B. APPENDIX: R-PLOT GENERATION

To provide an aid in the Risk Characterization section, NMFS developed a plot (referred to as a 'Risk-plot' or 'R-plot') displaying the various sources of data (i.e. exposure, response, and use) available as part of the consultation (e.g. EPA's BEs and risk assessments and NMFS's analyses). The R-plots are generated using the R programming language:

R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

This Appendix consists of several sections with information on the R-plot process:

- **R-plot Process Overview:** An overview of the R-plot process
- **Example R-Plot:** An example of an R-plot
- **Directory and Files:** A list of the directories and file needed by the R-plot process
- **R code lines to modify:** A description of inputs to change in the R code
- **Toxicity Data File:** An example of a toxicity data file used by the R code
- **HUC-12 List File:** An example of a HUC-12 list that may be used by the R code
- **Main R Code:** The primary R code file used to generate R-plots
- **Import R Code:** R code used to import data prior to running the main R code
- **Functions R Code:** R code defining several functions used by the other R code

R-plot Process Overview

The following is a brief overview of the R-plot process. The overview assumes an understanding of the data and some knowledge of the R programming environment.

The data displayed on the R-plots comes from several sources. For the R-plot process, these sources appear as various files that the R code relies on for generating the plots. A list of the files is provided in the **Directory and Files** section. A summary of the sources is detailed here:

1) Toxicity information for a species gathered from the available literature. For sublethal endpoints, such as growth, this is typically a range of LOECs or EC25s across the available studies. For endpoints such as mortality, this is can be a range of percent mortalities using an LC50 and slope chosen based on a species sensitivity distribution. For the R-plot process this information is provided by a tab-delimited text file generated by the user (an example is provided in **Toxicity Data File**).
2) Data on the species range and critical habitat (e.g. a list of HUC-12s), the uses of the pesticide (e.g. Vegetable and Ground Fruit), and their overlap (by HUC-12).

This information is from GIS analyses provided to NMFS by EPA (EPA 2017a; 2017b; 2017c). For the R-plot process, this information is organized into various files. Files such as species_huc12.csv (in the bin directory of the MagTool) provide a list of HUC-12s by species. The various CDL files (e.g. CDL_L48_2010.csv in the bin directory) provide a list of uses and their overlaps for each of the HUC-12s.

3) Exposure estimates generated using existing exposure models for each crop and use category (e.g. lettuce crop within the Vegetable and Ground Fruit use category). For the aquatic EECs the Pesticide Water Calculator (PWC) was used to generate thirty years of EECs for each HUC-2 and aquatic bin. For the R-plot process, for a compound the thirty annual peak EECs for different averaging periods are provided by a file (e.g. prometryn_eec.csv). For the terrestrial EECs, AgDRIFT and Terr-Plant were used to generate EECs and the results combined into a single file (TerrestrialEECs.csv).

The process of generating an R-plot typically starts with selecting a chemical and species. The selection of species determines which HUc-12s are extracted from the data files. The selection of chemical determines the relevant EECs and uses for the list of HUC-12s. The list of HUC-12s determines which HUC-2s are needed from the EEC data.

The plot only displays a single EEC range for a specific crop and use category (e.g. lettuce as a Vegetable and Ground Fruit use). If the list of HUC-12s spans multiple HUC-2s an adjustment can be made that computes a weighted average of the multiple HUC-2 EECs available. This adjustment is based on the proportion of the total area represented by each HUC-2. The more area of HUC-12s that belong to a HUC-2, the more that the EECs for that HUC-2 will contribute to the EECs displayed on the plot.

The R code compiles these sources of information into a single plot. An example of an R-plot is shown in **Example R-Plot**. The plot consists of five parts.

1) The upper portion displays the information present in the selected toxicity data file. This consists of multiple rows of endpoints each with a set of labeled markers. The meaning of each marker is up to the user (e.g. a LOEC, percent morality, etc.). The markers are positioned along the concentration axis below.
2) The center of the plot displays all the EEC data associated with the selected chemical, HUCs and relevant scenarios. For each crop (e.g. lettuce) of a use category (e.g. Vegetables and Ground Fruit) there will be a point for each averaging period and each aquatic bin (for aquatic R-plots) or exposure model and application method (for terrestrial R-plots). For aquatic EECs, each point represents the median peak annual EEC for one averaging period for a specific PWC scenario. Error bars around the point indicate the 5% and 95%tile of the distribution of thirty years of data. The EEC data is positioned using the same concentration axis as the toxicity data to allow direct comparison of exposure and effects.
3) The left side of the plot (i.e. the left Y-axis labels) list the use categories associated with the species range (or list of HUC12s) in order of their area (largest

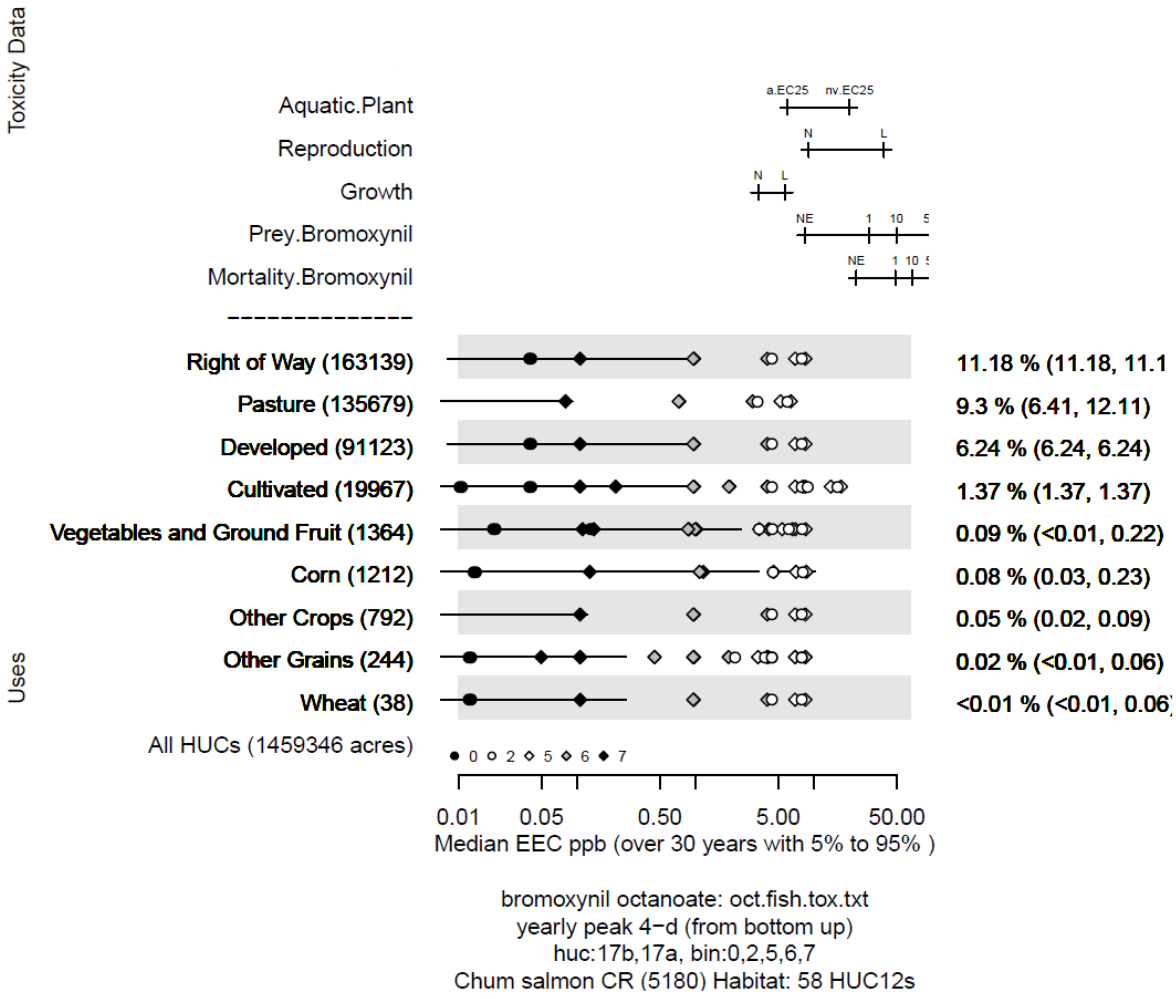
area at the top). The area of each use category is denoted in the parentheses.

