Reducing networks

Kelly Gallacher and Claire Miller and Marian Scott

2016-04-28

Contents

1 Set-up

1.1 Loading the package

The `stpca` package can be loaded as follows:

```
library(stpca)
```

2 Introduction

Monitoring network design is an important topic to ensure a balance between monitoring budgets and gathering enough information to monitor the status of a river network. One way to re-design a monitoring network is to reduce the number of samples taken over time, but another option is to consider how understanding of the river network might change if instead the number of monitoring sites is reduced. If the spatial sampling frequency is to be reduced, it is important to consider not only the amount by which the monitoring is to be reduced, but also how the retained sampling sites are selected.

This tutorial explains how to use the R functions developed as part of Kelly Gallacher's PhD (**???**) and the EPSRC SECURE network project 'Statistical software to identify spatiotemporal patterns and coherence over river networks' for identifying common patterns in river network data, a joint project between The University of Glasgow, The Environment Agency and The Scottish Environment Protection Agency.

This tutorial will use data files provided by the EA to illustrate how the `stpca` package can be used to:

- create sampled (reduced) networks using three sampling schemes
- fit spatial covariance models to the sampled networks
- visualize how understanding of the network changes when the number of monitoring sites is reduced.

Sampled networks are created by

- specifying the proportion of monitoring sites that should be retained in each sampled network
- specifying the sampling scheme that should be used to draw samples, without replacement, from the full monitoring network.

2.1 Sampling schemes

Sampled networks can be created in the `stpca` package using three sampling schemes:

- random sampling
- weighted sampling
- stratified sampling

Random sampling means that monitoring sites in the full network are selected for inclusion in a sampled network with equal probability. Sampling is carried out using `sample()` and with `replacement = FALSE` to ensure that no monitoring site is selected twice.

Weighted sampling means that monitoring sites in the full network are selected for inclusion in a sampled network with unequal probability. Sampling is performed without replacement. Weights must be probabilities in the range [0,1] but it is recommended that 0 and 1 are not used as this would either guarantee exclusion or inclusion of a monitoring site in every sampled network. In `demoNet` the probability of a monitoring site being included in a sampled network is found in the column `weights` in the `Obs` dataset.

Stratified sampling means that the sampled network reflects the composition of the full monitoring network. Stratified sampling is implemented using the `strata()` function in the `sampling` package and simple random sampling without replacement is used within each stratum, with each monitoring site within a single stratum having equal probablity of inclusion in the sampled network. In `demoNet` the strata information is found in the column `strata` in the `Obs` dataset.

2.2 Spatial covariance models

Spatial covariance models are part of a branch of statistics, often referred to as *geostatistics*. Many resources to explain geostatistics in detail can be found online but a very brief introduction of a geostatistical model will be given here to provide some context for the functions.

For data recorded at multiple sites across a geographic domain, it is reasonable to assume that the values are related in some way i.e. they are not independent. In geostatistics, the strength of relationship is usually assumed to depend on the distance between two monitoring sites. Distance is commonly measured as Euclidean distance (*as the crow flies*) but in river networks this might not be the best measure of distance to use. Recently, E. E. Peterson and ver Hoef (2010) showed how stream distance (*as the fish swims*) could be incorporated into geostatistical models, either as the only measure of distance between sites, or in combination with Euclidean distance.

This combination of distance measures is suitable for river networks where the variable of interest is related to the land in which the river network is embedded. For example, nitrate levels are related to agriculture (diffuse source pollution), but nitrate levels at a monitoring site could also be related to sites upstream due to pollution entering the network from a sewage outlet (point source pollution).

Geostatistical methods can be used to model the distance based relationship between monitoring sites. Several covariance models can be used for this but in the `stpca` package, the Gaussian model is used for Euclidean distance based relationships or, where appropriate, a hybrid covariance model with Epanechnikov Tail-up (stream distance) and Gausian (Euclidean distance) components is implemented.

The geostatistical models are fitted in the `sampNet()` function assuming no spatial trend, and that Euclidean distance based relationships are not direction dependent. For stream distance based relationships, it is assumed that monitoring sites are dependent on upstream monitoring sites and so this relationship is direction dependent. Further details about the specific geostatistical models fitted by the `sampNet` function can be found using `?sampNet`.

2.3 Data formats

Data can be entered in one of two formats: an SSN object (see below), or a `"data.frame"`.

`"data.frame"`: A data frame should include coordinates for monitoring site locations, a response variable (such as nitrates), and variables that are required for weighted and stratified sampling. A seconde `"data.frame"` will also be required containg coordinates for prediction locations, where the response variable is not observed.

SSN object: This is a data object created using ArcGIS software containg spatial information about a river network, as well as information about each stream segment and monitoring site. This SSN object is imported into R and contains information such as stream distances between flow connected monitoring sites, coordinates for monitoring site locations, a response variable (such as nitrates), an additive function value (see below), and variables that are required for weighted and stratified sampling. The SSN object should also include coordinates for prediction locations, where the response variable is not observed.

additive function value: This is calculated from information in the SSN object and is used to calculate spatial weights using the `createWeightS()` function. The spatial weights reflect flow direction and the relative influence of upstream monitoring sites on downstream sites. For example, a monitoring site located on a small tributary that is upstream from a monitoring site on the main stem of the river, will have less influence on the downstream site than a monitoring site located directly upstream on the main stem.

2.4 The data

Data for creating sampled networks can either be from an SSN object (see Erin E Peterson and ver Hoef (2014) for further details of SSN objects), or from an object of class `"data.frame"`. If data are in a `"data.frame"` then a second `"data.frame"` containing the coordinates of prediction locations will also be required.

