Command Reference: VariableLagK()

Lag and attenuate (route) a time series with parameters that vary by rate

Version 11.12.04, 2016-09-16

The `VariableLagK()` command can be used to lag and attenuate an input time series, resulting in a new time series. The command is commonly used to route an instantaneous (not mean) regular interval streamflow time series through a stretch of river (reach). For example, route 5-minute or 1-hour instantaneous streamflows measured as cubic feet per second, where the interval for data typically depends on the travel time in the reach and the analysis/modeling framework. `Lag` and `K` routing is a common routing method that combines the concepts of:

1. Lagging the inflow to simulate travel time in a reach and,
2. Attenuating the wave to simulate the storage-outflow relationship for the reach (see **Figure 1**).

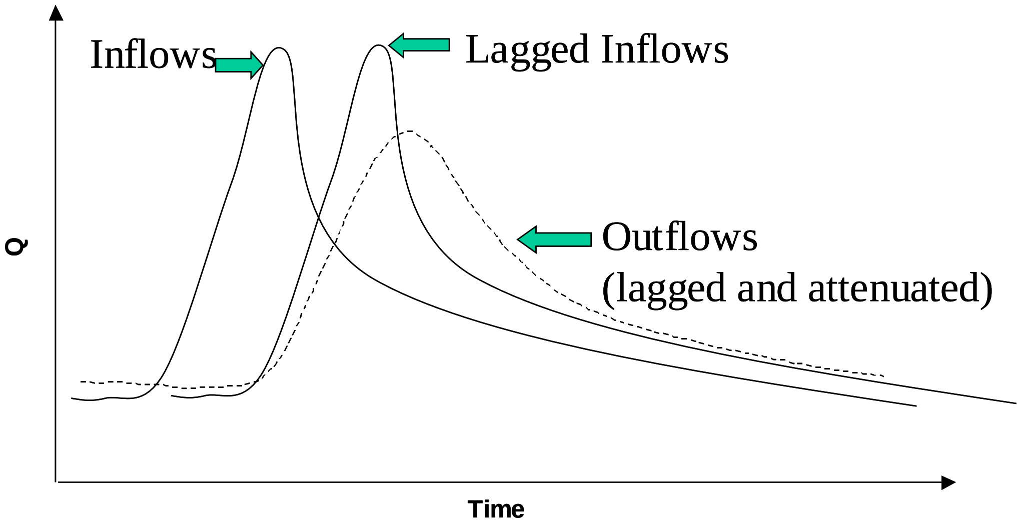


Figure 1: Lag and K Routing

At its fundamental level, the method solves the continuity equation using an approach similar to Muskingum routing (assuming that the Muskingum parameter representing wave storage is negligible). The governing equation for this routing method is given as:

$$Q_{in} - Q_{out} = \frac{\Delta S}{\Delta t}$$

where:

Q_{in} = instantaneous inflow [rate] lagged appropriately,
Q_{out} = instantaneous outflow [rate] lagged appropriately,
ΔS = change in storage in the reach [volume],
Δt = time difference (time series interval)

The relationship assumes an outflow-storage relationship of the form:

$$S = k \cdot Q_{out},$$

where:

k = attenuation for the outflow [time].

To ensure accurate results, k should be larger or equal to $\Delta t/2$. For discrete time steps these relationships translate into:

$$O_2 = \frac{I_1 + I_2 + \frac{2S_1}{\Delta t} - O_1}{\frac{2k}{\Delta t} + 1}, \qquad k \geq \frac{\Delta t}{2}$$

where: I_1 and I_2 are the lagged inflows into the reach at the previous and current time step, respectively,
O_1 and O_2 are the outflows out of the reach at the previous and current time step, respectively,
S_1 is the storage within the reach at the previous time step, defined as $S_1 = k \cdot O_1$, and
Δt is the time difference between the two time steps (time series interval)

Values for `Lag` and `K` can usually be established by comparing routed flows to downstream observations. Alternatively, the `Lag` can be estimated using the reach length and wave speed in the reach. Without any other information, `K` can be set to `Lag/2`.

