Introduction to the *streamDAG* Package

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

January 2, 2023

Contents

1 The *streamDAG* Package

The *streamDAG* package provides indices and tools for analyzing directed acyclic graph (DAG) representations of intermittent stream networks. The DAG framework allows a wide range of analytical approaches for stream graphs including classic measures from hydrology, ecology, and of course, graph theory. A focus of many *streamDAG* algorithms is the measurement of the local importance of individual nodes (stream locations) and arcs (stream segment) to network functionality, and network-level complexity and connectivity. While many of these approaches are purely topological, a non-trivial number of DAG indices, particularly weighted approaches, will provide outcomes identical to existing hydrological (non-graph-theoretic) measures for streams. Perennial streams (and even non-stream networks) can also be analyzed with *streamDAG* algorithms.

The *streamDAG* package is built under the basic idiom of the *igraph* package (Csardi & Nepusz, 2006), and most of its functions utilize *igraph* basis algorithms. The *streamDAG* package is currently only housed in the public 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:

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

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

```r
library(streamDAG)
```

2 Introductory Examples of Usage

2.1 Murphy Creek

We begin with an in-depth demonstration of the *streamDAG* package using Murphy Creek, a very simple intermittent stream in the Reynolds Creek experimental watershed in southwestern Idaho (Fig 1). From 6/3/2019 to 10/3/2019, stream presence data were acquired at 15 minute intervals from 25 Murphy Creek nodes, corresponding to 24 stream segment arcs. Bounding nodes were added to encompass the entire length of the network. This resulted in a final Murphy Creek network with 28 nodes and 27 arcs for analysis.

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

2.1.1 Data Outlay

Purely topological analyses can be conducted in *streamDAG* 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)
```

This code is contained as an option in the function `streamDAGs` which also codifies other intermittent stream *igraph* objects.

```
streamDAGs("mur_full")

IGRAPH 0673eed DN-- 28 27 --
+ attr: name (v/c)
+ edges from 0673eed (vertex names):
 [1] IN_N ->M1984 M1984->M1909 M1909->M1799 IN_S ->M1993 M1993->M1951
 [6] M1951->M1909 M1799->M1719 M1719->M1653 M1653->M1572 M1572->M1452
[11] M1452->M1377 M1377->M1254 M1254->M1166 M1166->M1121 M1121->M1036
[16] M1036->M918  M918 ->M823  M823 ->M759  M759 ->M716  M716 ->M624
[21] M624 ->M523  M523 ->M454  M454 ->M380  M380 ->M233  M233 ->M153
[26] M153 ->M91   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 spatial 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 ensure that the order of relevant rows and columns and elements 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).

Within ancillary datasets, different code identifiers can be used to designate arcs. For instance, for an arc $z = \vec{uv}$ where u is the tail of arc z and v is the head of z, we could use the code: `u--+v` or `u-->v` or `u --> v` or `u->v`, or even `u|v`. The important thing is that the ordering is consistent with the arcs in the corresponding graph object. For instance, here are the first six arc names for the graph object `murphy_spring`.

```
head(A(murphy_spring))

+ 6/27 edges from 066e095 (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 (in the first six rows) from `mur_lengths`.

```
head(mur_lengths)

            Arcs Lengths
1  IN_N -> M1984   20.30
2 M1984 -> M1909   75.00
3 M1909 -> M1799  108.99
4  IN_S -> M1993   68.30
5 M1993 -> M1951   27.60
6 M1951 -> M1909   14.40
```

Naming of nodes should be consistent with the node names in the corresponding graph object. For instance, here are the first six graph node names from `murphy_spring`.

```
head(V(murphy_spring))

+ 6/28 vertices, named, from 066e095:
[1] IN_N  M1984 M1909 IN_S  M1993 M1951
```

The naming (and order) correspond to the first six identifiers (column names in this case) for presence absence data from `mur_node_pres_abs`.

```
names(mur_node_pres_abs)[1:7][-1] # ignoring datestamp column 1

[1] "IN_N"  "M1984" "M1909" "IN_S"  "M1993" "M1951"
```

2.1.2 Spatial Plots

It is easy to depict 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).

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

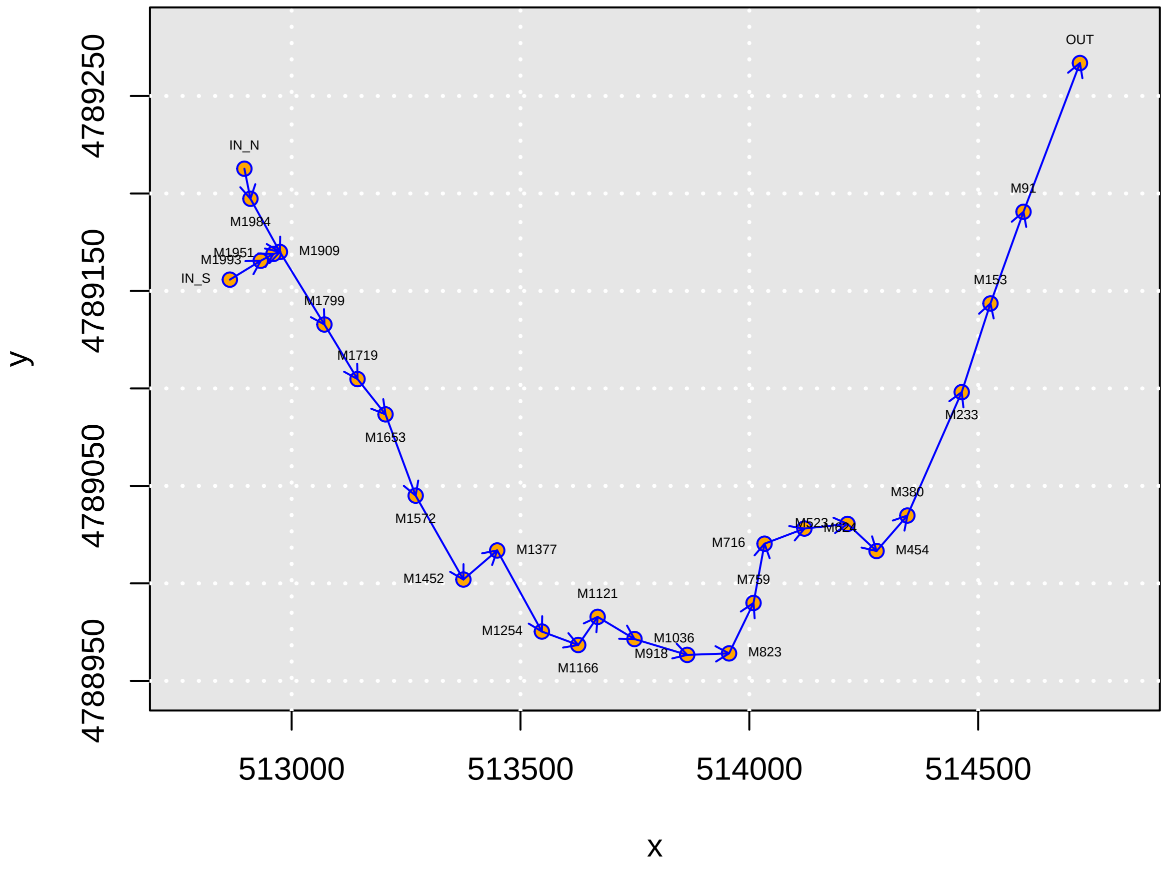

Figure 2: Spatially explicit graph of the completely wetted Murphy Cr. network, as it occurs in the spring.

ARC-GIS shapefiles can also be used to generate spatial plots with the function `spatial.plot.sf` . Use of shapefiles requires use of the libraries libraries *ggplot2* and *sf*. Resulting shapefile graphs 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 Section 2.3).

```
library(ggplot2); library(sf)
# Note that the directory "shape" also contains required ARC-GIS .shx,.cpg, and .prj files.
mur_sf <- st_read(system.file("shape/Murphy_Creek.shp", package="streamDAG"))

## Reading layer `Murphy_Creek' from data source
##   `C:\Users\ahoken\AppData\Local\R\win-library\4.2\streamDAG\shape\Murphy_Creek.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 2 features and 2 fields
## Geometry type: LINESTRING
## Dimension:     XY
## Bounding box:  xmin: 512864.7 ymin: 4788962 xmax: 514722.6 ymax: 4789265
## Projected CRS: NAD83 / UTM zone 11N

g1 <- spatial.plot.sf(x, y, names, shapefile = mur_sf)

## some ggplot customizations
library(ggrepel)
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"))
```

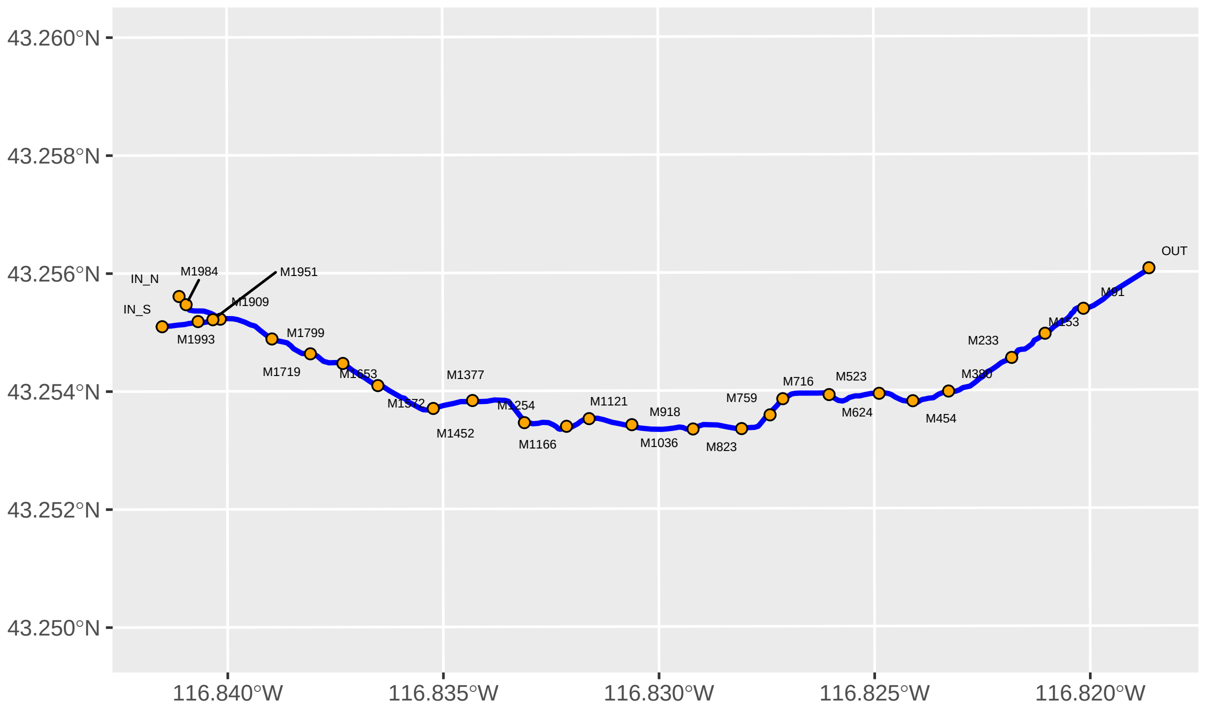

Figure 3: Example of using a shapefile with `spatial.plot`.

2.1.3 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 Creek 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
```

Modifying `murphy_spring` 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. There are several options for simultaneously plotting "dry"

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

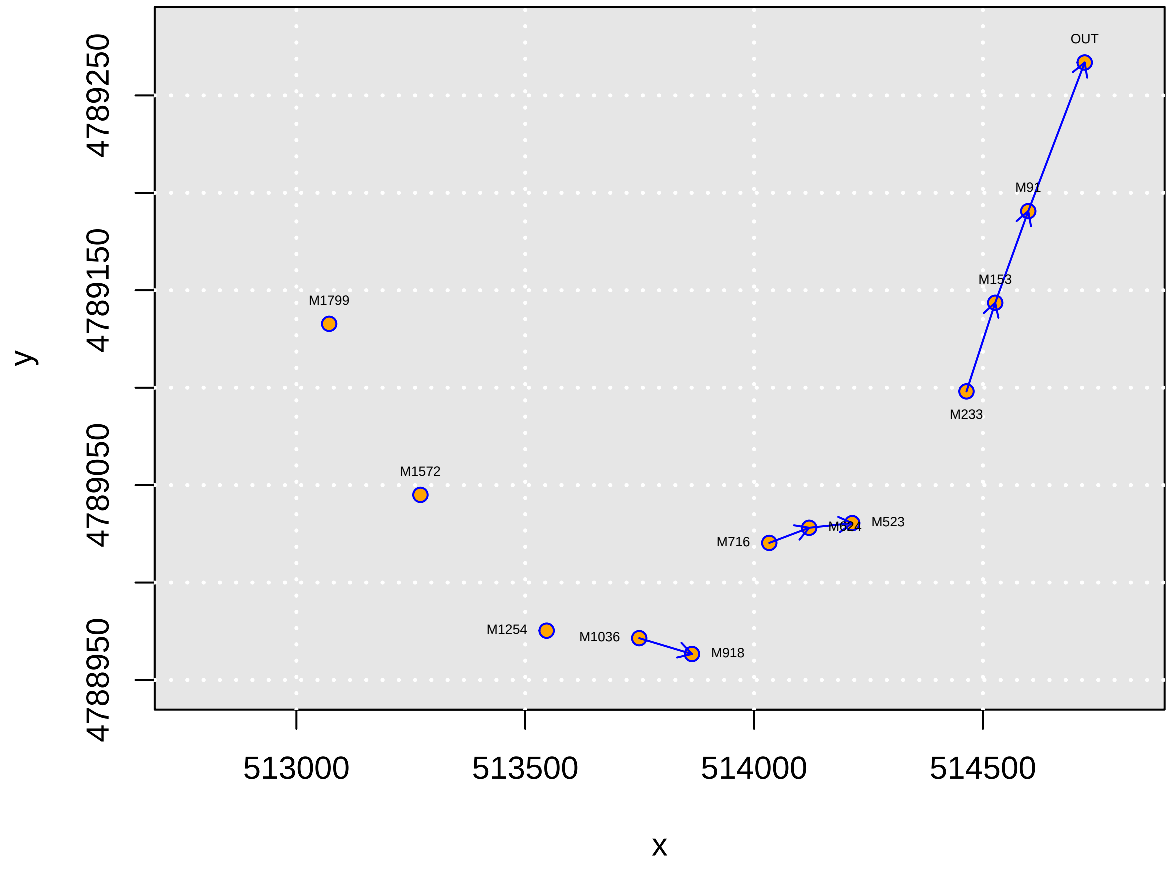

Figure 4: Plotting a modification of `murphy_spring` after application of `delete.nodes.pa`.

