

Analysis of a GRTS Survey Design for a Linear Resource

Thomas Kincaid

February 3, 2012

Contents

1 Preliminaries	1
2 Read the survey design and analytical variables data file	2
3 Analysis of site status evaluation variables	4
4 Analysis of stream condition variables	8
5 Analysis of stream condition variables correcting for population size	10
6 Analysis of quantitative variables	12

1 Preliminaries

This document presents analysis of a GRTS survey design for a linear resource. The linear resource used in the analysis is streams in the Upper Wabash basin in Indiana. The analysis will include calculation of three types of population estimates: (1) estimation of proportion and size (length of streams) for site evaluation status categorical variables; (2) estimation of proportion and size for stream condition categorical variables; and (3) estimation of the cumulative distribution function (CDF) and percentiles for quantitative variables. Testing for difference between CDFs from subpopulations also will be presented.

The initial step is to use the library function to load the spsurvey package. After the package is loaded, a message is printed to the R console indicating that the spsurvey package was loaded successfully.

Load the spsurvey package

```
> # Load the spsurvey package
> library(spsurvey)
>
```

Version 2.2 of the spsurvey package was loaded successfully.

2 Read the survey design and analytical variables data file

The next step is to read the data file, which includes both survey design variables and analytical variables. The `read.delim` function is used to read the tab-delimited file and assign it to a data frame named `IN_streams`. The `factor` function is used to convert the `Strahler_order` variable, which contains numeric values, to a factor. A factor is a data structure in R that is used to encode variables that contain a specified set of categorical values, which are referenced as levels. The `nrow` function is used to determine the number of rows in the `IN_streams` data frame, and the resulting value is assigned to an object named `nr`. Finally, the initial six lines and the final six lines in the `IN_streams` data frame are printed using the `head` and `tail` functions, respectively.

Read the survey design and analytical variables data file

```
> # Read the data file and determine the number of rows in the file
> IN_streams <- read.delim("IN_streams.tab")
> IN_streams$Strahler_order <- factor(IN_streams$Strahler_order)
> nr <- nrow(IN_streams)
>
```

Display the initial six lines in the data file.

```
> # Display the initial six lines in the data file
> head(IN_streams)
```

	siteID	xcoord	ycoord	wgt	Strahler_order		status	TNT
1	INRB98-001	7574978	12556251	180.5	1	Landowner	Denial	Target
2	INRB98-002	7490780	12580320	180.5	1		Sampled	Target
3	INRB98-003	7500380	12545405	57.7	2		Sampled	Target
4	INRB98-004	7543291	12557975	26.4	4	Landowner	Denial	Target
5	INRB98-005	7459504	12689766	29.6	3		Sampled	Target
6	INRB98-006	7515791	12649268	57.7	2	Physical	Barrier	Target

	IBI_status	IBI_score	QHEI_status	QHEI_score
1	Not Sampled	NA	Not Sampled	NA
2	Not Impaired	50	Impaired	48

```

3      Impaired      22 Not Impaired      65
4 Not Sampled      NA Not Sampled      NA
5 Not Impaired      38      Impaired      31
6 Not Sampled      NA Not Sampled      NA

```

```
>
```

Display the final six lines in the data file.

```

> # Display the final six lines in the data file
> tail(IN_streams)

```

	siteID	xcoord	ycoord	wgt	Strahler_order	status	TNT
95	INRB98-095	7503714	12628803	57.7	2	Landowner Denial	Target
96	INRB98-096	7496237	12662502	180.5	1	NonTarget	NonTarget
97	INRB98-097	7483938	12665060	29.6	3	Chemistry Only	Target
98	INRB98-098	7496841	12634665	180.5	1	NonTarget	NonTarget
99	INRB98-099	7443767	12609995	26.4	4	Sampled	Target
100	INRB98-100	7445717	12651622	26.4	4	Chemistry Only	Target

	IBI_status	IBI_score	QHEI_status	QHEI_score
95	Not Sampled	NA	Not Sampled	NA
96	Not Sampled	NA	Not Sampled	NA
97	Not Sampled	NA	Not Sampled	NA
98	Not Sampled	NA	Not Sampled	NA
99	Not Impaired	48	Not Impaired	78
100	Not Sampled	NA	Not Sampled	NA

```
>
```

