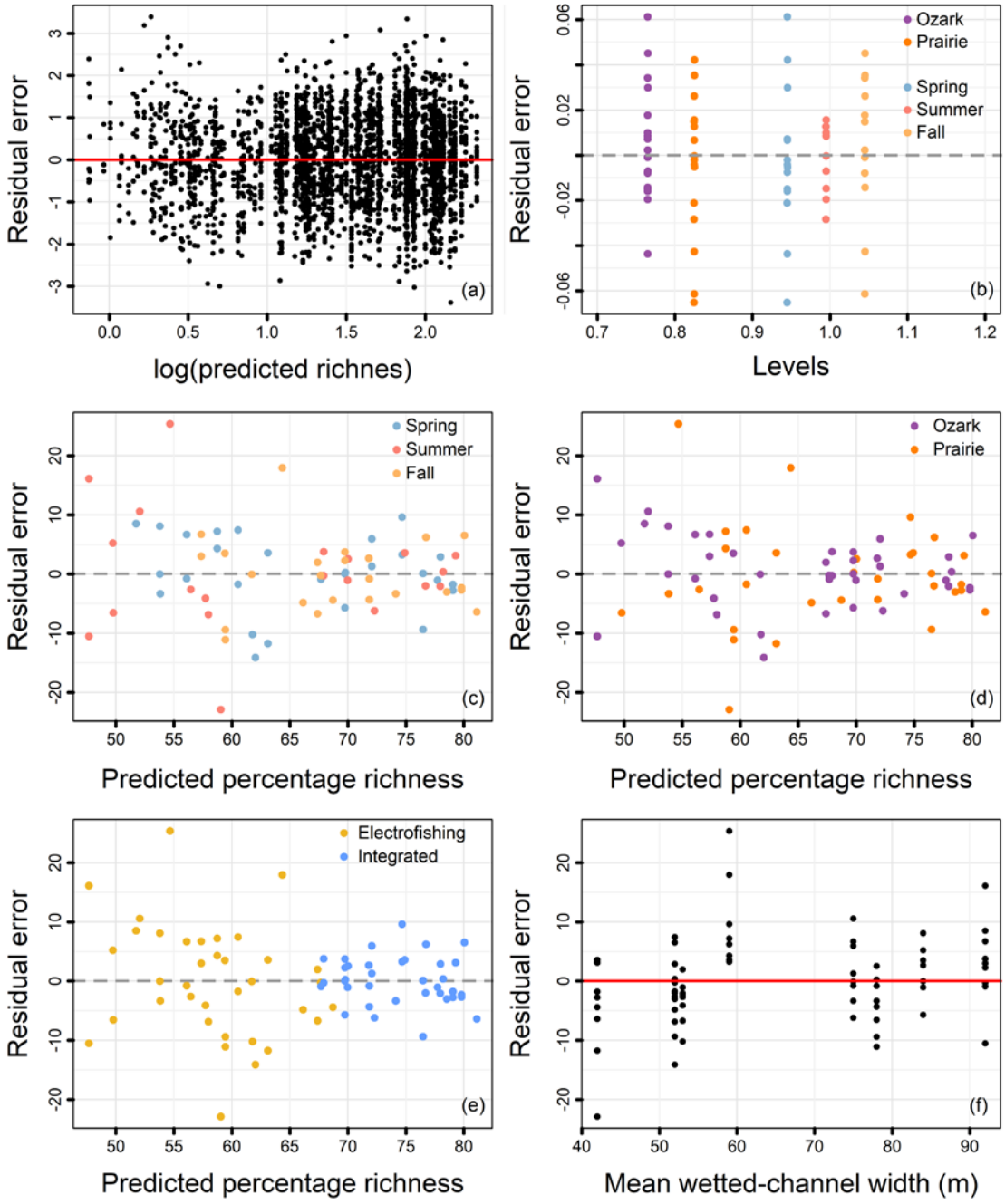
Supplementary material 1. Number of sub-samples with the full-effort and integrated-gear protocols in nine non-wadeable sites in Missouri (USA). A sub-sample by gear = 50 m electrofishing, 50 m trawling, one seine haul, one hoop net, one mini-fyke net, one trammel net. L = lower, U = upper, Lg = Large River, S = Small River, E = electrofishing, T = trawl, S = seine, H = hoop net, FN = mini-fyke net, ST = stationary trammel net.