There are several graphical approaches for distinguishing “dry” and “wet” stream locations. The simplest is to simply show “wet” nodes and arcs bounded by “wet” nodes as in Fig 4. One can also show “dry” node locations by specifying `show.dry = TRUE` (5).

```
spatial.plot(G1, x, y, names, plot.dry = TRUE, col = "orange", pt.bg = "blue")
```

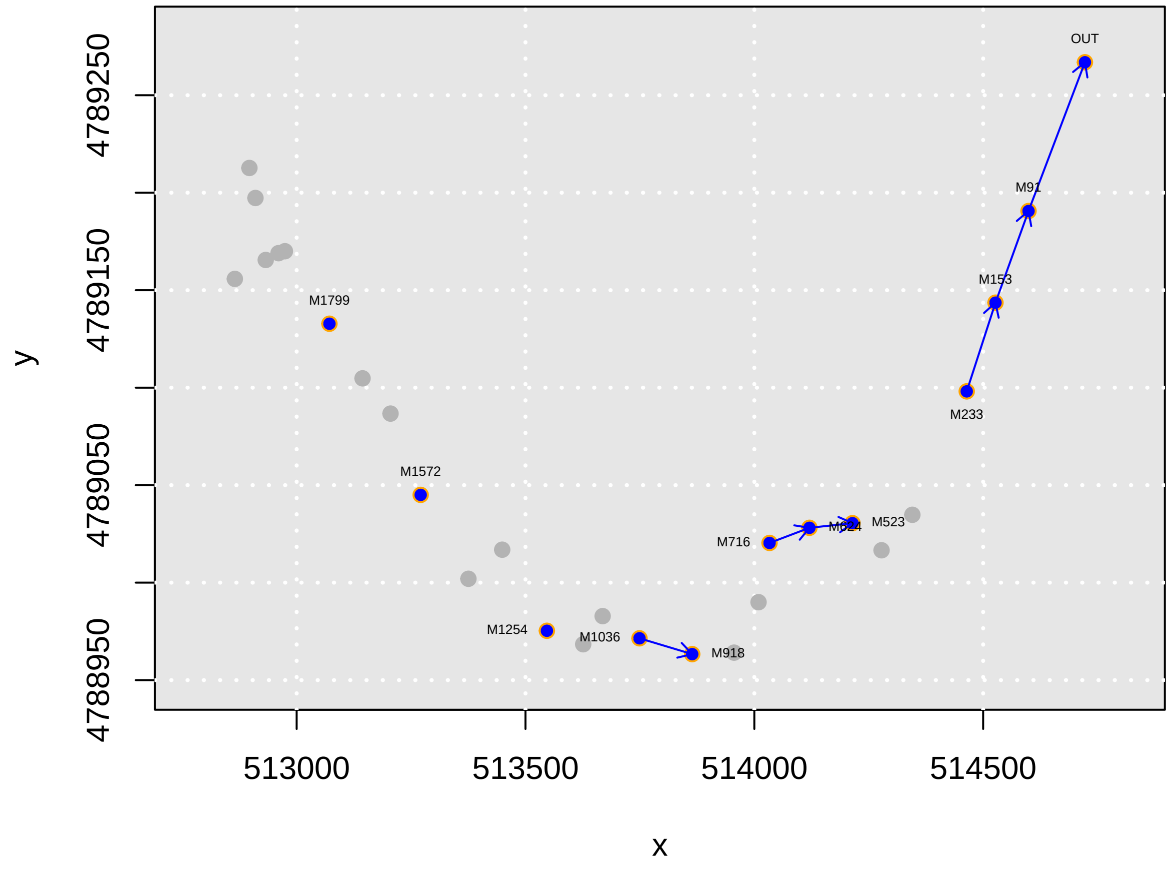

Figure 5: Dry nodes overlaid on Fig 4.

Finally, one can show “wet” nodes and associated arcs superimposed over the entire network, which includes, potentially, . This approach requires generation of `spatial.plot` object representing the entire network, and specification of this object using the argument `cnw` i.e., complete network (Fig 6).

```
spc <- spatial.plot(murphy_spring, x, y, names, plot = FALSE)
spatial.plot(G1, x, y, names, plot.dry = TRUE, col = "orange", pt.bg = "blue", cnw = spc)
```

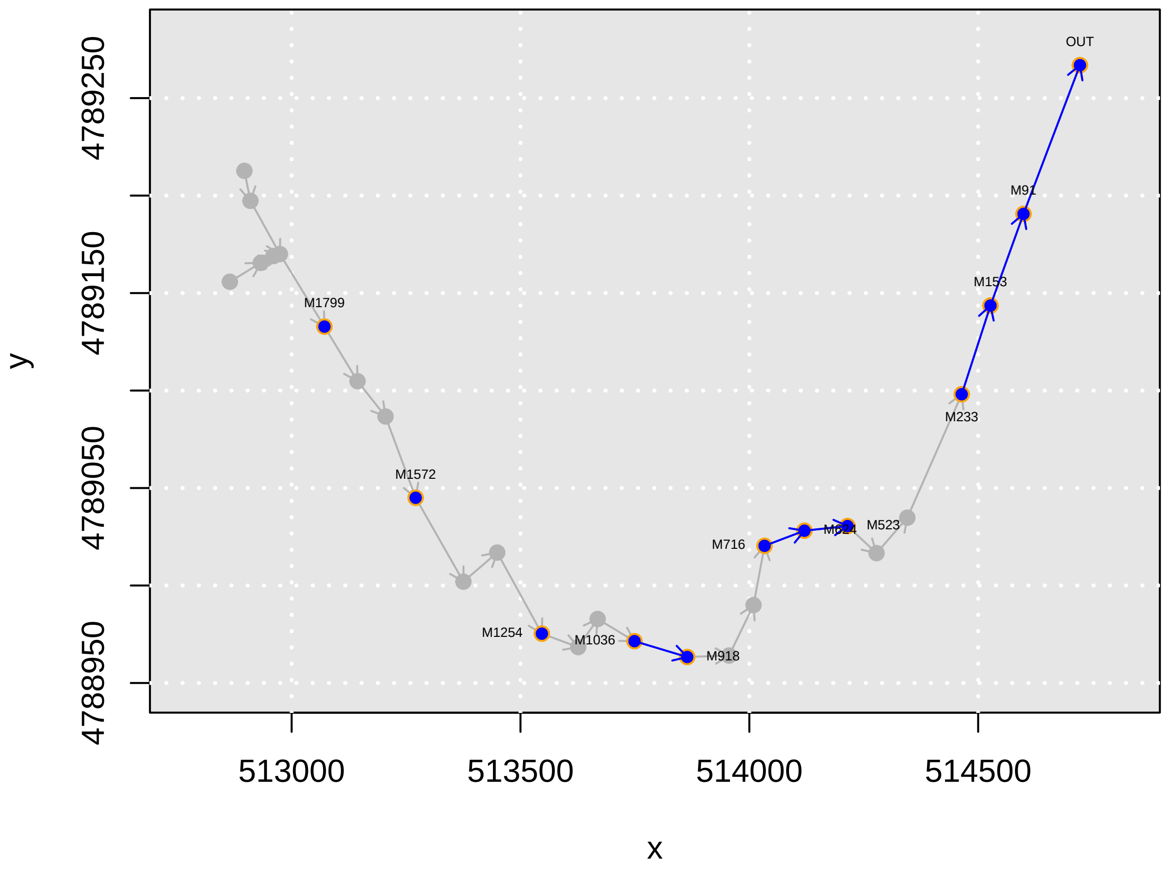

Figure 6: Dry portions of the network underlying wet nodes and associated arcs.

One can also modify graphs based on arc presence / absence data. The dataframe `mur_arc_pres_abs` contains simulated multivariate Bernoulli datasets 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
```

Modifying `murphy_spring` arcs based on the 6th simulated multivariate Bernoulli dataset of arc presence / absence, we have:

```
G2 <- delete.arcs.pa(murphy_spring, mur_arc_pres_abs[6,])
```

The resulting spatial plot is shown in Fig 7. Note that all nodes are plotted (wet and dry nodes are not distinguisshed), but plotted arcs are limited to those with recorded stream activity.