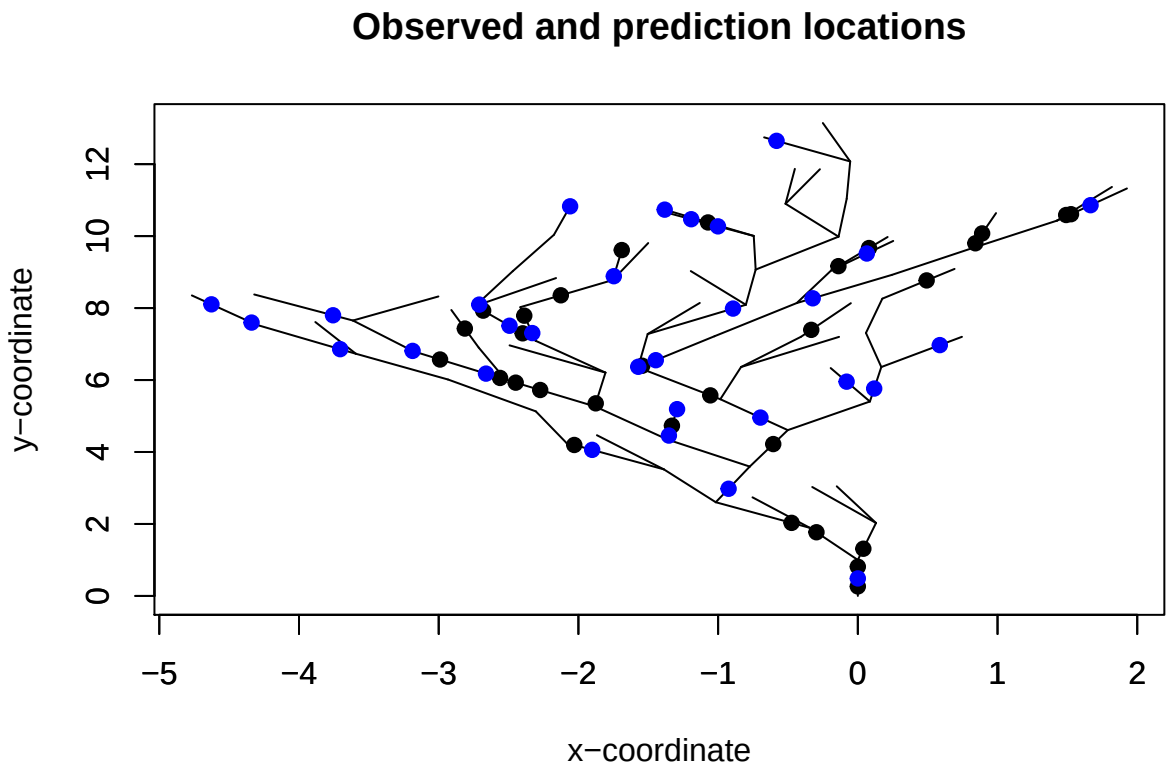
The data should include a column of observed values of a water quality parameter, such as nitrates. A river network with simulated observations at 30 monitoring sites is available in the `stpca` package and can be accessed as an SSN object using:

```
data(demoNet)
names(demoNet)
```

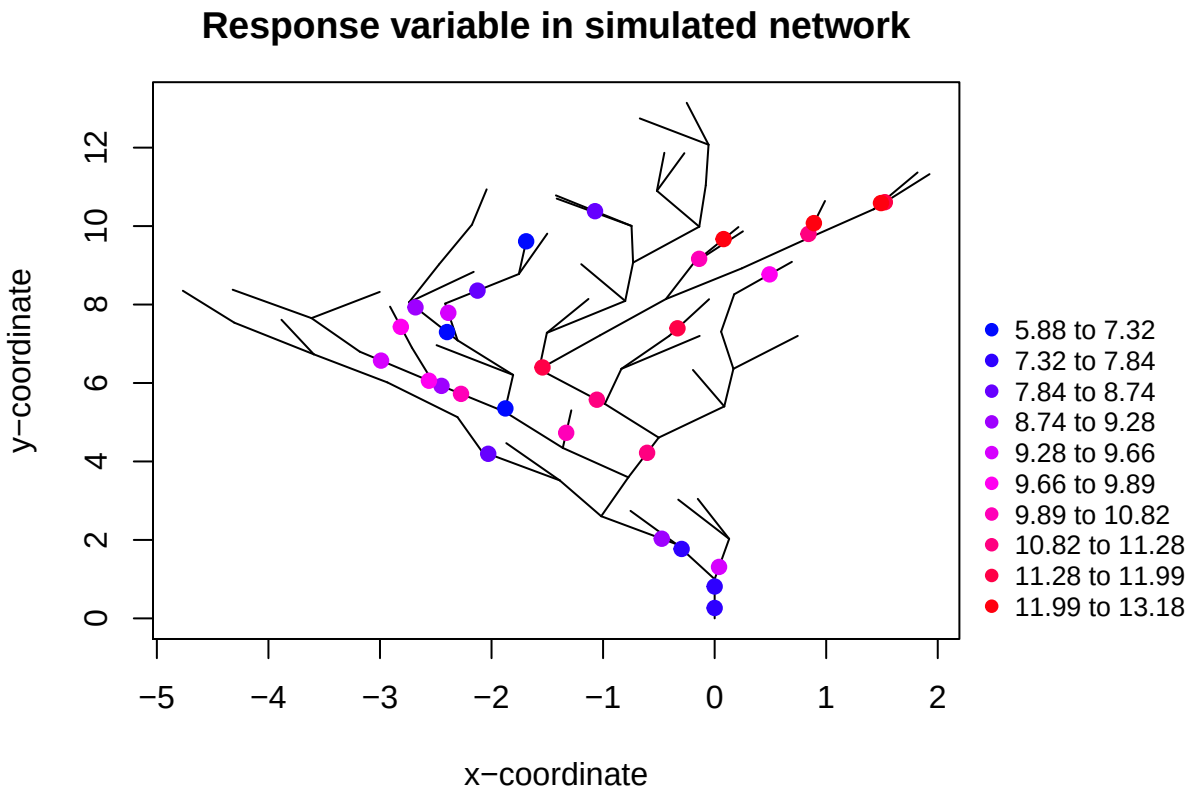
```
## $Obs
##  [1] "locID"      "upDist"     "pid"        "netID"      "rid"
##  [6] "ratio"      "shreve"     "addfunccol" "NEAR_X"     "NEAR_Y"
## [11] "Sim_Values" "strata"     "weight"
##
## $preds
##  [1] "locID"      "upDist"     "pid"        "netID"      "rid"
##  [6] "ratio"      "shreve"     "addfunccol" "NEAR_X"     "NEAR_Y"
## [11] "Sim_Values" "strata"     "weight"
```