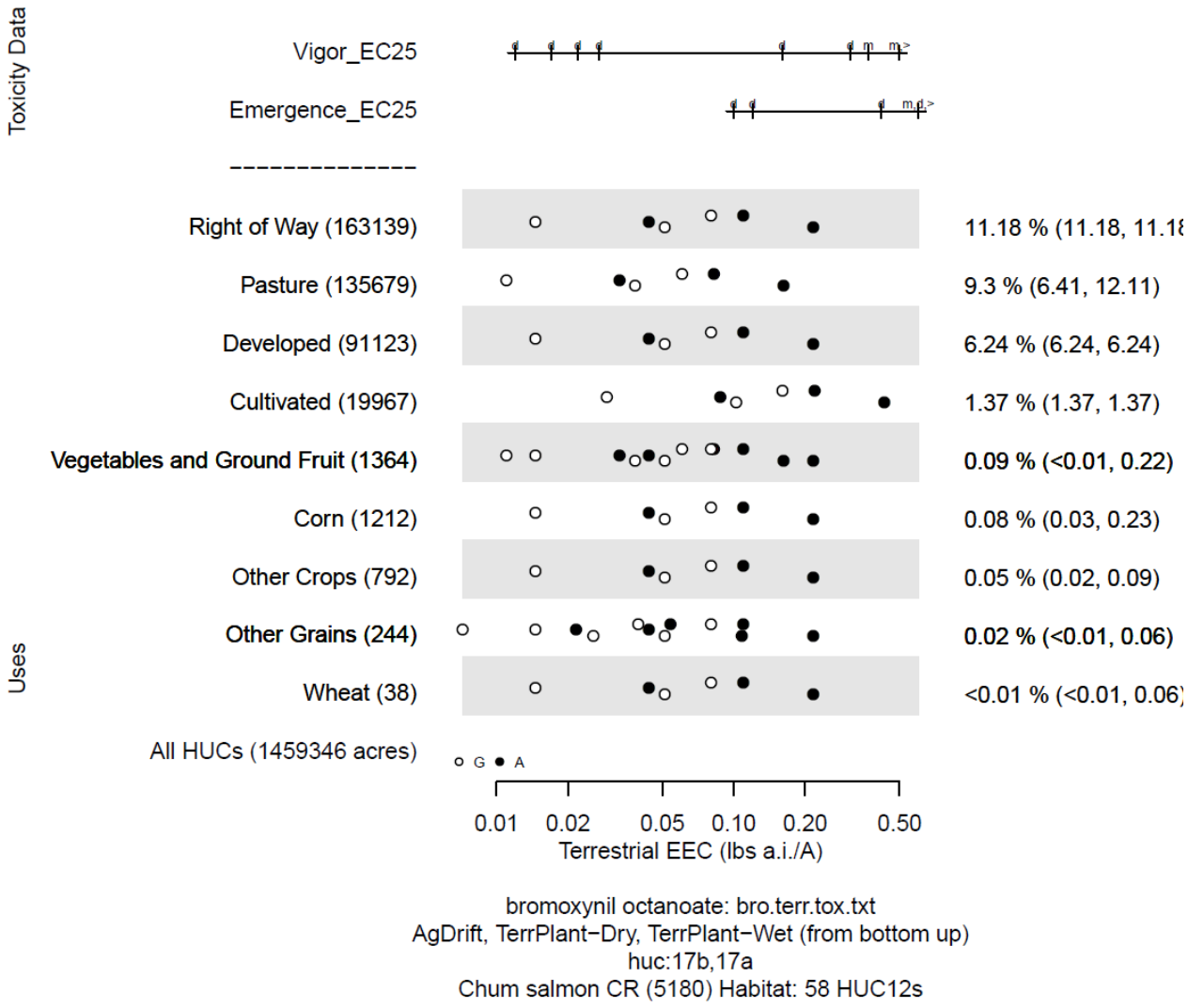
4) The right side of the plot (i.e. the right Y-axis labels) shows the percent of the range represented by each use. For each use, six years of overlap data were provided and the median, minimum, and maximum percent overlaps are shown.
5) The bottom of the plot has four lines of text that identify the specific information presented in the R-plot. The first line shows the chemical and toxicity data file. The second line shows which averaging periods were plotted. The third line lists the HUC-2s and, for the aquatic R-plots, the aquatic bins being displayed. The fourth line shows some species and range info (e.g. number of HUC-12s).

Running the R code requires a collection of files and folders (see **Directory and Files**). In brief, the R-plot process begins with converting several files (see **Directory and Files**) into R objects using one of the R files (*AqEECsImportMagToolDataC.R).* This needs to be done once each time the source data changes. Generating an R-plot then involves editing several lines within the main R file (*AqEECsSummariesWithOverlapZg.R* or *TerrEECsSummariesWithOverlapE.R*) to select the desired information (e.g. the chemical and species) and then running ('Source') the code. Details on these lines in the R code is provided in **R code lines to modify**. The resulting plot can then be saved as needed by the user. Generating a different R-plot entails editing a few lines as needed and repeating the process.

Example R-Plots

Examples of an aquatic R-plot and a terrestrial R-plot.




Toxicity Data
Vigor_EC25
Emergence_EC25
Uses
Right of Way (163139)
Pasture (135679)
Developed (91123)
Cultivated (19967)
Vegetables and Ground Fruit (1364)
Corn (1212)
Other Crops (792)
Other Grains (244)
Wheat (38)
All HUCs (1459346 acres)
11.18 % (11.18, 11.18
9.3 % (6.41, 12.11)
6.24 % (6.24, 6.24)
1.37 % (1.37, 1.37)
0.09 % (<0.01, 0.22)
0.08 % (0.03, 0.23)
0.05 % (0.02, 0.09)
0.02 % (<0.01, 0.06)
<0.01 % (<0.01, 0.06)
G A
0.01 0.02 0.05 0.10 0.20 0.50
Terrestrial EEC (lbs a.i./A)
bromoxynil octanoate: bro.terr.tox.txt
AgDrift, TerrPlant−Dry, TerrPlant−Wet (from bottom up)
huc:17b,17a
Chum salmon CR (5180) Habitat: 58 HUC12s

Directory and Files

List of files and directories associated with running the R-code. These files are provided in Attachment 2. They all need to be in the same directory. There are three R files (identified in italics below), the code of which is provided later in this document.

File/folder	Comments
AqEECsSummariesWithOverlapZg.R	Main R-code run to generate an aquatic plot
TerrEECsSummariesWithOverlapE.R	Main R-code run to generate a terrestrial plant plot
AqEECsFunctionsB.R	Functions loaded by main file
AqEECsImportMagToolDataC.R	R-code run to import data Creates R objects used by main R file Needs to be run at least once
Species Summary sheet updates_01-04-17.xlsx	Info from EPA used to create species_info.txt Export columns A-E as tab-delimited
oct.fish.tox.txt	Example of toxicity data file (provided below) Tab-delimited text file used by main R code
nurseryHUC12.txt	Example of HUC12 list (provided below) Tab-delimited text file
SalmonESUs.csv	Salmon ESU list with SpeciesID, ESU name, common name, and merged name for plot label.
TerrestrialEECs.csv	EECs generated using the terrestrial models.
Data Tables	Folder used to store any tables created
MagTool Data	Folder of files needed by *AqEECsImportMagToolDataC.R* Most files were provided by EPA
CDL_L48_2010.csv	
CDL_L48_2011.csv	
CDL_L48_2012.csv	
CDL_L48_2013.csv	
CDL_L48_2014.csv	
CDL_L48_2015.csv	
Critical_habitat_huc12.csv	
huc12_acres.csv	
huc_convert.csv	
input.csv	
species_huc12.csv	
chlorpyrifos_eec.csv	EECs provided previously by EPA (These files not utilized in this Opinion)
diazinon_eec.csv	
malathion_eec.csv	
species_info.txt	Tab-delimited file created from Species Summary sheet updates_01-04-17.xlsx Export columns A-E as tab-delimited
bromoxynil_octanoate_eec.csv	EECs generated by NMFS PWC runs

prometryn_eec.csv
bromoxynil_ParentDaughter_eec.csv

R Objects	Folder of R objects used by main R file
EECs.df	Files created by *AqEECsImportMagToolDataC.R*
cdl.names	
huc_acres.df	
huc_convert.df	
overlaps.ar	
speciesHUC.df	
species_info.df	
habitatHUC.df	

R code lines to modify

Several specific lines of the R code define the information plotted. The values for the variables specified in these lines determine the resulting R-plot (e.g. the species and chemical). These lines are located at the beginning of the R code just after the following comment.

```
# select data to summarize ***********  USER selection section of
inputs
```

Changing these values prior to running the R code is how different R-plots are generated.

An element from each of these lists is chosen in lines below.

```
chemlist<-
c("chlorpyrifos","diazinon","malathion","prometryn","bromoxynil
octanoate", "Parent", "Degradate1")
readoptions<-c("Species range","Read from file","Choose HUC2s")
```

Changing the index of chemlist determines which chemical is summarized (e.g. which EEC data is used).

```
chemtext<-chemlist[1] # ************** select chemical from
chemlist above e.g. 1 for cpy, 2 for dia, 3 for mal
```

The value of toxfilename specifies the file containing the toxicity data used to generate the toxicity display at the top of the R-plot. The file needs to exist in the same directory as the R code file. An example of a file is below.

```
toxfilename<-"pro.fish.tox.txt" # file with info for toxicity
data section of the plot
```

The index of readoptions determines which HUCs are used to generate the uses and overlaps shown on the R-plot. Species range uses the HUC12s associated with the species. Read from file uses a list of HUC12s specified in a file the user is asked to select. Choose HUC2s uses the HUC2s specified in a variable below.

```
readhucs<-readoptions[1] # ****************** how to get HUC12
and HUC2 lists, from readoptions list e.g. 1 for "Species range"
```

The value of speciesID determines which species is summarized (e.g. which HUC12s are used).

```
speciesID<-2514 # ************* Select species Entity ID
```

The value of plotHabitat determines whether the plot is based on HUCs in the species range (True) or the designated critical habitat (False).

```
plotHabitat<-FALSE # set to true to plot critical habitat rather
than species range
```

Use to force the plot to be based on selected HUC2s (False) rather than HUC12s (True).

```
useHUC12s<-True # ****************** use HUC12 info otherwise
rely on HUC2 and don't plot overlap
```

Lines used to determine which aquatic bins are summarized in the plot. Changing useSpeciesBins to False will plot only the bins specified by bintext and not all the bins

associated with a species. The bin 3 and 4 EECs associated with individual uses can be omitted by changing the lines to True. The aggregated bin 3 and bin 4 EECs across the collection of HUC12s (the 'watershed') are plotted separately.