		Full-effort protocol							Integrated-gear protocol						
River	Size	E	T	S	H	FN	ST	Total	E	T	S	H	FN	ST	Total
L. Gasconade	Lg	40	20	30	5	5	5	105	21	6	18	0	5	0	50
L. Grand	Lg	30	20	30	5	5	5	95	21	6	18	0	5	0	50
L. Meramec	Lg	30	20	30	5	5	5	95	21	6	18	0	5	0	50
U. Gasconade	Lg	30	20	30	5	5	5	95	21	6	18	0	5	0	50
Black	S	20	10	20	5	5	5	65	14	3	12	0	5	0	34
Lamine	S	20	10	20	5	5	5	65	14	3	12	0	5	0	34
Salt	S	20	10	20	5	5	5	65	14	3	12	0	5	0	34
U. Grand	S	20	10	20	5	5	5	65	14	3	12	0	5	0	34
U. Meramec	S	20	10	20	5	5	5	65	14	3	12	0	5	0	34

Note: Large and Small rivers were distinguished by mean wetted-channel widths ≥65 m and <65 m, respectively.

Supplementary material 2. Residual plots for models of (a) fish richness in sub-sample in 36 samples in Missouri (USA), (b) percentage richness detected by the integrated-gear protocol based on region (Ozark, Prairie) and season (spring, summer, fall), (c–f) percentage richness detected with 20 sub-samples by seasons, regions, protocol (integrated gear, electrofishing only), and mean wetted-channel width (MWCW).



Supplementary material 3. Transferable code with instructions for performing a multi-gear resampling procedure for assessing fish richness in the Black River, Missouri (USA) from summer 2015.

Introduction

Below is R code for running a multi-gear re-sampling procedure used in Dunn and Paukert *accepted*. This procedure provides naive estimates of species richness that would be detected by each combination of gear and effort nested within multi-gear survey designs. This code is accompanied by fish data from one of the most biologically diverse rivers in North America.

This code has two main parts:

1) Preparation of two matrices (Community, Indexing) that work together to efficiently resample data.
2) A nested `for` loop that resamples data (the sampler).

Setup: Install dependencies and obtain data

Dependencies

First, install dependencies. The code mainly uses base R.

```
install.packages("tidyr")
install.packages("dplyr")
install.packages("knitr")
install.packages("RCurl")
install.packages("foreign")
```

```
packages <- c("tidyr", "dplyr", "magrittr", "knitr", "RCurl", "foreign")
lapply(packages, require, character.only = T) # Output is false if package not installed
```

The data

Next, load data from the following link:

```
### Recreate url link
urlRemote_W  <- "https://raw.githubusercontent.com/"
pathGithub_W <- "Midsized-Rivers/Multigear_Resampler/master/Data/"
fileName_W   <- "BlackR_SU2015_W.csv"
url_W <- paste0(urlRemote_W, pathGithub_W, fileName_W)
Dat_W <- read.csv(textConnection(getURL(url_W)), header = T, stringsAsFactors = F)
```

This a "community" wide-style data frame with sub-samples as rows and the following columns:

- Column 1: Abbreviations for one of six gears
- Column 2: Section where a sub-sample occurred
- Column 3: Unique ID for each sub-sample
- Columns 4+: Each column has abundances for a given species (column name = [GEN]us[SPE]ies)

These data are from summer 2015 in the Black River, Missouri (USA) and were collected by CG Dunn. The survey used six gears and 65 sub-samples to detect 69 fish species! For example, one APLGRU (Freshwater Drum) was caught in [EL]ectrofishing Sub-sample 1 in Section 1. Table only shows first 5 rows and species.

Gear	Section	Subsample_ID	AMBARI	AMENAT	AMMVIV	APHSAY	APLGRU
EL	1	EL1_1	0	0	0	0	1
EL	1	EL1_2	0	0	0	0	0
EL	2	EL2_1	0	0	0	0	0
EL	2	EL2_2	0	0	0	0	0
EL	3	EL3_1	0	0	0	0	0