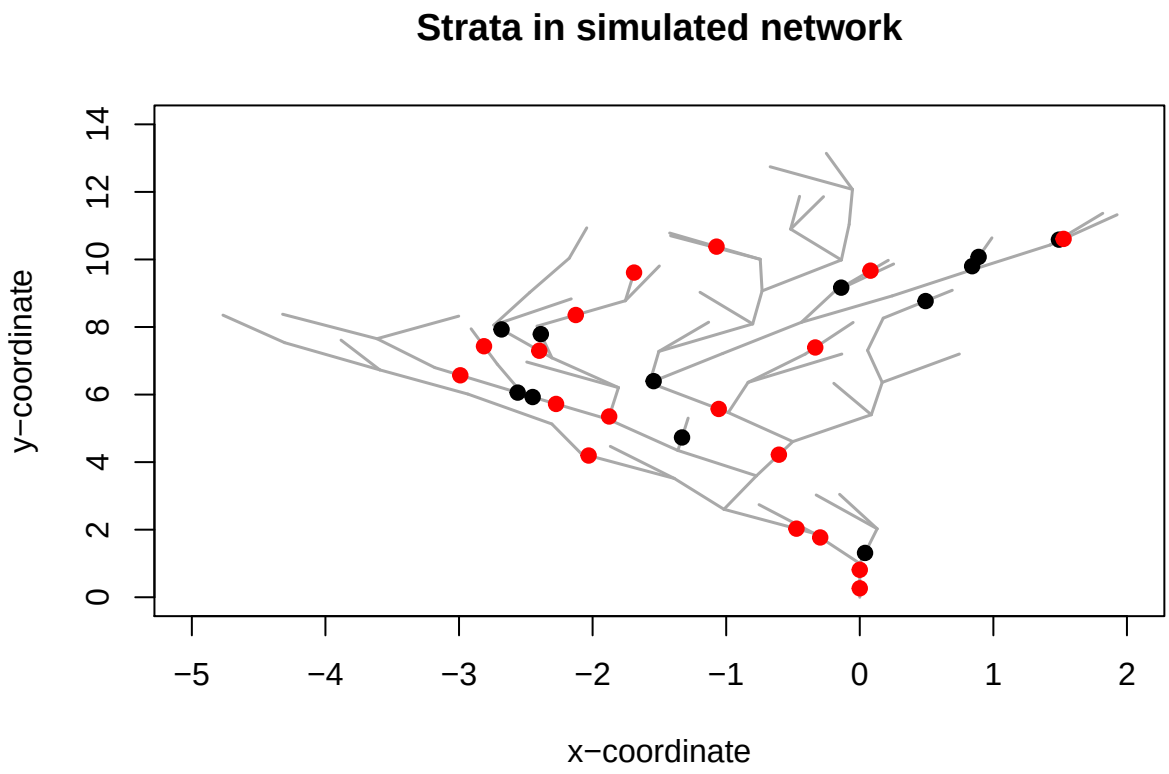
This SSN object has two datasets: `Obs` contains simulated 'observed' values and `preds` contains prediction locations on the simulated river network. Full details of this simulated SSN object can be found using `?demoNet`.

The river network is shown below, with monitoring locations (`"Obs"`) in black, and prediction locations (`"preds"`) in blue. There are 30 observed locations and 30 prediction locations.



The plot below shows the observed locations, coloured by simulated observed values, while the second plot shows the observed locations coloured by strata (1 = black, 2 = red).


Response variable in simulated network
y−coordinate
0
2
4
6
8
10
12
−5
−4
−3
−2
−1
0
1
2
x−coordinate
5.88 to 7.32
7.32 to 7.84
7.84 to 8.74
8.74 to 9.28
9.28 to 9.66
9.66 to 9.89
9.89 to 10.82
10.82 to 11.28
11.28 to 11.99
11.99 to 13.18



3 Creating sampled networks

The `sampNet` function can be used to create several sampled (reduced) monitoring networks, of varying sizes, using up to three sampling schemes. This allows investigation of the effect of the sampling scheme, as well as the effect of reducing the number of monitoring sites.

3.1 SSN object

Assuming the data are of `SpatialStreamNetwork` class, sampled networks can be created as follows:

```
# get the data
x <- importSSN(system.file("demoSSN/demoNet.ssn", package = "stpca"),
               predpts = "preds")
```

```
## binaryID.db already exists - no changes were made to binaryID.db table
```

```
# create sampled networks
test.samp <- sampNet(x, nsim = 3, prop.sites = c(0.5),
                     samp.scheme = c("random"),
                     sub.filepath = paste(tempdir(),"/subset1.ssn", sep = ""),
                     predpts = "preds", response.col = "Sim_Values",
                     addfunccol = "addfunccol", siteID = "locID")
```

```
## binaryID.db already exists - no changes were made to binaryID.db table
## [1] "random 50 1"
## binaryID.db already exists - no changes were made to binaryID.db table
## [1] "random 50 2"
## binaryID.db already exists - no changes were made to binaryID.db table
## [1] "random 50 3"
```

This function requires the specification of several arguments but these mostly tell the function which columns of the data object are of interest. The user must always specify `x`: the object containing the data, `nsim`: the number of sampled networks to create, `prop.sites`: the proportion of monitoring sites to retain in the sampled networks, and `samp.scheme`: the sampling method that should be used to create samples.