```
spatial.plot(G2, x, y, names)
```

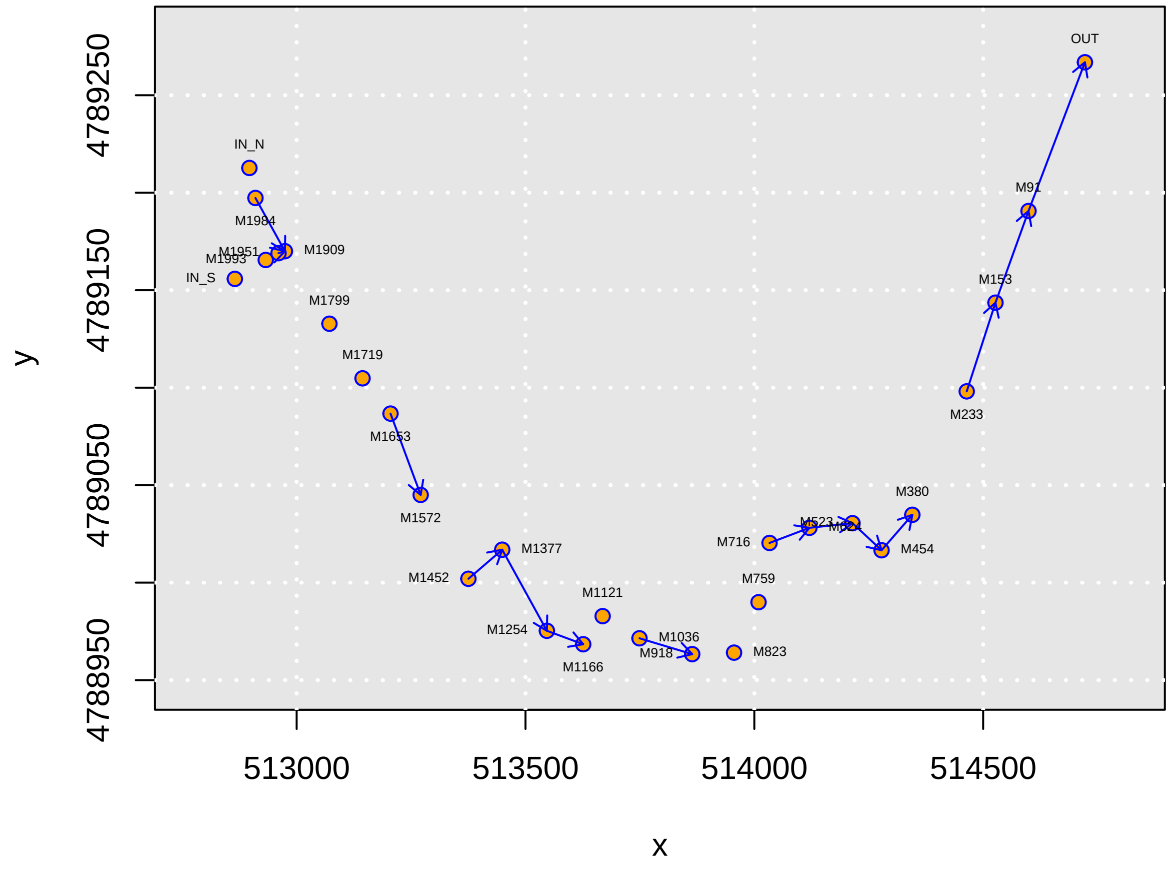

Figure 7: Plotting a modified version of `murphy_spring` after application of `delete.arcs.pa`.

2.1.4 Uweighted (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 separated 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.

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

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

                         IN_N M1984  M1909  IN_S M1993 M1951  M1799  M1719
alpha.cent               1.00  2.00   6.00  1.00  2.00  3.00   7.00   8.00
```

```
page.rank                0.01  0.01   0.03  0.01  0.01  0.02   0.03   0.04
imp.closeness.cent       0.00 27.00  90.00  0.00 27.00 40.50  78.75  77.40
betweenness.cent         0.00 23.00 110.00  0.00 24.00 46.00 126.00 140.00
n.paths                  0.00  1.00   5.00  0.00  1.00  2.00   6.00   7.00
upstream.network.length  0.00  1.00   5.00  0.00  1.00  2.00   6.00   7.00
in.path.length.mean       NaN  1.00   1.80   NaN  1.00  1.50   2.50   3.14
in.path.length.var         NA    NA   0.70    NA    NA  0.50   1.10   1.81
in.path.length.skew      0.00    NA   0.51  0.00    NA   NaN   0.00  -0.35
in.path.length.kurt     -0.50    NA  -0.61 -0.50    NA   NaN  -0.25  -0.30
in.eccentricity          0.00  1.00   3.00  0.00  1.00  2.00   4.00   5.00
mean.efficiency          0.00  0.04   0.12  0.00  0.04  0.06   0.11   0.11
                         M1653  M1572  M1452  M1377  M1254  M1166  M1121  M1036
alpha.cent                9.00  10.00  11.00  12.00  13.00  14.00  15.00  16.00
page.rank                 0.04   0.04   0.04   0.04   0.04   0.04   0.04   0.04
imp.closeness.cent       78.30  79.91  81.74  83.61  85.46  87.24  88.94  90.57
betweenness.cent        152.00 162.00 170.00 176.00 180.00 182.00 182.00 180.00
n.paths                   8.00   9.00  10.00  11.00  12.00  13.00  14.00  15.00
upstream.network.length   8.00   9.00  10.00  11.00  12.00  13.00  14.00  15.00
in.path.length.mean       3.75   4.33   4.90   5.45   6.00   6.54   7.07   7.60
in.path.length.var        2.79   4.00   5.43   7.07   8.91  10.94  13.15  15.54
in.path.length.skew      -0.46  -0.47  -0.44  -0.41  -0.37  -0.34  -0.30  -0.28
in.path.length.kurt      -0.60  -0.84  -1.01  -1.12  -1.19  -1.24  -1.27  -1.29
in.eccentricity           6.00   7.00   8.00   9.00  10.00  11.00  12.00  13.00
mean.efficiency           0.11   0.11   0.11   0.11   0.12   0.12   0.12   0.12
                          M918   M823   M759   M716   M624   M523   M454   M380
alpha.cent               17.00  18.00  19.00  20.00  21.00  22.00  23.00  24.00
page.rank                 0.04   0.04   0.04   0.04   0.04   0.04   0.04   0.04
imp.closeness.cent       92.12  93.60  95.01  96.36  97.64  98.88 100.06 101.20
betweenness.cent        176.00 170.00 162.00 152.00 140.00 126.00 110.00  92.00
n.paths                  16.00  17.00  18.00  19.00  20.00  21.00  22.00  23.00
upstream.network.length  16.00  17.00  18.00  19.00  20.00  21.00  22.00  23.00
in.path.length.mean       8.12   8.65   9.17   9.68  10.20  10.71  11.23  11.74
in.path.length.var       18.12  20.87  23.79  26.89  30.17  33.61  37.23  41.02
in.path.length.skew      -0.25  -0.23  -0.21  -0.20  -0.18  -0.17  -0.16  -0.15
in.path.length.kurt      -1.30  -1.31  -1.31  -1.31  -1.31  -1.31  -1.31  -1.30
in.eccentricity          14.00  15.00  16.00  17.00  18.00  19.00  20.00  21.00
mean.efficiency           0.13   0.13   0.13   0.13   0.13   0.14   0.14   0.14
                          M233   M153    M91    OUT
alpha.cent               25.00  26.00  27.00  28.00
page.rank                 0.04   0.04   0.04   0.04
imp.closeness.cent      102.29 103.34 104.35 105.33
betweenness.cent         72.00  50.00  26.00   0.00
n.paths                  24.00  25.00  26.00  27.00
upstream.network.length  24.00  25.00  26.00  27.00
in.path.length.mean      12.25  12.76  13.27  13.78
in.path.length.var       44.98  49.11  53.40  57.87
in.path.length.skew      -0.14  -0.13  -0.12  -0.11
in.path.length.kurt      -1.30  -1.30  -1.30  -1.29
in.eccentricity          22.00  23.00  24.00  25.00
mean.efficiency           0.14   0.14   0.14   0.14
```

A graphical summary based only on measures with complete cases and standardized outcomes is shown in Fig 8. 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(8,"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 = "topleft", cex = .7,
                           bty = "n", col = cols),
        ylab = "Standardized local measures")
```

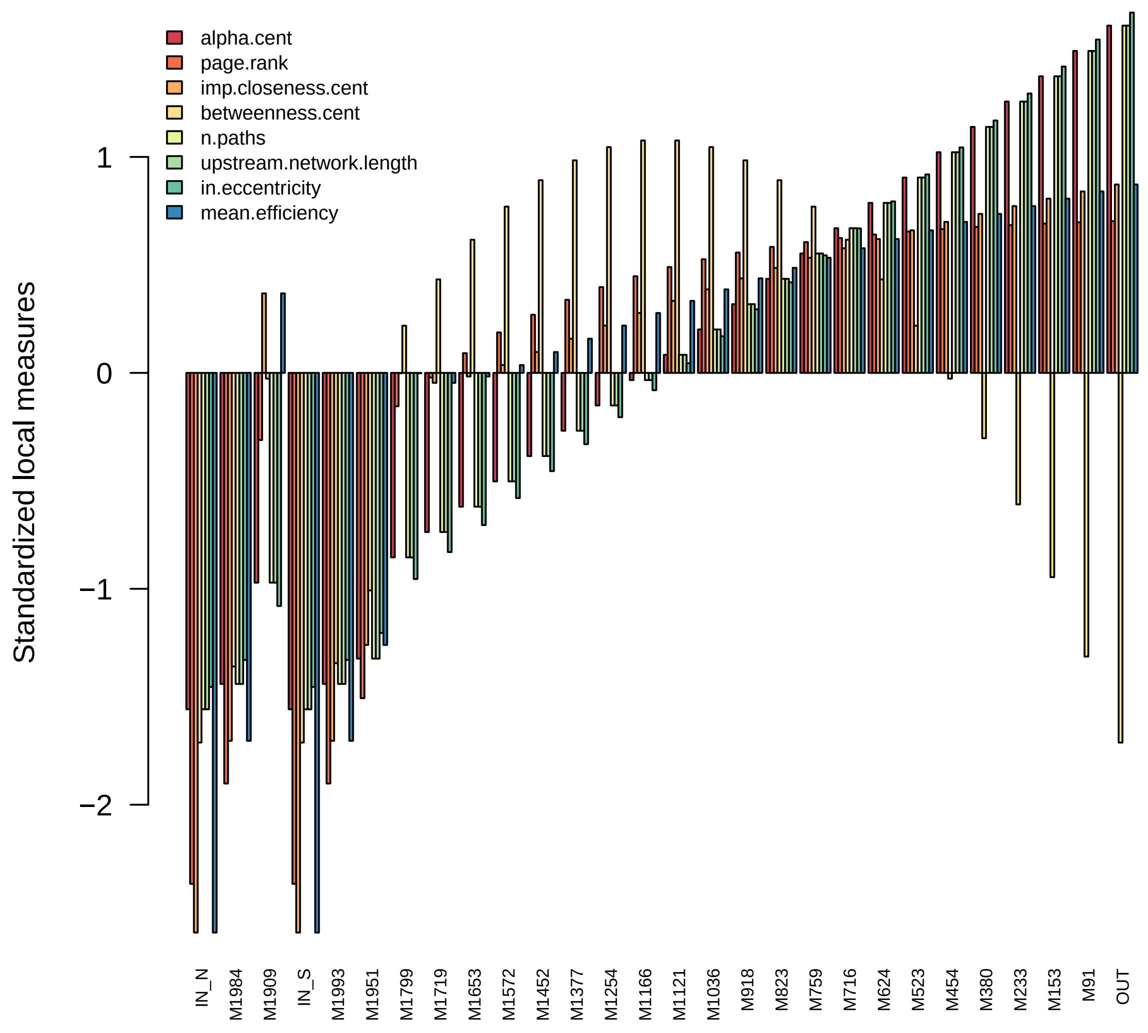

Figure 8: Local graph-theoretic summaries for `murphy_spring`.

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* from each other 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 in a visibility analysis based on indegree (Fig 9). Weighted (see below) node visibilities can also be obtained with `multi.path.visibility`.

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

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

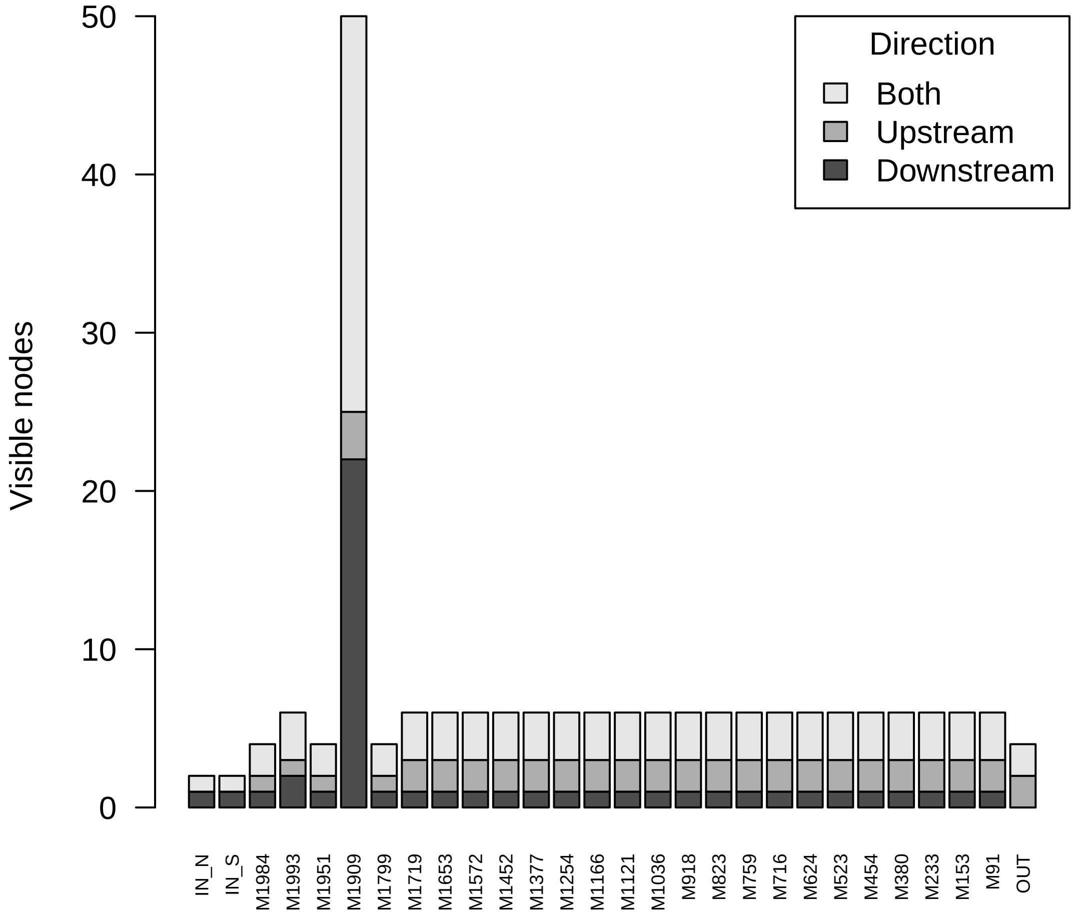

Figure 9: Nodal visibilities for `murphy_spring` based on nodal indegree.

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
Sources               2.00000000
n.paths.to.sink      27.00000000
mean.path.length     13.77777778
mean.a.centrality    14.28571429
Strahler.number       2.00000000
Shreve.number         2.00000000
Randic               13.20710678
first.Zagreb         28.00000000
second.Zagreb        14.50000000
ABC                   0.70710678
Geom.Arith           13.44280904
Harmonic             13.16666667
Harary               40.86258116
Global.efficiency     0.10810207
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 10 shows a 100 point time series that spans the entire 2019 sampling season. As in Fig 8, metrics are standardized to have a mean of zero and a variance of one. Note higher scores for most metrics occur 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.