The above discussion applies where the Lag and K parameters are single values (as implemented in the `LagK()` command). However, there are cases where the values vary by flow, which is handled by this command. The approach that is implemented is an adaptation of that described in National Weather Service River Forecast System LAG/K documentation:

- http://www.nws.noaa.gov/oh/hrl/nwsrfs/users_manual/part2/_pdf/24lagk.pdf
- http://www.nws.noaa.gov/ohd/hrl/nwsrfs/users_manual/part5/_pdf/533lagk.pdf

VariableLagK States

States can be saved to a table during analysis using the `StateSaveDateTime` and `StateSaveInterval` parameters, as long as state table parameters are also provided. States are saved at any date/time of processing that match the requested save date/time. State values in the state table are saved as a JSON string, as illustrated in the following example (line breaks inserted for readability):

```
{
      "lagInterval": "3Hour",
      "units": "CMS",
      "currentLaggedInflow": "1.0",
      "currentOutflow": ".4966",
      "currentStorage": "11.92",
      "qtLag": [ 0.0, 1.0, 11.93 ]
}
```

The following table explains the meaning of the states:

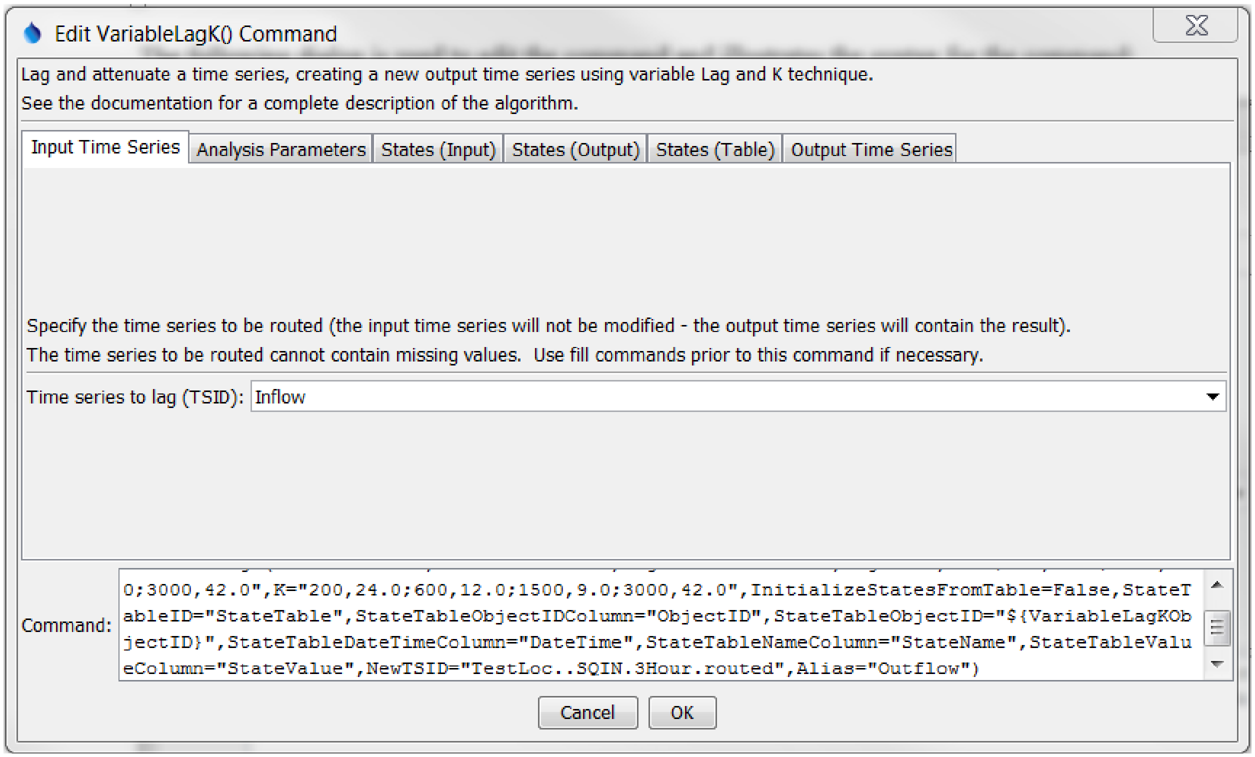
VariableLagK States

Parameter	Description	Default
`currentLaggedInflow`	The lagged inflow corresponding to the state save date/time.	`0`
`currentOutflow`	The outflow corresponding to the state save date/time.	`0`
`currentStorage`	The storage corresponding to the state save date/time.	`0`
`lagInterval`	The interval between `qtLag` values, which matches the input time series interval. This interval can be used as a check to confirm that states that were saved can be used later.	Assumed to be data interval for input time series.
`qtLag`	An array of QT lagged flows, where the last value corresponds to the save date/time and the first value is earliest in time. The number of values depends on the largest lag value from input. If specified with the `InitialQTLag` parameter the values will be positioned at the end of the internal array, with zeros filling earliest values if necessary.	`0` for all values