If `class(x)` is `"SpatialStreamNetwork"`, you must also specify some additional arguments: `sub.filepath` is a filepath to a location where SSN objects can be temporarily created and stored, `predpts` is the name of the data set containing predictions locations in the SSN object, `response.col` is the column in the data set of observed values in the SSN object containing the response variable of interest, `addfunccol` is the column name in the SSN object containing the additive function values, and `siteID` is the column name in the SSN object containing a unique identifier for each monitoring site.

`test.samp` will contain the following:

```
names(test.samp)
```

```
## [1] "random.50"     "results.names" "prop.sites"    "samp.scheme"
## [5] "all"           "columns"       "call"
```

`random.50` contains

```
## [1] "samp.site"     "cov.params"    "var.comps"     "crossval.pred"
## [5] "crossval.se"   "cross.stats"   "preds.value"   "preds.se"
```

and `all` contains

```
## [1] "all.model"       "all.covparams"   "all.varcomps"    "all.cross.stats"
## [5] "all.preds"       "class"
```

The structure of the output is described in detail in the help file for this function, which can be accessed using `?sampNet`. The main thing is that `random.50` (or similar) is a list containing results from `nsim` networks created using the sampling scheme (listed before the dot) and retaining the proportion of monitoring sites listed after the dot. `all` is a list containing results for the full monitoring network, and these can be used as baseline comparisons to the sampled networks. The rest of the output is used for plotting the results. The results can be plotted using the `plot_sampNet` function.

3.2 Data Frame

If your data are in a `"data.frame"` then you can create sampled networks using the following:

```
# get prediction points coordinates
library(SSN)

# get data as a data.frame
x <- getSSNdata.frame(demoNet, "Obs")
```

```
# get prediction locations
preds.coords <- getSSNdata.frame(demoNet, "preds")[,c("NEAR_X", "NEAR_Y")]

# create sampled networks
test.samp <- sampNet(x, nsim = 3, prop.sites = c(0.5),
                     samp.scheme = c("random"),
                     data.col = "Sim_Values", coords.col = c(9, 10),
                     preds.coords = preds.coords, siteID = "locID")
```

Here, `class(x)` is `"data.frame"` and `nsim`, `prop.sites`, `samp.scheme`, `data.col`, and `siteID` are specified in the same way as when `class(x)` is `SpatialStreamNetwork`.

If your data are in a data frame, then you must also specify `coords.col` (a vector of the column numbers containing X and Y coordinates for the monitoring sites) and a data.frame `preds.coords` containing X and Y coordinates for prediction locations.

4 Plotting the results

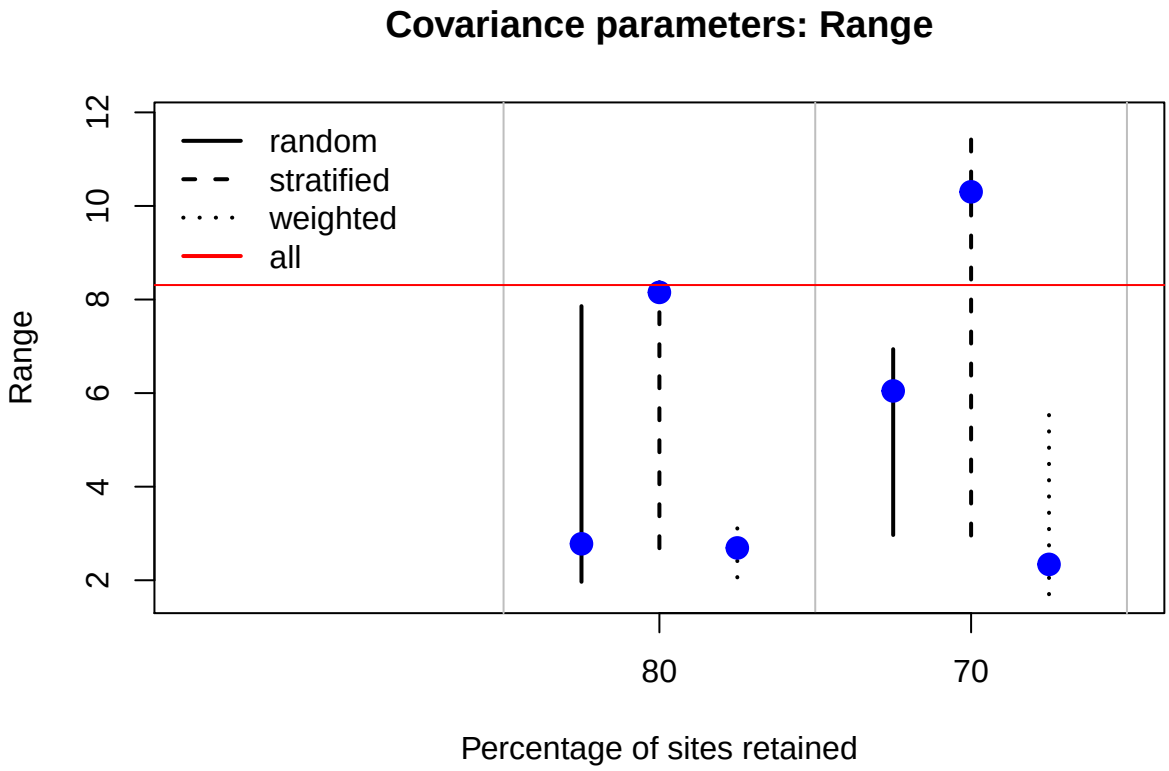
The results from `sampNet()` can be plotted using the `plot_sampNet()` function. An object has been provided in the `stpca` package, `sampDemo`, containing the results from the following application of `sampNet`. The data are entered into the `sampNet` function as a `data.frame` in this example.