The sample of streams in Indiana is displayed in Figure 1. The sample sites for each Strahler order are displayed using a unique color. First, the levels function is used to extract the set of unique Strahler order values, and the result is assigned to object strahler. Next, the rainbow function is called to select a set of four colors, and the result is assigned to object cols. The plot function is then used to produce the basic figure, but plotting of sample points is suppressed. The for function is used to loop through the set of four unique Strahler order values and plot the color-coded points for each Strahler order using the points function. Finally, the legend function is used to add a legend to the figure, and the title function is used to create a figure title.

```

> strahler <- levels(IN_streams$Strahler_order)
> cols <- rainbow(4)
> plot(IN_streams$xcoord, IN_streams$ycoord, type="n", xlab="x-coordinate",
+       ylab="y-coordinate")

```

```

> for(i in 1:4) {
+   ind <- IN_streams$Strahler_order == strahler[i]
+   points(IN_streams$xcoord[ind], IN_streams$ycoord[ind], pch=20, col=cols[i])
+ }
> legend(x="topright", inset=0.05, legend=paste("Order", strahler), pch=20,
+       cex=1, col=cols)
> title("Plot of Indiana Stream Sites Color-Coded by Strahler Order")

```

3 Analysis of site status evaluation variables

The first analysis that will be examined is calculation of extent estimates for site status evaluation variables. Extent is measured both by the proportion of the resource in status evaluation categories and by size of the resource in each category. For a linear resource like streams, size refers to the length of streams in a category. For calculating extent estimates (and for all of the analyses we will consider), the survey design weights are incorporated into the calculation process. Two site status variables will be examined: (1) status, which classifies streams into seven evaluation categories and (2) TNT, which classifies streams as either "Target" or "NonTarget". The table and addmargins functions are used to create tables displaying the count for each code (level) of the two status variables.

```
> addmargins(table(IN_streams$status))
```

A table displaying the number of values for each level of the status variable follows:

Chemistry Only	Landowner Denial	NonTarget	Physical Barrier
14	19	9	7
Sampled Target	Not Sampled	Unknown	Sum
48	2	1	100

```
> addmargins(table(IN_streams$TNT))
```

A table displaying the number of values for each level of the TNT variable follows:

NonTarget	Target	Sum
10	90	100

The cat.analysis function in the spsurvey package will be used to calculate extent estimates. Four data frames constitute the primary input to the cat.analysis function. The first column

