Introduction to the *streamDAG* Package

Ken Aho
Department of Biological Sciences
Idaho State University
ahoken@isu.edu

July 20, 2022

Contents

1 The *streamDAG* Package

The *streamDAG* package provides indices and tools for analyzing directed acyclic graph (DAG) representations of intermittent stream networks. The package is built under the basic idiom of the *igraph* package (Csardi & Nepusz, 2006). A focus of many *streamDAG* algorithms is the measurement of graph centrality, complexity, and connectivity. While many of these measures are purely topological, several weighted DAG indices will provide outcomes similar or identical to those of existing hydrological (non-graph-theoretic) measures for streams.

The *streamDAG* package is currently housed in a GitHub repository: `https://github.com/moondog1969/streamDAG`. The package can be installed from the R console directly after installing the package *devtools*. In particular, use:

```
library(devtools)
install_github("moondog1969/streamDAG")
```

After installing *streamDAG*, the package can be loaded into R conventionally.

```
library(streamDAG)
```

2 Introductory Examples of Usage

2.1 Data Outlay

We will demonstrate the *streamDAG* package using Murphy Cr., an intermittent stream in the Reynolds Cr. experimental watershed in southwestern Idaho (Fig 1) where stream presence was recorded at 25 nodes, corresponding to 27 stream segments (including remaining segments above and below the measured terminal nodes).

Figure 1: The Reynolds Cr. experimental watershed in SW Idaho.

Purely topological analyses can be conducted using only an *igraph* codified stream network. Much more flexibility is possible, however, by defining actual spatial coordinates and graph weighting data, including stream lengths, and information about stream segment presence (wet) or absence (dry). Below is a codification of Murphy Creek based on nodes established by Warix *et al.* (2021). The code `IN_N --+ M1984` indicates that the stream flows from node `IN_N` to node `M1984`, and so on.

```
murphy_spring <- graph_from_literal(IN_N --+ M1984 --+ M1909, IN_S --+ M1993,
M1993 --+ M1951 --+ M1909 --+ M1799 --+ M1719 --+ M1653 --+ M1572 --+ M1452,
M1452 --+ M1377 --+ M1254 --+ M1166 --+ M1121 --+ M1036 --+ M918 --+ M823,
M823 --+ M759 --+ M716 --+ M624 --+ M523 --+ M454 --+ M380 --+ M233 --+ M153,
M153 --+ M91 --+ OUT)
```

The *streamDAG* package contains additional Murphy Cr data including nodal spatial coordinates (UTMs), stream edge (segment) lengths, and stream edge presence absence data. Instream lengths and distances can be obtained from a number of sources including ARC-GIS and the the R package *SSN*. Stream presence can be ascertained using a number of methods, including conductivity and temperature sensors.

```
data(mur_coords) # Node UTM coords
data(mur_lengths) # Arc (stream segment) lengths
data(mur_node_pres_abs) # Node presence / absence data with explicit datetimes
data(mur_arc_pres_abs) # Arc (stream segment) simulated presence / absence data
```

Care should be taken to insure that the order of relevant rows and columns in ancillary data correspond to the order of nodes and arcs defined in the underlying graph, `G` with the functions `igraph::V` (which gives nodes) and `A` or `igraph::E` (which give arcs). Although different identifiers can be used to designate arc direction, e.g., `--+`, `--` , `|`, etc., exact names must be used for node names. For instance, here are the first six graph arc names.

```
head(A(murphy_spring))

+ 6/27 edges from 976473e (vertex names):
[1] IN_N ->M1984 M1984->M1909 M1909->M1799 IN_S ->M1993 M1993->M1951
[6] M1951->M1909
```

Note that these correspond to the identifiers for the first six stream lengths from `mur_lengths`.

```
head(mur_lengths[,1])

[1] "IN_N -> M1984"  "M1984 -> M1909" "M1909 -> M1799" "IN_S -> M1993"
[5] "M1993 -> M1951" "M1951 -> M1909"
```

2.2 Spatial Plots

It is easy to make a spatially explicit stream DAG using the *streamDAG* function `spatial.plot`. We can make a spatial plot by augmenting graph data with nodal spatial coordinates (Fig 2).

```r
x <- mur_coords[,2]; y <- mur_coords[,3]
names = mur_coords[,1]
spatial.plot(murphy_spring, x, y, names)
```

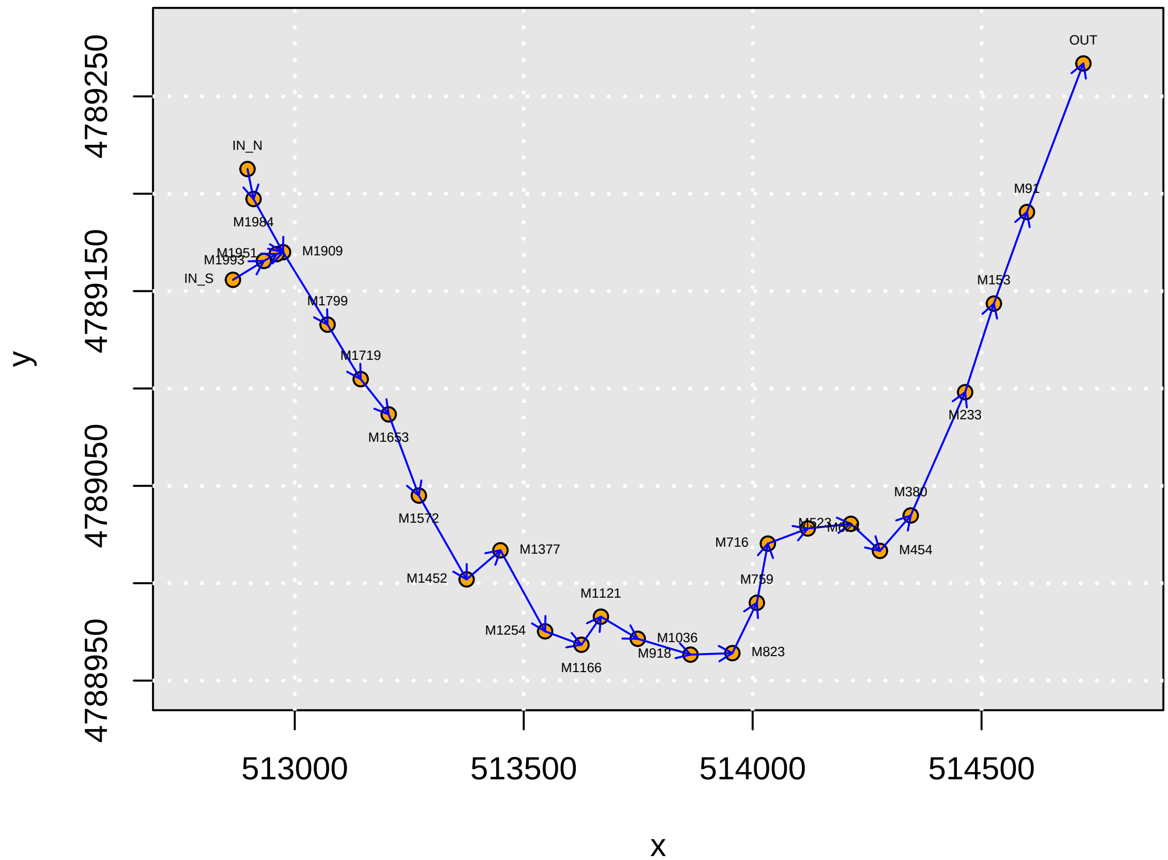

Figure 2: Spatially explicit graph of the completely wetted Murphy Cr. network, as it occurs in the spring.

Or based on ARC-GIS shapefiles. The later can be customized using *ggplot2* modifiers (Fig 3). Use of shapefiles will eliminate some of the easy to easy-to-use features in `spatial.plot` including directional arrows indicating flow and deletion of arcs and nodes with presence / absence data (see below).

```r
library(ggplot2); library(sf); library(ggrepel)
mur_sf <- st_read("Murphy_shapefile/Murphy/Murphy_Creek.shp", quiet = TRUE)
# Note that the directory "Murphy" also contains required ARC-GIS .shx,.cpg,.dbf, and .prj files.
g1 <- spatial.plot(murphy_spring, x, y, names, plot.shapefile = TRUE, shapefile = mur_sf)

## some ggplot customizations
g1 + expand_limits(y = c(4788562,4789700)) +
  theme(plot.margin = margin(t = 0, r = 10, b = 0, l = 0)) +
  geom_text_repel(data = mur_coords, aes(x = x, y = y, label = Object.ID), colour = "black",
                  size = 1.6, box.padding = unit(0.3, "lines"), point.padding =
                    unit(0.25, "lines"))
```