`units`	Data units for states, which can be used to check against VariableLagK command that is attempting to use the states.	Assumed to be consistent with input time series.

The saved states can then be used to restart the analysis, as follows:

- **Output period** – The output period used to create the output time series is defined by the `OutputStart` and `OutputEnd` parameters. If not specified, the full time series period (or global output period) is used.
- **Date/time for initial states** – The date/time corresponding to `OutputStart` is used to retrieve states that have been previously saved to the state table.
- **Initialize states** – States are initialized for the algorithm as follows:
    - If the state table should be used to initialize states (`InitializeStatesFromTable=True`), get the initial states from the state table by matching the `StateTableObjectID`, `OutputStart`, and `TableStateNameColumn` (for example state name of `VariableLagK`).
    - If no states have been saved in the state table, then the initial values will be set to the values from parameters `InitialLaggedInflow`, `InitialOutflow`, `InitialStorage`, and `InitialQTLag`.
    - If no initial states are specified, the values will default to zero. This will cause a start-up condition that will converge to accurate routed output after several intervals.

The following dialog is used to edit the command and illustrates the syntax for the command:



Edit VariableLagK() Command

Lag and attenuate a time series, creating a new output time series using variable Lag and K technique.
See the documentation for a complete description of the algorithm.

Specify the time series to be routed (the input time series will not be modified - the output time series will contain the result).
The time series to be routed cannot contain missing values. Use fill commands prior to this command if necessary.