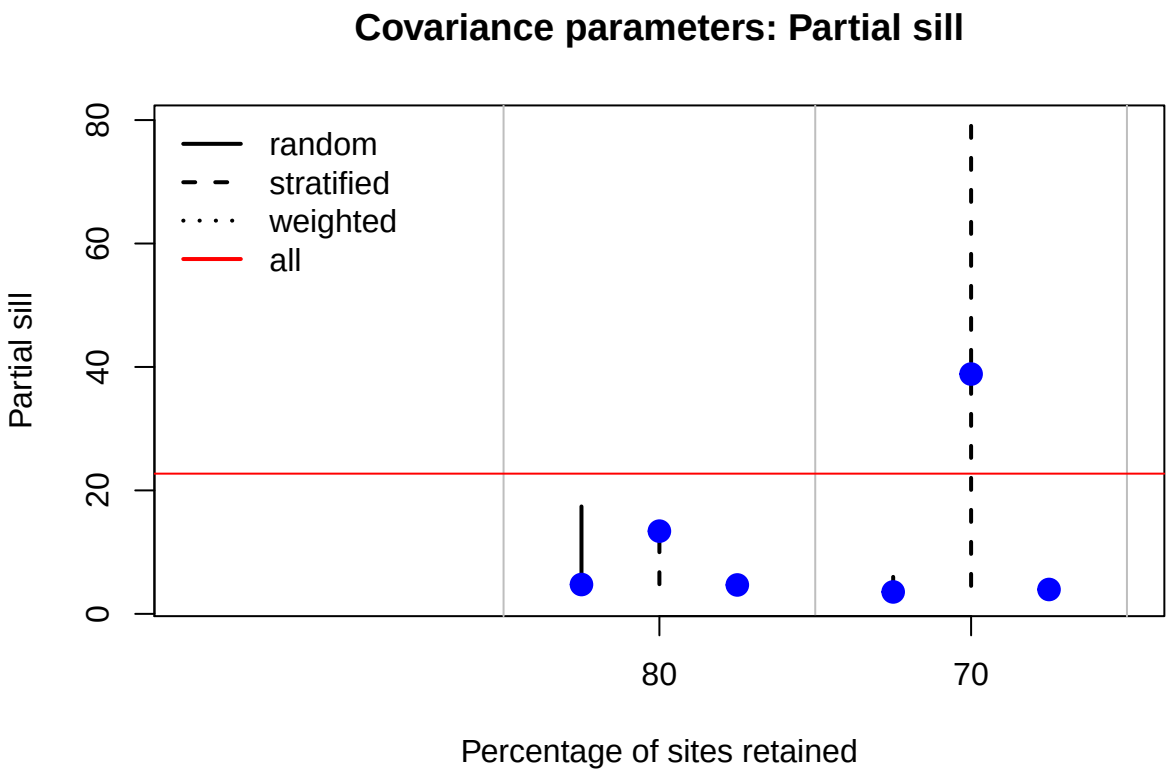
```
set.seed(618999)
data(demoNet)

# Observed values for the simulated network.
x <- getSSNdata.frame(demoNet, "Obs")

# Prediction locations for the simulated network
preds.coords <- getSSNdata.frame(demoNet, "preds")[,c("NEAR_X", "NEAR_Y")]

# Create the sampled networks
sampDemo <- sampNet(x, nsim = 5, prop.sites = c(0.8, 0.7),
                    samp.scheme = c("random", "stratified", "weighted"),
                    data.col = "Sim_Values", coords.col = c(9, 10),
                    preds.coords = preds.coords, siteID = "locID",
                    strat.col = "strata", weight.col = "weight")
```