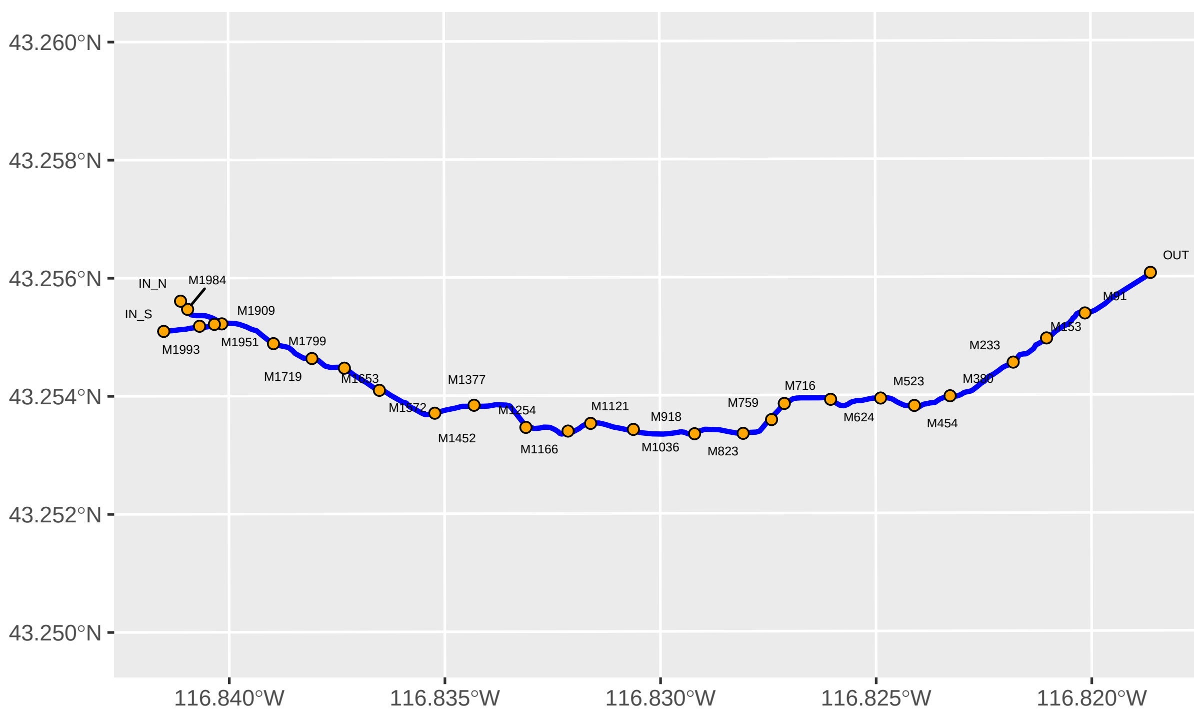

Figure 3: Example of using a shapefile with `spatial.plot`.

2.3 Tracking Intermittency

The activity of stream nodes and/or arcs (segments) can be easily tracked in stream graphs based on STIC or conductivity data using the *streamDAG* functions `delete.arcs.pa` and `delete.nodes.pa`.

For instance, the dataset `mur_node_pres_abs` contains a subset of nodal presence absence data for Murphy Cr in 2019. Below we see rows for time series observations 650 to 655.

```
mur_node_pres_abs[650:655,]

            Datetime IN_N M1984 M1909 IN_S M1993 M1951 M1799 M1719 M1653 M1572
6491  8/9/2019 22:30    0     0     0    0     0     0     1     0     0     1
6501  8/10/2019 1:00    0     0     0    0     0     0     1     0     0     1
6511  8/10/2019 3:30    0     0     0    0     0     0     1     0     0     1
6521  8/10/2019 6:00    0     0     0    0     0     0     1     0     0     1
6531  8/10/2019 8:30    0     0     0    0     0     0     1     0     0     1
6541 8/10/2019 11:00    0     0     0    0     0     0     1     0     0     1
     M1452 M1377 M1254 M1166 M1121 M1036 M918 M823 M759 M716 M624 M523 M454
6491     0     0     1     0     0     1    1    0    0    1    1    1    0
6501     0     0     1     0     0     1    1    0    0    1    1    1    0
6511     0     0     1     1     0     1    1    0    0    1    1    1    0
6521     0     0     1     1     0     1    1    0    0    1    1    1    0
6531     0     0     1     1     0     1    1    0    0    1    1    1    1
6541     0     0     1     1     0     1    1    0    0    1    1    1    1
     M380 M233 M153 M91 OUT
6491    0    1    1   1   1
6501    0    1    1   1   1
6511    0    1    1   1   1
6521    0    1    1   1   1
6531    0    1    1   1   1
6541    0    1    1   1   1
```

Subsetting `murphy_spring` arcs based on the nodal observations at 8/9/2019 22:30 we have:

```
npa <- mur_node_pres_abs[650,][,-1]
G1 <- delete.nodes.pa(murphy_spring, npa)
```

The resulting spatial plot is shown as Fig 4. Note that nodes without water are now omitted from the graph. Arcs missing one or more bounding nodes are also omitted.

```
spatial.plot(G1, x, y, names)
```

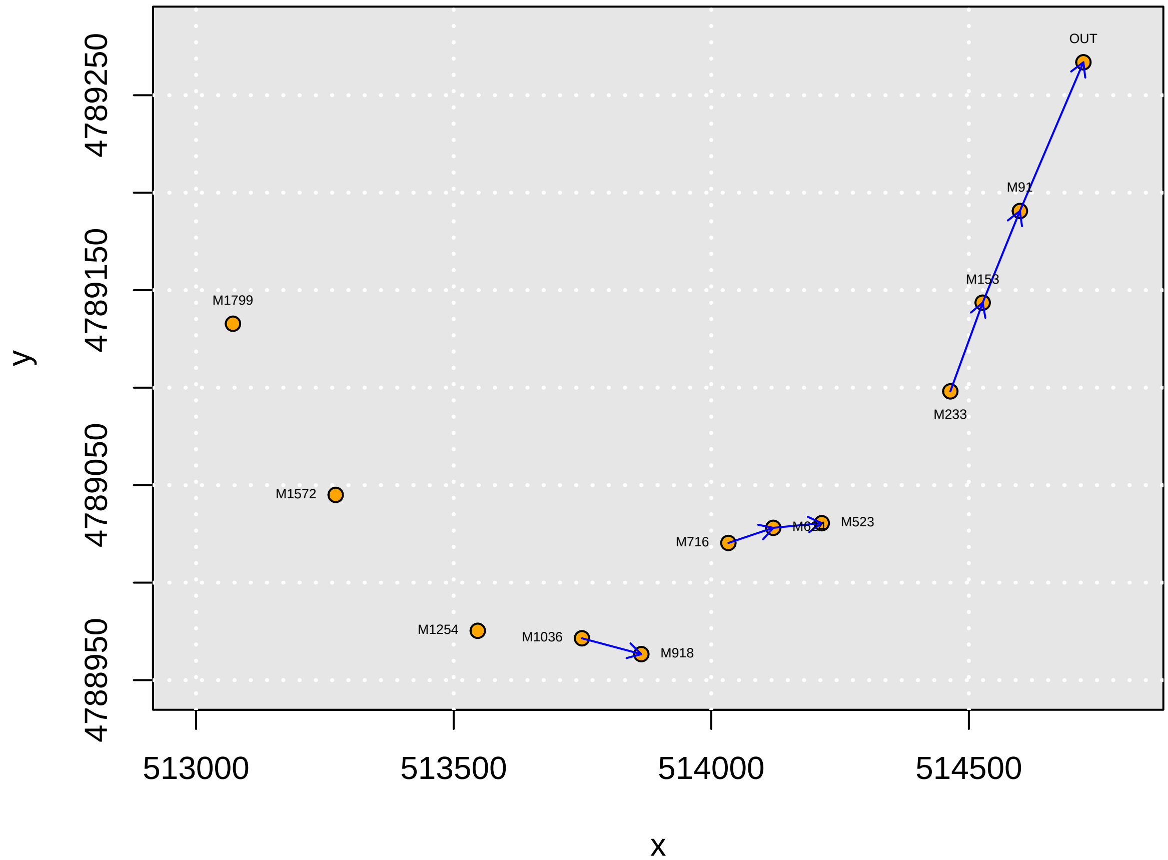

Figure 4: Plotting a subgraph of `murphy_spring` after application of `delete.nodes.pa`.

One can also subset graphs based on arc presence / absence data. The dataframe `mur_arc_pres_abs` contains simulated multivariate Bernoulli data for Murphy Cr. arcs based on 2019 nodal data.

```
head(mur_arc_pres_abs) # 1st 6 rows of data

  IN_N-->M1984 M1984-->M1909 M1909-->M1799 IN_S-->M1993 M1993-->M1951
1            1             0             1            0             0
2            1             0             1            0             1
3            1             0             1            0             0
4            0             1             1            1             0
5            0             0             1            0             0
6            0             1             0            0             0
  M1951-->M1909 M1799-->M1719 M1719-->M1653 M1653-->M1572 M1572-->M1452
1             0             1             0             1             1
```

```
2            0            0            1            1            1
3            1            0            1            0            0
4            1            1            1            1            1
5            0            0            0            1            1
6            1            0            0            1            0
  M1452-->M1377 M1377-->M1254 M1254-->M1166 M1166-->M1121 M1121-->M1036
1            0            1            1            0            0
2            0            1            0            1            1
3            1            1            1            0            0
4            1            1            0            1            1
5            0            1            0            1            1
6            1            1            1            0            0
  M1036-->M918 M918-->M823 M823-->M759 M759-->M716 M716-->M624 M624-->M523
1           1           0           0           1           1           1
2           1           1           1           1           1           1
3           0           1           0           1           1           1
4           1           1           1           1           1           0
5           1           0           0           1           1           1
6           1           0           0           0           1           1
  M523-->M454 M454-->M380 M380-->M233 M233-->M153 M153-->M91 M91-->OUT
1           1           0           0           0          1         1
2           1           1           1           0          1         1
3           1           1           0           1          1         1
4           1           1           1           1          1         1
5           1           1           1           1          1         1
6           1           1           0           1          1         1
```

Subsetting `murphy_spring` arcs based on the 6th simulated multivariate dataset we have:

```
G <- delete.arcs.pa(murphy_spring, mur_arc_pres_abs[6,])
```

The resulting spatial plot is shown in Fig 5. Note that all nodes are plotted, but plotted arcs are limited to those with recorded activity.

```r
spatial.plot(G, x, y, names)
```