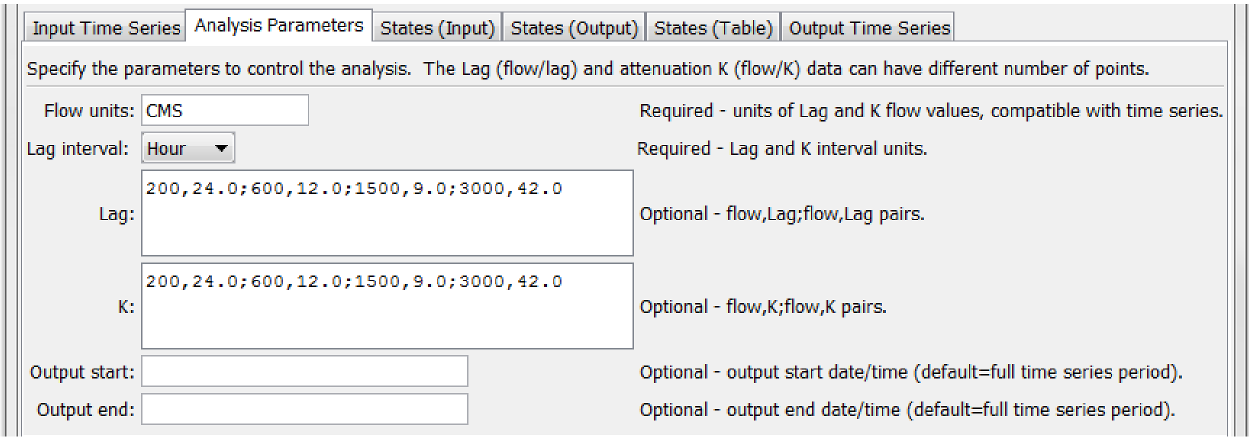
Time series to lag (TSID): Inflow

Command:

```
0;3000,42.0",K="200,24.0;600,12.0;1500,9.0;3000,42.0",InitializeStatesFromTable=False,StateT
ableID="StateTable",StateTableObjectIDColumn="ObjectID",StateTableObjectID="${VariableLagKOb
jectID}",StateTableDateTimeColumn="DateTime",StateTableNameColumn="StateName",StateTableValu
eColumn="StateValue",NewTSID="TestLoc..SQIN.3Hour.routed",Alias="Outflow")
```

ViriableLagK

VariableLagK() Command Editor Showing Input Time Series Parameter



Specify the parameters to control the analysis. The Lag (flow/lag) and attenuation K (flow/K) data can have different number of points.

Flow units: CMS Required - units of Lag and K flow values, compatible with time series.

Lag interval: Hour Required - Lag and K interval units.

Lag: 200,24.0;600,12.0;1500,9.0;3000,42.0 Optional - flow,Lag;flow,Lag pairs.

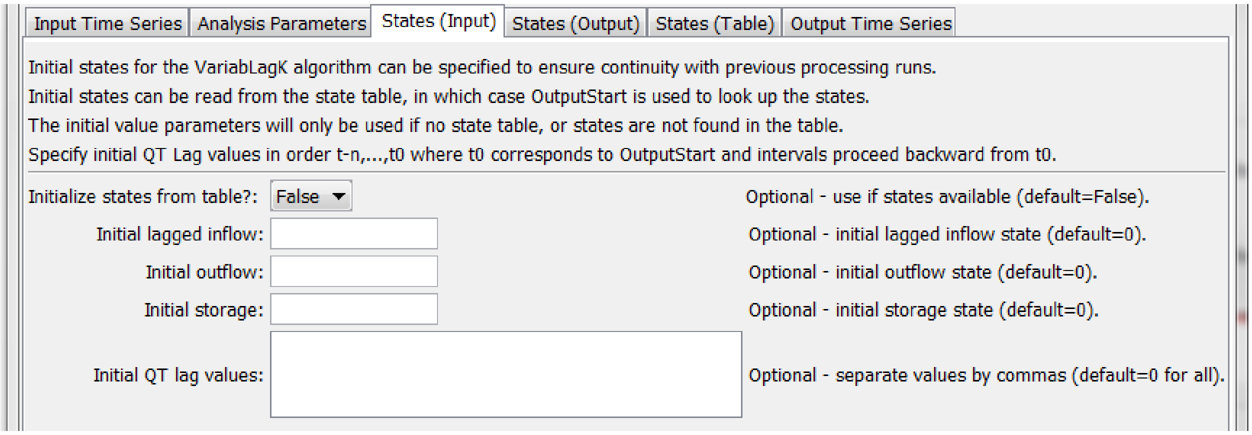
K: 200,24.0;600,12.0;1500,9.0;3000,42.0 Optional - flow,K;flow,K pairs.

Output start: Optional - output start date/time (default=full time series period).

Output end: Optional - output end date/time (default=full time series period).