Plot of Indiana Stream Sites Color-Coded by Strahler Order

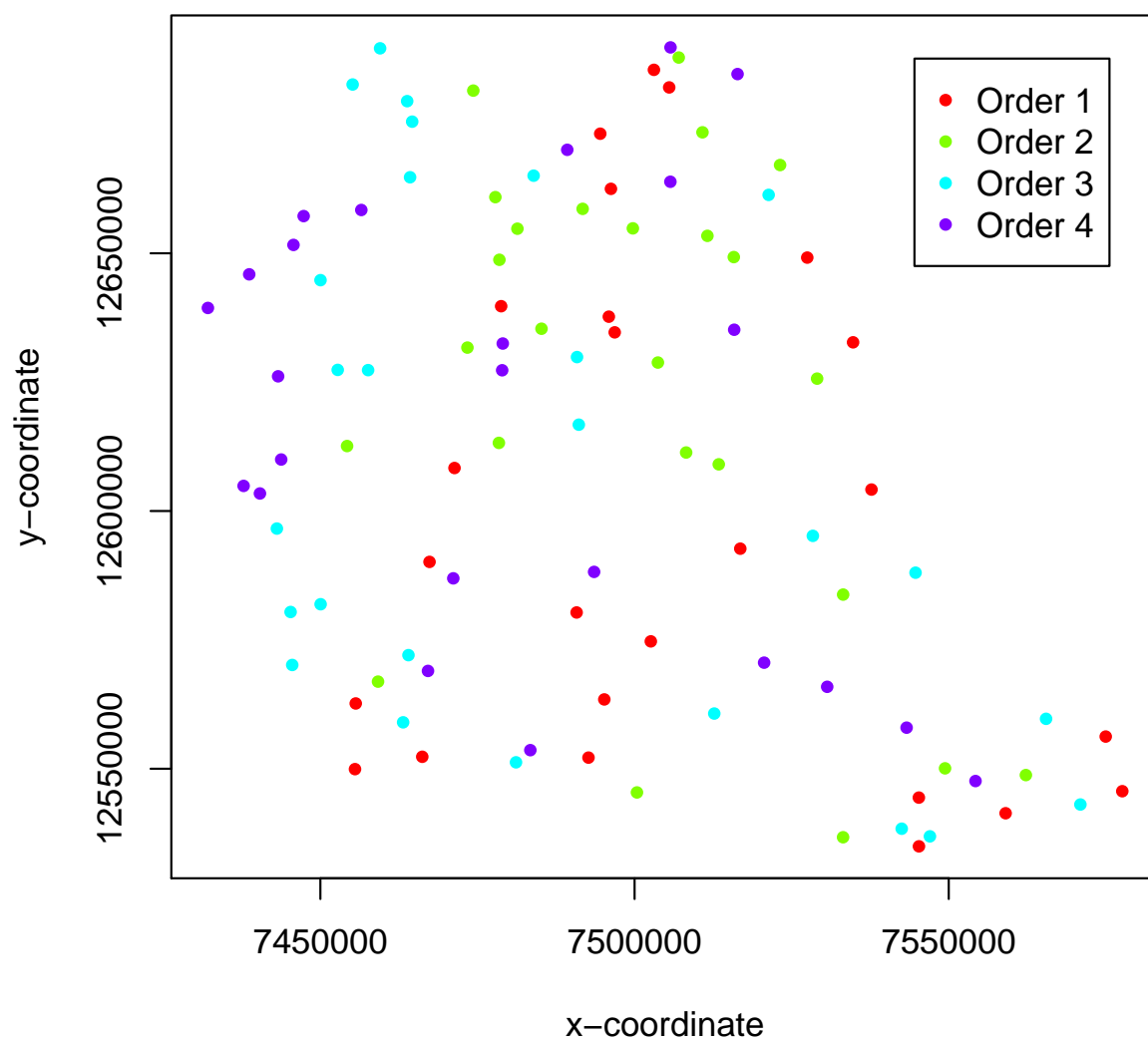


Figure 1: Indiana Stream Sample Sites.

(variable) in the four data frames provides the unique identifier (site ID) for each sample site and is used to connect records among the data frames. The siteID variable in the IN_streams data frame is assigned to the siteID variable in the data frames. The four data frames that will be created are named as follows: sites, subpop, design, and data.cat. The sites data frame identifies sites to use in the analysis and contains two variables: (1) siteID - site ID values and (2) Use - a logical vector indicating which sites to use in the analysis. The rep (repeat) function is used to assign the value TRUE to each element of the Use variable. Recall that nr is an object containing the number of rows in the IN_streams data frame. The subpop data frame defines populations and, optionally, subpopulations for which estimates are desired. Unlike the sites and design data frames, the subpop data frame can contain an arbitrary number of columns. The first variable in the subpop data frame identifies site ID values and each subsequent variable identifies a type of population, where the variable name is used to identify type. A type variable identifies each site with a character value. If the number of unique values for a type variable is greater than one, then the set of values represent subpopulations of that type. When a type variable consists of a single unique value, then the type does not contain subpopulations. For this analysis, the subpop data frame contains three variables: (1) siteID - site ID values, (2) Upper_Wabash - which will be used to calculate estimates for all of the sample sites combined, and (3) Strahler_Order - which will be used to calculate estimates for each Strahler order individually. The Strahler_order variable in the IN_streams data frame is assigned to the Strahler_Order variable in the subpop data frame. The design data frame consists of survey design variables. For the analysis under consideration, the design data frame contains the following variables: (1) siteID - site ID values; (2) wgt - final, adjusted, survey design weights; (3) xcoord - x-coordinates for location; and (4) ycoord - y-coordinates for location. The wgt, xcoord, and ycoord variables in the design data frame are assigned values using variables with the same names in the IN_streams data frame. Like the subpop data frame, the data.cat data frame can contain an arbitrary number of columns. The first variable in the data.cat data frame identifies site ID values and each subsequent variable identifies a response variable. The two response variables are Status and Target_NonTarget, which are assigned the status and TNT variables, respectively, in the IN_streams data frame. Missing data (NA) is allowed for the response variables, which are the only variables in the input data frames for which NA values are allowed.