Important note: Many of these options are based on R-plots for previous Biological Opinions. For the prometryn and bromoxynil assessments, bins 3 and 4 should not be specified since EECs were not generated for those aquatic bins. For example, `useSpeciesBins` should be set to False and `OmitBin3, and OmitBin4` should be set to True. `bintext` can include a value of '0' to plot EECs of direct runoff. For terrestrial R-plots, these values should be 2 and 5 to plot the aerial and ground application EECs as different 'bins'.

```
useSpeciesBins<-FALSE # ***************** plot all the bins
assigned to a species or specific bins
OmitBin3<-TRUE  # ***************** don't plot bin 3 values for
individual uses
OmitBin4<-TRUE  # ***************** don't plot bin 4 values for
individual uses
bintext<-c("0","2","6","7") # *************** bins to plot if not
based on species
```

The HUC-2s specified in huctext are used to determine the EEC data summarized if the HUC-2s aren't specified by the species data.

```
huctext<-c("18a") # *************** list of HUC2s for EEC data if
not specified by the species HUC12 list
```

For aquatic R-plots, the values in avgper determine which EEC averaging periods are plotted. Any combination of 1, 2, 3, and 4 can be specified. For terrestrial R-plots, the values determine which terrestrial model EECs are plotted (0, 1, 2, corresponding to AgDRIFT, TerrPlant-Dry, and TerrPlant-Wet),

```
avgper<-c(1,2) # ************** EEC averaging periods to plot, 1-
4 correspond to 1-d, 4-d, 21-d, 60-d
```

The value of multiHUC2adj determines how to adjust for multiple HUC2 regions in the species range. Set to False will average the HUC-2 EECs without respect to area, while True will account for the proportion of the total range represented by each HUC-2 as described above. This will involve more computation time.

```
multiHUC2adj<-TRUE # if multiple HUC2s adjust EECs based on
proportion of total area
```

Toxicity Data File

Example of toxicity data file used to generate toxicity data at the top of the plot. Needs to be a tab-delimited file with two columns for each endpoint. Endpoints are plotted in reverse order (i.e. first column pair appears at the bottom of the toxicity data). The first row has the name of the endpoint in the first column that will be used on the plot with the second column being ignored. The remaining rows contain data that will be plotted. The first column has the concentration and the second column has the label for the point.

Mortality.Bromoxynil	M.labels	Prey.Bromoxynil	P.labels	Growth	G.labels	Reproduction	R.labels	Aquatic.Plant	P.labels
48.48737418	1	29.1947687	1	5.7	L	39	L	20	nv.EC25
67.11465407	10	49.82897989	10	3.4	N	9	N	6	a.EC25
100	50	96	50						
148.9987565	90	184.9526123	90						
206.2392565	99	315.6729925	99						
22.8	NE	8.4	NE						

HUC12 List File

Example of a HUC12 list. Needs to be a tab-delimited file. The first row has the name of the region. This is used by the R code to present a list from which the user selects the desired HUC12s to plot. The remaining rows are the HUC12s to use to generate the plot.

Charlotte Harbor	Everglades	Nursery
31001010907	30902041000	31001010907
31001010906	30902040800	31001010906
31001011106	30902040700	31001011106
31001011107	30902041100	31001011107
31001020506	30902021300	31001020506
31001011108	30902021400	31001011108
31001030202	30902061609	31001030202
31001030101	30902061608	31001030101
31001030201	30902030200	31001030201
31002010303	30902030100	31002010303
30902050502	30902030300	30902050502
31001030102		31001030102
30902050602		30902050602
31001030103		31001030103
30902050601		30902050601
30902050408		30902050408
31001030300		31001030300
30902050603		30902050603
31001030104		31001030104
30902050607		30902050607
30902050605		30902050605
30902050606		30902050606
31001030105		31001030105
30902040102		30902040102
30902040103		30902040103
30902040107		30902040107
30902040105		30902040105
30902040106		30902040106
30902040208		30902040208
		30902041000
		30902040800
		30902040700
		30902041100
		30902021300
		30902021400
		30902061609
		30902061608
		30902030200
		30902030100
		30902030300

Aquatic R-plot R Code

The primary R code file for generating R-plots to assess impacts in the aquatic environment. Any changes to the code should be made (e.g. modify the USER selection lines), the code should then be saved, and then the code should be "Sourced". This allows the first line of the code to determine the folder the code is in and therefore how to find the support folders needed.