ViriableLagK_Analysis

VariableLagK() Command Editor Showing Analysis Parameters

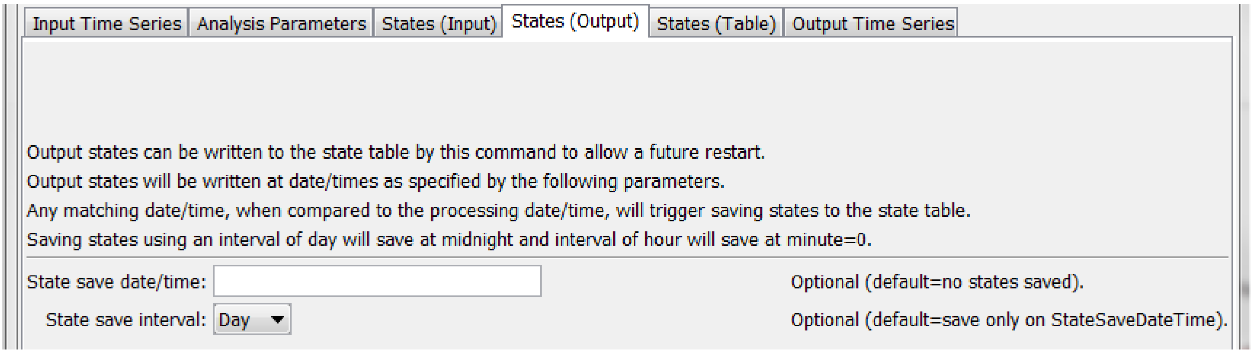


Initial states for the VariabLagK algorithm can be specified to ensure continuity with previous processing runs.
Initial states can be read from the state table, in which case OutputStart is used to look up the states.
The initial value parameters will only be used if no state table, or states are not found in the table.
Specify initial QT Lag values in order t-n,...,t0 where t0 corresponds to OutputStart and intervals proceed backward from t0.

Initialize states from table?:	False	Optional - use if states available (default=False).
Initial lagged inflow:		Optional - initial lagged inflow state (default=0).
Initial outflow:		Optional - initial outflow state (default=0).
Initial storage:		Optional - initial storage state (default=0).
Initial QT lag values:		Optional - separate values by commas (default=0 for all).

ViriableLagK_StatesInput

VariableLagK() Command Editor Showing Input State Parameters

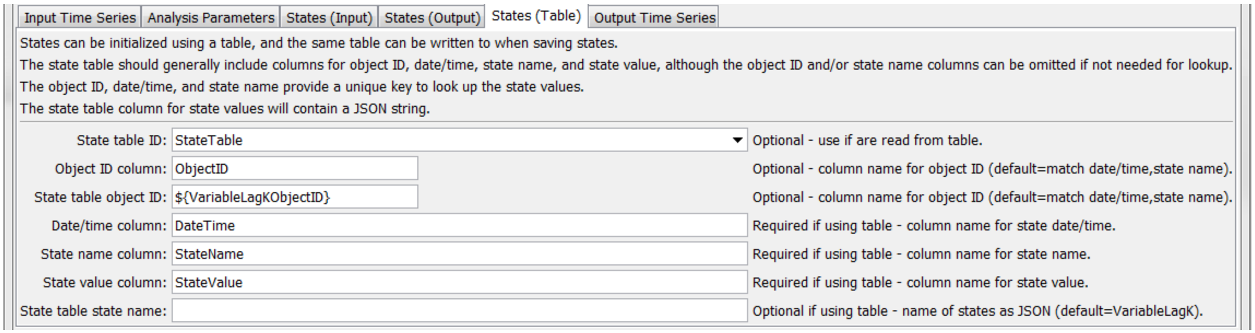


Output states can be written to the state table by this command to allow a future restart.
Output states will be written at date/times as specified by the following parameters.
Any matching date/time, when compared to the processing date/time, will trigger saving states to the state table.
Saving states using an interval of day will save at midnight and interval of hour will save at minute=0.

State save date/time:		Optional (default=no states saved).
State save interval:	Day	Optional (default=save only on StateSaveDateTime).