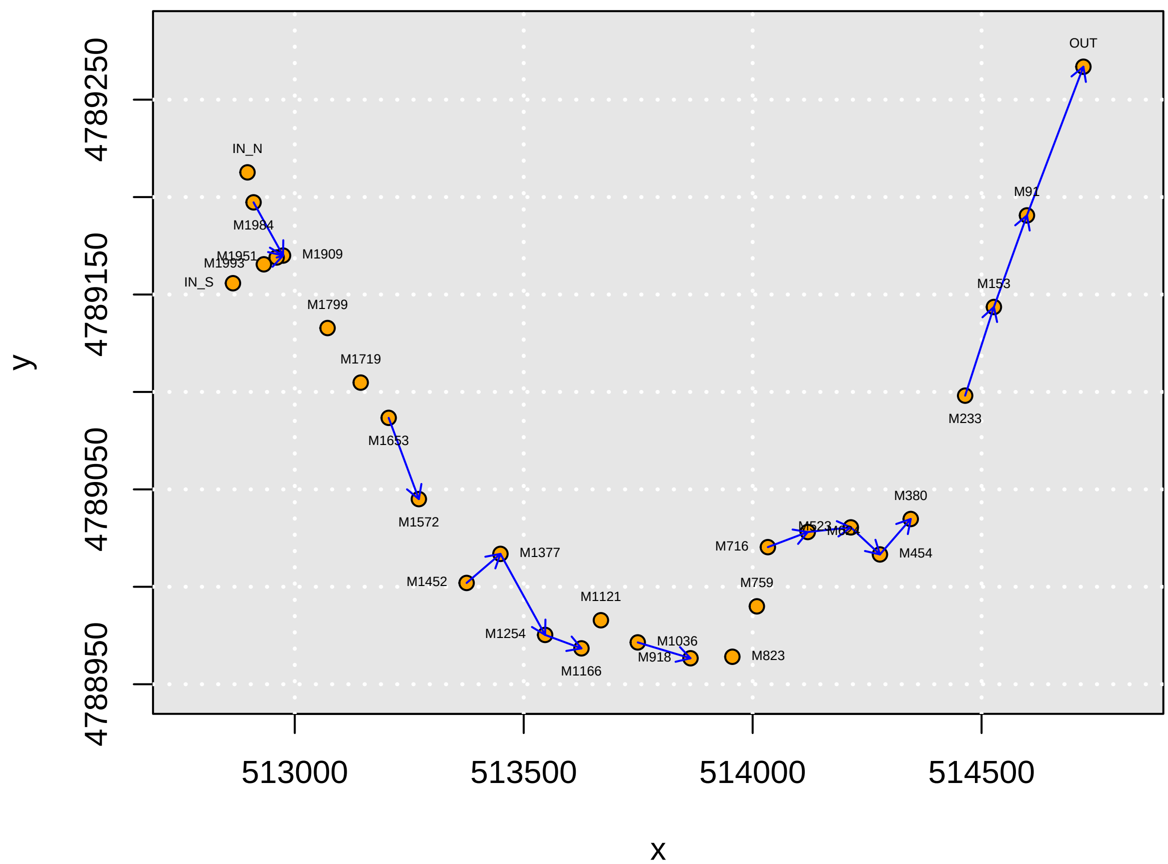

Figure 5: Plotting a subgraph of `murphy_spring` after application of `delete.arcs.pa`.

2.4 Purely Topological Measures for Stream DAGs

There are many measures useful for describing and distinguishing intermittent stream networks that are based solely on graph topological features (i.e., the presence or absence of nodes and adjoining arcs). These can be subset into local measures that describe the characteristics of individual stream nodes or arcs, and global measures that summarize the characteristics of an entire network, i.e., the entire graph.

2.4.1 Local measures

A number of local measures are included in the *streamDAG* function `local.summary`. The function only requires an *igraph* graph object.

```r
local <- local.summary(murphy_spring)
round(local, 2)
```

```
                          IN_N M1984 M1909 IN_S M1993 M1951 M1799 M1719 M1653
indegree                   0.0     1  2.00  0.0     1   1.0  1.00  1.00  1.00
alpha.centrality           1.0     2  6.00  1.0     2   3.0  7.00  8.00  9.00
imp.closeness.centrality   0.0    27 90.00  0.0    27  40.5 78.75 77.40 78.30
n.paths                    0.0     1  5.00  0.0     1   2.0  6.00  7.00  8.00
path.length.mean           NaN     1  1.80  NaN     1   1.5  2.50  3.14  3.75
path.length.var             NA    NA  0.70   NA    NA   0.5  1.10  1.81  2.79
path.length.skew           0.0    NA  0.51  0.0    NA   NaN  0.00 -0.35 -0.46
path.length.kurt          -0.5    NA -0.61 -0.5    NA   NaN -0.25 -0.30 -0.60
height                     0.0     1  3.00  0.0     1   2.0  4.00  5.00  6.00
                          M1572 M1452 M1377 M1254 M1166 M1121 M1036  M918  M823
indegree                   1.00  1.00  1.00  1.00  1.00  1.00  1.00  1.00  1.00
alpha.centrality          10.00 11.00 12.00 13.00 14.00 15.00 16.00 17.00 18.00
imp.closeness.centrality 79.91 81.74 83.61 85.46 87.24 88.94 90.57 92.12 93.60
n.paths                    9.00 10.00 11.00 12.00 13.00 14.00 15.00 16.00 17.00
path.length.mean           4.33  4.90  5.45  6.00  6.54  7.07  7.60  8.12  8.65
path.length.var            4.00  5.43  7.07  8.91 10.94 13.15 15.54 18.12 20.87
path.length.skew          -0.47 -0.44 -0.41 -0.37 -0.34 -0.30 -0.28 -0.25 -0.23
path.length.kurt          -0.84 -1.01 -1.12 -1.19 -1.24 -1.27 -1.29 -1.30 -1.31
height                     7.00  8.00  9.00 10.00 11.00 12.00 13.00 14.00 15.00
                           M759  M716  M624  M523   M454   M380   M233   M153
indegree                   1.00  1.00  1.00  1.00   1.00   1.00   1.00   1.00
alpha.centrality          19.00 20.00 21.00 22.00  23.00  24.00  25.00  26.00
imp.closeness.centrality 95.01 96.36 97.64 98.88 100.06 101.20 102.29 103.34
n.paths                   18.00 19.00 20.00 21.00  22.00  23.00  24.00  25.00
path.length.mean           9.17  9.68 10.20 10.71  11.23  11.74  12.25  12.76
path.length.var           23.79 26.89 30.17 33.61  37.23  41.02  44.98  49.11
path.length.skew          -0.21 -0.20 -0.18 -0.17  -0.16  -0.15  -0.14  -0.13
path.length.kurt          -1.31 -1.31 -1.31 -1.31  -1.31  -1.30  -1.30  -1.30
height                    16.00 17.00 18.00 19.00  20.00  21.00  22.00  23.00
                            M91    OUT
indegree                   1.00   1.00
alpha.centrality          27.00  28.00
imp.closeness.centrality 104.35 105.33
n.paths                   26.00  27.00
path.length.mean          13.27  13.78
path.length.var           53.40  57.87
path.length.skew          -0.12  -0.11
path.length.kurt          -1.30  -1.29
height                    24.00  25.00
```

A graphical summary based only on measures with complete cases and standardized outcomes is shown in Fig 6. Nodes along the x-axis are sorted based on their order in the `murphy_spring` *igraph* object, which roughly corresponds to their order from sources to sink. In general, nodes increase in information and importance as distance to the sink decreases. Note, however, the "unusual" importance of M1909 due to its location at a confluence (Fig 2).

```
library(RColorBrewer)
cols <- brewer.pal(5,"Spectral")

cc <- complete.cases(local)
local.cc <- local[cc,]
scaled.local <- scale(t(local.cc))

barplot(t(scaled.local), col = cols, las = 2, cex.names = .6,
        beside = T, legend.text = row.names(local)[cc],
        args.legend = list(x = "topright", cex = .7,
                           bty = "n", col = cols),
        ylab = "Standardized local measures")
```