Create the sites data frame.

```
> sites <- data.frame(siteID=IN_streams$siteID,
+                      Use=rep(TRUE, nr))
```

Create the subpop data frame.

```
> subpop <- data.frame(siteID=IN_streams$siteID,
+                      Upper_Wabash=rep("Upper Wabash", nr),
+                      Strahler_Order=IN_streams$Strahler_order)
```

Create the design data frame.

```
> design <- data.frame(siteID=IN_streams$siteID,
+                       wgt=IN_streams$wgt,
+                       xcoord=IN_streams$xcoord,
+                       ycoord=IN_streams$ycoord)
```

Create the data.cat data frame.

```
> data.cat <- data.frame(siteID=IN_streams$siteID,
+                         Status=IN_streams$status,
+                         Target_NonTarget=IN_streams$TNT)
```

Use the cat.analysis function to calculate extent estimates for the site status evaluation variables.

```
> # Calculate extent estimates for the site status evaluation variables
> Extent_Estimates <- cat.analysis(sites, subpop, design, data.cat)
>
```

The extent estimates for all basins combined are displayed using the print function. The object produced by cat.analysis is a data frame containing thirteen columns. The first five columns identify the population (Type), subpopulation (Subpopulation), response variable (Indicator), levels of the response variable (Category), and number of values in a category (NResp). A category labeled "Total" is included for each combination of population, subpopulation, and response variable. The next four columns in the data frame provide results for the proportion estimates: the proportion estimate (Estimate.P), standard error of the estimate (StdError.P), lower confidence bound (LCB95Pct.P), and upper confidence bound (UCB95Pct.P). Argument conf for cat.analysis allows control of the confidence bound level. The default value for conf is 95, hence the column names for confidence bounds contain the value 95. Supplying a different value to the conf argument will be reflected in the confidence bound names. Confidence bounds are obtained using the standard error and the Normal distribution multiplier corresponding to the confidence level. The final four columns in the data frame provide results for the size (units) estimates: the units estimate (Estimate.U), standard error of the estimate (StdError.U), lower confidence bound (LCB95Pct.U), and upper confidence bound (UCB95Pct.U).

```
> # Print the extent estimates for all basins combined
> print(Extent_Estimates[c(1:8, 32:34),])
```

	Type	Subpopulation	Indicator	Category	NResp
1	Upper_Wabash	Upper Wabash	Status	Chemistry Only	14
2	Upper_Wabash	Upper Wabash	Status	Landowner Denial	19
3	Upper_Wabash	Upper Wabash	Status	NonTarget	9
4	Upper_Wabash	Upper Wabash	Status	Physical Barrier	7

5	Upper_Wabash	Upper Wabash	Status	Sampled	48
6	Upper_Wabash	Upper Wabash	Status Target Not	Sampled	2
7	Upper_Wabash	Upper Wabash	Status	Unknown	1
8	Upper_Wabash	Upper Wabash	Status	Total	100
32	Upper_Wabash	Upper Wabash Target_NonTarget		NonTarget	10
33	Upper_Wabash	Upper Wabash Target_NonTarget		Target	90
34	Upper_Wabash	Upper Wabash Target_NonTarget		Total	100

	Estimate.P	StdError.P	LCB95Pct.P	UCB95Pct.P	Estimate.U	StdError.U	LCB95Pct.U
1	6.560	1.664	3.299	9.821	482.7	110.1	266.8
2	17.877	3.759	10.510	25.243	1315.4	286.9	753.1
3	22.078	5.028	12.222	31.933	1624.5	422.9	795.6
4	5.543	2.375	0.888	10.199	407.9	176.9	61.1
5	46.441	5.054	36.536	56.345	3417.2	432.5	2569.6
6	1.143	0.745	0.000	2.603	84.1	54.2	0.0
7	0.359	0.295	0.000	0.937	26.4	21.6	0.0
8	100.000	0.000	100.000	100.000	7358.2	539.5	6300.7
32	22.436	5.029	12.580	32.292	1650.9	423.5	820.9
33	77.564	5.029	67.708	87.420	5707.3	464.9	4796.1
34	100.000	0.000	100.000	100.000	7358.2	539.5	6300.7