```
# 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 = 18, 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))

scaled.global <- scale(global)

# plot
matplot(scaled.global, xaxt = "n", type = "l", lty = 1:5, col = hcl.colors(18, palette = "spectral"),
        ylab = "Standardized global measures")
legend("bottomright", lty = 1:5, col = hcl.colors(18, palette = "spectral"),
       legend = row.names(g), cex = .55)
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)
```

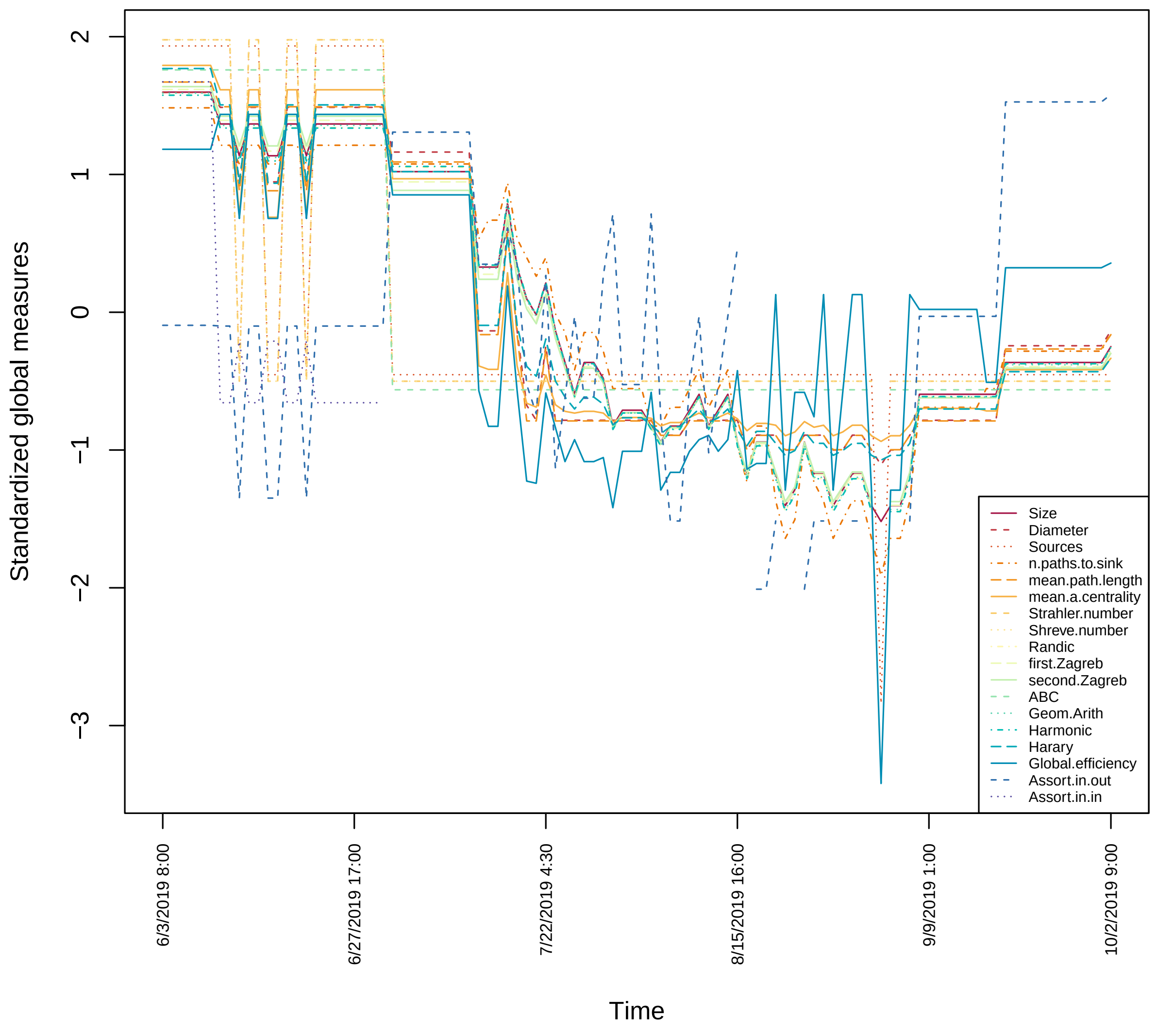

Figure 10: Global summaries for texttmurphy_spring.

2.1.5 Weighted (Reality-Driven) 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 11 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)
o2 <- order(prob)
o3 <- order(o2)

col <- hcl.colors(27, palette = "Vik", rev = T)[o3]

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

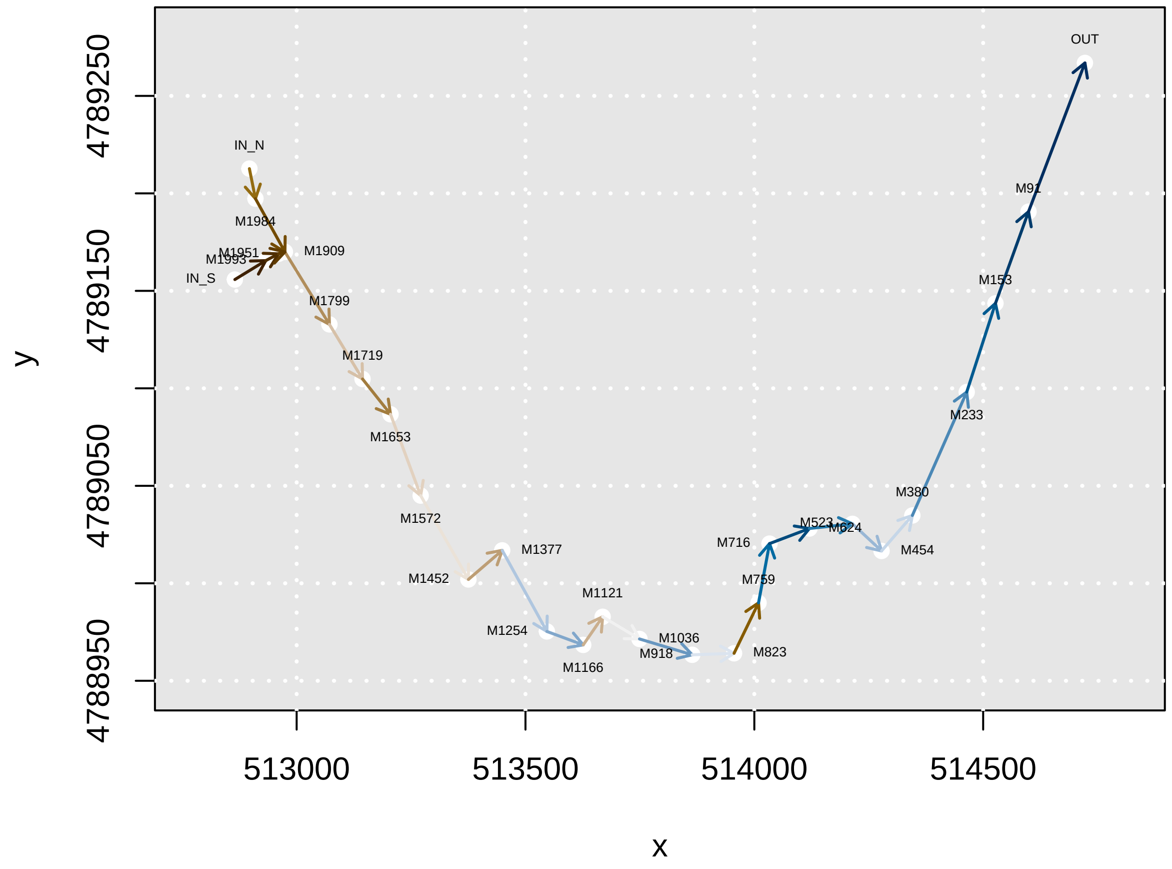

Figure 11: Murphy Cr. arcs colored by their probabilities of surface water presence.

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 12.

```r
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))
```

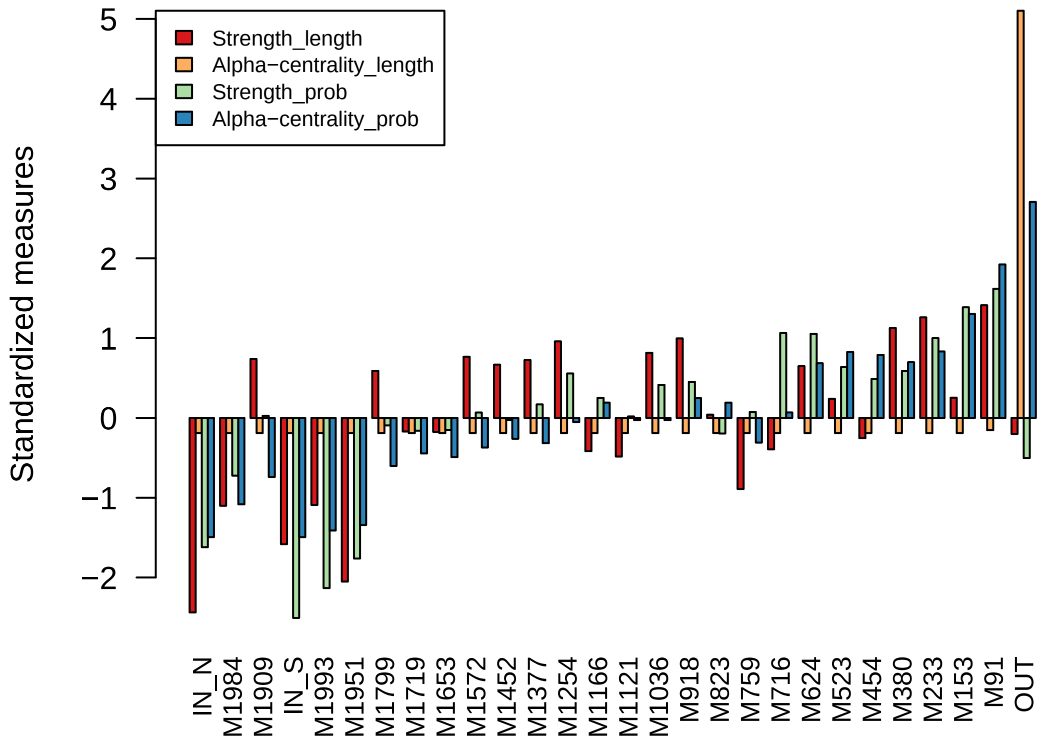

Figure 12: Node strength and alpha-centrality using stream segment length and stream segment probability of activity (separately) as weights.

Weights address bias that may occur in designating nodes in stream networks. For instance path lengths can be made arbitrarily large by adding more nodes to paths. This effect, however, will be largely addressed if arcs are weighted by their actual field measured lengths.

```
G3 <- murphy_spring
E(G3)$weight <- mur_lengths[,2]
library(asbio)
nodes <- attributes(V(G3))$names
list.paths <- vector(mode='list', length = length(nodes)); names(list.paths) <- nodes

for(i in 1:length(nodes)){
list.paths[[i]] <- path.lengths(G3, node = nodes[i])
}

mean <- as.matrix(unlist(lapply(list.paths, mean)))
median <- as.matrix(unlist(lapply(list.paths, median)))
var <- as.matrix(unlist(lapply(list.paths, var)))
skew <- as.matrix(unlist(lapply(list.paths, skew)))
kurt <- as.matrix(unlist(lapply(list.paths, kurt)))

path.summary <- data.frame(Mean = mean, Median = median, Variance = var, Skew = skew, Kurtosis = kurt)

no.na <- na.omit(path.summary); scale.no.na <- scale(no.na)

barplot(t(scale.no.na), beside = TRUE, las = 2, ylim = c(-3.75, 4),
        col = brewer.pal(5,"Spectral"),
        legend.text = c("Mean", "Median", "Variance", "Skew", "Kurtosis"),
        args.legend = list(x = "bottomright", cex = .7, bty = "n", title = "Path length"),
        ylab = "Standardized measures")
```

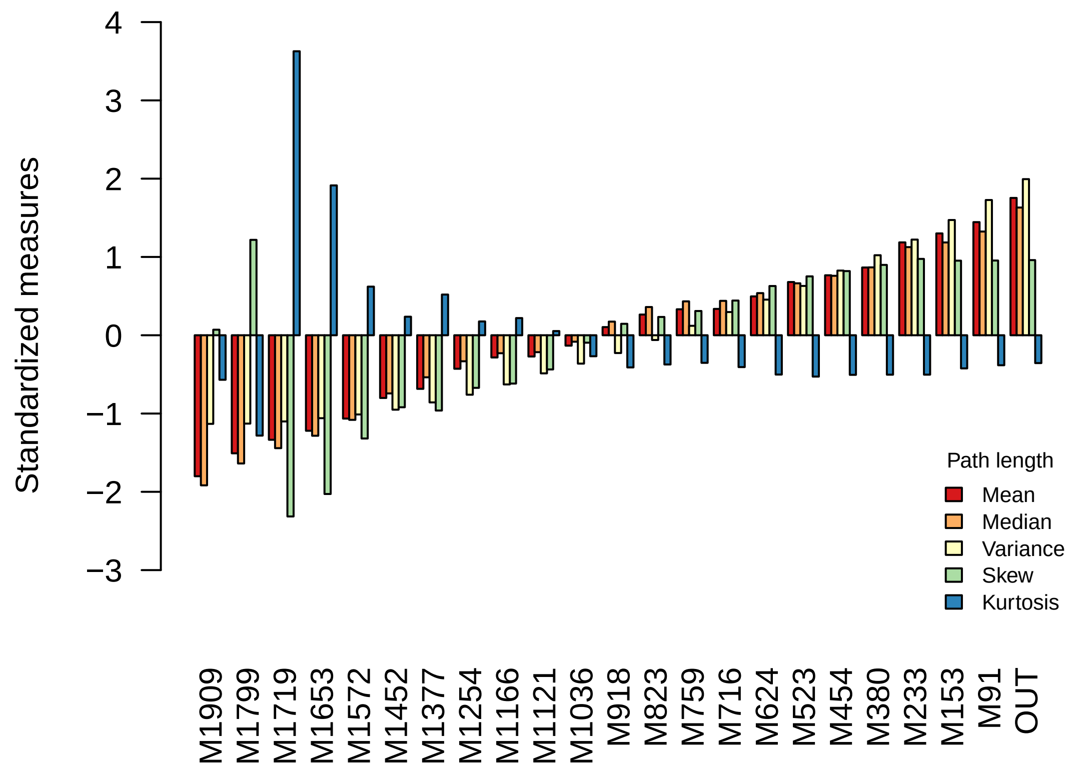

Figure 13: In-path length summaries after weighting arcs by actual in-stream lengths.

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 inverse 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 14, 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.