ViriableLagK_StatesOutput

VariableLagK() Command Editor Showing Output State Parameters

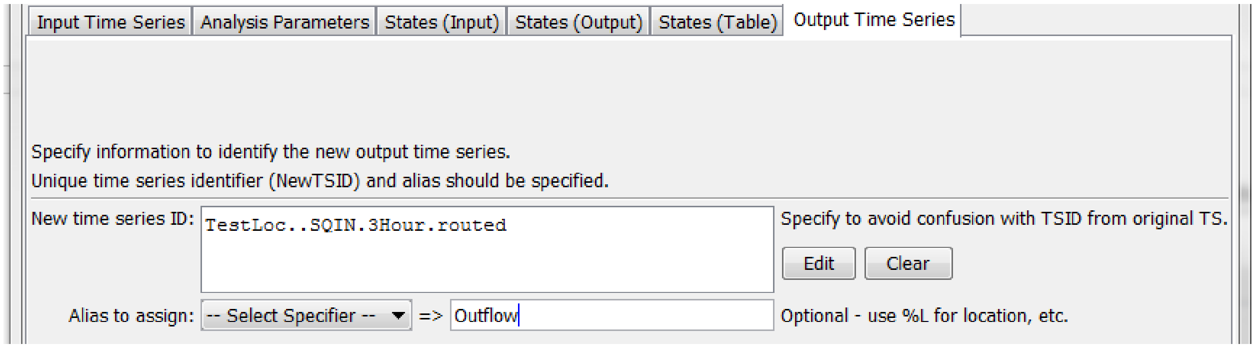


States can be initialized using a table, and the same table can be written to when saving states.
The state table should generally include columns for object ID, date/time, state name, and state value, although the object ID and/or state name columns can be omitted if not needed for lookup.
The object ID, date/time, and state name provide a unique key to look up the state values.
The state table column for state values will contain a JSON string.

State table ID:	StateTable	Optional - use if are read from table.
Object ID column:	ObjectID	Optional - column name for object ID (default=match date/time,state name).
State table object ID:	${VariableLagKObjectID}	Optional - column name for object ID (default=match date/time,state name).
Date/time column:	DateTime	Required if using table - column name for state date/time.
State name column:	StateName	Required if using table - column name for state name.
State value column:	StateValue	Required if using table - column name for state value.
State table state name:		Optional if using table - name of states as JSON (default=VariableLagK).

ViriableLagK_StatesTable

VariableLagK() Command Editor Showing State Table Parameters



Specify information to identify the new output time series.
Unique time series identifier (NewTSID) and alias should be specified.

New time series ID: TestLoc..SQIN.3Hour.routed

Specify to avoid confusion with TSID from original TS.

Alias to assign: -- Select Specifier -- => Outflow

Optional - use %L for location, etc.