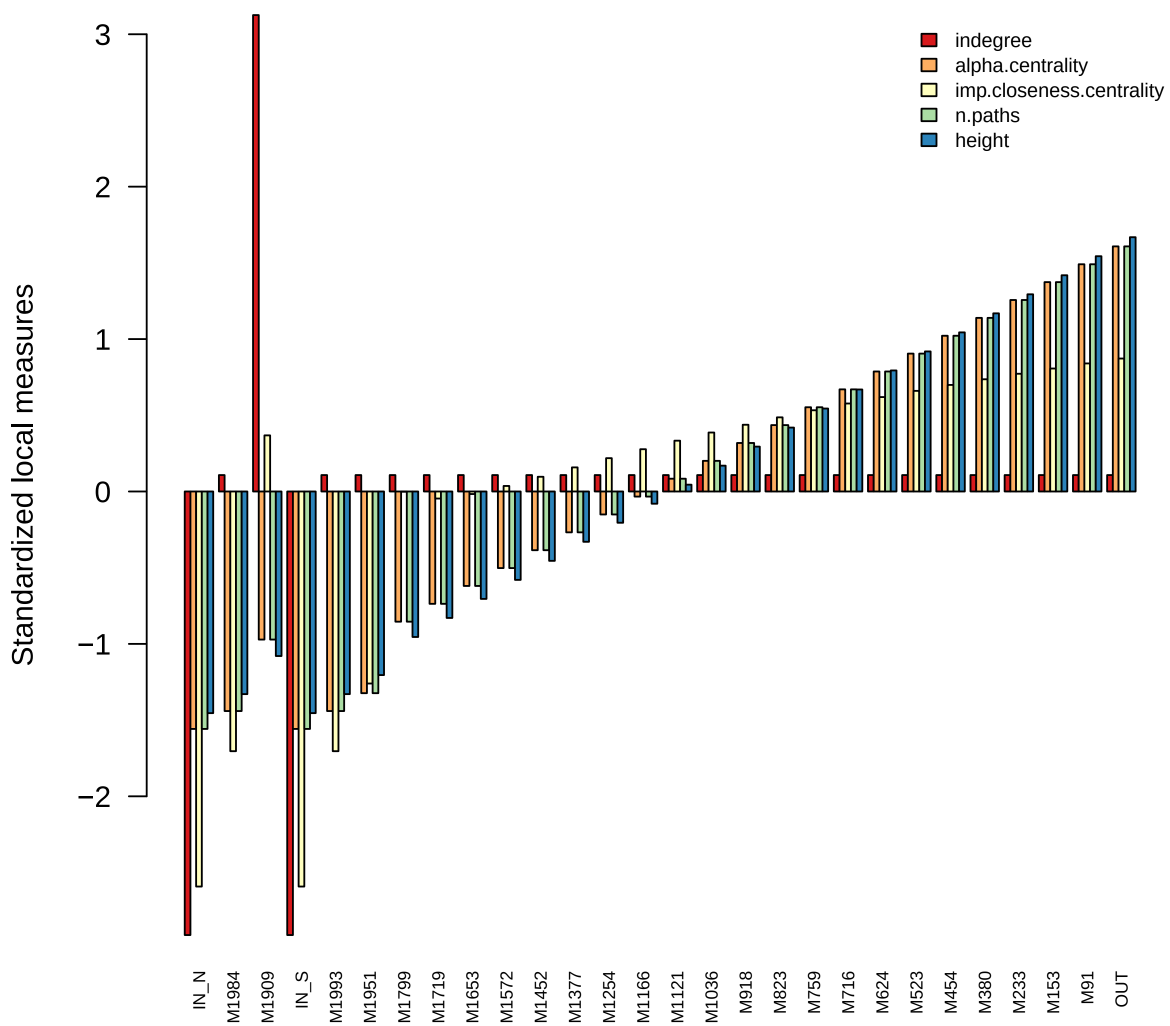

Figure 6: Local graph-theoretic summaries for `murphy_spring`.

Horizontal visibility graphs A less frequently used, but potentially important tool for measuring nodal importance is the horizontal visibility graph (Luque *et al.*, 2009). Two nodes will be *visible* if, when node data (e.g., degrees) are plotted as horizontal bars along the abscissa axis, and placed along the ordinate based on their location in the stream path, the bars can be connected with a horizontal line (Luque *et al.*, 2009). Note that the importance of M1909, as the only node where stream segments converge is again strongly emphasized (Fig 7). Weighted (see below) node visibilities can also be obtained with `multi.path.visibility`.

```
vis <- multi.path.visibility(murphy_spring, source = c("IN_N","IN_S"),
sink = "OUT")

barplot(vis$visibility.summary, las = 2, cex.names = .6, ylab = "Visible nodes",
        legend.text = c("Out", "In", "Both"),
        args.legend = list(x = "topright", title = "Degree"))
```

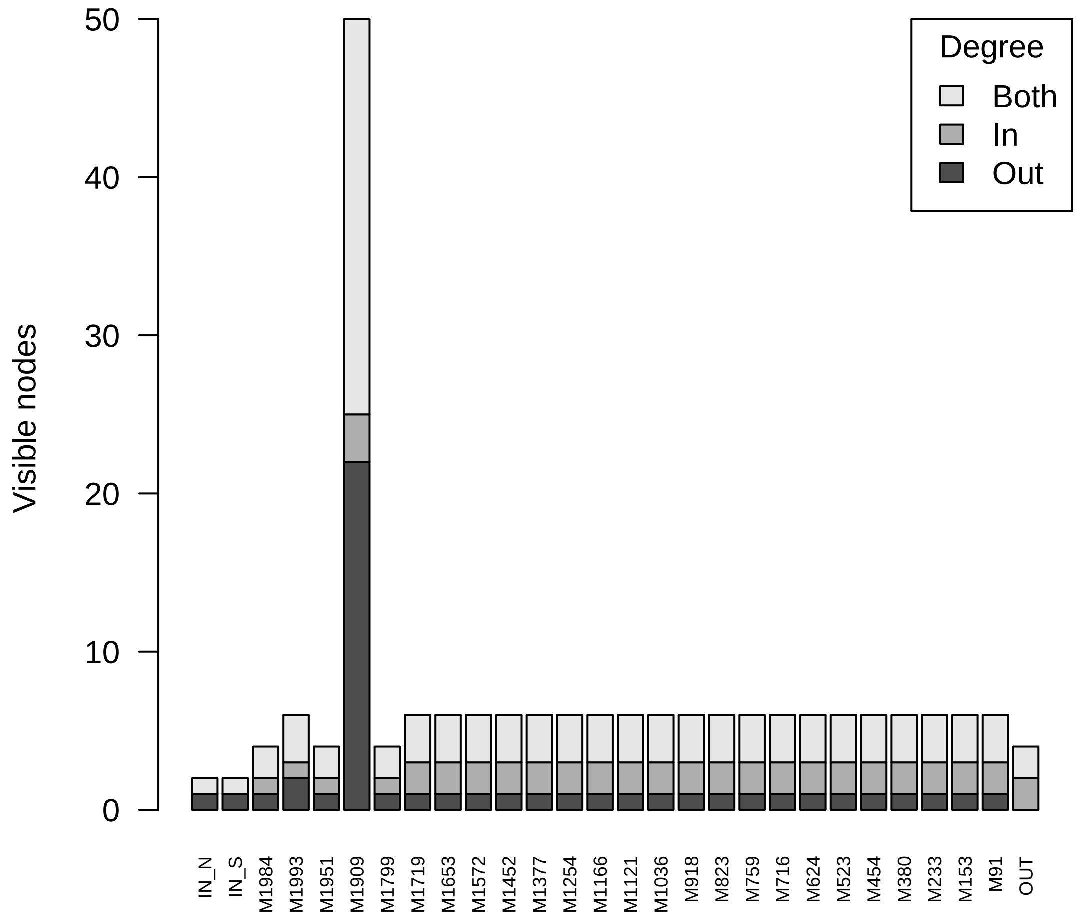

Figure 7: Nodal visibilities for `murphy_spring`.

2.4.2 Global measures

Global graph-theoretic measures allow consideration of a stream network in its entirety. Many popular global graph-theoretic measures can be called using the *streamDAG* function `global.summary`. These metrics have been designed expressly to quantify network connectivity, complexity, and, in the case of assortativity, degree trends.

```
g <- global.summary(murphy_spring, sink = "OUT")
g

                  Global.metrics
Size                 27.00000000
Diameter             25.00000000
n.paths.to.sink      27.00000000
Mean.a.centrality    14.28571429
mean.path.lengths    13.77777778
Randic               13.20710678
first.Zagreb         28.00000000
second.Zagreb        14.50000000
ABC                   0.70710678
GA                   13.44280904
Harm                 13.16666667
Assort.in.out        -0.02192645
Assort.in.in          0.03162278
```

It may be informative to track changes in global metrics (and local metrics) over time. Fig 8 shows a 100 point time series that spans the entire 2019 sampling season. As in Fig 6, metrics are standardized to have a mean of zero and a variance of one. Note higher scores for most metrics during the spring and a re-wet period during the fall, indicating higher network connectivity. An exception is in-out assortativity, which increases for a time during the drying period due to increasing homogenization of graph characteristics.

```r
# create a subset of node presence / absence data
subset <- mur_node_pres_abs[seq(1,1163, length = 100),]
subset.nodate <- subset[,-1]

# walk global.summary through node presence / absence data
global <- matrix(ncol = 13, nrow = nrow(subset))
for(i in 1:nrow(subset)){
  global[i,] <- global.summary(delete.nodes.pa(murphy_spring, subset.nodate[i,]), sink = "OUT")
}

# standardize measures
scaled.global <- scale(global)
par(mar = c(7,4.2,1.5,2))

# plot
matplot(scaled.global, xaxt = "n", type = "l", col = hcl.colors(13, palette = "spectral"),
        ylab = "Standardized global measures")
legend("bottomright", lty = 1:5, col = hcl.colors(13, palette = "spectral"),
       legend = row.names(g), cex = .6)
axis(side = 1, at = c(1,21,41,61,81,100), labels = subset[,1][c(1,21,41,61,81,100)],
     las = 2, cex.axis = .7)
mtext(side = 1, "Time", line = 6)
```