```
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")
```

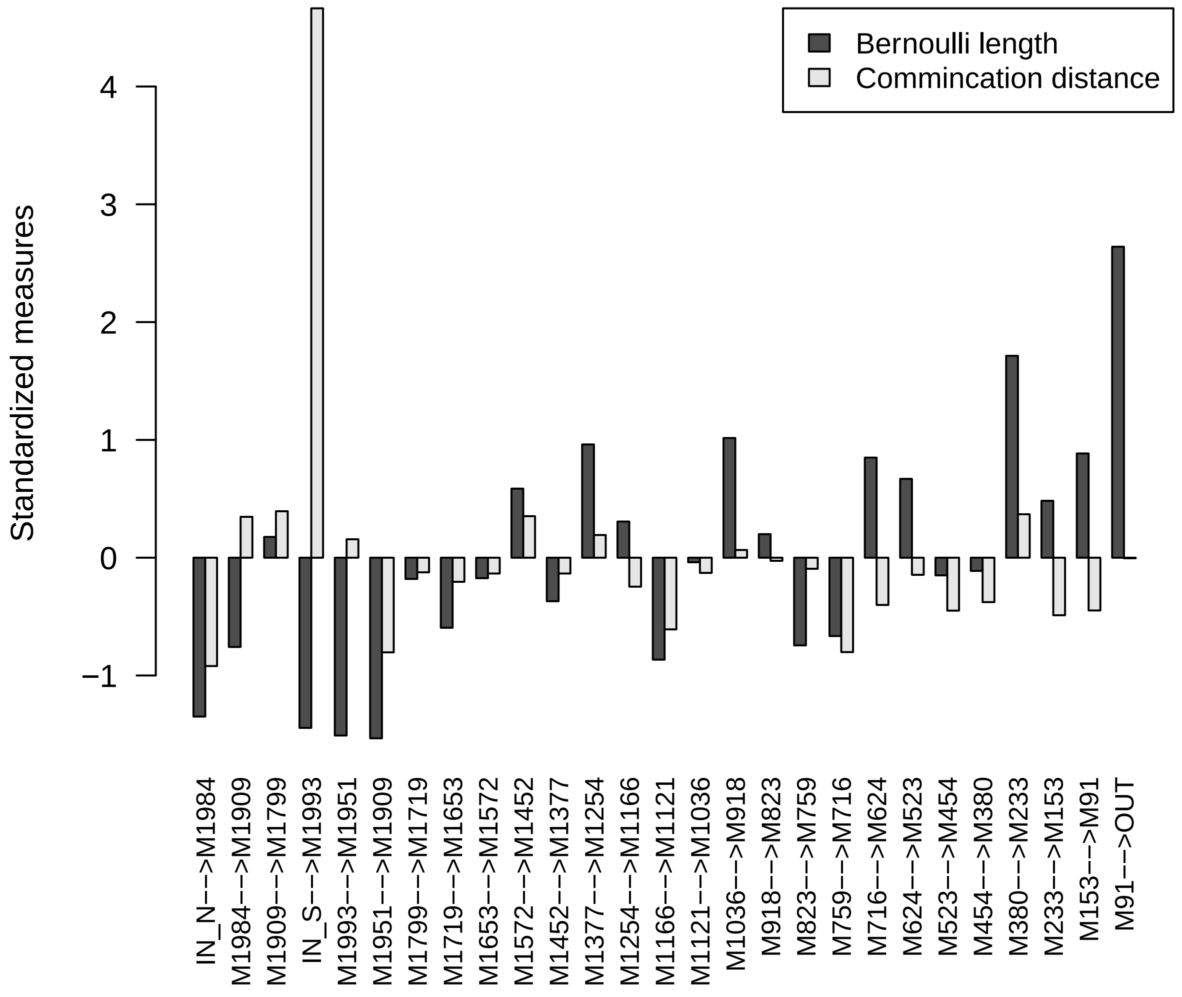

Figure 14: Bernoulli length and communication distance using stream segment length and stream segment probability of activity (collectively) as weights.

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)), network-level average Bernoulli stream length (Botter & Durighetto, 2020) and average network communication distance. Here we calculate network level average Bernoulli stream length and average network communication distance for Murphy Creek. Units, are the units of measured in-stream lengths; in this case, meters.

```r
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 the average Euclidean distance of nodes, for the completely wetted network, represented in `murphy_spring` (Fig 2).

```r
# 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 graphs resulting from the stream node presence / absence time series data used earlier (Fig 15).

```
# walk global.summary through node presence / absence data

icsl <- 1:nrow(subset) -> intact.to.sink -> a.cent -> harary -> global.eff

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))$vname
  w <- which(namelv %in% a)
  length.sub <- mur_lengths[,2][w]
  E(temp.graph)$weights <- length.sub
  icsl[i] <- ICSL(temp.graph)
  global.eff <- global.efficiency(temp.graph)
  intact.to.sink[i] <- size.intact.to.sink(temp.graph, "OUT")
  a.cent[i] <- mean(alpha.centrality(temp.graph), na.rm = T)
  harary[i] <- harary(temp.graph)
}

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

# 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(5, palette = "spectral"),
        ylab = "Standardized global measures", lty = 1:2)
legend("topright", lty = 1:2, col = hcl.colors(5, palette = "spectral"),
       legend = c("ICSL", "global efficiency", "intact stream length to sink", "alpha-centrality", "Harary"), 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)
```

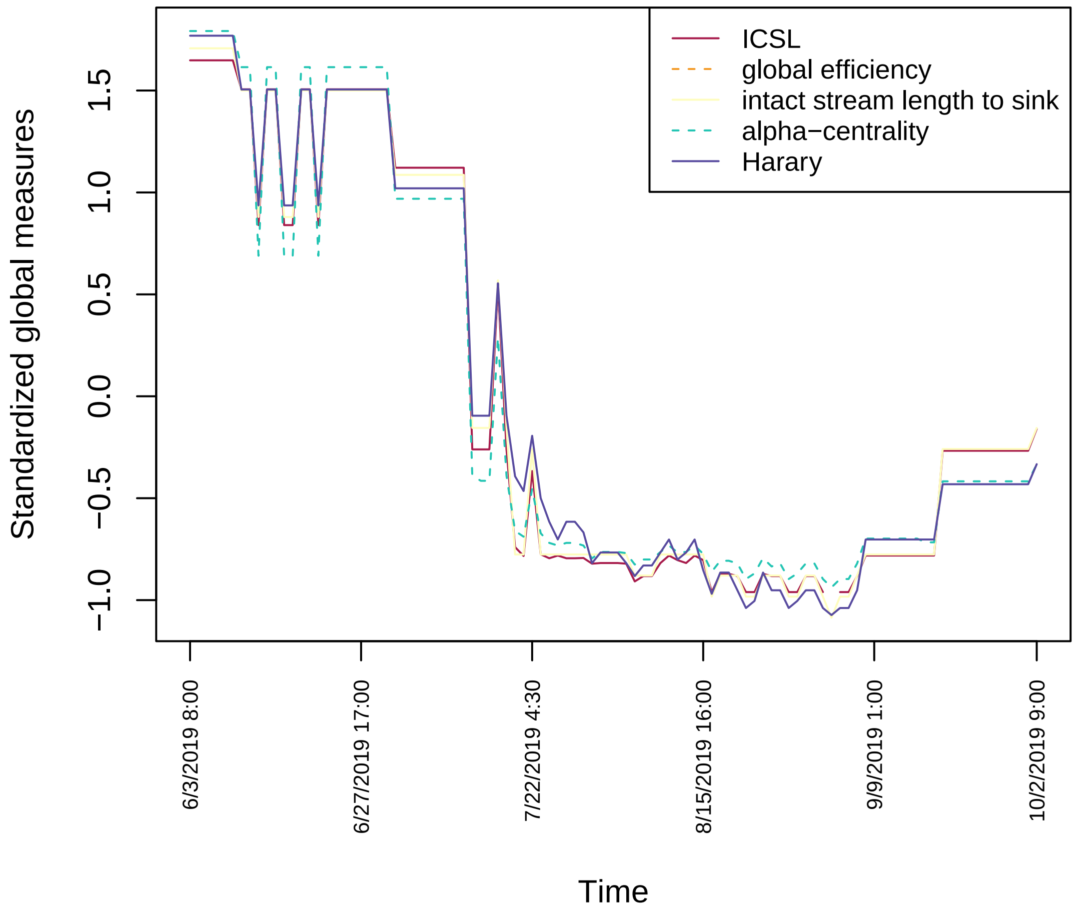

Figure 15: Global weighted network connectivity measures for Murphy Cr. over time.

The dataframe `mur_seasons_arc_pa` contains simulated arc presence/absence data for the spring, summer, and fall, represented as equal subdivisions of the sampling period. Specifically, the three time periods were: spring (6/3/2019 - 7/13/2019), and summer (7/13/2019 - 8/23/2019), and fall (8/23/2019 - 10/2/2019).

```r
data(mur_seasons_arc_pa)
```

Fig 16 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.

```r
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)
```

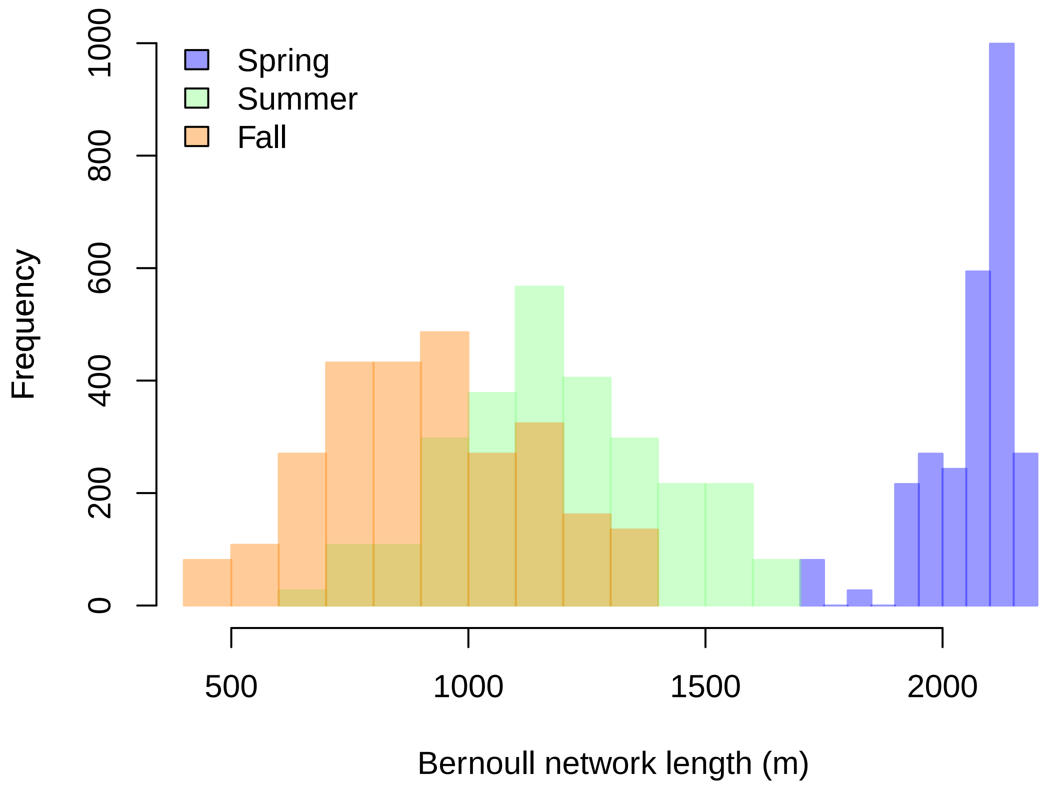

Figure 16: Distributions of Bernoulli network lengths for the seasonal designations.

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
```

2.1.6 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:

```
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.