AqEECsSummariesWithOverlapZg.R

```
# establish dirs
wd<-dirname(sys.frame(1)$ofile) # get dir of sourced R script and other
dir
rdir<-"R Objects" # dir name for R objects within dir of R script
sdir<-"Data Tables" # dir name to store data tables within dir of R script
pdir<-"R Plots" # dir name to store R-plots within dir of R script

# load functions and data if needed
if (!exists("getHUC")) source(file.path(wd,"AqEECsFunctionsB.R")) # load
functions
if (!exists("getHabitatHUC")) source(file.path(wd,"AqEECsFunctionsB.R")) #
load functions
if (!exists("EECs.df")) load(file.path(wd,rdir,"EECs.df")) # load data
if (!exists("overlaps.ar")) load(file.path(wd,rdir,"overlaps.ar")) # load
data
if (!exists("cdl.names")) load(file.path(wd,rdir,"cdl.names")) # load data
if (!exists("huc_convert.df")) load(file.path(wd,rdir,"huc_convert.df")) #
load data
if (!exists("huc_acres.df")) load(file.path(wd,rdir,"huc_acres.df")) #
load data
if (!exists("speciesHUC.df")) load(file.path(wd,rdir,"speciesHUC.df")) #
load data
if (!exists("habitatHUC.df")) load(file.path(wd,rdir,"habitatHUC.df")) #
load data
if (!exists("species_info.df")) load(file.path(wd,rdir,"species_info.df"))
# load data

chemlist<-c("chlorpyrifos","diazinon","malathion","prometryn","bromoxynil
octanoate", "Parent", "Degradate1")
huclist<-
c("1","2","3","4","5","6","7","8","9","10a","10b","11a","11b","12a","12b",
"13","14","15a","15b","16a","16b","17a","17b","18a","18b","19a","19b","20a
","20b","21")
binlist<-c("0","2","3","4","5","6","7")
readoptions<-c("Species range","Read from file","Choose HUC2s")

# select data to summarize ***********  USER selection section of inputs
usedialog<-FALSE # ******************** use dialog boxes to enter data,
leave FALSE for now
chemtext<-chemlist[5] # *************** select chemical from chemlist
above e.g. 1 for cyp, 2 for dia, 3 for ma
toxfilename<-"oct.fish.tox.txt" # file with info for toxicity data section
of the plot
```

```
readhucs<-readoptions[1] # ****************** how to get HUC12 and HUC2
lists, from readoptions list e.g. 1 for "Species range"
speciesID<-5180 # ************* Select species Entity ID
plotHabitat<-TRUE # set to true to plot critical habitat rather than
sepcies range
useHUC12s<-T # ****************** use HUC12 info otherwise rely on HUC2
and don't plot overlap
useSpeciesBins<-F # *************** plot all the bins assigned to a
species or specific bins
OmitBin3<-T  # *************** don't plot bin 3 values for individual
uses
OmitBin4<-T  # *************** don't plot bin 4 values for individual
uses
bintext<-c("2","7") # *************** bins to plot if not based on species
huctext<-c("17b") # *************** list of HUC2s for EEC data if not
specified by the species HUC12 list
avgper<-c(2) # ************** EEC averaging periods to plot, 1-4
correspond to 1-d, 4-d, 21-d, 60-d
multiHUC2adj<-TRUE # if multiple HUC2s adjust EECs based on proportion of
total area
minX<-0.01 # set to >0 to force minimum of X axis to desired value
# ***********************************

toxdatafile<-file.path(wd,toxfilename)
ESUfile<-file.path(wd,"SalmonESUs.csv")
ESUnames<-read.csv(ESUfile)

if (usedialog) {
      chemtext<-select.list(chemlist,preselect=chemtext,title="Select
pesticide")
      toxdatafile<-file.choose()
      bintext<-select.list(binlist,preselect=bintext,title="Select desired
bins",multiple=TRUE)
      readhucs<-select.list(readoptions,preselect=readhucs,title="Select
HUCs")
      toxfilename<-basename(toxdatafile)
}
if (readhucs=="Read from file") {
      foo<-read.delim(file.choose()) # use file.choose() for windows
tk_choose.files() for Mac
      listname<-select.list(names(foo))
      speciesID<-paste0(listname," (List)")
      HUCs<-sort(na.omit(foo[,grep(listname,names(foo))])) # make sure
list is sorted in HUC12 order
      huc2list<-subset(huc_convert.df, HUC12 %in% HUCs)
      huc2list<-huc2list[order(huc2list$HUC12),]
      huctext<-unique(as.character(huc2list$HUC02))
      useHUC12s<-TRUE
      if (length(HUCs)==0) {
            stop("No HUC12s in list")
      }
}
if (readhucs=="Species range") {
      if (useSpeciesBins) {
```

```
            bintext<-
as.character(unique(species_info.df[species_info.df$EntityID==speciesID,]$
Bin))
      }
      if (plotHabitat==FALSE) {
            HUCs<-sort(getHUC(speciesID)) # species entityID to get list
of HUCs in order
            HUCtype<-"Range"
      } else {
            HUCs<-sort(getHabitatHUC(speciesID)) # species entityID to get
list of HUCs in order
            HUCtype<-"Habitat"
      }
      if (length(ESUnames[ESUnames$ID==speciesID,6])>0) {
            speciesLabel<-ESUnames[ESUnames$ID==speciesID,6]
      } else {
            speciesLabel<-
species_info.df[species_info.df$EntityID==speciesID,][1,5]
      }
      speciesID<-paste0(speciesLabel," (",speciesID,") ",HUCtype)
      huc2list<-subset(huc_convert.df, HUC12 %in% HUCs)
      huc2list<-huc2list[order(huc2list$HUC12),]
      huctext<-unique(as.character(huc2list$HUC02))
      useHUC12s<-TRUE
      if (length(HUCs)==0) {
            stop("No HUC12s in list")
      }
}
if (readhucs=="Choose HUC2s") {
      if (usedialog) {
            huctext<-select.list(huclist,preselect=huctext,title="Select
desired HUC2s",multiple=TRUE)
      }
      if (useSpeciesBins) {
            bintext<-
as.character(unique(species_info.df[species_info.df$EntityID==speciesID,]$
Bin))
      }
      speciesID<-paste0(huctext,collapse="_")
      useHUC12s<-FALSE
}

toxdata.df<-read.delim(toxdatafile) # read tox data from tab-delimited
file, columns in pairs with conc col and label col
numToxRows<-dim(toxdata.df)[2]/2

# ************************************
# get use overlap data based on HUC12s or set to dummy fill data
GroupedHUCdata<-data.frame(cdl.names[3:27])
if (useHUC12s) {
      tempHUCs<-subset(huc_acres.df, HUC12 %in% HUCs) # get list of HUC12s
with acres
      tempHUCs<-tempHUCs[order(tempHUCs$HUC12),] # order list by HUC12 to
make sure and match order of HUC12s in other data objects
```

```
      HUCsTotalAcres<-tempHUCs[,2]
      HUCsRows<-as.numeric(row.names(tempHUCs))
      HUCsPercents<-overlaps.ar[,HUCsRows,3:27] # overlaps omit HUC12 and
acres

      GroupedUseAcres<-array(0,dim=c(6,25)) # array to collect acres for
each HUC12 based on use for each of 6 years
      if (length(HUCsRows)==1) {
            GroupedUseAcres<-HUCsPercents*HUCsTotalAcres/100 # acres foe
each of 6 years for each use
      } else {
            for (i in 1:length(HUCsRows)) {
                  GroupedUseAcres<-GroupedUseAcres +
HUCsPercents[,i,]*HUCsTotalAcres[i]/100 # add acres for each of 6 years by
use for the ith HUC in the list
            }
      }

      GroupedUsePercents<-100*GroupedUseAcres/sum(HUCsTotalAcres)
      GroupedHUCdata<-
cbind(GroupedHUCdata,apply(GroupedUsePercents,2,median),apply(GroupedUsePe
rcents,2,min),apply(GroupedUsePercents,2,max))
} else {
      GroupedHUCdata<-cbind(GroupedHUCdata,array(-1,dim=25),array(-
1,dim=25),array(-1,dim=25)) # dummy data to fill object
      HUCsRows<-0
      HUCsTotalAcres<-0
}
names(GroupedHUCdata)<-c("scenario","PercentUse","min","max")

# **********************************
# get EEC info
pks.df<-EECs.df[EECs.df$Pesticide==chemtext,-1]
scentext<- levels(pks.df$Scenario)
croptext<- levels(pks.df$Crop)

temp.df<-subset(pks.df,HUC2 %in% huctext & Bin %in% bintext)
if (dim(temp.df[temp.df$Scenario=="Other Rowcrops",])[1]>0) {
      temp.df[temp.df$Scenario=="Other Rowcrops",]$Scenario<-"Other
RowCrops" # make Scenario names consistent
}
rm(pks.df)

temp.df<-temp.df[temp.df$X1.day>0,] # omit rows with 0 EEC

# check for multiple huc2 EECs
# if desired pad EECs from each HUC2 to reflect contribution to overall
distribution proportional to area
if (multiHUC2adj & length(huctext)>1 & useHUC12s) {
      hucsWithAcres<-merge(huc2list,tempHUCs)
      huc2Acres<-
aggregate(hucsWithAcres[,3],by=list(hucsWithAcres$HUC02),FUN="sum")
      huc2Acres<-cbind(huc2Acres,round(100*huc2Acres$x/sum(huc2Acres$x)))
      names(huc2Acres)<-c("HUC2","Acres","Percent")
```

```
      for (i in 1:dim(huc2Acres)[1]) {
            t.df<-
temp.df[as.character(temp.df$HUC2)==as.character(huc2Acres$HUC2[i]),]
            for (j in 1:(huc2Acres$Percent[i]-1)) {
                  temp.df<-rbind(temp.df,t.df)
            }
      }
}

temp_medians<-
aggregate(temp.df[,8:11],by=list(temp.df$Bin,temp.df$Scenario,temp.df$Crop
),FUN="median")
temp_quants<-
aggregate(temp.df[,8:11],by=list(temp.df$Bin,temp.df$Scenario,temp.df$Crop
),FUN="quant95")

# *********************************
# combined plot of use and EECs
temp_y<-array(-1,dim=length(temp_medians$Group.2)) # vector for the median
percent use data over 6 years
temp_min<-array(-1,dim=length(temp_medians$Group.2)) # vector for the min
percent use data over 6 years
temp_max<-array(-1,dim=length(temp_medians$Group.2)) # vector for the max
percent use data over 6 years
if (useHUC12s) {
      for (i in 1:length(temp_y)) {
            temp_y[i]<-
sum(na.omit(GroupedHUCdata[GroupedHUCdata$scenario==as.character(temp_medi
ans$Group.2[i]),])[,2])
            temp_min[i]<-
sum(na.omit(GroupedHUCdata[GroupedHUCdata$scenario==as.character(temp_medi
ans$Group.2[i]),])[,3])
            temp_max[i]<-
sum(na.omit(GroupedHUCdata[GroupedHUCdata$scenario==as.character(temp_medi
ans$Group.2[i]),])[,4])
      }
}

temp_data<-cbind(temp_medians,temp_quants[4:7]) # combine median and range
EECs
temp_data<-cbind(temp_data,temp_y,temp_min,temp_max) # combine percent
areas

if (useHUC12s & max(temp_y)>0) {
      temp_data<-temp_data[temp_data$temp_y>0,] # remove uses with zero
percent area
      temp_order<-as.numeric(factor(temp_data$temp_y)) # establish order
based on percent area
} else {
      temp_order<-as.numeric(factor(temp_data$Group.2)) # establish order
based on Scenario
}
temp_data<-cbind(temp_data,temp_order) # add order for y plotting of EEC
data
```

```
maxed_uses<-