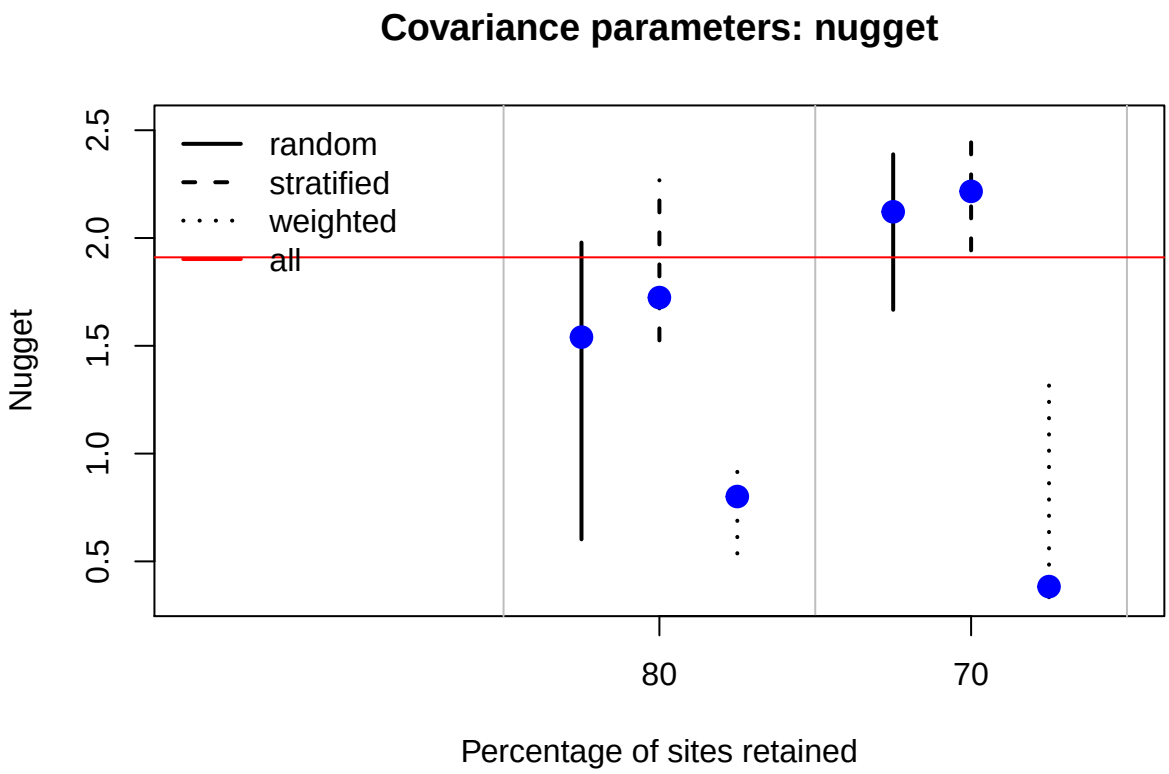
This example uses all three possible sampling schemes and creates sampled networks where two different proportions of monitoring sites are retained. Since weighted and stratified sampling schemes are used here, it is necessary to specify two further columns: `strat.col` is the column in the `data.frame` (or dataset of observed values in an SSN object) containing a factor variable with strata groups information, and `weight.col` contains weights in the range [0,1] for weighted sampling.

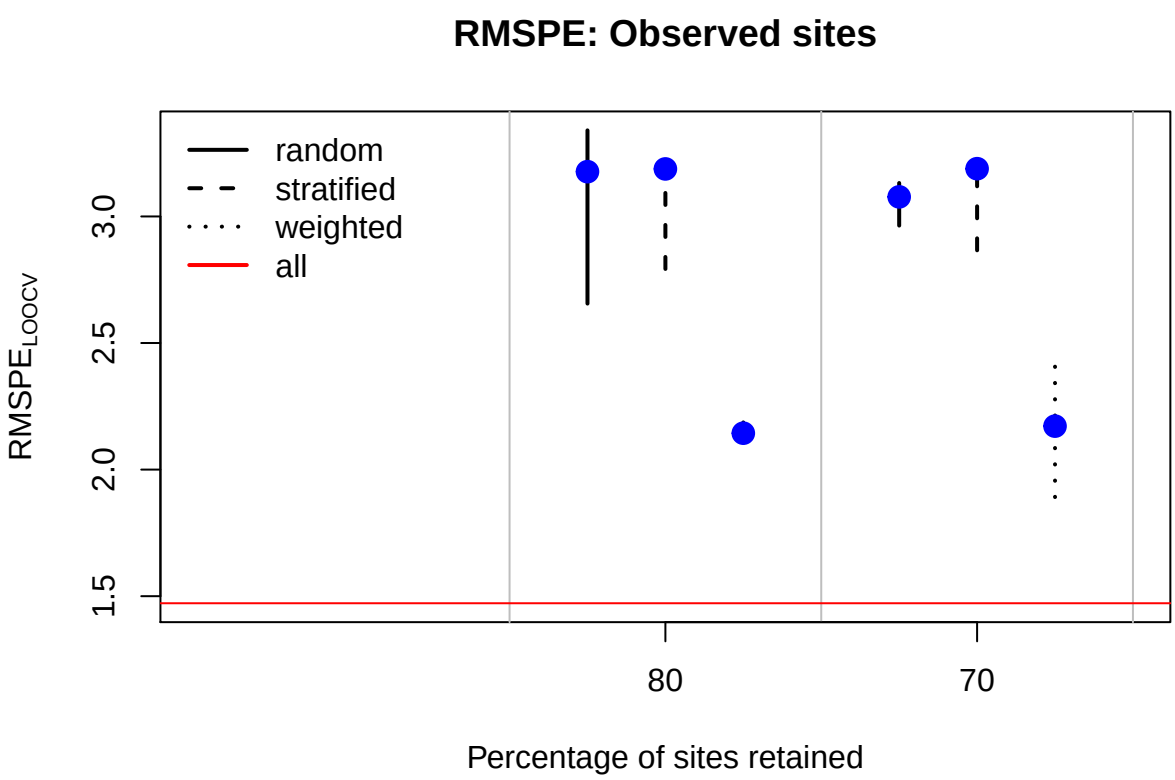
The results can be plotted using the `plot_sampNet()` function, shown below. You must specify `samp.results` which is the name of the object containing the results from `sampNet`. Other options are available to allow only some of the plots to be produced, or to plot the y-axis of certain plots on a log scale. Further information about these can be found by accessing the help file using `?plot_sampNet`.