ViriableLagK_Output

VariableLagK() Command Editor Showing Output Parameters

The command syntax is as follows:

```
VariableLagK(Parameter=Value,…)
```

Command Parameters

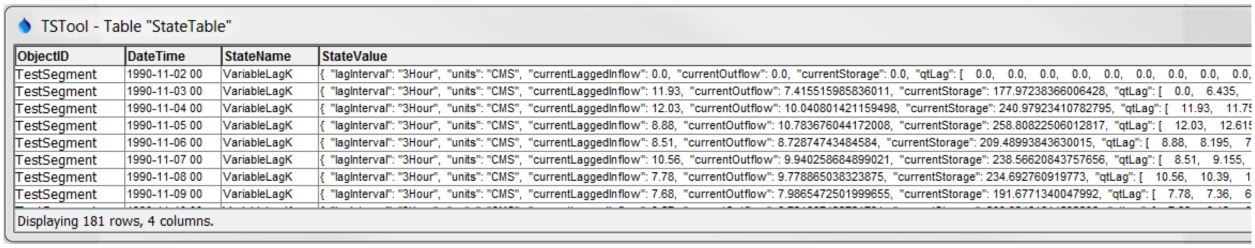
Parameter	Description	Default
`TSID`	Identifier or alias for the time series to be routed. It is assumed that this series describes an instantaneous flow. Due to the lagging, the first data values required for the computation of O_2 are not available within this time series and are therefore set to values set in the `InflowStates` parameter. Can be specified with `${Property}`.	None – must be specified.
`FlowUnits`	The units of the flow data specified in the `Lag` and `K` tables. These units must be compatible with the time series units. The table values will be converted to the time series units if the units are not the same. Can be specified with `${Property}`.	None – must be specified.
`LagInterval`	The base interval for the time data specified in the `Lag` and `K` tables. The interval must be compatible with the time series base interval. The table values will be converted to the time series time interval if the intervals are not the same. For example, table data specified in `Hour` base interval will be converted to `Minute` if the time series being routed contains NMinute data.	None – must be specified.
`Lag`	Flow value and lag time pairs to control routing. The units of the data values are as specified by the `FlowUnits` parameter (see above). The units of the lag are time as specified by the `LagInterval` parameter. The `Lag` value is not required to be evenly divisible by the time step interval; values in the time series between time steps will be linearly interpolated. Can be specified with `${Property}`. Use commas and semi-colons to separate values, for example: `100.0,10;200.0,20`	None – must be specified.

Parameter	Description	Default
`K`	Flow value and K time pairs to control routing. The attenuation factor K is applied to the wave. The units of `K` are time as specified by the `LagInterval` parameter. Can be specified with `${Property}`. Use commas and semi-colons to separate values, for example: `100.0,5;200.0,10`	None – must be specified.
`OutputStart`	The starting date/time for the output time series. States will be initialized relative to this date/time, which is referred to as t_0. Can be specified with `${Property}`.	Full input time series period is processed.
`OutputEnd`	The ending date/time for the output time series. Can be specified with `${Property}`.	Full input time series period is processed.
`InitializeStatesFromTable`	Indicate whether to initialize states from the state table (can be specified with `${Property}`): • `False` – do not attempt to initialize states from the state table, in which case the following parameters are used to initialize, if specified • `True` – attempt to initialize states from the state table and if states are not found, use the following parameters to initialize, if specified	`False`
`InitialLaggedInflow`	Initial state value for lagged inflow. Can be specified using `${Property}`.	`0`
`InitialOutflow`	Initial state value for outflow. Can be specified using `${Property}`.	`0`
`InitialStorage`	Initial state value for storage. Can be specified using `${Property}`.	`0`
`InitialQTLag`	Comma-delimited list of initial QT Lag array values. The order of the values is earliest to latest. The array can specify up to (`Lag`/multiplier) + 1 values (1 is for bounding end point). For example, a 10 minute time series data interval with a `LAG` of 30 must be provided with 30/10 + 1 = 4 inflow state values. Specifying values that are not consistent with the Lag and K parameters will result in oscillation. Can be specified with `${Property}`.	`0` for all values.
`StateSaveDateTime`	Specify a date/time string, for example YYYY-MM-DD HH for a date/time to save states. The precision of the date/time should be consistent with time series interval. Can be specified with `${Property}`.	Do not save states.
`StateSaveInterval`	Specify an interval to save states: • `Hour` – save states when hour is zero • `Day` – save states at midnight of each day	Do not save states.
`StateTableID`	The table identifier for the state table. Can be specified using `${Property}`.	Default states are used.
`StateTableObjectIDColumn`	State table column name for object ID, used to the computation unit with states. For example, the object ID	Will not be used to lookup states.