unique(temp_data[temp_data$temp_order==max(temp_data$temp_order),]$Group.2
) # identify max uses (multiple could be at 100%)
for (i in 1:length(maxed_uses)) {
      temp_data[temp_data$Group.2==maxed_uses[i],]$temp_order<-
temp_data[temp_data$Group.2==maxed_uses[i],]$temp_order + (i-1) #
increment order to separate uses
}

miny<-min(temp_data$temp_y)
max1<-max(temp_data[,8:11]) # max of quant data

if (minX<=0) {
      min1<-min(temp_data[,8:11]) # autoscale X minimum to min of quant
data
} else {
      min1<-minX # overide auto scale
}

if (useHUC12s & max(temp_y)>0) {
      # omit bin 3 data except for 100% Uses unless no data remains
afterward
      if (OmitBin3 & dim(temp_data[!(temp_data$Group.1==3 &
temp_data$temp_y<99),])[1]>0) {
            temp_data<-temp_data[!(temp_data$Group.1==3 &
temp_data$temp_y<99),]
      }
      # omit bin 4 data except for 100% Uses unless no data remains
afterward
      if (OmitBin4 & dim(temp_data[!(temp_data$Group.1==4 &
temp_data$temp_y<99),])[1]>0) {
            temp_data<-temp_data[!(temp_data$Group.1==4 &
temp_data$temp_y<99),]
      }
}

# ********************************************************
# plot data with uniform spacing along Y axis
# plot median, 5%, 95% on continuous Y axis

savePlot<-menu(c("Save","View"),graphics=TRUE,title="Save or View Plot?")

if (savePlot==1) {
      plotName<-file.path(wd,pdir,paste0(speciesID,"_",chemtext,".pdf"))
      pdf(file=plotName,width=6.5,height=5.5,pointsize=9)
      par(mar=c(8.1,16.5,0.5,9),cex.axis=1, bty="n")
} else {
      par(mar=c(8.1,16.5,0.5,10),cex.axis=1, bty="n")
}

maxy<-max(temp_data$temp_order) # number of uses to plot need to add more
for bins 3 and 4 and tox data (2 plus 1 gap plus numToxRows)
ypad<-1 # use 3 to make room for 2 rows of bins 3 and 4
```

```
plot(temp_data$X1.day, temp_data$temp_order, type="n", ylim = c(0,
maxy+ypad+0.2+numToxRows), xlim = c(min1,max1), log = "x", xlab ="",
yaxt="n", ylab="")

for (i in seq(1,maxy+ypad-1,by=2)) {
#     abline(h=i-0.35,lty="dotted")
      rect(min1,i-0.35,max1,1+i-0.35,border="transparent",col="grey90")
}

binmark<-c(21,21,21,21,23,23,23) # list of symbols for bins 0,2-7
bincol<-c("black","white","grey","black","white","grey","black") # list of
colors for bin 0,2-7 symbols

avgperlabels<-c("1-d","4-d","21-d","60-d")
avgperinfo<-paste0("yearly peak ", paste0(avgperlabels[avgper],collapse=",
"), " (from bottom up)")
if (useHUC12s) {
      HUClabel<-paste0(speciesID,": ",length(HUCsRows), " HUC12s")
} else {
      HUClabel<-"HUC2 info only"
}

titletext<-
c(chemtext,avgperinfo,paste0("huc:",paste0(huctext,collapse=",),",
bin:",paste0(bintext,collapse=",")),HUClabel)

# add averaging periods 4:7 for 1, 4, 21, 60 d
for (i in avgper+3) {
      arrows(unlist(temp_data[,i+4])[,1], temp_data$temp_order + (i-4)/10,
unlist(temp_data[,i+4])[,2], temp_data$temp_order + (i-4)/10, length = 0,
angle = 90, code = 3)
      for (j in c(7:2,0)) {
            points(temp_data[temp_data$Group.1==j,i],
temp_data[temp_data$Group.1==j,]$temp_order + (i-4)/10,
pch=binmark[grep(j,c(0,2:7))], bg=bincol[grep(j,c(0,2:7))])
      }
}

# add levels "Bin 3" and "Bin 4" to temp_data columns 2 and 3
levels(temp_data[,2])<-c(levels(temp_data[,2]),"Bin 3","Bin 4")
levels(temp_data[,3])<-c(levels(temp_data[,3]),"Bin 3","Bin 4")

# add aggregate data for bins 3 and 4 if in bintext and overlap info is
available
if (useHUC12s & max(temp_y)>0) {
      t.bins<-as.numeric(bintext[bintext %in% c("3","4")]) # list bins 3
and/or 4 if in bintext
      if (length(t.bins>0)) {
            for (t.b in t.bins) {
                  t.df<-
aggregate(temp_medians[temp_medians$Group.1==t.b,4:7],by=list(temp_medians
[temp_medians$Group.1==t.b,]$Group.2),FUN="max") # maximum run for each
use and bin 3
                  names(t.df)[1]<-"scenario"
```

```
                  t.df<-merge(t.df,GroupedHUCdata)
                  t.df<-t.df[t.df$PercentUse<99,]

                  # get row names of temp_medians that contain the data in
t.df
                  t.rows<-
as.numeric(row.names(temp_medians[temp_medians$Group.1==t.b &
temp_medians$X60.day==t.df$X60.day[1] &
temp_medians$Group.2==t.df$scenario[1],]))[1] # start with first row of
t.df (pick only one row if dup X60day eecs)
                  for (i in 2:dim(t.df)[1]) {
                        t.rows<-
c(t.rows,as.numeric(row.names(temp_medians[temp_medians$Group.1==t.b &
temp_medians$X60.day==t.df$X60.day[i] &
temp_medians$Group.2==t.df$scenario[i],]))[1]) # pick only one row if dup
60-d eecs
                  }

                  tq.df<-temp_quants[t.rows,] # get quantile data from same
runs
                  t.df<-cbind(t.df,tq.df[,4:7])

                  temp_data<-rbind(temp_data,temp_data[1,]) # append row to
temp_data
                  newrow<-dim(temp_data)[1] # row number of row added
                  # update new row in temp_data for bin 3 or 4 info
                  temp_data[newrow,1]<-t.b
                  temp_data[newrow,2:3]<-paste0("Bin ",t.b)
                  temp_data[newrow,12:14]<-100
                  temp_data[newrow,15]<-maxy + t.b - 2

                  # calculate bin 3 and 4 averaging periods 4:7 for 1, 4,
21, 60 d
                  for (i in avgper+3) {
                        t.median<-sum(t.df[,i-2]*t.df[,6]/100)
                        t.max<-sum(unlist(t.df[,5+i])[,2]*t.df[,6]/100)
                        t.min<-sum(unlist(t.df[,5+i])[,1]*t.df[,6]/100)

                        # update new row in temp_data
                        temp_data[newrow,i]<-t.median
                        temp_data[newrow,i+4][1]<-t.min
                        temp_data[newrow,i+4][2]<-t.max

                        # plot handling off scale points with < or >
                        if (t.max<min1) {
                              points(min1,maxy + t.b - 2 + (i-4)/10,pch=60)
                        } else {
                              if (t.min>max1) {
                                    points(max1,maxy + t.b - 2 + (i-
4)/10,pch=62)
                              } else {
                                    arrows(t.min, maxy + t.b - 2 + (i-
4)/10, t.max, maxy + t.b - 2 + (i-4)/10, length = 0, angle = 90, code = 3)
```

```
                                    points(t.median, maxy + t.b - 2 + (i-
4)/10, pch=binmark[t.b-1], bg=bincol[t.b-1])
                                }
                            }
                        }
                    }
                }
# add various labels
axis(2,at=c(0,temp_data$temp_order,maxy+seq(1:(ypad+numToxRows))),labels=c
(paste0("All HUCs
(",format(sum(HUCsTotalAcres),digits=1,nsmall=0,trim=TRUE,scientific=FALSE
)," acres)"),as.character(paste0(temp_data$Group.2,"
(",format(sum(HUCsTotalAcres)*temp_data$temp_y/100,digits=1,nsmall=0,trim=
TRUE,scientific=FALSE),")")),rep("",ypad-1),"--------------
",names(toxdata.df[seq(1,numToxRows*2-1,by=2)])),las=1,padj=0.4,lwd=0) #
addition of Bin 3 and 4 and tox data

if (useHUC12s & max(temp_y)>0) {
      l.med<-array(data="",dim=length(temp_data$temp_y))
      l.min<-array(data="",dim=length(temp_data$temp_y))
      l.max<-array(data="",dim=length(temp_data$temp_y))
      prec<-2
      minprec<-10^(-1*prec)
# round percent areas
      for (i in 1:length(temp_data$temp_y)) {
            if (temp_data$temp_y[i]<minprec) {
                  l.med[i]<-paste0("<",minprec)
            } else {
                  l.med[i]<-round(temp_data$temp_y[i],digits=prec)
            }
            if (temp_data$temp_min[i]<minprec) {
                  l.min[i]<-paste0("<",minprec)
            } else {
                  l.min[i]<-round(temp_data$temp_min[i],digits=prec)
            }
            if (temp_data$temp_max[i]<minprec) {
                  l.max[i]<-paste0("<",minprec)
            } else {
                  l.max[i]<-round(temp_data$temp_max[i],digits=prec)
            }
      }
      axis(4,at=temp_data$temp_order,labels=paste0(l.med," % (",l.min,",
",l.max,")"),las=1,lwd=0) # percent area on right Y-axis
#     mtext("Median percent of total acres", side = 4, line=13, adj=0,
at=1)
#     mtext("(over 6 years with min, max)", side = 4, line=14, adj=0,
at=1)
}

# mtext("Uses (median total acres over 6 yrs)", side = 2, line=15, adj=0,
at=1)
mtext("Uses", side = 2, line=15, adj=0, at=1)
```

```
mtext("Toxicity Data", side = 2, line=15, adj=1)

mtext("Median EEC ppb (over 30 years with 5% to 95% )", side=1, line=2)
mtext(paste0(titletext[1],": ",toxfilename),side=1, line=4)
mtext(titletext[2],side=1, line=5)
mtext(titletext[3],side=1, line=6)
mtext(titletext[4],side=1, line=7)

# add additional data to show magnitude of effect info
for (i in seq(1,numToxRows*2-1,by=2)) {
      if (!is.na(toxdata.df[1,i])) {
            if (max(na.omit(toxdata.df[,i]))<min1) {
                  points(min1,maxy+ypad+(i+1)/2,pch=60)
            } else {
                  if (min(na.omit(toxdata.df[,i]))>max1) {
                        points(max1,maxy+ypad+(i+1)/2,pch=62)

                  } else {

      points(toxdata.df[,i],rep(maxy+ypad+(i+1)/2,length(toxdata.df[,i])),
pch=3)

      arrows(min(na.omit(toxdata.df[,i])),maxy+ypad+(i+1)/2,max(na.omit(to
xdata.df[,i])),maxy+ypad+(i+1)/2,length=0,angle=90,code=3)
                        text(toxdata.df[,i],
rep(maxy+ypad+(i+1)/2+0.4,length(toxdata.df[,i])), toxdata.df[,i+1],
cex=0.6)
                  }
            }
      }
}

# add legend
legend(x="bottomleft",pch=binmark[c(0,2:7) %in%
bintext],pt.bg=bincol[c(0,2:7) %in%
bintext],bty="n",horiz=TRUE,legend=bintext,cex=0.7)

# add X limit line
# abline(v=2000, lty=2)

if (savePlot==1) {
      dev.off()
}

# save table of EECs to file if desired
if (menu(c("Yes","No"),graphics=TRUE,title="Save Table?")==1) {
      tableName<-file.path(wd,sdir,paste0(speciesID,"_",chemtext,".csv"))
      temp_table<-temp_data
      names(temp_table)[1:3]<-c("Bin","Use","Crop")
      write.csv(temp_table,file=tableName,row.names=FALSE)
}
```