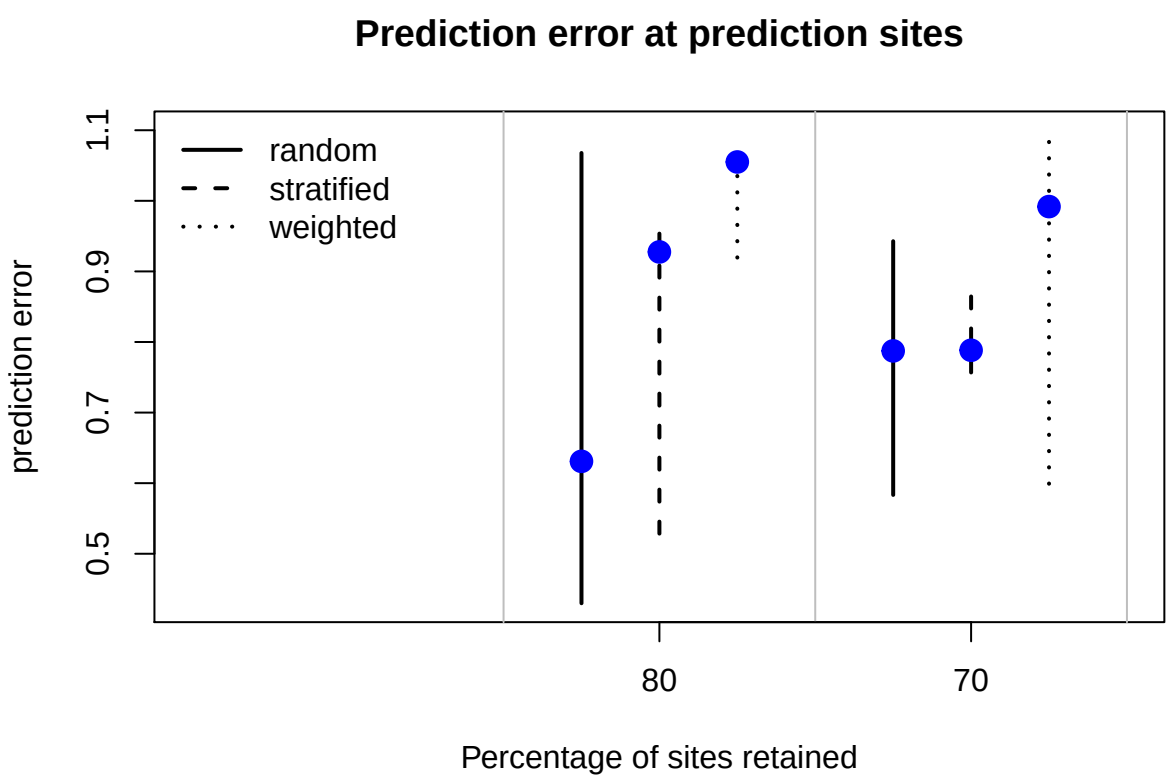
```
data(sampDemo)
plot_sampNet(samp.results = sampDemo)
```

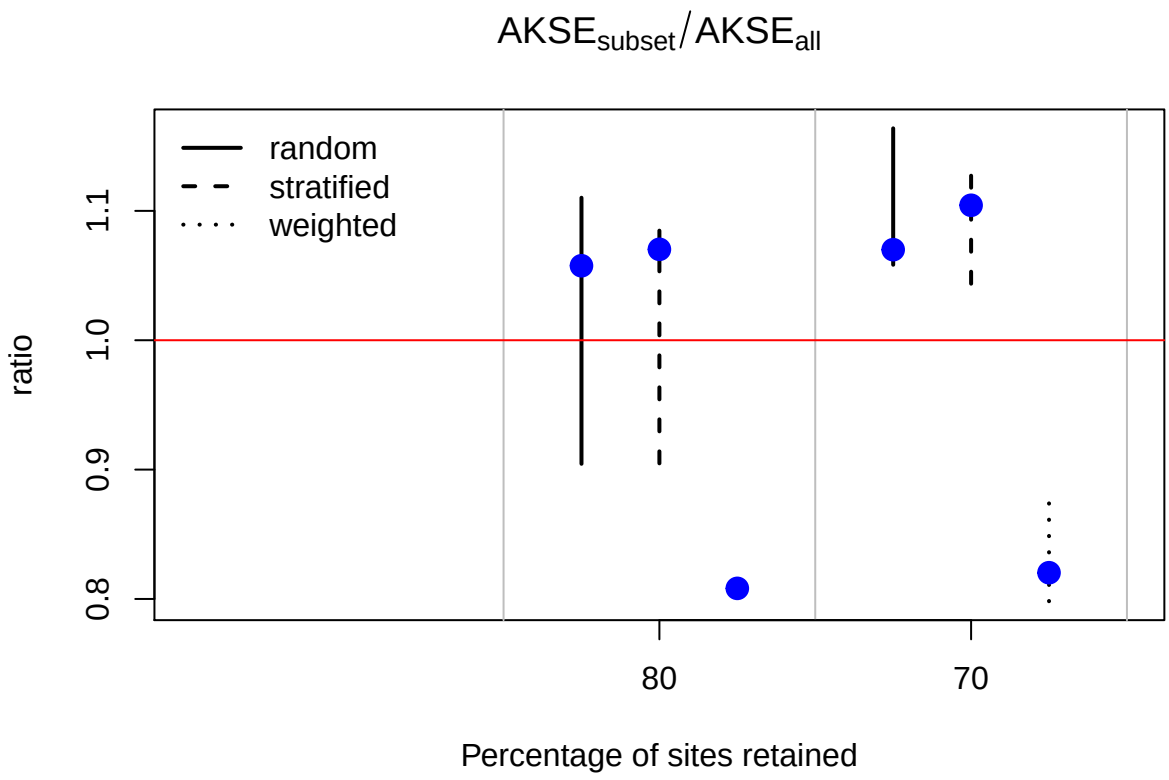

Covariance parameters: Range
random
stratified
weighted
all
Range
2
4
6
8
10
12
80
70
Percentage of sites retained


Covariance parameters: Partial sill
random
stratified
weighted
all
Partial sill
0
20
40
60
80
80
70
Percentage of sites retained


Covariance parameters: nugget
random
stratified
weighted
all
Nugget
0.5
1.0
1.5
2.0
2.5
80
70
Percentage of sites retained


RMSPE: Observed sites
random
stratified
weighted
all
RMSPE_LOOCV
1.5
2.0
2.5
3.0
80
70
Percentage of sites retained


Prediction error at prediction sites
random
stratified
weighted
prediction error
0.5
0.7
0.9
1.1
80
70
Percentage of sites retained



These plots can be used to compare the results between different sizes of monitoring networks, and between different sampling schemes. This will enable the user to assess the extent to which a monitoring network could be reduced, and to understand how the choice of sampling scheme can influence results.