User-specified parameters

Below are parameters that need to be customized for different datasets and survey designs. The default for the Black River uses six gears: electrofishing (EL), trawling (TR), seining (SE), mini-fyke nets (FN), stationary trammel nets (TN), and hoop nets (HN). The site was divided into 10 equal sections (see main text Fig. 2). Sub-samples were taken in each section for EL (2 sub-samples per section), TR (1), and SE (2). FN, TN, and HN each had 1 net in five of the 10 sections.

```
##### Parameters: order of Gears, Sections, Subsamples need to align
Gears       <- c("EL", "TR", "SE", "FN", "TN", "HN")   # Gear names/abbreviations
Sections    <- c(10, 10, 10, 5, 5, 5)                  # Total sections with each gear
Subsamples  <- c(2, 1, 2, 1, 1, 1)                     # Sub-samples per section

### Derived structures from parameters
Gear_Matrix <- data.frame(Gears, Sections, Subsamples) # Stores info about survey design
Protocols_N <- prod(Gear_Matrix[, "Sections"] + 1)     # Number of sub-protocols
```

Fill in the exact label in quotes given to sub-samples where no fish were detected.

```
Nofish          <- "NO_NO_"  # These are labeled as "NO_NO_" in the Black R dataset
Dat_W[, Nofish] <- NULL    # Remove "Nofish" column so it isn't counted as a species
```

Part 1: Create Community and Indexing matrices

Community Matrix

Streamline the Community data frame storing fish data by converting it to a matrix and all data within to integers. Integers and matrices require far less memory than a data frame with character strings and numerics, meaning code runs faster. First, each species is given a unique integer ID, which is used as an indicator signaling a given species was caught within a given sub-sample. Replacing presences (1's) with column integers is not essential, but it can be useful for calculating other metrics requiring species identification (e.g., richness by traits). Note that this bit starts at Column 4 because that is where fish data begin in the Community data frame.

```
##### Streamlined Community Matrix
Dat_W[, 4:ncol(Dat_W)] <- (Dat_W[, 4:length(Dat_W)] > 0) * 1           # Convert to P/A
colnames(Dat_W)[4:length(Dat_W)] <- 1:length(Dat_W[, 4:length(Dat_W)]) # Rename columns
Species <- length(colnames(Dat_W)[4:ncol(Dat_W)])                      # Total species

### Replace presences with unique integer for each species
Caught <- which(Dat_W[4:length(Dat_W)] == 1, arr.ind = T)     # Sub-samples where caught
Dat_W[, 4:length(Dat_W)][Caught] <-
  as.numeric(colnames(Dat_W[, 4:length(Dat_W)])[Caught[, 2]])
```

Total number of species in this dataset.

```
paste("Total species = ", Species, sep = "")
```

```
## [1] "Total species = 69"
```

Next, convert gear and sub-sample values to unique integers.

```
for(i in 1:length(Gears)){                              # Replace gears with integer
  Condition <- (Dat_W[, "Gear"] == Gears[i])
  Dat_W[Condition, "Gear"] <- i
}
Comm_Matrix <- Dat_W[order(Dat_W[, "Gear"]),]           # Important to reorder by gear!!
```

```
Comm_Matrix$Subsample_ID <- 1:nrow(Comm_Matrix)       # Replace sub-samples with integers
Comm_Matrix <- apply(data.matrix(Comm_Matrix), c(1, 2), as.integer) # All integer matrix
```

The Freshwater Drum occurrence from above now reads as "5", corresponding to its integer column name.

```
  Comm_Matrix[1:5, 1:8]                          # Displays subset of rows & cols
```

```
##   Gear Section Subsample_ID 1 2 3 4 5
## 1    1       1            1 0 0 0 0 5
## 2    1       1            2 0 0 0 0 0
## 3    1       2            3 0 0 0 0 0
## 4    1       2            4 0 0 0 0 0
## 5    1       3            5 0 0 0 0 0
```

Indexing Matrix

Next, create the Indexing Matrix. The Indexing Matrix tells the sampler the number of sections-worth of sub-samples to select according to each sub-protocol. Consequently, dimensions of the Indexing Matrix are Sub-protocols (rows) x Gears (cols). This should automatically generate based on parameters previously specified in the Gear Matrix.

```
##### Organize protocols in the Indexing Matrix
### Store Sub-sample gear ranges from Community Matrix
GR <- aggregate(Comm_Matrix[,"Subsample_ID"], by = list(Comm_Matrix[,"Gear"]),
                FUN = range, simplify = T)[, 2]

### parameters needed to organize Indexing Matrix
Ind_P <- data.frame("Times" = rep(NA, times = nrow(Gear_Matrix)),
                    "Each"  = rep(NA, times = nrow(Gear_Matrix)))
Ind_P[1, ] <- c(Protocols_N/(Gear_Matrix[1, "Sections"] + 1), 1) # Initial row
for(i in 2:nrow(Gear_Matrix)){                                  # Subsequent rows
  Ind_P[i, "Times"] <- Ind_P[i-1, "Times"] / (Gear_Matrix[i, "Sections"] + 1)
  Ind_P[i, "Each"] <- Protocols_N / (Ind_P[i, "Times"]*(Gear_Matrix[i,"Sections"] + 1))
}

### Create Indexing Matrix based on parameters
Index_Matrix <- matrix(NA, nrow = Protocols_N, ncol = nrow(Gear_Matrix))
for(i in 1:nrow(Ind_P)){
  Index_Matrix[, i] <- rep(0:Gear_Matrix[i,"Sections"],
                           times = Ind_P[i, "Times"],
                           each = Ind_P[i, "Each"])
}
```