Terrestrial R-plot R Code

The primary R code file for generating an R-plot to assess impacts on terrestrial riparian vegetation. Any changes to the code should be made (e.g. modify the USER selection lines), the code should then be saved, and then the code should be "Sourced". This allows the first line of the code to determine the folder the code is in and therefore how to find the support folders needed.

Note that this R code is based on *AqEECsSummariesWithOverlapZg.R.* Much of the code is not used in the generation of terrestrial R-plots (e.g. plotting aquatic bins 3 and 4). For the purpose of this code, aquatic bins 2 and 5 are used to plot EECs associated with ground and aerial applications (respectively). Other aquatic bins are ignored (and should not be specified in the USER section). Similarly, the 1-d, 4-d, and 21-d averaging periods are used to plot three different terrestrial EEC model outputs (AgDRIFT, TerrPlant Dry, and TerrPlant Wet).

TerrEECsSummariesWithOverlapE.R

```
# establish dirs
wd<-dirname(sys.frame(1)$ofile) # get dir of sourced R script and other
dir
rdir<-"R Objects" # dir name for R objects within dir of R script
sdir<-"Data Tables" # dir name to store data tables within dir of R script
pdir<-"R Plots" # dir name to store R-plots within dir of R script

# load functions and data if needed
if (!exists("getHUC")) source(file.path(wd,"AqEECsFunctionsB.R")) # load
functions
if (!exists("getHabitatHUC")) source(file.path(wd,"AqEECsFunctionsB.R")) #
load functions
# if (!exists("TerrEECs.df")) load(file.path(wd,rdir,"TerrEECs.df")) #
load data
if (!exists("overlaps.ar")) load(file.path(wd,rdir,"overlaps.ar")) # load
data
if (!exists("cdl.names")) load(file.path(wd,rdir,"cdl.names")) # load data
if (!exists("huc_convert.df")) load(file.path(wd,rdir,"huc_convert.df")) #
load data
if (!exists("huc_acres.df")) load(file.path(wd,rdir,"huc_acres.df")) #
load data
if (!exists("speciesHUC.df")) load(file.path(wd,rdir,"speciesHUC.df")) #
load data
if (!exists("habitatHUC.df")) load(file.path(wd,rdir,"habitatHUC.df")) #
load data
if (!exists("species_info.df")) load(file.path(wd,rdir,"species_info.df"))
# load data

TerrEECs.df<-read.csv(file.path(wd,"TerrestrialEECs.csv"))
names(TerrEECs.df)[1]<-"Pesticide"

chemlist<-c("chlorpyrifos","diazinon","malathion","prometryn","bromoxynil
octanoate")
huclist<-
c("1","2","3","4","5","6","7","8","9","10a","10b","11a","11b","12a","12b",
```

```
"13","14","15a","15b","16a","16b","17a","17b","18a","18b","19a","19b","20a
","20b","21")
binlist<-c("0","2","3","4","5","6","7")
readoptions<-c("Species range","Read from file","Choose HUC2s")

# select data to summarize ***********  USER selection section of inputs
usedialog<-FALSE # ******************** use dialog boxes to enter data,
leave FALSE for now
chemtext<-chemlist[5] # *************** select chemical from chemlist
above e.g. 1 for cyp, 2 for dia, 3 for ma
toxfilename<-"bro.terr.tox.txt" # file with info for toxicity data section
of the plot
readhucs<-readoptions[1] # ******************* how to get HUC12 and HUC2
lists, from readoptions list e.g. 1 for "Species range"
speciesID<-5180 # ************* Select species Entity ID
plotHabitat<-TRUE # set to true to plot critical habitat rather than
species range
useHUC12s<-T # ******************* use HUC12 info otherwise rely on HUC2
and don't plot overlap
useSpeciesBins<-F # ***************** plot all the bins assigned to a
species or specific bins
OmitBin3<-T  # ***************** don't plot bin 3 values for individual
uses
OmitBin4<-T  # ***************** don't plot bin 4 values for individual
uses
bintext<-c("2","5") # *************** bins to plot if not based on species
huctext<-c("18a") # *************** list of HUC2s for EEC data if not
specified by the species HUC12 list
avgper<-c(1,2,3) # ************** EEC averaging periods to plot, 1-4
correspond to 1-d, 4-d, 21-d, 60-d
multiHUC2adj<-TRUE # if multiple HUC2s adjust EECs based on proportion of
total area
# ************************************

toxdatafile<-file.path(wd,toxfilename)
ESUfile<-file.path(wd,"SalmonESUs.csv")
ESUnames<-read.csv(ESUfile)

if (usedialog) {
      chemtext<-select.list(chemlist,preselect=chemtext,title="Select
pesticide")
      toxdatafile<-file.choose()
      bintext<-select.list(binlist,preselect=bintext,title="Select desired
bins",multiple=TRUE)
      readhucs<-select.list(readoptions,preselect=readhucs,title="Select
HUCs")
      toxfilename<-basename(toxdatafile)
}
if (readhucs=="Read from file") {
      foo<-read.delim(file.choose()) # use file.choose() for windows
tk_choose.files() for Mac
      listname<-select.list(names(foo))
      speciesID<-paste0(listname," (List)")
```

```
      HUCs<-sort(na.omit(foo[,grep(listname,names(foo))])) # make sure
list is sorted in HUC12 order
      huc2list<-subset(huc_convert.df, HUC12 %in% HUCs)
      huc2list<-huc2list[order(huc2list$HUC12),]
      huctext<-unique(as.character(huc2list$HUC02))
      useHUC12s<-TRUE
      if (length(HUCs)==0) {
            stop("No HUC12s in list")
      }
}
if (readhucs=="Species range") {
      if (useSpeciesBins) {
            bintext<-
as.character(unique(species_info.df[species_info.df$EntityID==speciesID,]$
Bin))
      }
      if (plotHabitat==FALSE) {
            HUCs<-sort(getHUC(speciesID)) # species entityID to get list
of HUCs in order
            HUCtype<-"Range"
      } else {
            HUCs<-sort(getHabitatHUC(speciesID)) # species entityID to get
list of HUCs in order
            HUCtype<-"Habitat"
      }
      if (length(ESUnames[ESUnames$ID==speciesID,6])>0) {
            speciesLabel<-ESUnames[ESUnames$ID==speciesID,6]
      } else {
            speciesLabel<-
species_info.df[species_info.df$EntityID==speciesID,][1,5]
      }
      speciesID<-paste0(speciesLabel," (",speciesID,") ",HUCtype)
      huc2list<-subset(huc_convert.df, HUC12 %in% HUCs)
      huc2list<-huc2list[order(huc2list$HUC12),]
      huctext<-unique(as.character(huc2list$HUC02))
      useHUC12s<-TRUE
      if (length(HUCs)==0) {
            stop("No HUC12s in list")
      }
}
if (readhucs=="Choose HUC2s") {
      if (usedialog) {
            huctext<-select.list(huclist,preselect=huctext,title="Select
desired HUC2s",multiple=TRUE)
      }
      if (useSpeciesBins) {
            bintext<-
as.character(unique(species_info.df[species_info.df$EntityID==speciesID,]$
Bin))
      }
      speciesID<-paste0(huctext,collapse="_")
      useHUC12s<-FALSE
}
```

```
toxdata.df<-read.delim(toxdatafile) # read tox data from tab-delimited
file, columns in pairs with conc col and label col
numToxRows<-dim(toxdata.df)[2]/2
minTox<-min(toxdata.df[,c(1,3)],na.rm=TRUE)
maxTox<-max(toxdata.df[,c(1,3)],na.rm=TRUE)

# ************************************
# get use overlap data based on HUC12s or set to dummy fill data
GroupedHUCdata<-data.frame(cdl.names[3:27])
if (useHUC12s) {
      tempHUCs<-subset(huc_acres.df, HUC12 %in% HUCs) # get list of HUC12s
with acres
      tempHUCs<-tempHUCs[order(tempHUCs$HUC12),] # order list by HUC12 to
make sure and match order of HUC12s in other data objects
      HUCsTotalAcres<-tempHUCs[,2]
      HUCsRows<-as.numeric(row.names(tempHUCs))
      HUCsPercents<-overlaps.ar[,HUCsRows,3:27] # overlaps omit HUC12 and
acres

      GroupedUseAcres<-array(0,dim=c(6,25)) # array to collect acres for
each HUC12 based on use for each of 6 years
      if (length(HUCsRows)==1) {
            GroupedUseAcres<-HUCsPercents*HUCsTotalAcres/100 # acres foe
each of 6 years for each use
      } else {
            for (i in 1:length(HUCsRows)) {
                  GroupedUseAcres<-GroupedUseAcres +
HUCsPercents[,i,]*HUCsTotalAcres[i]/100 # add acres for each of 6 years by
use for the ith HUC in the list
            }
      }

      GroupedUsePercents<-100*GroupedUseAcres/sum(HUCsTotalAcres)
      GroupedHUCdata<-
cbind(GroupedHUCdata,apply(GroupedUsePercents,2,median),apply(GroupedUsePe
rcents,2,min),apply(GroupedUsePercents,2,max))
} else {
      GroupedHUCdata<-cbind(GroupedHUCdata,array(-1,dim=25),array(-
1,dim=25),array(-1,dim=25)) # dummy data to fill object
      HUCsRows<-0
      HUCsTotalAcres<-0
}
names(GroupedHUCdata)<-c("scenario","PercentUse","min","max")

# ************************************
# get EEC info using TerrEECs instead
pks.df<-TerrEECs.df[TerrEECs.df$Pesticide==chemtext,-1]
scentext<- levels(pks.df$Scenario)
croptext<- levels(pks.df$Crop)

temp.df<-subset(pks.df,HUC2 %in% huctext & Bin %in% bintext)
if (dim(temp.df[temp.df$Scenario=="Other Rowcrops",])[1]>0) {
      temp.df[temp.df$Scenario=="Other Rowcrops",]$Scenario<-"Other
RowCrops" # make Scenario names consistent
```

```
}
rm(pks.df)

temp.df<-temp.df[temp.df$X1.day>0,] # omit rows with 0 EEC

# check for multiple huc2 EECs
# if desired pad EECs from each HUC2 to reflect contribution to overall
distribution proportional to area
if (multiHUC2adj & length(huctext)>1 & useHUC12s) {
      hucsWithAcres<-merge(huc2list,tempHUCs)
      huc2Acres<-
aggregate(hucsWithAcres[,3],by=list(hucsWithAcres$HUC02),FUN="sum")
      huc2Acres<-cbind(huc2Acres,round(100*huc2Acres$x/sum(huc2Acres$x)))
      names(huc2Acres)<-c("HUC2","Acres","Percent")
      for (i in 1:dim(huc2Acres)[1]) {
            t.df<-
temp.df[as.character(temp.df$HUC2)==as.character(huc2Acres$HUC2[i]),]
            for (j in 1:(huc2Acres$Percent[i]-1)) {
                  temp.df<-rbind(temp.df,t.df)
            }
      }
}

temp_medians<-
aggregate(temp.df[,8:11],by=list(temp.df$Bin,temp.df$Scenario,temp.df$Crop
),FUN="median")
temp_quants<-
aggregate(temp.df[,8:11],by=list(temp.df$Bin,temp.df$Scenario,temp.df$Crop
),FUN="quant95")

# ********************************
# combined plot of use and EECs
temp_y<-array(-1,dim=length(temp_medians$Group.2)) # vector for the median
percent use data over 6 years
temp_min<-array(-1,dim=length(temp_medians$Group.2)) # vector for the min
percent use data over 6 years
temp_max<-array(-1,dim=length(temp_medians$Group.2)) # vector for the max
percent use data over 6 years
if (useHUC12s) {
      for (i in 1:length(temp_y)) {
            temp_y[i]<-
sum(na.omit(GroupedHUCdata[GroupedHUCdata$scenario==as.character(temp_medi
ans$Group.2[i]),])[,2])
            temp_min[i]<-
sum(na.omit(GroupedHUCdata[GroupedHUCdata$scenario==as.character(temp_medi
ans$Group.2[i]),])[,3])
            temp_max[i]<-
sum(na.omit(GroupedHUCdata[GroupedHUCdata$scenario==as.character(temp_medi
ans$Group.2[i]),])[,4])
      }
}