Parameter	Description	Default
	might be a stream reach identifier or location. Can be specified with `${Property}`.	
`StateTableObjectID`	The object identifier, used to match the row in the state table. Can be specified with `${Property}`.	Will not be used to lookup states.
`StateTableDateTimeColumn`	State table column containing state date/time. Can be specified with `${Property}`.	Required for table.
`StateTableNameColumn`	State table column containing state name. Can be specified with `${Property}`.	Will not be used to lookup states.
`StateTableValueColumn`	State table column containing state value. Can be specified with `${Property}`.	Required for table.
`StateTableStateName`	Name of the state in the `StateTableNameColumn` for states, used to match rows. The text should match that of the **VariableLagK States** section above. Can be specified with `${Property}`.	Will not be used to lookup states.
`NewTSID`	Identifier for the new (routed) time series. This is required to ensure that the internal identifier for the time series is unique and accurate for the data. The interval of the identifier must be the same as for the time series specified by `TSID`. Can be specified with `${Property}`.	None – must be specified.
`Alias`	The alias that will be assigned to the new time series, which can use `${Property}` and `${ts:Property}`.	No alias will be assigned.

A sample command file is as follows (commands to read time series are omitted):

```
# Test routing at 3 hour interval
StartLog(LogFile="Results/Test_VariableLagK_3hr.TSTool.log")
# Read NWSCard input file
ReadNwsCard(InputFile="Data\3HR_INPUT.SQIN",Alias=”Inflow”)
#
# Route using the same routing parameters used in the mcp3 input deck
# (metric units: Lag(hrs) K(hrs) Q(cms)
# Lag
# K
#    24.0   200.0    12.0  600.00     9.0  1500.0    42.0  3000.0
#    24.0   200.0    12.0  600.00     9.0  1500.0    42.0  3000.0
#
NewTable(TableID="StateTable",
  Columns="ObjectID,string;DateTime,datetime;StateName;string;StateValue;string")
SetProperty(PropertyName="VariableLagKObjectID",PropertyType=String,
  PropertyValue="TestSegment")
VariableLagK(TSID="Inflow",FlowUnits="CMS",LagInterval="Hour",
  Lag="200,24.0;600,12.0;1500,9.0;3000,42.0",
  K="200,24.0;600,12.0;1500,9.0;3000,42.0",InitializeStatesFromTable=False,
  StateTableID="StateTable",StateTableObjectIDColumn="ObjectID",
  StateTableObjectID="${VariableLagKObjectID}",StateTableDateTimeColumn="DateTime",
  StateTableNameColumn="StateName",StateTableValueColumn="StateValue",
  NewTSID="TestLoc..SQIN.3Hour.routed",Alias="Outflow")
```



TSTool - Table "StateTable"

ObjectID	DateTime	StateName	StateValue
TestSegment	1990-11-02 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 0.0, "currentOutflow": 0.0, "currentStorage": 0.0, "qtLag": [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TestSegment	1990-11-03 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 11.93, "currentOutflow": 7.415515985836011, "currentStorage": 177.97238366006428, "qtLag": [0.0, 6.435,
TestSegment	1990-11-04 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 12.03, "currentOutflow": 10.040801421159498, "currentStorage": 240.97923410782795, "qtLag": [11.93, 11.75
TestSegment	1990-11-05 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 8.88, "currentOutflow": 10.783676044172008, "currentStorage": 258.80822506012817, "qtLag": [12.03, 12.615
TestSegment	1990-11-06 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 8.51, "currentOutflow": 8.72874743484584, "currentStorage": 209.48993843630015, "qtLag": [8.88, 8.195, 7
TestSegment	1990-11-07 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 10.56, "currentOutflow": 9.940258684899021, "currentStorage": 238.56620843757656, "qtLag": [8.51, 9.155,
TestSegment	1990-11-08 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 7.78, "currentOutflow": 9.778865038323875, "currentStorage": 234.692760919773, "qtLag": [10.56, 10.39, 1
TestSegment	1990-11-09 00	VariableLagK	{ "lagInterval": "3Hour", "units": "CMS", "currentLaggedInflow": 7.68, "currentOutflow": 7.9865472501999655, "currentStorage": 191.6771340047992, "qtLag": [7.78, 7.36, 6

Displaying 181 rows, 4 columns.

ViriableLagK_StateTable_Out

Output State Table

This page is intentionally blank.