	UCB95Pct.U
1	698.5
2	1877.7
3	2453.4
4	754.6
5	4264.8
6	190.4
7	68.8
8	8415.6
32	2480.9
33	6618.4
34	8415.6

>

The write.table function is used to store the extent estimates as a comma-separated value (csv) file. Files in csv format can be read by programs such as Microsoft Excel.

```
> write.table(Extent_Estimates, file="Extent_Estimates.csv", sep=",",
+             row.names=FALSE)
```

4 Analysis of stream condition variables

The second analysis that will be examined is estimating resource proportion and size for stream condition variables. Two stream condition variables will be examined: (1) IBI_Status,

which classifies streams by IBI (index of biotic integrity) status categories and (2) QHEI_Status, which classifies streams by QHEI (qualitative habitat evaluation index) status categories. The table and addmargins functions are used to create tables displaying the count for each level of the two stream condition variables.

```
> addmargins(table(IN_streams$IBI_status))
```

A table displaying the number of values for each level of the IBI status variable follows:

Impaired	Not Impaired	Not Sampled	Sum
12	36	52	100

```
> addmargins(table(IN_streams$QHEI_status))
```

A table displaying the number of values for each level of the QHEI status variable follows:

Impaired	Not Impaired	Not Sampled	Sum
14	34	52	100

As for extent estimates, the cat.analysis function will be used to calculate condition estimates. The sites data frame for this analysis differs from the one used to calculate extent estimates. The Use logical variable in sites is set equal to the value "Sampled", so that only sampled sites are used in the analysis. The subpop and design data frames created in the prior analysis can be reused for this analysis. The data.cat data frame contains the two stream condition variables: IBL_Status and QHEI_Status. Variables IBL_status and QHEI_status in the IN_streams data frame are assigned to IBL_Status and QHEI_Status, respectively.

Create the sites data frame.

```
> sites <- data.frame(siteID=IN_streams$siteID,
+                     Use=IN_streams$status == "Sampled")
```

Create the data.cat data frame.

```
> data.cat <- data.frame(siteID=IN_streams$siteID,
+                        IBI_Status=IN_streams$IBI_status,
+                        QHEI_Status=IN_streams$QHEI_status)
```

Use the cat.analysis function to calculate estimates for the stream condition variables.

```
> # Calculate estimates for the categorical variables
> Condition_Estimates <- cat.analysis(sites, subpop, design, data.cat)
>
```

Print the stream condition estimates for all sites combined.

```
> # Print the condition estimates for all basins combined
> print(Condition_Estimates[c(1:3, 16:19),])
```

	Type	Subpopulation	Indicator	Category	NResp	Estimate.P	
1	Upper_Wabash	Upper Wabash	IBI_Status	Impaired	12	27.7	
2	Upper_Wabash	Upper Wabash	IBI_Status	Not Impaired	36	72.3	
3	Upper_Wabash	Upper Wabash	IBI_Status	Total	48	100.0	
16	Upper_Wabash	Upper Wabash	QHEI_Status	Impaired	14	40.9	
17	Upper_Wabash	Upper Wabash	QHEI_Status	Not Impaired	34	59.1	
18	Upper_Wabash	Upper Wabash	QHEI_Status	Total	48	100.0	
19	Strahler_Order	1	QHEI_Status	Impaired	6	54.5	
	StdError.P	LCB95Pct.P	UCB95Pct.P	Estimate.U	StdError.U	LCB95Pct.U	UCB95Pct.U
1	6.61	14.7	40.6	945	247	460	1430
2	6.61	59.4	85.3	2472	346	1794	3150
3	0.00	100.0	100.0	3417	363	2706	4128
16	8.37	24.5	57.3	1398	358	697	2099
17	8.37	42.7	75.5	2019	305	1422	2616
18	0.00	100.0	100.0	3417	363	2706	4128
19	14.59	26.0	83.1	1083	290	515	1651

```
>
```