temp_data<-cbind(temp_medians,temp_quants[4:7]) # combine median and range
EECs
```

```
temp_data<-cbind(temp_data,temp_y,temp_min,temp_max) # combine percent
areas

if (useHUC12s & max(temp_y)>0) {
      temp_data<-temp_data[temp_data$temp_y>0,] # remove uses with zero
percent area
      temp_order<-as.numeric(factor(temp_data$temp_y)) # establish order
based on percent area
} else {
      temp_order<-as.numeric(factor(temp_data$Group.2)) # establish order
based on Scenario
}
temp_data<-cbind(temp_data,temp_order) # add order for y plotting of EEC
data

maxed_uses<-
unique(temp_data[temp_data$temp_order==max(temp_data$temp_order),]$Group.2
) # identify max uses (multiple could be at 100%)
for (i in 1:length(maxed_uses)) {
      temp_data[temp_data$Group.2==maxed_uses[i],]$temp_order<-
temp_data[temp_data$Group.2==maxed_uses[i],]$temp_order + (i-1) #
increment order to separate uses
}

miny<-min(temp_data$temp_y)
min1<-min(temp_data[,8:10]) # min of quant data omitting 60-d
max1<-max(temp_data[,8:10]) # max of quant data omitting 60-d

# if (maxTox>min1) {min1<-minTox}
# if (minTox>max1) {max1<-maxTox}

min1<-min(c(minTox,min1))
max1<-max(c(maxTox,max1))

# min1<-0.1 # overide auto scale if desired

if (useHUC12s & max(temp_y)>0) {
      # omit bin 3 data except for 100% Uses unless no data remains
afterward
      if (OmitBin3 & dim(temp_data[!(temp_data$Group.1==3 &
temp_data$temp_y<99),])[1]>0) {
            temp_data<-temp_data[!(temp_data$Group.1==3 &
temp_data$temp_y<99),]
      }
      # omit bin 4 data except for 100% Uses unless no data remains
afterward
      if (OmitBin4 & dim(temp_data[!(temp_data$Group.1==4 &
temp_data$temp_y<99),])[1]>0) {
            temp_data<-temp_data[!(temp_data$Group.1==4 &
temp_data$temp_y<99),]
      }
}

# ******************************************************
```

```
# plot data with uniform spacing along Y axis
# plot median, 5%, 95% on continuous Y axis

savePlot<-menu(c("Save","View"),graphics=TRUE,title="Save or View Plot?")

if (savePlot==1) {
      plotName<-
file.path(wd,pdir,paste0(speciesID,"_",chemtext,"_Terrestrial.pdf"))
      pdf(file=plotName,width=6.5,height=5.5,pointsize=9)
      par(mar=c(8.1,16.5,0.5,9),cex.axis=1, bty="n")
} else {
      par(mar=c(8.1,16.5,0.5,10),cex.axis=1, bty="n")
}

maxy<-max(temp_data$temp_order) # number of uses to plot need to add more
for bins 3 and 4 and tox data (2 plus 1 gap plus numToxRows)
ypad<-1 # use 3 to make room for 2 rows of bins 3 and 4
plot(temp_data$X1.day, temp_data$temp_order, type="n", ylim = c(0,
maxy+ypad+0.3+numToxRows), xlim = c(min1,max1), log = "x", xlab ="",
yaxt="n", ylab="")

for (i in seq(1,maxy+ypad-1,by=2)) {
#     abline(h=i-0.35,lty="dotted")
      rect(min1,i-0.35,max1,1+i-0.35,border="transparent",col="grey90")
}

binmark<-c(21,21,21,21,21,23,23) # list of symbols for bins 0,2-7
bincol<-c("black","white","grey","black","black","grey","black") # list of
colors for bin 0,2-7 symbols

avgperlabels<-c("AgDrift","TerrPlant-Dry","TerrPlant-Wet","60-d")
avgperinfo<-paste0("", paste0(avgperlabels[avgper],collapse=", "), " (from
bottom up)")
if (useHUC12s) {
      HUClabel<-paste0(speciesID,": ",length(HUCsRows), " HUC12s")
} else {
      HUClabel<-"HUC2 info only"
}

titletext<-
c(chemtext,avgperinfo,paste0("huc:",paste0(huctext,collapse=",")),HUClabel
)

# add averaging periods 4:7 for 1, 4, 21, 60 d
for (i in avgper+3) {
      arrows(unlist(temp_data[,i+4])[,1], temp_data$temp_order + (i-4)/10,
unlist(temp_data[,i+4])[,2], temp_data$temp_order + (i-4)/10, length = 0,
angle = 90, code = 3)
      for (j in c(7:2,0)) {
            points(temp_data[temp_data$Group.1==j,i],
temp_data[temp_data$Group.1==j,]$temp_order + (i-4)/10,
pch=binmark[grep(j,c(0,2:7))], bg=bincol[grep(j,c(0,2:7))])
      }
}
```

```
# add levels "Bin 3" and "Bin 4" to temp_data columns 2 and 3
levels(temp_data[,2])<-c(levels(temp_data[,2]),"Bin 3","Bin 4")
levels(temp_data[,3])<-c(levels(temp_data[,3]),"Bin 3","Bin 4")

# add aggregate data for bins 3 and 4 if in bintext and overlap info is
available
if (useHUC12s & max(temp_y)>0) {
     t.bins<-as.numeric(bintext[bintext %in% c("3","4")]) # list bins 3
and/or 4 if in bintext
     if (length(t.bins>0)) {
          for (t.b in t.bins) {
               t.df<-
aggregate(temp_medians[temp_medians$Group.1==t.b,4:7],by=list(temp_medians
[temp_medians$Group.1==t.b,]$Group.2),FUN="max") # maximum run for each
use and bin 3
               names(t.df)[1]<-"scenario"

               t.df<-merge(t.df,GroupedHUCdata)
               t.df<-t.df[t.df$PercentUse<99,]

               # get row names of temp_medians that contain the data in
t.df
               t.rows<-
as.numeric(row.names(temp_medians[temp_medians$Group.1==t.b &
temp_medians$X60.day==t.df$X60.day[1] &
temp_medians$Group.2==t.df$scenario[1],]))[1] # start with first row of
t.df (pick only one row if dup X60day eecs)
               for (i in 2:dim(t.df)[1]) {
                    t.rows<-
c(t.rows,as.numeric(row.names(temp_medians[temp_medians$Group.1==t.b &
temp_medians$X60.day==t.df$X60.day[i] &
temp_medians$Group.2==t.df$scenario[i],]))[1]) # pick only one row if dup
60-d eecs
               }

               tq.df<-temp_quants[t.rows,] # get quantile data from same
runs
               t.df<-cbind(t.df,tq.df[,4:7])

               temp_data<-rbind(temp_data,temp_data[1,]) # append row to
temp_data
               newrow<-dim(temp_data)[1] # row number of row added
               # update new row in temp_data for bin 3 or 4 info
               temp_data[newrow,1]<-t.b
               temp_data[newrow,2:3]<-paste0("Bin ",t.b)
               temp_data[newrow,12:14]<-100
               temp_data[newrow,15]<-maxy + t.b - 2

               # calculate bin 3 and 4 averaging periods 4:7 for 1, 4,
21, 60 d
               for (i in avgper+3) {
                    t.median<-sum(t.df[,i-2]*t.df[,6]/100)
                    t.max<-sum(unlist(t.df[,5+i])[,2]*t.df[,6]/100)
```

```
                        t.min<-sum(unlist(t.df[,5+i])[,1]*t.df[,6]/100)

                        # update new row in temp_data
                        temp_data[newrow,i]<-t.median
                        temp_data[newrow,i+4][1]<-t.min
                        temp_data[newrow,i+4][2]<-t.max

                        # plot handling off scale points with < or >
                        if (t.max<min1) {
                              points(min1,maxy + t.b - 2 + (i-4)/10,pch=60)
                        } else {
                              if (t.min>max1) {
                                    points(max1,maxy + t.b - 2 + (i-
4)/10,pch=62)
                              } else {
                                    arrows(t.min, maxy + t.b - 2 + (i-
4)/10, t.max, maxy + t.b - 2 + (i-4)/10, length = 0, angle = 90, code = 3)
                                    points(t.median, maxy + t.b - 2 + (i-
4)/10, pch=binmark[t.b-1], bg=bincol[t.b-1])
                              }
                        }
                  }
            }
      }
}

# add various labels
axis(2,at=c(0,temp_data$temp_order,maxy+seq(1:(ypad+numToxRows))),labels=c
(paste0("All HUCs
(",format(sum(HUCsTotalAcres),digits=1,nsmall=0,trim=TRUE,scientific=FALSE
)," acres)"),as.character(paste0(temp_data$Group.2,"
(",format(sum(HUCsTotalAcres)*temp_data$temp_y/100,digits=1,nsmall=0,trim=
TRUE,scientific=FALSE),")")),rep("",ypad-1),"--------------
",names(toxdata.df[seq(1,numToxRows*2-1,by=2)])),las=1,padj=0.4,lwd=0) #
addition of Bin 3 and 4 and tox data

if (useHUC12s & max(temp_y)>0) {
      l.med<-array(data="",dim=length(temp_data$temp_y))
      l.min<-array(data="",dim=length(temp_data$temp_y))
      l.max<-array(data="",dim=length(temp_data$temp_y))
      prec<-2
      minprec<-10^(-1*prec)
# round percent areas
      for (i in 1:length(temp_data$temp_y)) {
            if (temp_data$temp_y[i]<minprec) {
                  l.med[i]<-paste0("<",minprec)
            } else {
                  l.med[i]<-round(temp_data$temp_y[i],digits=prec)
            }
            if (temp_data$temp_min[i]<minprec) {
                  l.min[i]<-paste0("<",minprec)
            } else {
                  l.min[i]<-round(temp_data$temp_min[i],digits=prec)
            }
```

```
            if (temp_data$temp_max[i]<minprec) {
                  l.max[i]<-paste0("<",minprec)
            } else {
                  l.max[i]<-round(temp_data$temp_max[i],digits=prec)
            }
      }
      axis(4,at=temp_data$temp_order,labels=paste0(l.med," % (",l.min,",
",l.max,")"),las=1,lwd=0) # percent area on right Y-axis
#     mtext("Median percent of total acres", side = 4, line=13, adj=0,
at=1)
#     mtext("(over 6 years with min, max)", side = 4, line=14, adj=0,
at=1)
}

# mtext("Uses (median total acres over 6 yrs)", side = 2, line=15, adj=0,
at=1)
mtext("Uses", side = 2, line=15, adj=0, at=1)
mtext("Toxicity Data", side = 2, line=15, adj=1)

mtext("Terrestrial EEC (lbs a.i./A)", side=1, line=2)
mtext(paste0(titletext[1],": ",toxfilename),side=1, line=4)
mtext(titletext[2],side=1, line=5)
mtext(titletext[3],side=1, line=6)
mtext(titletext[4],side=1, line=7)

# add additional data to show magnitude of effect info
for (i in seq(1,numToxRows*2-1,by=2)) {
      if (!is.na(toxdata.df[1,i])) {
            if (max(na.omit(toxdata.df[,i]))<min1) {
                  points(min1,maxy+ypad+(i+1)/2,pch=60)
            } else {
                  if (min(na.omit(toxdata.df[,i]))>max1) {
                        points(max1,maxy+ypad+(i+1)/2,pch=62)

                  } else {

      points(toxdata.df[,i],rep(maxy+ypad+(i+1)/2,length(toxdata.df[,i])),
pch=3)

      arrows(min(na.omit(toxdata.df[,i])),maxy+ypad+(i+1)/2,max(na.omit(to
xdata.df[,i])),maxy+ypad+(i+1)/2,length=0,angle=90,code=3)
                        text(toxdata.df[,i],
rep(maxy+ypad+(i+1)/2+0.2,length(toxdata.df[,i])), toxdata.df[,i+1],
adj=c(0.5,1), cex=0.6)
                  }
            }
      }
}

# add legend
legend(x="bottomleft",pch=binmark[c(0,2:7) %in%
bintext],pt.bg=bincol[c(0,2:7) %in%
bintext],bty="n",horiz=TRUE,legend=c("G","A"),cex=0.7)
```

```
# add X limit line
# abline(v=2000, lty=2)

if (savePlot==1) {
      dev.off()
}

# save table of EECs to file if desired
if (menu(c("Yes","No"),graphics=TRUE,title="Save Table?")==1) {
      tableName<-
file.path(wd,sdir,paste0(speciesID,"_",chemtext,"_Terrestrial.csv"))
      temp_table<-temp_data
      names(temp_table)[1:3]<-c("Bin","Use","Crop")
      write.csv(temp_table,file=tableName,row.names=FALSE)
}
```