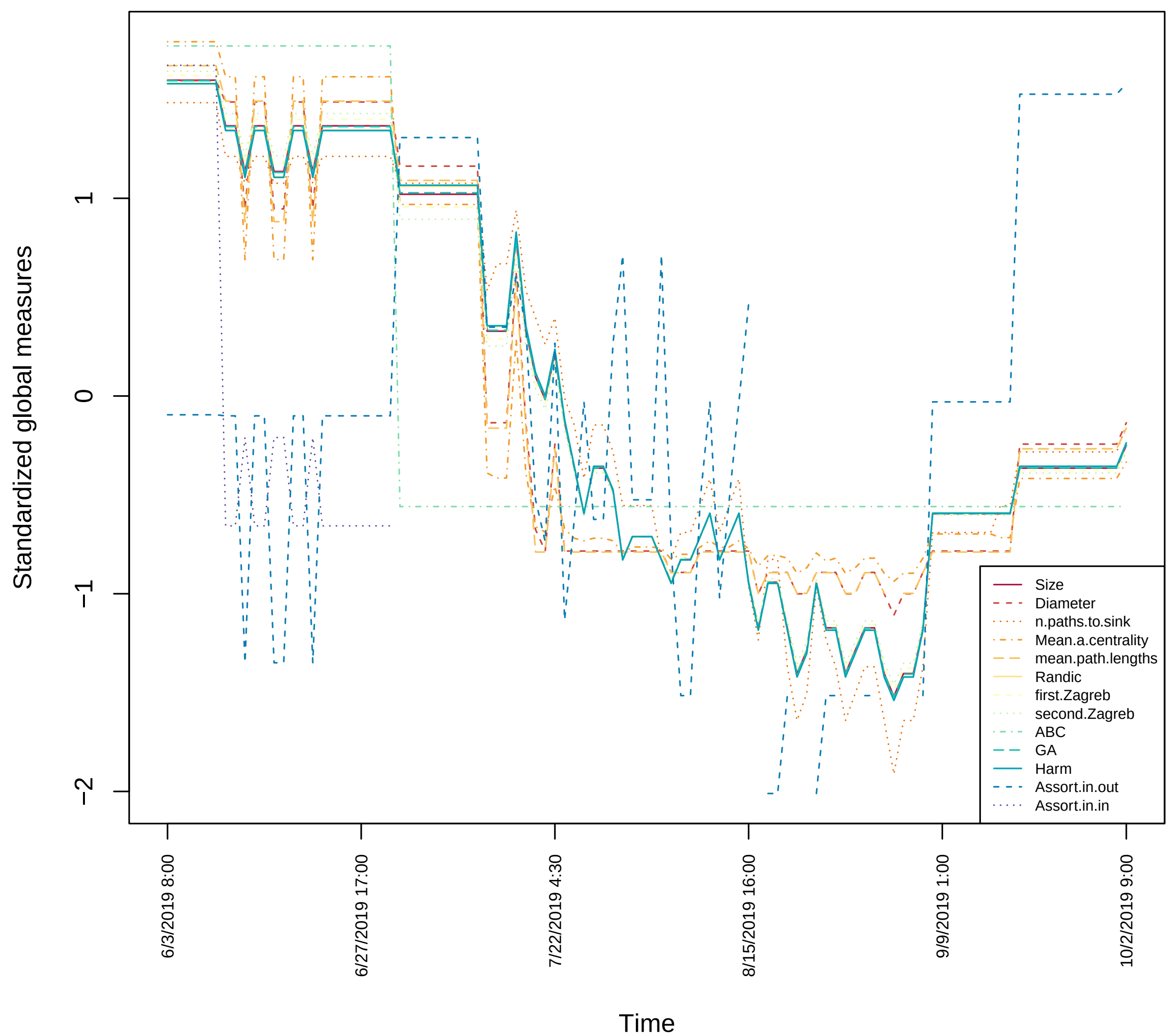

Figure 8: Global summaries for `murphy_spring`.

2.5 Reality-Driven Weighted DAG Approaches

Purely topological measures may be useful in describing the importance of individual stream nodes along with network-level connectivity and complexity. However, they will be strongly affected by user-defined node designations and abstracted from many important characteristics of stream networks. To account for this, increased realism in stream DAGs can be achieved by adding information to nodes and/or arcs in the form of *weights*. In fact, weighted DAG measures will result in indices similar or identical to existing connectivity metrics from the hydrological literature, e.g., Integral Connectivity Scale Length, (ICSL; Western *et al.* (2001)), Bernoulli stream length (Botter & Durighetto, 2020). Weighting information particularly relevant to intermittent stream DAGs include flow rates, stream lengths, and arc or node probabilities of activity. In Fig 9 Murphy Cr. arcs are colored based on their average probabilities for persistence in 2019. As with non-weighted metrics, both local and global summaries are possible.

```r
prob <- apply(mur_arc_pres_abs, 2, mean)
o <- order(prob, decreasing = TRUE)
col <- hcl.colors(27, palette = "Vik")[o]

spatial.plot(murphy_spring, x, y, names, arrow.col = col, arrow.lwd = 1.5,
             col = "white", pt.bg = "white")
```

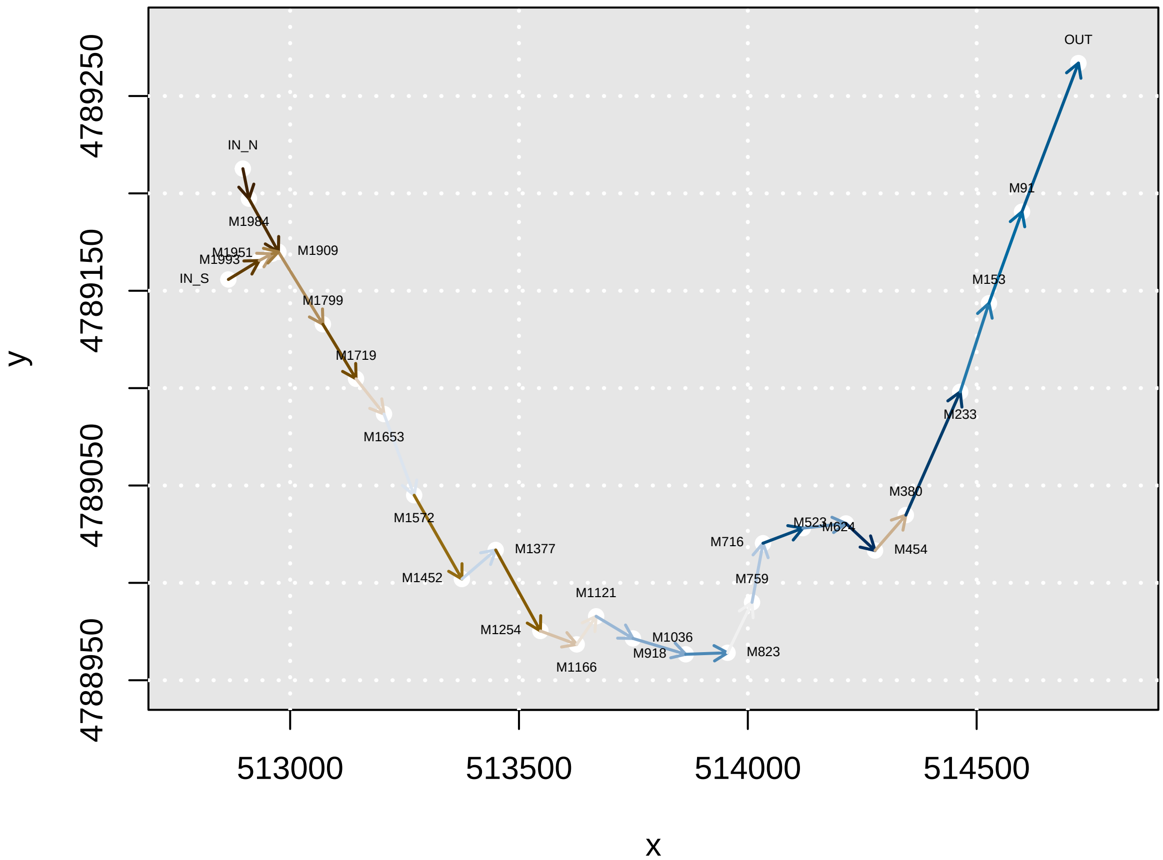

Figure 9: Murphy Cr. arcs colored by their probabilities of surface water presence.

2.5.1 Local measures

Conventional weighted measures of nodal importance include strength (weighted degree) and weighted alpha-centrality. Code for calculating these measures using stream length and stream probability as weights are shown below through use of the functions `igraph::strength` and `igraph::alpha.centrality` with respect to the completely wetted Murphy Cr. network (Fig 2). A summary plot is shown as Fig 10. Increased nuance in strength compared to raw in-degree and nodal weighted alpha centrality compared to raw alpha-centrality are evident when comparing Figs 6 and 10.

```
G3 <- murphy_spring
E(G3)$weight <- mur_lengths[,2]
s1 <- strength(G3)
a1 <- alpha.centrality(G3)

E(G3)$weight <- prob
s2 <- strength(G3)
a2 <- alpha.centrality(G3)

weighted.local <- cbind(s1, a1, s2, a2)
s.weighted.local <- scale(weighted.local) # standardize outcomes

barplot(t(s.weighted.local), beside = T, names = V(G3)$name,
        col = brewer.pal(4,"Spectral"), ylab = "Standardized measures",
        las = 2, cex.names = .8, legend.text = c("Strength_length",
                                                 "Alpha-centrality_length",
                                                 "Strength_prob",
                                                 "Alpha-centrality_prob"),
        args.legend = list(x = "topleft", cex = .7))
```