Use the write.table function to write the condition estimates as a csv file.

```
> write.table(Condition_Estimates, file="Condition_Estimates.csv", sep="," ,
+             row.names=FALSE)
```

5 Analysis of stream condition variables correcting for population size

The frame is a data structure containing spatial location data in addition to other attributes regarding a resource of interest and is used to create a survey design. A frame often takes the form of a shapefile. The frame can be used to obtain size values (e.g., length of streams) for the populations and subpopulations examined in an analysis. Examination of the Estimates.U column in the Condition_Estimates data frame produced by cat.analysis reveals

that the estimated Total value for both condition variables and each combination of population value and subpopulation value does not sum to the corresponding frame size value. For example, the Total entry in the Estimate.U column for the IBI_status variable, population "Upper_Wabash" and subpopulation "Upper Wabash" is 3,417 kilometers(rounded to a whole number). The corresponding frame size value is 7,358 kilometers. The popsize (population size) argument to cat.analysis provides a mechanism for forcing the Total category to equal a desired value. First, the c (combine) function is used to create a named vector of frame size values for each basin. Output from the c function is assigned to an object named framesize. The popsize argument is a list, which is a particular type of R object. The popsize list must include an entry for each population type included in the subpop data frame, i.e., Upper_Wabash and Strahler_Order for this analysis. The sum function applied to framesize is assigned to the Upper_Wabash entry in the popsize list. Recall that the Strahler order population type contains subpopulations, i.e., Strahler order categories. When a population type contains subpopulations, the entry in the popsize list also is a list. The as.list function is applied to framesize, and the result is assigned to the Strahler_Order entry in the popsize list.

Assign frame size values.

```
> framesize <- c("1"=4514.450, "2"=1443.260, "3"=740.146, "4"=660.294)
```

Use the cat.analysis function to calculate estimates for the stream condition variables.

```
> Condition_Estimates_popsiz <- cat.analysis(sites, subpop, design, data.cat,
+   popsize=list(Upper_Wabash=sum(framesize),
+   Strahler_Order=as.list(framesize)))
```

Print the stream condition estimates for all sites combined.

```
> # Print the stream condition estimates for all sites combined
> print(Condition_Estimates_popsiz[c(1:3, 16:19),])
```

	Type	Subpopulation	Indicator	Category	NResp	Estimate.P	
1	Upper_Wabash	Upper Wabash	IBI_Status	Impaired	12	27.7	
2	Upper_Wabash	Upper Wabash	IBI_Status	Not Impaired	36	72.3	
3	Upper_Wabash	Upper Wabash	IBI_Status	Total	48	100.0	
16	Upper_Wabash	Upper Wabash	QHEI_Status	Impaired	14	40.9	
17	Upper_Wabash	Upper Wabash	QHEI_Status	Not Impaired	34	59.1	
18	Upper_Wabash	Upper Wabash	QHEI_Status	Total	48	100.0	
19	Strahler_Order	1	QHEI_Status	Impaired	6	54.5	
	StdError.P	LCB95Pct.P	UCB95Pct.P	Estimate.U	StdError.U	LCB95Pct.U	UCB95Pct.U
1	6.61	14.7	40.6	2035	486	1082	2989
2	6.61	59.4	85.3	5323	486	4369	6276

3	NA	NA	NA	7358	NA	NA	NA
16	8.37	24.5	57.3	3010	616	1803	4216
17	8.37	42.7	75.5	4349	616	3142	5555
18	NA	NA	NA	7358	NA	NA	NA
19	14.59	26.0	83.1	2462	658	1172	3753

>

Use the write.table function to write the condition estimates as a csv file.

```
> write.table(Condition_Estimates_popsizes, file="Condition_Estimates_popsizes.csv",
+             sep=";", row.names=FALSE)
```

6 Analysis of quantitative variables

The third analysis that will be examined is estimating the CDF and percentiles for quantitative variables. Two quantitative variables will be examined: (1) IBI_score - IBI score and (2) QHEI_score - QHEI score. The summary function is used to summarize the data structure of the two quantitative variables.

```
> summary(IN_streams$IBI_score)
```