```
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
```

We can use this information to depict arc posteriors for the probability of stream presence Fig 17, and the inverse probability of stream presence Fig 18. Multiplying the former distributions by their respective stream lengths will give average Bernoulli stream lengths for the segments. Multiplying the latter distributions by their respective stream lengths will give average communication distances for the segments.

```r
means <- b$alpha/(b$alpha + b$beta)
col <- gray(means/max(means))
par(mfrow = c(6,5), oma = c(4,4.5, 0.1, 1), mar = c(0,0,1.2,0.6))

for(i in 1:27){
  x <- seq(0.,1,by = .001)
  y <- dbeta(x, b$alpha[i], b$beta[i])
  n <- length(x)
  plot(x, yaxt = ifelse(i %in% c(1,6,11,16,21,26), "s", "n"),
       xaxt = ifelse(i %in% 23:27, "s", "n"), type = "n", xlim = c(0,1), ylim = c(0,5.5), cex.axis = .8)

  polygon(c(x, x[n:1]), c(y, rep(0,n)), col = col[i], border = "grey")
  segments(means[i], 0, means[i], dbeta(means[i], b$alpha[i], b$beta[i]), lty = 2)
  mtext(side = 3, names(b$beta)[i], cex = .5)
}

#axis labels
mtext(side = 2, outer = T, expression(paste(italic(f),"(",theta[italic(k)],"|",italic(x[k]),")")), line = 2.5)
mtext(side = 1, outer = T, expression(paste(theta[italic(k)],"|",italic(x[k]))), line = 2.5)
```

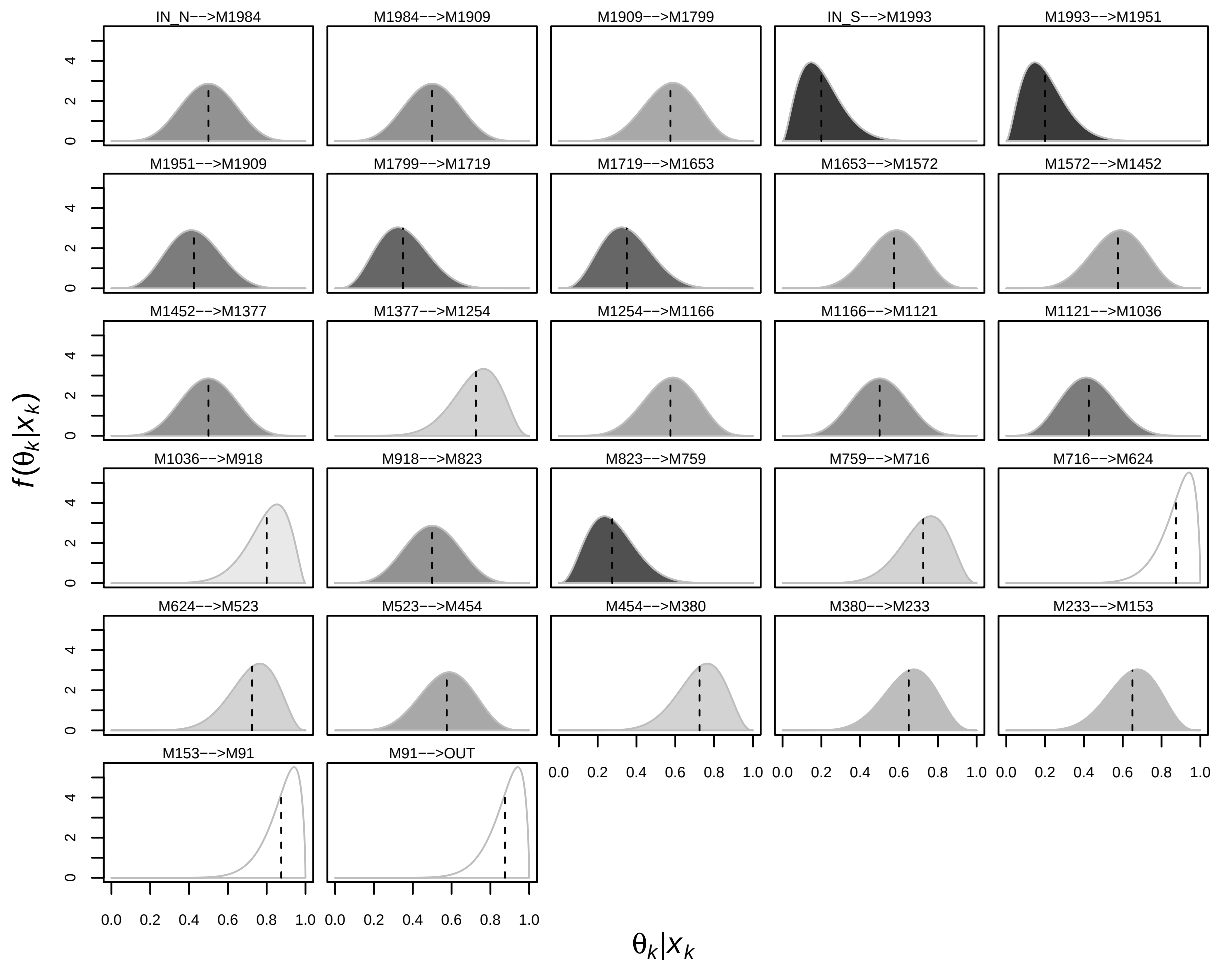

Figure 17: Graphical summaries of posterior beta distributions for Murphy Creek stream segments from 06/01/2019 to 10/01/2019. The posteriors represent distributions of probabilities of stream activity. Arc distributions are colored by their mean values (darker distributions have smaller means). The posterior means are overlain on the distributions with dashed lines.

```r
means <- (b$alpha + b$beta - 1)/(b$alpha - 1)
col <- gray(means/max(means))
par(mfrow = c(6,5), oma = c(4,4.5, 0.1, 1), mar = c(0,0,1.2,0.6))

for(i in 1:27){

  if(i %in% 1:10){lim <- c(0,1.1)} else {
    if(i %in% 11:15){lim <- c(0,2.3)} else {lim <- c(0,5)}}
  x <- seq(1,30,by = .01)
  y <- dinvbeta(x, b$alpha[i], b$beta[i])
  n <- length(x)
  plot(x, yaxt = ifelse(i %in% c(1,6,11,16,21,26), "s", "n"),
       xaxt = ifelse(i %in% 23:27, "s", "n"), type = "n", xlim = c(1,15), ylim = lim, cex.axis = .8, log = "x")

  polygon(c(x, x[n:1]), c(y, rep(0,n)), col = col[i], border = "grey")
  segments(means[i], 0, means[i], dinvbeta(means[i], b$alpha[i], b$beta[i]), lty = 2)
  mtext(side = 3, names(b$beta)[i], cex = .5)
}

#axis labels
  mtext(side = 2, outer = T, expression(paste(italic(f),"(",theta[italic(k)],"|",italic(x[k]),")",""^{-1})), line = 2.5)
  mtext(side = 1, outer = T, expression(paste("(",theta[italic(k)],"|",italic(x[k]),")",""^{-1})), line = 2.5)
```

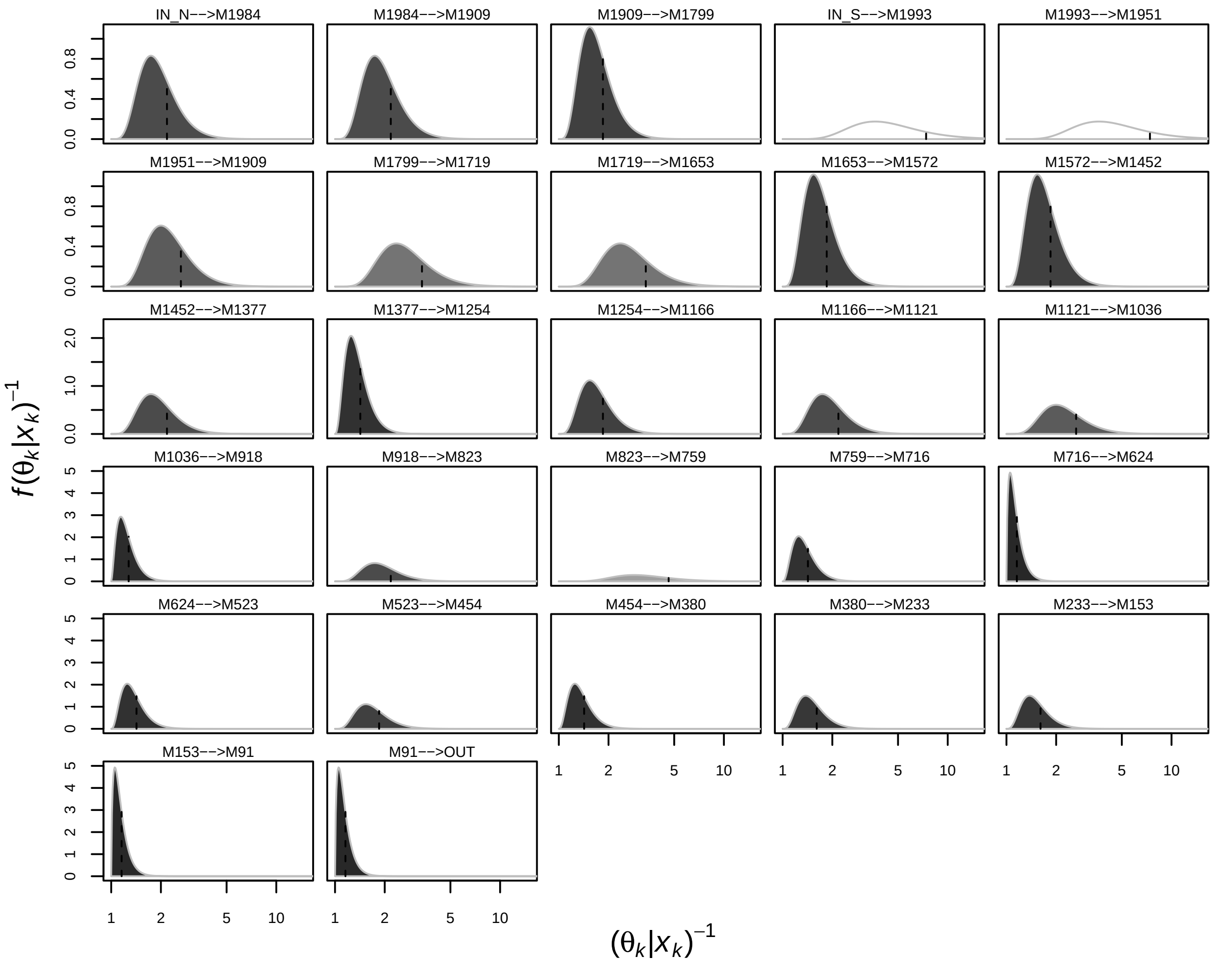

Figure 18: Graphical summaries of posterior inverse beta distributions for Murphy Creek stream segments from 06/01/2019 to 10/01/2019. The posteriors represent distributions of inverse probabilities of stream activity. Arc distributions are colored by their distributional mean values (darker distributions have smaller inverse probabilities). The posterior means are overlain on the distributions with dashed lines.

2.2 Konza Prairie

For comparison, we now briefly consider Konza Prairie, a more complex intermittent stream network in the northern Flint Hills region of Kansas, USA (39.11394°N, 96.61153°W). The network contains 42 nodes and 41 arcs with three major reaches and eight source nodes. Codification of the complete Konza Prairie (and the complete Murphy Creek network) are contained in the *streamDAG* function `streamDAGs`.

```r
kon_full <- streamDAGs("konza_full")
```

A `spatial.plot` of the full wetted network is shown in Fig 19.

