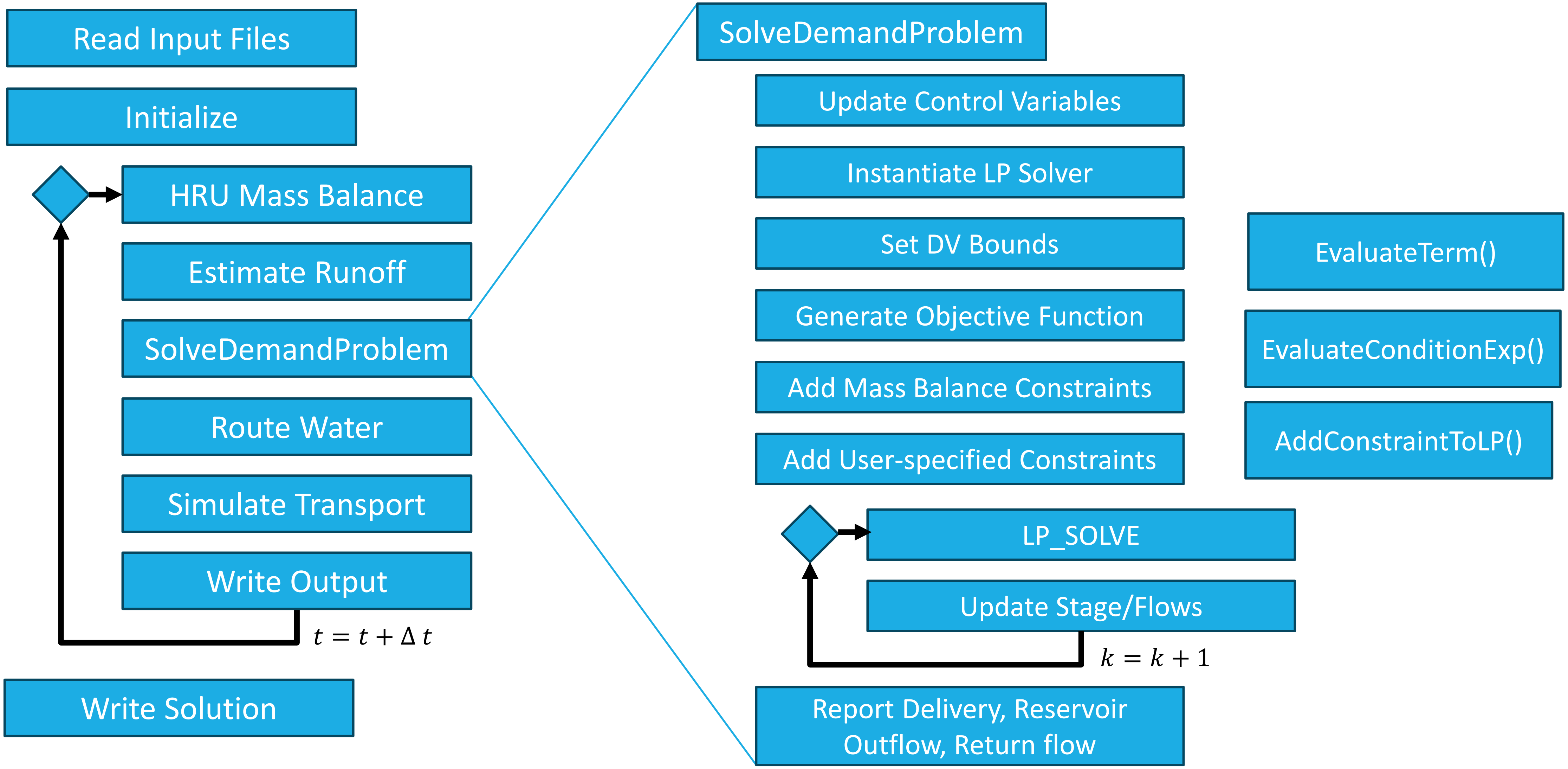
Extra Slides

Initialization – pre RVM read

- Catalogue Enabled Basins
- Populate Decision Variable array
- Gather Demand Data
- IdentifyUpstreamDemands()

Initialization – post RVM read

- Initialize History Arrays
- Create Slack Variables
- AddReservoirConstraints()
- Create Slack Vars for User Goals
- Initialize Time Series
- Print Summary to Screen


Read Input Files
Initialize
HRU Mass Balance
Estimate Runoff
SolveDemandProblem
Route Water
Simulate Transport
Write Output
$t = t + \Delta t$
Write Solution
SolveDemandProblem
Update Control Variables
Instantiate LP Solver
Set DV Bounds
Generate Objective Function
Add Mass Balance Constraints
Add User-specified Constraints
LP_SOLVE
Update Stage/Flows
$k = k + 1$
Report Delivery, Reservoir Outflow, Return flow
EvaluateTerm()
EvaluateConditionExp()
AddConstraintToLP()

Recent Advances: BMI (new as of July!)