Summarize the data structure of the IBI score variable:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0.0	31.5	36.0	36.1	42.0	54.0	52.0

```
> summary(IN_streams$QHEI_score)
```

Summarize the data structure of the QHEI score variable:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
25.0	47.8	60.0	59.6	71.2	87.0	52.0

The cont.analysis function will be used to calculate estimates for quantitative variables. Input to the cont.analysis function is the same as input for the cat.analysis function except that the data frame containing response variables is named cont.data rather than cat.data. The sites, subpop, and design data frames created in the analysis of stream condition variables can be reused for this analysis. The data.cont data frame contains the two quantitative variables: IBI_Score and QHEI_Score, which contain the numeric scores for the IBI and QHEI variables,

respectively. Variables IBI_score and QHEI_score in the IN_streams data frame are assigned to IBI_Score and QHEI_Score, respectively. The popsize argument is included in the call to cont.analysis.

Create the data.cont data frame.

```
> data.cont <- data.frame(siteID=IN_streams$siteID,  
+                           IBI_Score=IN_streams$IBI_score,  
+                           QHEI_Score=IN_streams$QHEI_score)
```

Use the cont.analysis function to calculate CDF and percentile estimates for the quantitative variables.

```
> CDF_Estimates <- cont.analysis(sites, subpop, design, data.cont,  
+   popsize=list(Upper_Wabash=sum(framesize),  
+   Strahler_Order=as.list(framesize)))
```

The object produced by cont.analysis is a list containing two objects: (1) CDF, a data frame containing the CDF estimates and (2) Pct, a data frame containing percentile estimates plus estimates of population values for mean, variance, and standard deviation. Format for the CDF data frame is analogous to the data frame produced by cat.analysis. For the CDF data frame, however, the fourth column is labeled Value and contains the value at which the CDF was evaluated. Unlike the data frames produced by the other analysis functions we have examined, the Pct data frame contains only nine columns since there is a single set of estimates rather than two sets of estimates. In addition, the fourth column is labeled Statistic and identifies either a percentile or the mean, variance, or standard deviation. Finally, since percentile estimates are obtained by inverting the CDF estimate, the percentile estimates do not have a standard error value associated with them.

Use the write.table function to write the CDF estimates as a csv file.

```
> write.table(CDF_Estimates$CDF, file="CDF_Estimates.csv", sep="," ,  
+   row.names=FALSE)
```

The cont.cdfplot function in spsurvey can be used to produce a PDF file containing plots of the CDF estimates. The primary arguments to cont.cdfplot are a character string containing a name for the PDF file and the CDF data frame in the CDF_Estimates object.

Produce a PDF file containing plots of the CDF estimates.

```
> cont.cdfplot("CDF_Estimates.pdf", CDF_Estimates$CDF)  
>
```

Print the percentile estimates for IBI score for all sites combined.

```
> # Print the percentile estimates for IBI score for all sites combined
> print(CDF_Estimates$Pct[1:10,])
```

	Type	Subpopulation	Indicator	Statistic	NResp	Estimate
1	Upper_Wabash	Upper Wabash	IBI_Score	5Pct	1	0.0
2	Upper_Wabash	Upper Wabash	IBI_Score	10Pct	2	23.4
3	Upper_Wabash	Upper Wabash	IBI_Score	25Pct	8	28.7
4	Upper_Wabash	Upper Wabash	IBI_Score	50Pct	23	34.2
5	Upper_Wabash	Upper Wabash	IBI_Score	75Pct	31	39.6
6	Upper_Wabash	Upper Wabash	IBI_Score	90Pct	41	44.2
7	Upper_Wabash	Upper Wabash	IBI_Score	95Pct	44	48.9
21	Upper_Wabash	Upper Wabash	IBI_Score	Mean	48	34.2
31	Upper_Wabash	Upper Wabash	IBI_Score	Variance	48	112.1
41	Upper_Wabash	Upper Wabash	IBI_Score	Std. Deviation	48	10.6
		StdError	LCB95Pct	UCB95Pct		
1			0.00	24.6		
2			0.00	26.7		
3			24.22	32.2		
4			31.39	37.1		
5			35.91	43.9		
6			40.78	51.7		
7			41.67	54.0		
21	1.74391897533996		30.77	37.6		
31	45.0756162161629		23.78	200.5		
41	2.12837932435036		6.42	14.8		

```
>
```