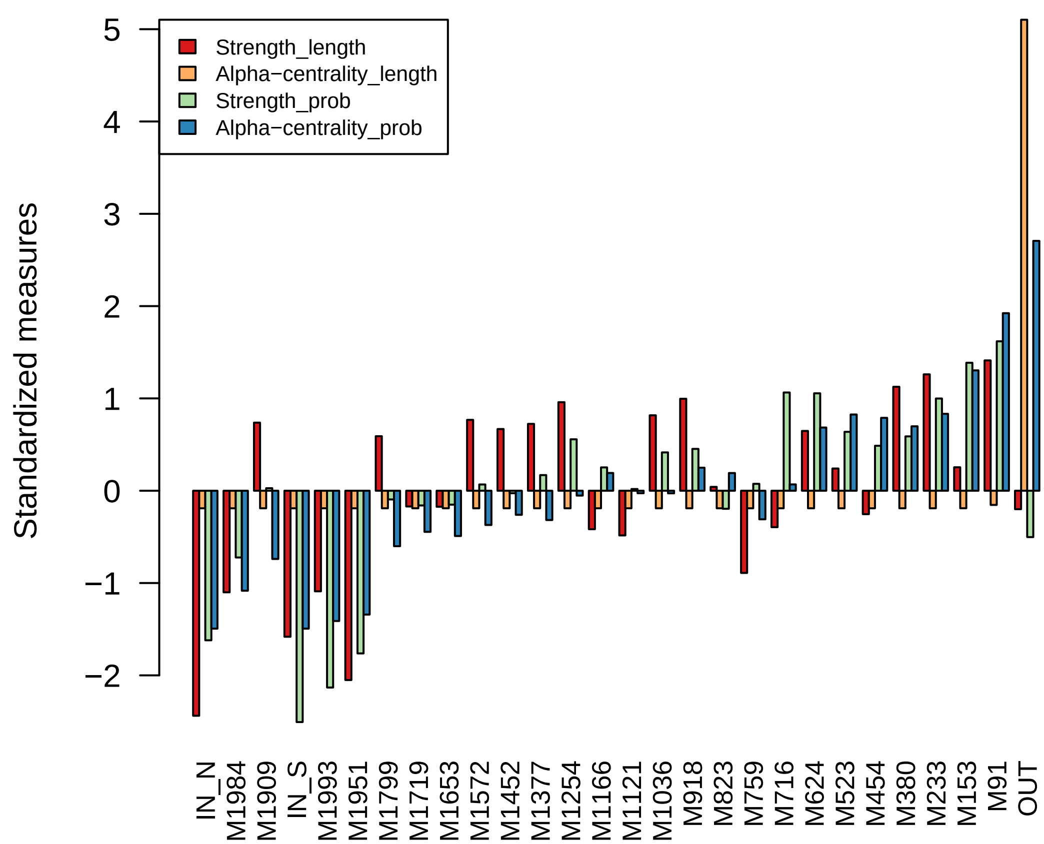

Figure 10: Node strength and alpha-centrality using stream segment length and stream segment probability of acticity (seperately) as weights.

Stream-focused measures that consider both arc probability and arc length include Bernoulli stream length (i.e., stream segment length multiplied by the probability of stream presence) and communication distance (i.e., stream segment length multiplied by the probability of stream presence). Thus, while most local graph measures are defined with respect to graph nodes (despite the fact that some nodal metrics (e.g., strength and alpha centrality) have arc weights), Bernoulli length and communication distance are defined with respect to graph arcs.

Note that in Fig 11, Bernoulli stream length and communication distance are negatively correlated because of their basis on the probability of arc presence and inverse arc presence, respectively. Large communication distance at an arc implies a higher probability of a stream bottleneck at that location.

```r
bsl <- bern.length(mur_lengths[,2], prob) # Bernoulli length
bcd <- bern.length(mur_lengths[,2], 1/prob) # Comm dist.

both <- cbind(bsl, bcd)
scale.both <- scale(both) # standardize outcomes

par(mar = c(7,4.5,1.5,1.5)) # allow full arc names to be seen

barplot(t(scale.both), beside = T,las = 2, cex.names = .8,
        legend.text = c("Bernoulli length", "Commincation distance"),
        args.legend = list(x = "topright", cex = .9),
        ylab = "Standardized measures")
```

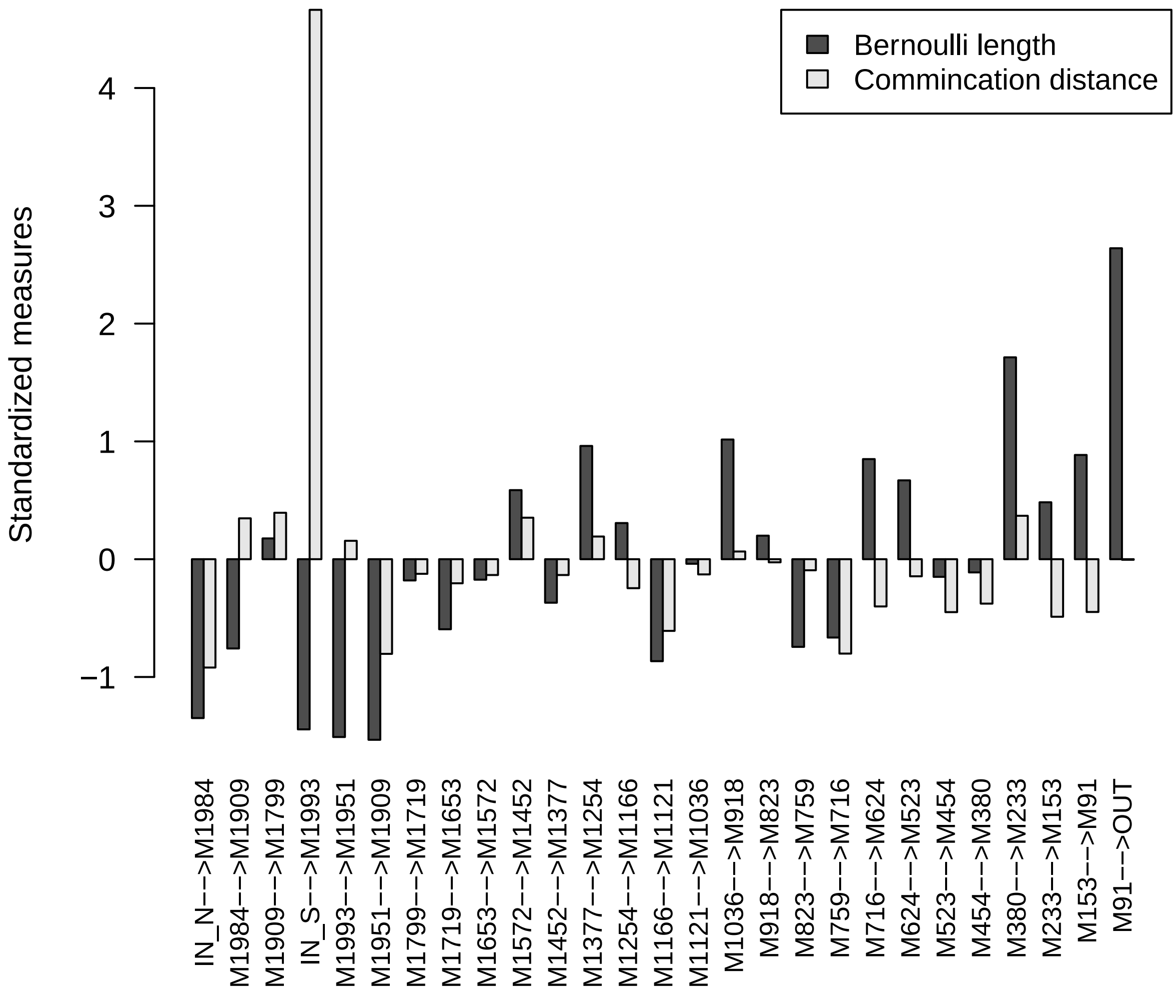

Figure 11: Bernoulli length and communication distance using stream segment length and stream segment probability of acticity (collectively) as weights.

2.5.2 Global measures

Many existing network-level stream connectivity metrics can be viewed as weighted stream DAG measures. These include Integral Connectivity Scale Length (ICSL; Western *et al.* (2001)) and Bernoulli stream length (Botter & Durighetto, 2020) and communication distance. Here we calculate network level average Bernoulli stream length and communication distance for Murphy Creek (units are in meters):

```
bern.length(mur_lengths[,2], prob, mode = "global") # Bernoulli length

[1] 1493.665

bern.length(mur_lengths[,2], 1/prob, mode = "global") # Comm dist.

[1] 3999.231
```

Here is stream-length based ICSL (average in-stream distance of nodes), and average Euclidean distance of nodes, for the completely wetted network, represented in `murphy_spring` (Fig 2).

```
# in-stream average nodal distance
ICSL(murphy_spring, lengths = mur_lengths[,2])

[1] 784.4886

# average nodal Euclidean distance
ICSL(murphy_spring, coords = mur_coords[,2:3], names = mur_coords[,1])

[1] 708.0446
```

As with unweighted metrics, it may be informative to track weighted global (and local) metrics over time. Below we consider: ICSL, intact stream length to the node, and average alpha-centrality (with stream lengths as arc weights) for Murphy Creek subgraphs resulting from the subset stream node presence / absence time series data used earlier (Fig 12).

```r
# walk global.summary through node presence / absence data
icsl <- 1:nrow(subset) -> intact.to.sink -> a.cent

for(i in 1:nrow(subset)){
  temp.graph <- delete.nodes.pa(murphy_spring, subset.nodate[i,])
  # replace direction symbol for igraph comparability
  namelv <- gsub(" -> ", "|", mur_lengths[,1])
  a <- attributes(E(temp.graph))$vname
  w <- which(namelv %in% a)
  length.sub <- mur_lengths[,2][w]
  icsl[i] <- ICSL(temp.graph, lengths = length.sub)
  E(temp.graph)$weights <- length.sub
  intact.to.sink[i] <- size.intact.to.sink(temp.graph, "OUT")
  a.cent[i] <- mean(alpha.centrality(temp.graph), na.rm = T)
}

global <- cbind(icsl, intact.to.sink, a.cent)

# standardize measures
scaled.global <- scale(global)
par(mar = c(7,4.2,1.5,2))

# plot
matplot(scaled.global, xaxt = "n", type = "l", col = hcl.colors(13, palette = "spectral"),
        ylab = "Standardized global measures")
legend("topright", lty = 1:5, col = hcl.colors(13, palette = "spectral"),
       legend = c("ICSL", "intact stream length to sink", "alpha-centrality"), cex = .8)
axis(side = 1, at = c(1,21,41,61,81,100), labels = subset[,1][c(1,21,41,61,81,100)],
     las = 2, cex.axis = .7)
mtext(side = 1, "Time", line = 6)
```