```r
data(kon_coords)
spatial.plot(kon_full, kon_coords[,3], kon_coords[,2], names = kon_coords[,1])
```

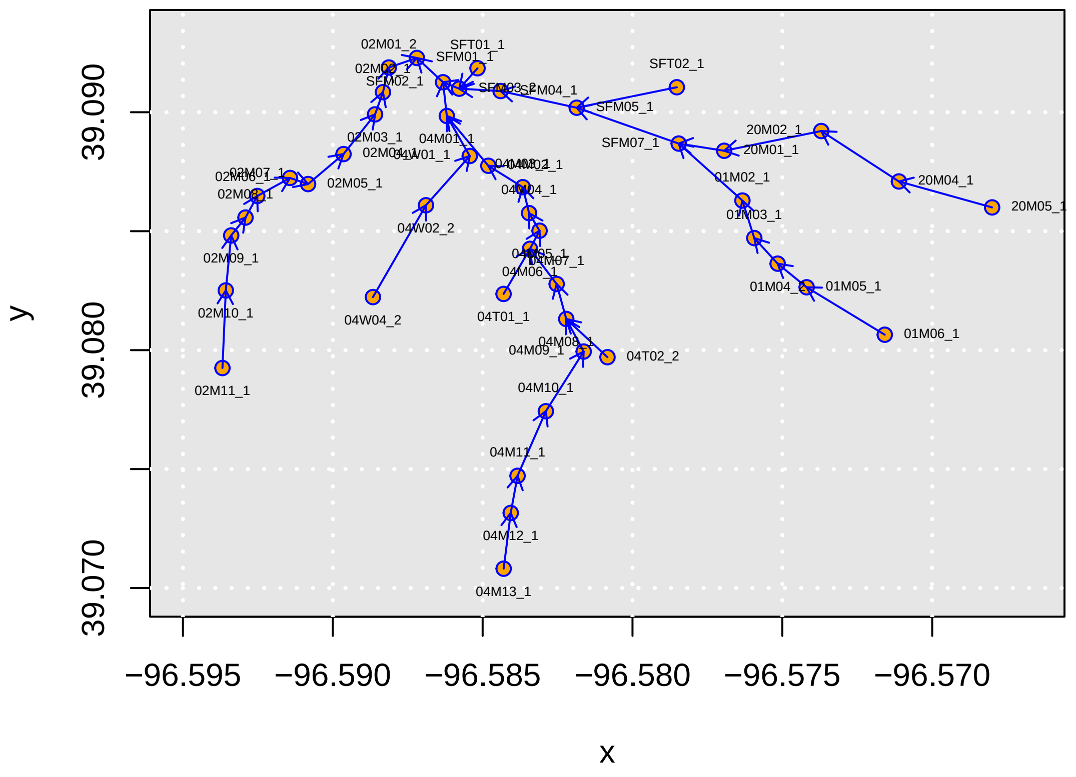

Figure 19: Spatially explicit graph of the completely wetted Konza Prairie network.

When applying the definition of matrix multiplication to an adjacency matrix $\mathbf{A}$, the i, j entry in $\mathbf{A}^k$ will give the number of paths in the graph from node i to node j of length k. The result of the computation of $\mathbf{A}^k$ (in paths of length k) is provided by `A.mult`. The actual matrix $\mathbf{A}^k$ can also be obtained from the function.

```
A.mult(kon_full, power = 6, text.summary = TRUE)

Paths of length 6:
 [1] "04M13_1 to 04M07_1" "04M12_1 to 04M06_1" "04M11_1 to 04M05_1"
 [4] "04M10_1 to 04M04_1" "04M09_1 to 04M03_1" "04T02_2 to 04M03_1"
 [7] "04M08_1 to 04M02_1" "04M07_1 to 04M01_1" "04T01_1 to 04M01_1"
[10] "04M06_1 to SFM02_1" "20M02_1 to SFM02_1" "01M03_1 to SFM02_1"
[13] "02M11_1 to 02M05_1" "02M10_1 to 02M04_1" "02M09_1 to 02M03_1"
[16] "02M08_1 to 02M02_1" "02M07_1 to 02M01_2" "04M05_1 to SFM01_1"
[19] "02M06_1 to SFM01_1" "20M01_1 to SFM01_1" "01M02_1 to SFM01_1"
[22] "01M06_1 to SFM05_1" "20M05_1 to SFM04_1" "01M05_1 to SFM04_1"
[25] "20M04_1 to SFM03_2" "01M04_2 to SFM03_2"
```

There are 26 paths of length six in the full Konza network.

The complete Konza network has a Strahler order of three (Fig 20) and a Shreve order of nine (Fig 21). From either perspective the Murphy Cr. network has a stream order of two .

```r
sok <- stream.order(G = streamDAGs("konza_full"), sink = "SFM01_1", method = "strahler")

colk <- as.character(factor(sok, labels = topo.colors(3, rev = TRUE)))
spatial.plot(G = streamDAGs("konza_full"), x = kon_coords[,3], y = kon_coords[,2],
             names = kon_coords[,1], pt.bg = colk, cex = 1.5, cex.text = 0, plot.bg = "white")
legend("bottomright", title = "Strahler\nStream order", legend = unique(sok),
pch = 21, pt.cex = 1.5, pt.bg = unique(colk), ncol = 1, bty = "n", title.cex = 0.9)
```

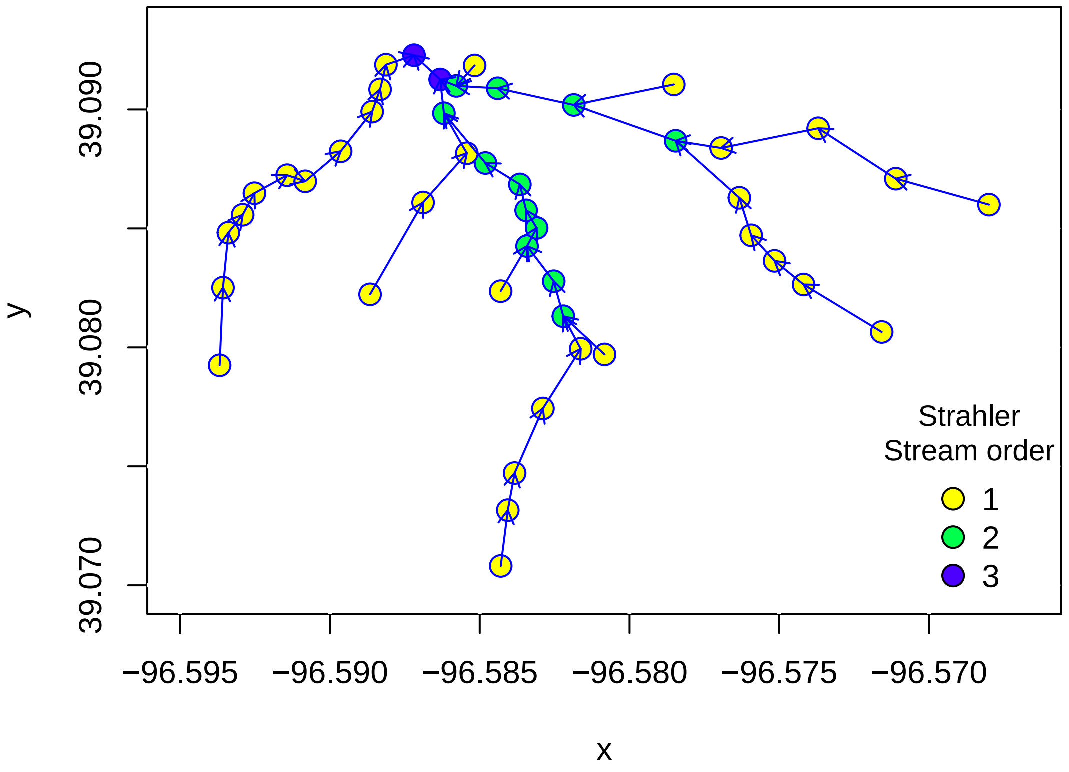

Figure 20: Strahler numbers for the complete Konza network.

```
sok <- stream.order(G = streamDAGs("konza_full"), sink = "SFM01_1", method = "shreve")
colk <- as.character(factor(sok, labels = topo.colors(6, rev = TRUE)))
spatial.plot(G = streamDAGs("konza_full"), x = kon_coords[,3], y = kon_coords[,2],
             names = kon_coords[,1], pt.bg = colk, cex = 1.5, cex.text = 0, plot.bg = "white")
legend("bottomright", title = "Shreve\nStream order", legend = unique(sok),
pch = 21, pt.cex = 1.5, pt.bg = unique(colk), ncol = 2, bty = "n", title.cex = 0.9)
```

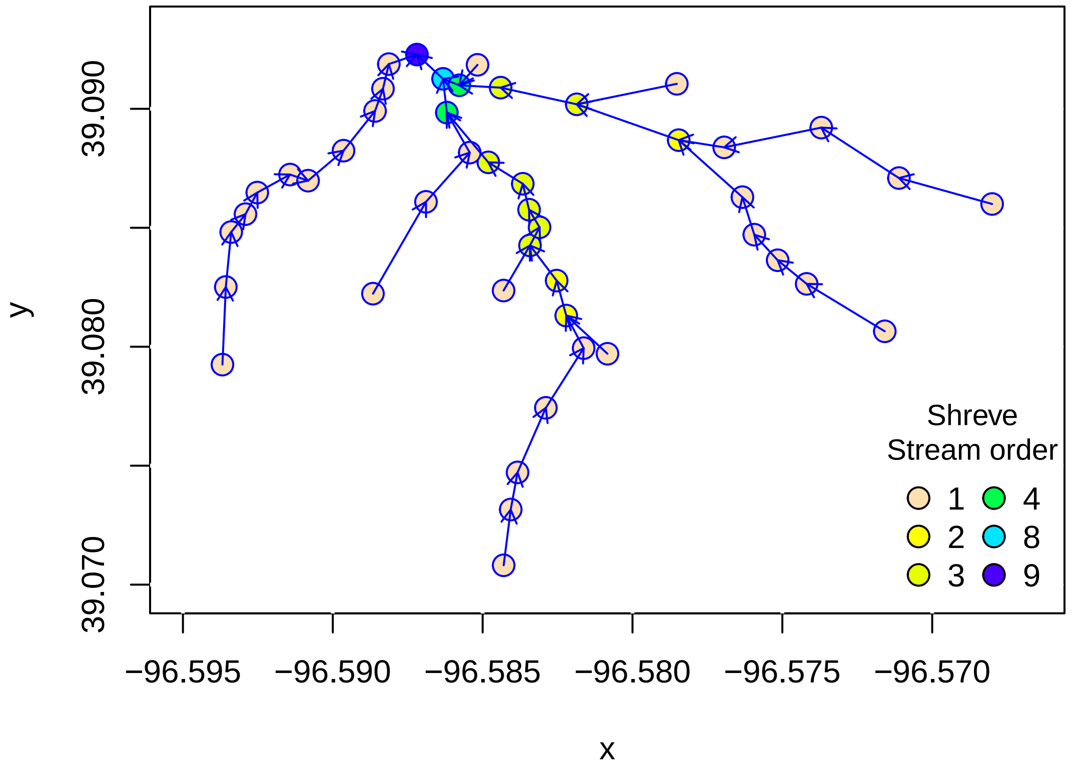

Figure 21: Shreve numbers for the complete Konza network.

2.2.1 Unweighted Local Measures

Increased nodal complexity of the complete Konza network, compared to Murphy Creek, is evident in the local metric summary in Fig 22.

```
local <- local.summary(kon_full)
cols <- brewer.pal(8,"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")
```

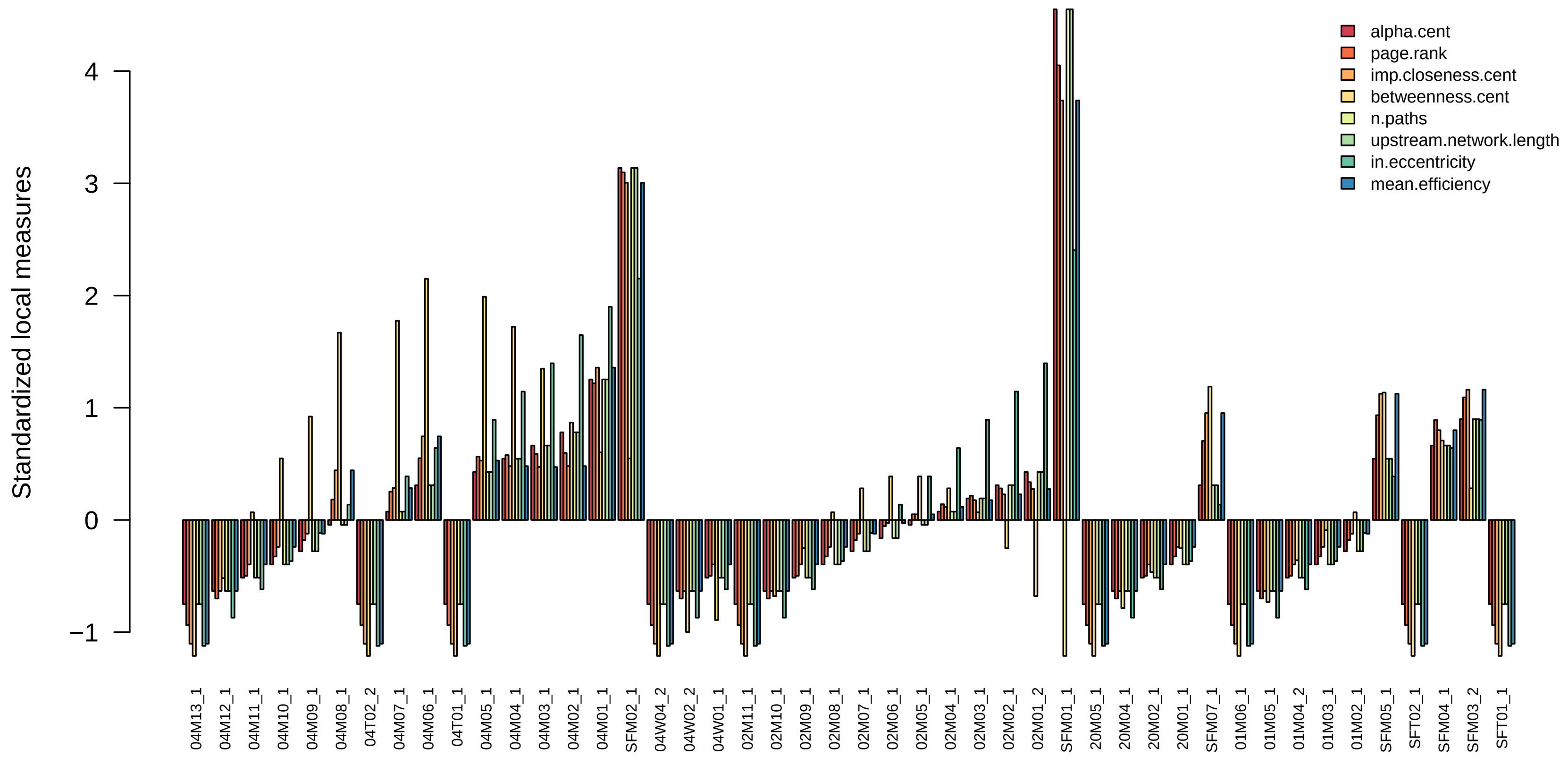

Figure 22: Local graph-theoretic summaries for Konza Prairie.

The importance of nodes at convergence points, is emphasized in a horizontal visibility graph summary (Fig 23).