Use the write.table function to write the percentile estimates as a csv file.

```
> write.table(CDF_Estimates$Pct, file="Percentile_Estimates.csv", sep="," ,
+             row.names=FALSE)
```

The cont.cdfctest function in spsurvey can be used to test for statistical difference between the CDFs from subpopulations. For this analysis we will test for statistical difference between the CDFs for the four Strahler order categories. The cont.cdfctest function will test all possible pairs of Strahler order categories. Arguments to cont.cdfctest are the same as arguments to cont.analysis. Since we are interested only in testing among Strahler order categories, the subpop data frame is subsetting to include only the siteID and Strahler_Order variables. Note that the popsize argument was modified from prior examples to include only the entry for Strahler_Order.

```
> CDF_Tests <- cont.cdfctest(sites, subpop[,c(1,3)], design, data.cont,
+                             popsize=list(Strahler_Order=as.list(framesize)))
```

During execution of the program, a warning message was generated. The warning message is stored in a data frame named 'warn.df'. Enter the following command to view the warning message: warnprnt()

The print function is used to display results for IBI score of the statistical tests for difference between CDFs for Strahler order categories. The object produced by cont.cdfctest is a data frame containing eight columns. The first column (Type) identifies the population. The second and third columns (Subpopulation_1 and Subpopulation_2) identify the subpopulations. The fourth column (Indicator) identifies the response variable. Column five contains values of the test statistic. Six test statistics are available, and the default statistic is an F-distribution version of the Wald statistic, which is identified in the data frame as "Wald-F". The default statistic is used in this analysis. For further information about the test statistics see the help file for the cdf.test function in spsurvey, which includes a reference for the test for differences in CDFs. Columns six and seven (Degrees_of_Freedom_1 and Degrees_of_Freedom_2) provide the numerator and denominator degrees of freedom for the Wald test. The final column (p_Value) provides the p-value for the test.

```
> # Print results of the statistical tests for difference between CDFs from
> # Strahler order categories for IBI score
> print(CDF_Tests[1:15,])
```

	Type	Subpopulation_1	Subpopulation_2	Indicator	Wald_F
1	Strahler_Order	1	2	IBI_Score	0.3519
2	Strahler_Order	1	3	IBI_Score	0.3138
3	Strahler_Order	1	4	IBI_Score	3.5350
4	Strahler_Order	2	3	IBI_Score	0.0651
5	Strahler_Order	2	4	IBI_Score	3.5644
6	Strahler_Order	3	4	IBI_Score	2.6747
7	Strahler_Order	1	2	QHEI_Score	0.9906
8	Strahler_Order	1	3	QHEI_Score	1.6332
9	Strahler_Order	1	4	QHEI_Score	5.6277
10	Strahler_Order	2	3	QHEI_Score	0.4057
11	Strahler_Order	2	4	QHEI_Score	3.5103
12	Strahler_Order	3	4	QHEI_Score	1.9688
NA	<NA>	<NA>	<NA>	<NA>	NA
NA.1	<NA>	<NA>	<NA>	<NA>	NA
NA.2	<NA>	<NA>	<NA>	<NA>	NA
	Degrees_of_Freedom_1	Degrees_of_Freedom_2	p_Value		
1	2	21	0.7074		
2	2	23	0.7337		
3	2	17	0.0520		
4	2	25	0.9371		
5	2	19	0.0485		
6	2	21	0.0923		

7	2	21	0.3881
8	2	23	0.2172
9	2	17	0.0133
10	2	25	0.6708
11	2	19	0.0504
12	2	21	0.1646
NA	NA	NA	NA
NA.1	NA	NA	NA
NA.2	NA	NA	NA

>

Use the `write.table` function to write CDF test results as a csv file.

```
> # Write CDF test results as a csv file
> write.table(CDF_Tests, file="CDF_Tests.csv", sep="," , row.names=FALSE)
>
```