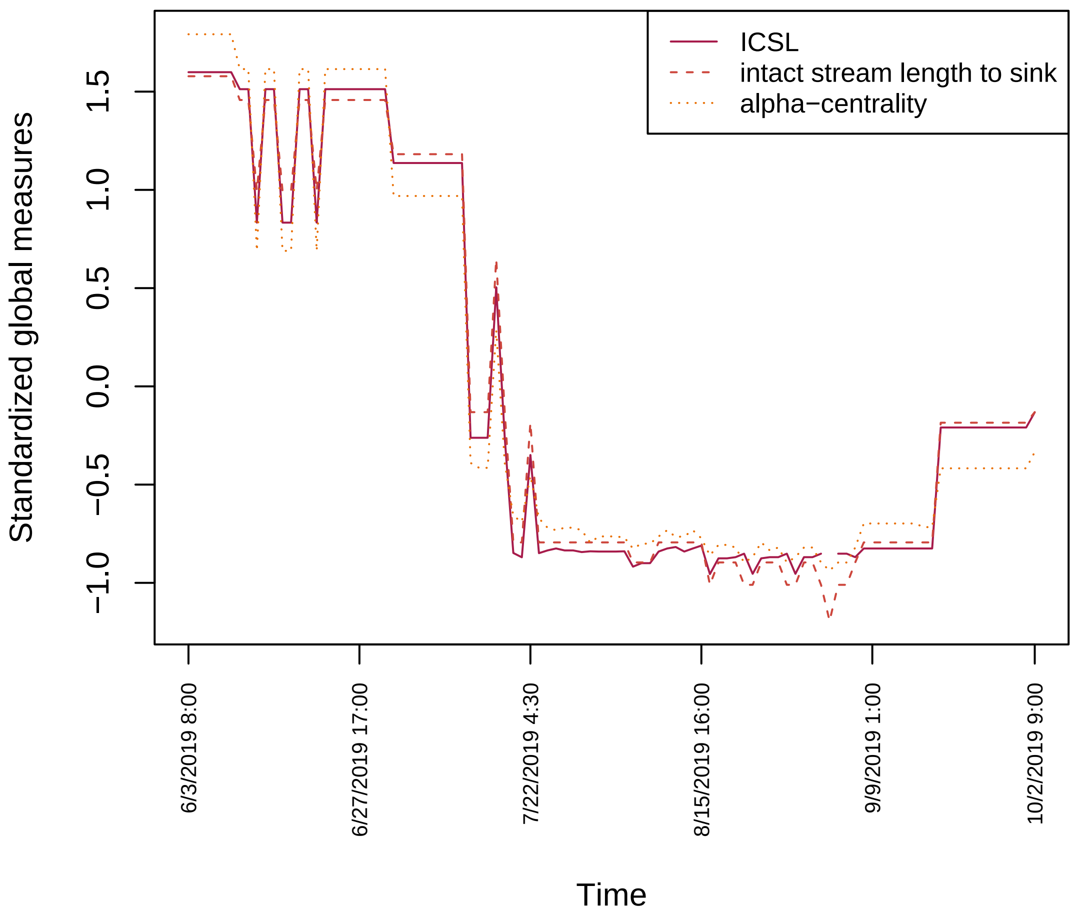

Figure 12: Global weighted network connectivity measures for Murphy Cr. over time.

The dataframe `mur_seasons_arc_pa` contains simulated arc presence/absence data for the spring, summer, and fall.

```
data(mur_seasons_arc_pa)
```

Fig 13 shows histograms of distributions of Bernoulli stream lengths in the spring, summer, and fall. Note that the fall rewet is not captured because of the coarse cutoffs used for seasons.

```
springL <- matrix(nrow = 100, ncol = 27) -> summerL -> fallL

for(i in 1:100){
springL[i,] <-
  bern.length(mur_lengths[,2], mur_seasons_arc_pa[,1:27][mur_seasons_arc_pa$Season == "Spring",][i,], "global")
summerL[i,] <-
  bern.length(mur_lengths[,2], mur_seasons_arc_pa[,1:27][mur_seasons_arc_pa$Season == "Summer",][i,], "global")
fallL[i,] <-
  bern.length(mur_lengths[,2], mur_seasons_arc_pa[,1:27][mur_seasons_arc_pa$Season == "Fall",][i,], "global")
}

xlim <- range(c(springL, summerL, fallL), na.rm = T)
h <- hist(springL, plot = F)
ylim <- range(h$counts)
col <- rgb(c(0,0.5,1), c(0,1,0.5), c(1,0.5,0), c(0.4,0.4,0.4))

hist(springL, xlim = xlim, ylim = ylim, main = "", xlab = "Bernoull network length (m)", col = col[1],
     border = col[1])
par(new = TRUE)
hist(summerL, xlim = xlim, ylim = ylim, axes = F, main = "", xlab = "", col = col[2], border = col[2])
par(new = TRUE)
hist(fallL, xlim = xlim, ylim = ylim, axes = F, main = "", xlab = "", col = col[3], border = col[3])

legend("topleft", fill = col, legend = c("Spring", "Summer", "Fall"), bty = "n", cex = 1)
```

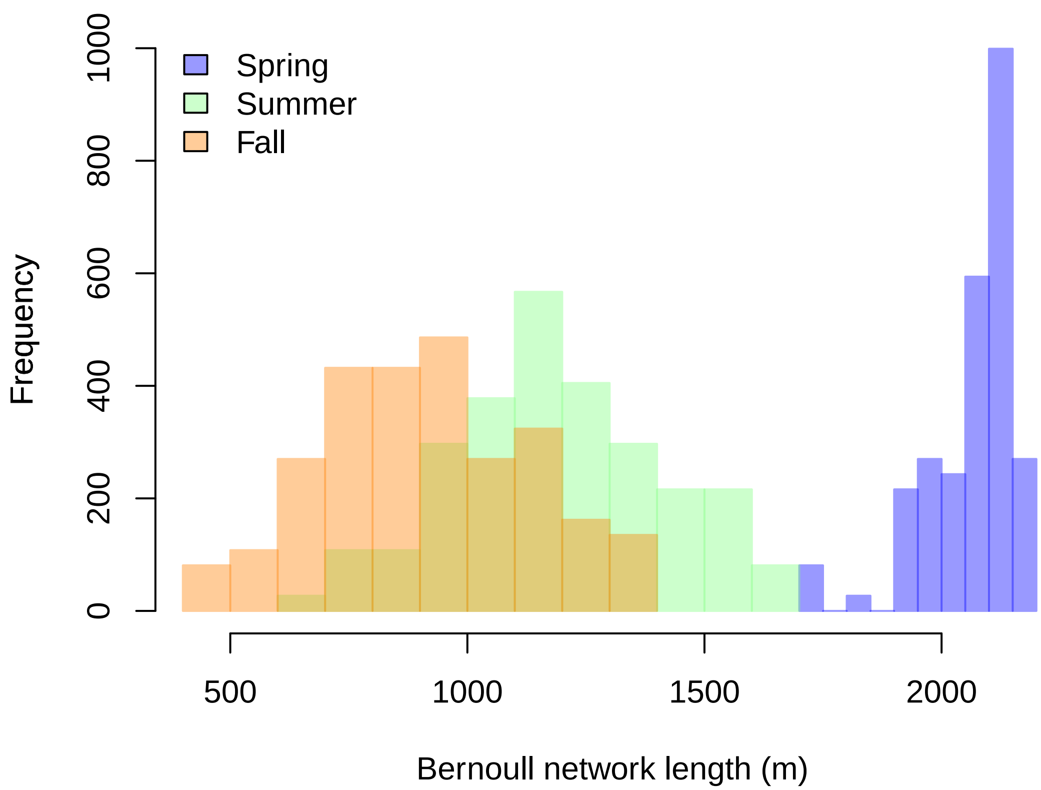

Figure 13: Distributions of Bernoulli network lengths for the seasonal designations.

Here are average network-level Bernoulli stream lengths and communication distances in the spring, summer and fall. Note the presence of infinitely large network-level communication distances in the fall and summer due to the presence of network blockages.

```
mean(springL) # mean spring network length

[1] 2063.295

mean(summerL) # mean summer network length

[1] 1190.534

mean(fallL) # mean fall network length

[1] 909.1867

# mean spring network communication distance
bern.length(mur_lengths[,2],
            1/colMeans(mur_seasons_arc_pa[,1:27][mur_seasons_arc_pa$Season == "Spring",],
                       na.rm = TRUE), "global")

[1] 2748.009

# mean summer network communication distance
bern.length(mur_lengths[,2],
            1/colMeans(mur_seasons_arc_pa[,1:27][mur_seasons_arc_pa$Season == "Summer",],
                       na.rm = TRUE), "global")

[1] Inf

# mean fall network communication distance
bern.length(mur_lengths[,2],
            1/colMeans(mur_seasons_arc_pa[,1:27][mur_seasons_arc_pa$Season == "Fall",],
                       na.rm = TRUE), "global")

[1] Inf
```

2.6 Bayesian Extensions

Bayesian extensions are possible for Bernoulli length and communication distance by viewing the probabilities of stream presence at arcs as random variables. The underlying theory for these approaches is described in Aho et al. (in prep). Briefly, given a beta-distribution prior (and a binomial likelihood), the posterior beta distribution for the probability of stream presence for the kth arc can have the form:

$$\theta_k \mid \boldsymbol{x}_k \sim BETA\left(w \cdot n \cdot \hat{p}_k + \sum \boldsymbol{x}_k, w \cdot n\,(1-\hat{p}_k) + n - \sum \boldsymbol{x}_k\right) \tag{1}$$

where w is the weight given to the prior relative to the current data, and p_k is the mean of the prior beta distribution. The posterior distribution for the inverse probability of stream presence for the kth arc will follow an inverse beta distribution (see Aho et al. (in prep)) with the same parameters shown in Eq 1. Multiplying the kth posterior for the probability of stream presence and the kth posterior for the inverse probability of stream presence by the kth stream length will provide posteriors for Bernoulli stream length and communication distance, respectively.