```
vis <- multi.path.visibility(kon_full, source =
                               c("SFT01_1", "02M11_1","04W04_2",
                                 "04T01_1", "04M13_1",
                                 "SFT02_1","01M06_1", "20M05_1"),
                             sink = "SFM01_1", autoprint = F)
barplot(vis$visibility.summary, las = 2, cex.names = .6, ylab = "Visible nodes",
        legend.text = c("Downstream", "Upstream", "Both"),
        args.legend = list(x = "topleft", title = "Direction"))
```

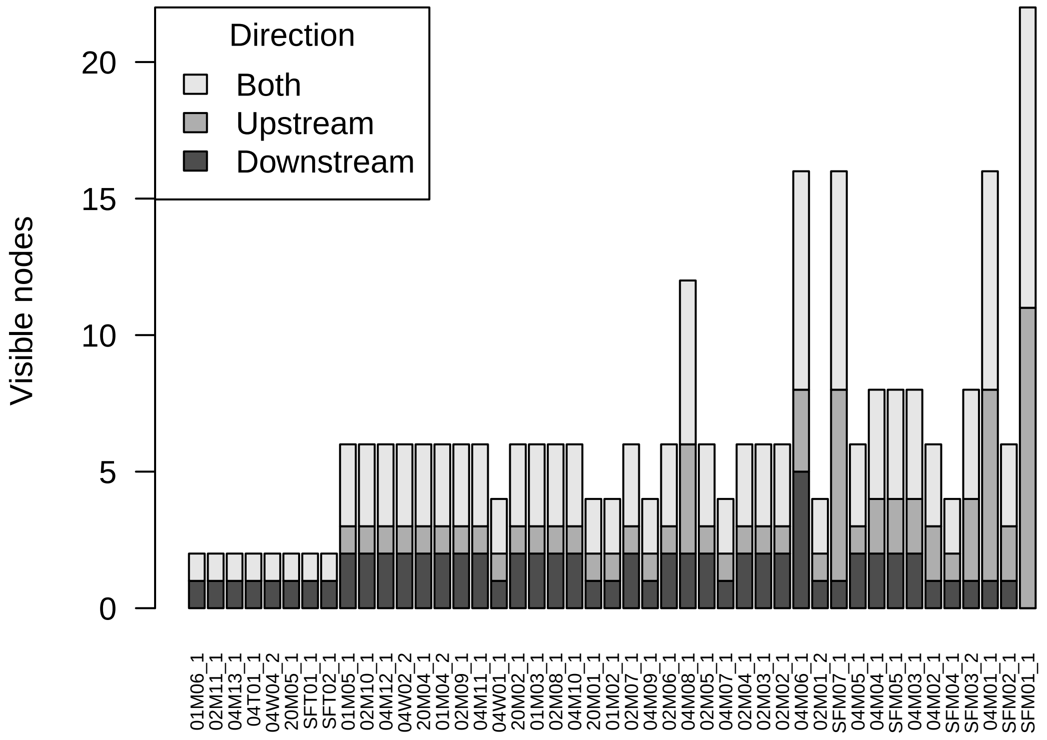

Figure 23: Nodal visibilities for for Konza Prairie based on nodal indegree.

2.2.2 Unweighted Global Measures

In 2021 the Konza network changed rapidly and dramatically from 05/21/2021 (before spring snow melt) to 05/28/2021 (during spring snow melt) to 06/04/2021 (drying following snow melt) (Fig 24).

```r
K0521 <- streamDAGs("KD0521"); K0528 <- streamDAGs("KD0528"); K0604 <- streamDAGs("KD0604")
kx <- kon_coords[,3]; ky <- kon_coords[,2]; kn <- kon_coords[,1]
par(mfrow = c(3,1), mar = c(0,0,1,0), oma = c(5,4,1,1))
spatial.plot(K0521, kx, ky, kn, xaxt = "n", cex.text = 0); legend("topright", bty = "n", legend = "A", cex = 2)
spatial.plot(K0528, kx, ky, kn, xaxt = "n", cex.text = 0); legend("topright", bty = "n", legend = "B", cex = 2)
spatial.plot(K0604, kx, ky, kn, cex.text = 0); legend("topright", bty = "n", legend = "C", cex = 2)
mtext(side = 1, "Longitude", outer = T, line = 3.5); mtext(side = 2, "Latitude", outer = T, line = 3)
```

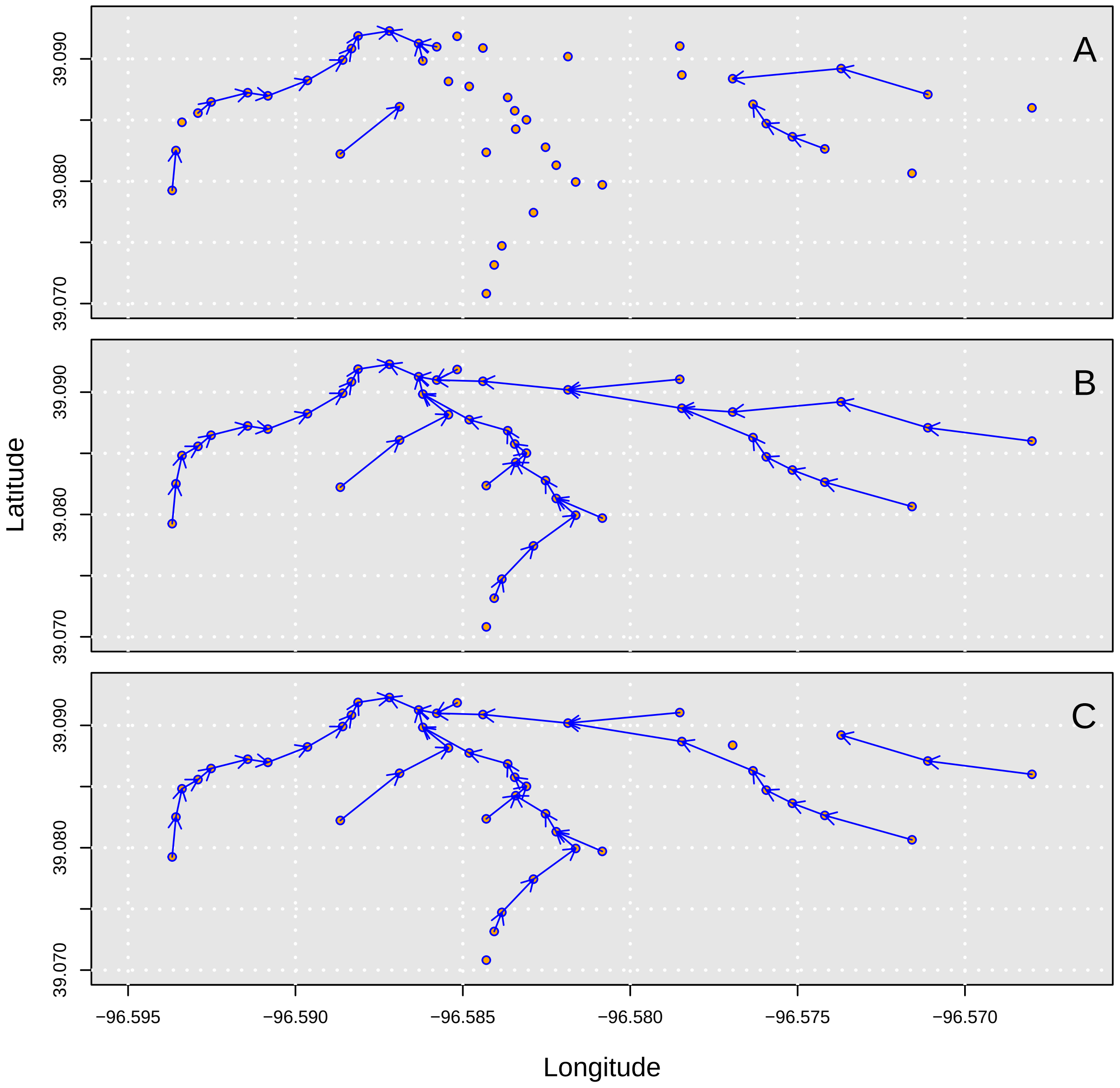

Figure 24: Spatial plot representations for Konza Prairie for (A) 05/21/2021 (B) 05/28/2021, and (C) 06/04/2021.

This is reflected in the global graph summaries for those dates (Fig 25).

```
g0521 <- global.summary(K0521, sink = "SFM01_1")
g0528 <- global.summary(K0528, sink = "SFM01_1")
g0604 <- global.summary(K0604, sink = "SFM01_1")
global <- cbind(g0521, g0528, g0604)
scaled.global <- scale(t(global))

matplot(scaled.global, xaxt = "n", type = "l", col = hcl.colors(15, palette = "spectral"),
        ylab = "Standardized global measures")
legend("topleft", lty = 1:5, col = hcl.colors(15, palette = "spectral"),
       legend = row.names(global), cex = .5, bg = "white")
axis(side = 1, at = c(1,2,3), labels = c("5/21/2021","5/28/2021","6/04/2021"),las = 2)
```

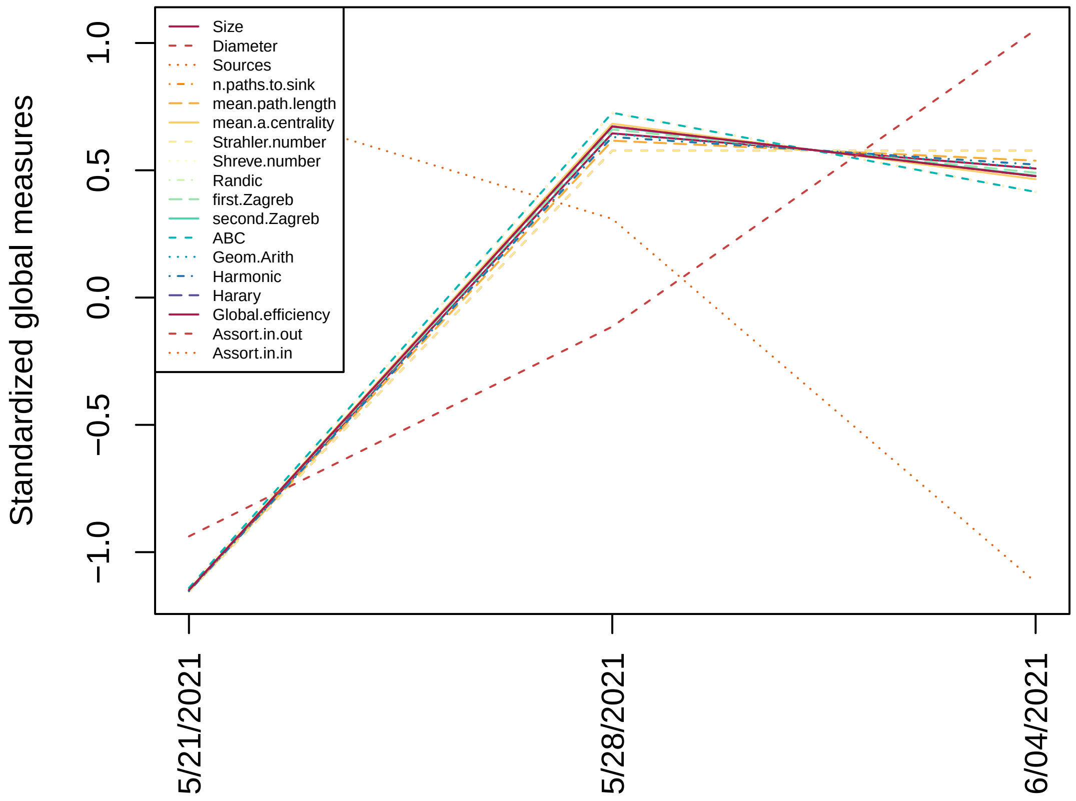

Figure 25: Global summaries for Konza Prairie for 05/21/2021, 05/28/2021, and 06/04/2021.

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. Let $x_{k,i}$ be a possible outcome from the kth arc, $k = 1, 2, \ldots, m$, with bounding nodes u and v, for the ith time frame, $i = 1, 2, 3, \ldots, n$. There are three possibilities:

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

This conversion is facilitated by the *streamDAG* function `arc.pa.from.nodes` which provides arc activity probabilities (using the rule above) based on bounding node 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).

```
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.