Here are some checks to make sure the Indexing Matrix is correctly configured. The first identifies redundant, identical protocols (this should be 0). The second makes sure rows of the Indexing Matrix equal total sub-protocols.

```
### Checks for correct Indexing Matrix: Check 1) = 0, Check 2) = Protocols_N
sum(duplicated(tidyr::unite(data.frame(Index_Matrix),          # Check 1
                            sep = "_", "Code")))
```

```
## [1] 0
```

```
length(unique(unlist(tidyr::unite(data.frame(Index_Matrix),    # Check 2
                                  sep ="_", "Code"))))
```

```
## [1] 287496
```

Look at the Indexing Matrix; the first row is a "no effort" sub-protocol.

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    0    0    0    0    0    0
## [2,]    1    0    0    0    0    0
```

The last row is the full survey design (i.e., max effort).

```
##           [,1] [,2] [,3] [,4] [,5] [,6]
## [287495,]    9   10   10    5    5    5
## [287496,]   10   10   10    5    5    5
```

Intermediate rows are intermediate combinations of gear and effort (sub-protocols 100,000-100,002).

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    9    4    1    3    0    2
## [2,]   10    4    1    3    0    2
## [3,]    0    5    1    3    0    2
```

Part 2: Configure the sampler

Parameters for the sampler

First, specify the number of randomizations (bootstraps) per sub-protocol. For the Black River, 1 randomization for all sub-protocols (287,496) takes roughly 23 seconds, so 1,000 bootstraps will take roughly 6 hours. Smaller datasets and simpler designs require less run time. The default is 2 randomizations.

```
##### Prepare for lift off
### Parameters for bootstrapping
Bootstraps <- 2                 # Number of randomization for each protocol
Protocols  <- Protocols_N       # Total number of protocols
Cols       <- ncol(Comm_Matrix) # Restricts columns to search for species

### Make sure code is integer-matrix format
attributes(Comm_Matrix)[2:3] <- NULL                                  # Strip names
Subsamples <- as.matrix(as.integer(Subsamples))                       # Convert to integer
sapply(list(GR, Comm_Matrix, Index_Matrix, Subsamples), is.integer) # Integer check
```

```
## [1] TRUE TRUE TRUE TRUE
```

```
sapply(list(GR, Comm_Matrix, Index_Matrix, Subsamples), is.matrix)  # Matrix check
```

```
## [1] TRUE TRUE TRUE TRUE
```

Begin the sampler

The outer loop (rows) controls different sub-protocols, while the inner loop (columns) replicates the sampler.

```
x <- matrix(as.integer(1:length(Gears)), ncol = 1)
mg_samplerF1 <- function(x){sample(GR[x, 1]:GR[x, 2],
                                   Index_Matrix[i, x]*Subsamples[x], FALSE)}
mg_samplerF2 <- function(i){unlist(lapply(x, FUN = mg_samplerF1), use.names = F)}

Richness <- matrix(NA, nrow = Protocols, ncol = Bootstraps, dimnames = NULL)
system.time(for(i in 1:Protocols){
  for(j in 1:Bootstraps){
     Richness[i, j] <- sum(colSums(Comm_Matrix[mg_samplerF2(i), 4:Cols, drop = F]) > 0)
```

```
  }
})
```

```
##    user  system elapsed
##    67.7     0.1    68.4
```