This process is facilitated by the *streamDAG* function `beta.posterior`. Assume that we wish to apply a naive Bayesian prior, $\theta_k \sim BETA(1,1)$, to the probability of stream segment activity at Murphy Cr., for all segments. The distribution $BETA(1,1)$ is equivalent to a continuous uniform distribution in 0,1, and will have the mean, $E(\theta_k) = 0.5$. Assume further that wish to give the priors 1/3 of the weight of observed binomial outcomes. As data we will use the first 10 rows from `mur_arc_pres_abs`. We have:

```r
data <- mur_arc_pres_abs[1:10,]
b <- beta.posterior(p.prior = 0.5, dat = data, length = mur_lengths[,2], w = 1/3)
```

The `beta.posterior` function returns a list with the following components:

- `alpha`: The α shape parameters for the beta and inverse beta posteriors.
- `beta`: The β shape parameters for the beta and inverse beta posteriors.
- `mean`: The means of the beta posteriors.
- `var`: The variances of the beta posteriors.
- `mean.inv`: The means of the inverse-beta posteriors.
- `var.inv`: The variances of the inverse-beta posteriors.
- `Com.dist`: If `length` is supplied, the mean communication distances of the network.
- `Length`: If `length` is supplied, the mean stream length of the network.
- `x`: The observed number of Bernoulli successes over n trials from `dat`. .

For instance, here are the resulting shape parameters for the beta posterior distributions for the probability of stream presence and the inverse beta posterior distributions for the probability of stream presence.

```r
b$alpha

 IN_N-->M1984 M1984-->M1909 M1909-->M1799  IN_S-->M1993 M1993-->M1951
     6.666667      6.666667      7.666667      2.666667      2.666667
M1951-->M1909 M1799-->M1719 M1719-->M1653 M1653-->M1572 M1572-->M1452
     5.666667      4.666667      4.666667      7.666667      7.666667
M1452-->M1377 M1377-->M1254 M1254-->M1166 M1166-->M1121 M1121-->M1036
     6.666667      9.666667      7.666667      6.666667      5.666667
 M1036-->M918   M918-->M823   M823-->M759   M759-->M716   M716-->M624
    10.666667      6.666667      3.666667      9.666667     11.666667
  M624-->M523   M523-->M454   M454-->M380   M380-->M233   M233-->M153
     9.666667      7.666667      9.666667      8.666667      8.666667
   M153-->M91     M91-->OUT
    11.666667     11.666667

b$beta

 IN_N-->M1984 M1984-->M1909 M1909-->M1799  IN_S-->M1993 M1993-->M1951
     6.666667      6.666667      5.666667     10.666667     10.666667
M1951-->M1909 M1799-->M1719 M1719-->M1653 M1653-->M1572 M1572-->M1452
     7.666667      8.666667      8.666667      5.666667      5.666667
M1452-->M1377 M1377-->M1254 M1254-->M1166 M1166-->M1121 M1121-->M1036
     6.666667      3.666667      5.666667      6.666667      7.666667
 M1036-->M918   M918-->M823   M823-->M759   M759-->M716   M716-->M624
     2.666667      6.666667      9.666667      3.666667      1.666667
  M624-->M523   M523-->M454   M454-->M380   M380-->M233   M233-->M153
     3.666667      5.666667      3.666667      4.666667      4.666667
   M153-->M91     M91-->OUT
     1.666667      1.666667
```

3 Estimating Arc Presence Probabilities

Intermittent stream arc presence / absence data are generally not available because presence / absence data are obtained at particular points in the stream, e.g., nodes. Given a relatively even spatial distribution of nodes, one possibility is to estimate the probability of arc presence as the mean of the presence / absence values of the bounding nodes. Thus, for the kth arc with bounding nodes u and v, for the ith time frame, $i = 1, 2, 3, \ldots, n$, there are three possibilities:

$$x_{k,i} = \begin{cases} 1, & \text{both } u \text{ and } v \text{ are active (wet)} \\ 0, & \text{both } u \text{ and } v \text{ are inactive (wet)} \\ 0.5, & \text{one of } u \text{ or } v \text{ is active} \end{cases}$$

This conversion is facilitated by the *streamDAG* function `arc.pa.from.nodes` which provides arc activity probabilities (using the rule above) based on bounding nodes presence/absence values. For instance, below are the 404th and 405th nodal stream presence observations from Murphy Cr.

```
mur_node_pres_abs[404:405,]

          Datetime IN_N M1984 M1909 IN_S M1993 M1951 M1799 M1719 M1653 M1572
4031  7/15/2019 7:30    1     1     0    0     0     1     1     1    NA     1
4041 7/15/2019 10:00    1     1     0    0     0     0     1     1    NA     1
     M1452 M1377 M1254 M1166 M1121 M1036 M918 M823 M759 M716 M624 M523 M454
4031     1     1     1    NA     1     1    1    1    1    1    1    1    1
4041     0     1     1    NA     1     1    1    1    1    1    1    1    1
     M380 M233 M153 M91 OUT
4031    1    1    1   1   1
4041    1    1    1   1   1
```

Here we estimate arc probabilities from the nodal data.

```
arc.pa.from.nodes(murphy_spring, mur_node_pres_abs[404:405,][,-1])

     IN_N -> M1984 M1984 -> M1909 M1909 -> M1799 IN_S -> M1993 M1993 -> M1951
[1,]             1            0.5            0.5             0            0.5
[2,]             1            0.5            0.5             0            0.0
     M1951 -> M1909 M1799 -> M1719 M1719 -> M1653 M1653 -> M1572 M1572 -> M1452
[1,]            0.5              1              1              1            1.0
[2,]            0.0              1              1              1            0.5
     M1452 -> M1377 M1377 -> M1254 M1254 -> M1166 M1166 -> M1121 M1121 -> M1036
[1,]            1.0              1              1              1              1
[2,]            0.5              1              1              1              1
     M1036 -> M918 M918 -> M823 M823 -> M759 M759 -> M716 M716 -> M624
[1,]             1            1           1            1           1
[2,]             1            1           1            1           1
     M624 -> M523 M523 -> M454 M454 -> M380 M380 -> M233 M233 -> M153
[1,]            1            1            1            1            1
[2,]            1            1            1            1            1
     M153 -> M91 M91 -> OUT
[1,]           1          1
[2,]           1          1
```

Here we estimate the marginal arc probabilities and arc correlation structures using the entire `mur_node_pres_abs` dataset.

```
conversion <- arc.pa.from.nodes(murphy_spring, mur_node_pres_abs[,-1])
marginal <- colMeans(conversion, na.rm = TRUE)
corr <- cor(conversion, use = "pairwise.complete.obs")
```

Impossible correlations (given marginal probabilities) are adjusted with the *streamDAG* function `R.bounds` (see Aho et al. in prep).

```r
corrected.corr <- R.bounds(marginal, corr)
```

Multivariate Bernoulli outcomes can now be simulated using functions from the package *mipfp* (Barthélemy & Suesse, 2018).

```r
library(mipfp)
p.joint.all <- ObtainMultBinaryDist(corr = corrected.corr, marg.probs = marginal,
                                    , tol = 0.001, tol.margins = .001, iter = 100)
out <- RMultBinary(n = 5, p.joint.all, target.values = NULL)$binary.sequences
```

Note that even for relatively small stream networks (e.g., Murphy Cr. with 28 nodes and 27 arcs), the generation of multivariate Bernoulli distributions using `mipfp::ObtainMultBinaryDist` and simulation of multivariate Bernoulli random outcomes using `mipfp::RMultBinary` is computationally cumbersome. Thus, to simplify computational procedures we recommend simulating outcomes only for arcs that demonstrate stream presence spatial dependence, e.g., arcs with outcomes that *are not* always 0 or 1 for an observational period.

References

Barthélemy, J. & Suesse, T. (2018) mipfp: An R package for multidimensional array fitting and simulating multivariate Bernoulli distributions. *Journal of Statistical Software, Code Snippets*, **86**, 1–20.

Botter, G. & Durighetto, N. (2020) The stream length duration curve: A tool for characterizing the time variability of the flowing stream length. *Water resources research*, **56**, e2020WR027282.

Csardi, G. & Nepusz, T. (2006) The igraph software package for complex network research. *InterJournal*, **Complex Systems**, 1695.

Luque, B., Lacasa, L., Ballesteros, F. & Luque, J. (2009) Horizontal visibility graphs: Exact results for random time series. *Physical Review E*, **80**, 046103.

Warix, S.R., Godsey, S.E., Lohse, K.A. & Hale, R.L. (2021) Influence of groundwater and topography on stream drying in semi-arid headwater streams. *Hydrological Processes*, **35**, e14185.

Western, A.W., Blöschl, G. & Grayson, R.B. (2001) Toward capturing hydrologically significant connectivity in spatial patterns. *Water Resources Research*, **37**, 83–97.