The plots are basic summaries of all combinatins of `prop.sites` and `samp.scheme` from `sampNet()`. The median result from the `nsim` sampled networks is shown as a blue dot and the vertical bars span between the lower and upper quartiles. In several of the plots where appropriate, a horizontal red line indicates the corresponding result from the full monitoring network.

The first three or five plots (data are from a `data.frame` or SSN object, respectively) will show summaries of the covariance function parameters. If data are from a `data.frame`, these plots are the range, partial sill, and nugget parameters. If data are from an SSN object, then these plots will be the Tail-up range, Euclidean range, Tail-up partial sill, Euclidean partial sill, and nugget parameters.

The last three plots are the root mean squared prediction error (RMSPE) at observed monitoring sites, prediction error at prediction sites, and the average kriging standard error ratio. Descriptions of these can be found in the help file, accessed using `?plot_sampNet`.

4.1 Interpreting the plots

Next, we will consider how to interpret the plots, bearing in mind that these summaries are based on only five sampled networks for each combination of `prop.sites` and `samp.scheme`. Note that the interpretation of these plots are not general rules for the differences between sizes of sampled networks or sampling schemes, but are included here to provide suggestions as to features of the plot that might be of interest to the user.

Range parameter This plot suggests that range parameter estimated from sampled networks is generally smaller than for the full network, except for the weighted sampling scheme. There is no clear difference between the different sizes of sampled networks.

Partial sill This plot suggest that the partial sill estimated from sampled networks created using the weighted sampling scheme is closest to the partial sill estimated from the full network, for both sizes of network considered, since the median partial sill for the weighted sampling scheme is closer to the red line than the median values for the other sampling schemes.

Nugget This plot shows that nugget estimated from sampled networks created using the weighted sampling scheme is most different to the nugget estimated from the full network, for both sizes of network considered, since the median nugget for the weighted sampling scheme is further from the red line than the median values for the other sampling schemes. The weighted sampling scheme appears to underestimate the nugget compared to the full monitoring network.

RMSPE (observed sites) This plot shows that models fitted to sampled networks created using the weighted sampling scheme fit the data better than models fitted to sampled networks created using random or stratified sampling schemes, since the median values for the weighted sampling scheme are closer to the red line than values for the random or stratified sampling schemes.

Prediction error (prediction sites) This plot does not have a red reference line since difference are calculated as the differences between predictions made using a model based on the full monitoring network and predictions made using models based on sampled networks (see `?plot_sampNet` for further details). The plot shows that the weighted sampling scheme performs worse than the random and stratified sampling schemes since the predicition error is highest, on average, for sampled networks created using the weighted sampling scheme.

AKSE ratio This plot is used to understand how uncertainty around predictions at prediction sites changes as a result of decreasing the size of the monitoring network. The plot suggests that the standard error of predictions is about 20% lower for the sampled networks created under the weighted sampling scheme compared to the standard error of predictions from a model based on the full monitoring network. This plot also suggests that there is lower uncertainty in predictions from models based on sampled networks retaining 70% of the monitoring sites, compared to predictions from models based on sampled networks retaining 80% of the monitoring sites, since the median ratio is lower for the smaller networks! This is likely due to the very small number of sampled networks created and should not be taken as a general rule! Further details for the AKSE ratio can be found in the help files.

Full details of how to calculate RMSPE, prediction error, and AKSE ration can be found using `?plot_sampNet`.

5 Troubleshooting

5.1 `sampNet()`

5.1.1 Function is slow

The run time for this function increases as the number of monitoring sites increases, and will also depend on `nsim`, as well as the number of sampling schemes used and the length of the vector `prop.sites`. For example, for a dataset of ~500 sites, with `nsim = 500`, `prop.sites = c(0.9, 0.8, 0.5, 0.2, 0.1)`, and `samp.scheme = c("random", "weighted", "stratified")`, it took approximately 8 days to produce all of the sampled networks when data were contained in an SSN object.

If your data are stored in an SSN object, then the function can take a long time to run because for each `nsim/prop.sites/samp.scheme` combination, a new SSN object is produced and this requires calculation of a river distance matrix which is a time consuming process. Also, the spatial covariance model fitted to the SSN object takes longer to fit than the spatial covariance model fitted to a `"data.fame"` since a more complicated model is being fitted.

5.2 `plot_sampNet()`

5.2.1 Error appears

Sometimes the following Error appears:

```
Error in solve.default(v$varcov, xmat) :    system is computationally singular: reciprocal
condition number = 1.21203e-16
```

This happens when the spatial covariance model cannot be fitted to the sampled network. This might be because the sampled network contains very few monitoring sites. For example, when applying `sampNet()` to `demoNet`, the full network contained only 30 monitoring sites and the sampled network retaining 70% of monitoring sites contained only 21 sites. These are very small numbers of observations to modelled using a spatial covariance function. It is suggested that if this error message appears that you try increasing the proportion of sites to retain in the sampled networks.

References

Peterson, E. E., and J. M. ver Hoef. 2010. "A Mixed-Model Moving Average Approach to Geostatistical Modeling in Stream Networks." *Ecology* 91: 644–51.

Peterson, Erin E, and Jay M ver Hoef. 2014. "STARS: An ArcGIS Toolset Used to Calculate the Spatial Information Needed to Fit Spatial Statistical Models to Stream Network Data." *J Stat Softw* 56 (2): 1–17.