Inspect results

Summarize results across bootstraps. Because the Indexing Matrix aligns with the output matrix, we can join these together. This is configured for the Black River.

```
### Combine Index Matrix with re-sampler output
Results <- data.frame("Gear1" = Index_Matrix[, 1],
                      "Gear2" = Index_Matrix[, 2],
                      "Gear3" = Index_Matrix[, 3],
                      "Gear4" = Index_Matrix[, 4],
                      "Gear5" = Index_Matrix[, 5],
                      "Gear6" = Index_Matrix[, 6],
                      "Mean" = apply(Richness, MARGIN = 1, mean),
                      "SD" = apply(Richness, MARGIN = 1, sd))
Results$Perc_Rich <- (Results$Mean / Species)*100 # Percentage of total species
```

Finally, quickly look up results for interesting sub-protocols. For example, a protocol solely using 1 km of electrofishing (10 sections-worth) on average only detected 37 of 69 (54%) of species.

```
### For example: 1 km of electrofishing
dplyr::filter(Results,
       Gear1 == 10, # Electrofishing (0-10 sections, 1 section = 100 m)
       Gear2 == 0,  # Trawling (0-10 sections, 1 section = 50 m)
       Gear3 == 0,  # Seining (0-10 sections, 1 section = 2 hauls)
       Gear4 == 0,  # Mini-fyke nets (0-5 sections, 1 section = 1 net)
       Gear5 == 0,  # Stationary trammel nets (0-5 sections, 1 section = 1 net)
       Gear6 == 0)  # Hoop nets (0-5 sections, 1 section = 1 net)
```

```
##   Gear1 Gear2 Gear3 Gear4 Gear5 Gear6 Mean SD Perc_Rich
## 1    10     0     0     0     0     0   37  0      53.6
```

Whereas, the equivalent effort (20 sub-samples) with an efficient multi-gear design detected roughly 50 species.

```
##   Gear1 Gear2 Gear3 Gear4 Gear5 Gear6 Mean   SD Perc_Rich
## 1     5     2     3     2     0     0 46.5 7.78      67.4
```

Literature cited

Dunn, C.G., and C.P. Paukert. 2020. A flexible survey design for monitoring spatiotemporal fish richness in nonwadeable rivers: optimizing efficiency by integrating gears. Canadian Journal of Fisheries and Aquatic Sciences. dx.doi.org/10.1139/cjfas-2019-0315

Supplementary material 4. Parameters estimates ($\hat{\beta}$) and standard errors (SE) for best-supported models in questions1, 4, 5. Question 1: which gears detected the most species? Question 4: did the most-efficient protocol detect a consistent percentage of species across regions and seasons? Question 5: was the integrated-gear protocol more effective and consistent than traditional effort with an electrofishing-only protocol?

Question (response)	Variable	log(β)	log(SE)
Question 1 (log[richness])	Intercept = E-fishing, Ozarks, Summer	1.69	0.09
	Prairie	-0.42	0.14
	Fall	0.42	0.08
	Spring	0.31	0.08
	Mini-fyke net	0.30	0.09
	Hoop net	-1.35	0.14
	Seine haul	0.35	0.05
	Stationary trammel net (STN)	-1.31	0.14
	Trawl run	-0.31	0.07
	Prairie, Fall	-0.20	0.12
	Prairie, Spring	-0.14	0.11
	Prairie, Mini-fyke net	0.28	0.09
	Prairie, Hoop net	0.83	0.14
	Prairie, Seine haul	-0.05	0.06
	Prairie, STN	0.67	0.13
	Prairie, Trawl run	0.07	0.08
	Fall, Mini-fyke net	-0.33	0.12
	Spring, Mini-fyke net	-0.48	0.12
	Fall, Hoop net	-0.68	0.19
	Spring, Hoop net	-0.26	0.17
	Fall, Seine haul	-0.42	0.07
	Spring, Seine haul	-0.36	0.07
	Fall, STN	-0.18	0.17
	Spring, STN	-0.25	0.17
	Fall, Trawl run	-0.30	0.09
	Spring, Trawl run	-0.35	0.09
	Variable	β	SE
Question 4 (percentage richness)	Intercept	90.01	0.47
	Variable	β	SE
Question 5 (percentage richness)	Intercept = E-fishing, Ozarks, Summer	71.37	5.90
	Integrated-gear (IG) protocol	20.24	3.01
	Spring	4.06	3.37
	Fall	9.67	3.47
	Prairie	-1.51	2.27
	Mean wetted-channel width (m)	-0.26	0.07
	IG protocol, Spring	-4.29	3.67
	IG protocol, Fall	-7.84	3.79