Raven can now be compiled as .dll library

Uses BMI: Basic Model Interface

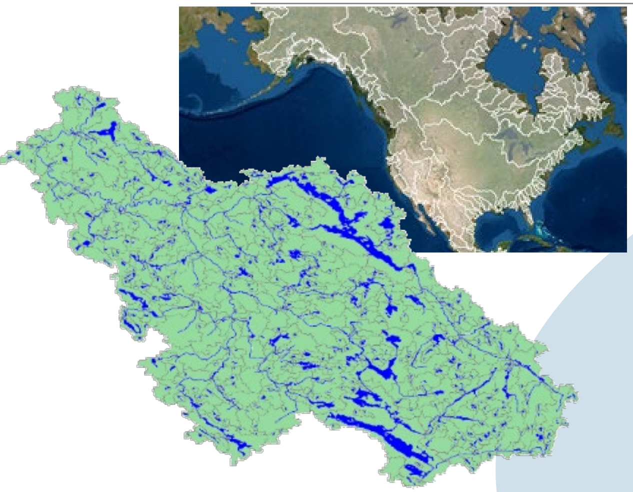
- Protocol shared by many existing earth systems models (USGS, NASA, NWS…)

Wraps Raven functionality such that it can be directly called by other applications

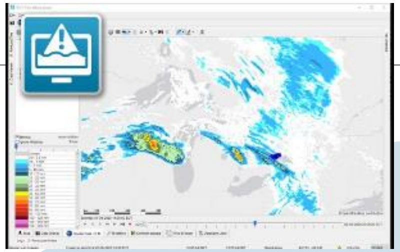
Plans to integrate with U.S. NextGen Framework with support from University of Manitoba colleagues

```
#include "BMI.h"
#include "Model.h"

class CRavenBMI : public bmixx::Bmi
{
  private:
    CModel      *pModel;
    optStruct    Options;
    time_struct tt;

  public:
    CRavenBMI();
    ~CRavenBMI();

    // Model control functions.
    void Initialize(std::string config_file);
    void Update();
    void UpdateUntil(double time);
    void Finalize();

    // Model information functions.
    std::string GetComponentName();
    int GetInputItemCount();
    int GetOutputItemCount();
    std::vector<std::string> GetInputVarNames();
    std::vector<std::string> GetOutputVarNames();

    // Variable information functions
    int GetVarGrid(std::string name);
    std::string GetVarType(std::string name);
    std::string GetVarUnits(std::string name);
```

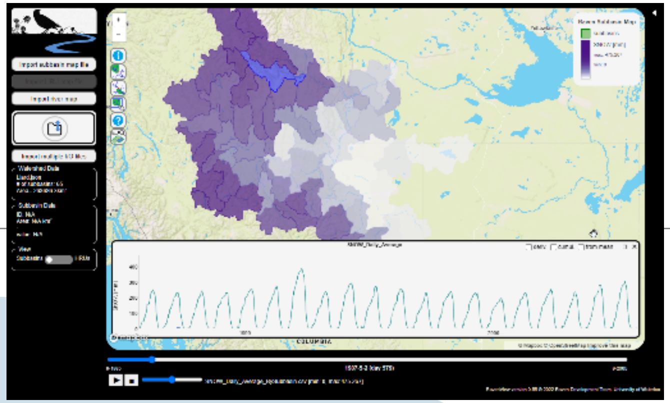
the Raven Ecosystem



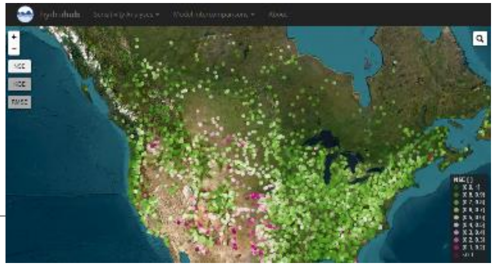
BasinMaker
lake-river discretization toolkit



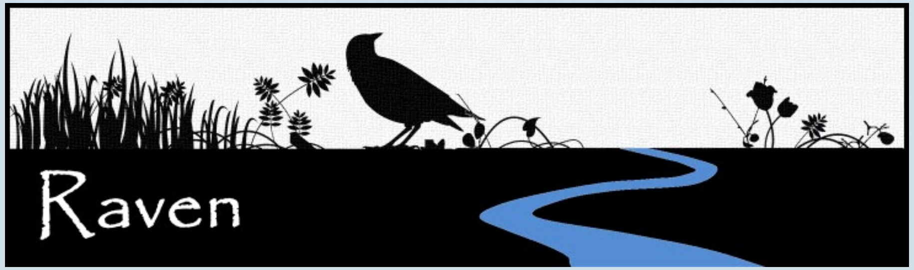
Delft FEWS
flood and early warning system



RavenView
online output visualization



HydroHub
model download & intercomparison

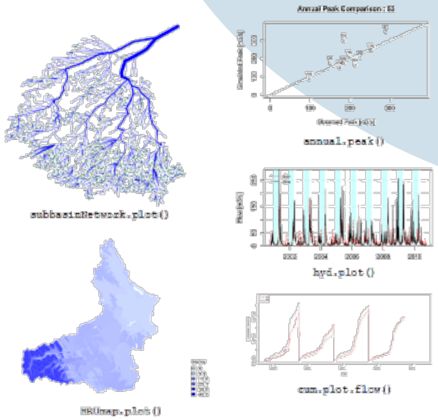


Raven Thermal Model
stream/lake temperature simulation

QRaven

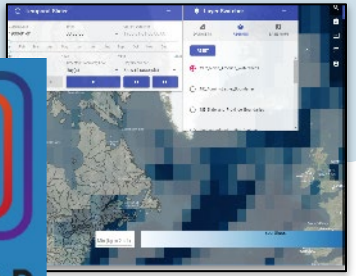
QRaven
QGIS plugin

RavenR



RavenR
hydrologic analysis library

PAVICS



PAVICS-Hydro / RavenPy

Robin Vegetation Growth Library
wildfire/forestry disturbance impacts

Magpie

Magpie
Google colab workflow

Control Structures

Raven supports:
- User-supplied stage-discharge curves
- Basic weirs
- Pumps
- Orifices

Each of these control structures can turn on and off or have its properties change via the use of multiple ***operating regimes***

Operating regimes define when and under what hydrological conditions different structure setups are operated
- Also can be used to define constraints on flow or flow ramping

Used for EXPLICIT simulation of actual reservoir operations
- Testing long term operational strategies
- Forecasting short term operational choices