Import R Code

R code to import various data files and generate R objects for use in *AqEECsSummariesWithOverlapZg.R* and *TerrEECsSummariesWithOverlapE.R.* Needs to be run at least once to generate R objects. Also, needs to be run anytime that the data used to generate plots has been updated (e.g. new EECs or GIS data are generated). Needs to be saved and then 'sourced' in order for the first line to determine where the file is located and find the folder to save the R objects.

AqEECsImportMagToolDataC.R

```
# establish directories
wd<-dirname(sys.frame(1)$ofile) # get dir of sourced R script and other
dir
mdir<-"MagTool Data" # dir name with MagTool data within dir with script
rdir<-"R Objects" # dir name for R objects within dir of R script

# _____________ EECs.df
# import EEC files into a single dataframe
# set file names for EEC files
c.file<-"chlorpyrifos_eec.csv"
d.file<-"diazinon_eec.csv"
m.file<-"malathion_eec.csv"
p.files<-c(c.file,d.file,m.file)
p.names<-c("chlorpyrifos","diazinon","malathion")

# get dir name for EEC files assuming they are in the same dir
eec.dir<-file.path(wd,mdir) # or use dirname(file.choose())

for (i in 1:length(p.files)) {
  temp<-read.csv(file.path(eec.dir,p.files[i])) # read EEC csv
  temp<-cbind(p.names[i],temp) # prepend column with pesticide name
  names(temp)[1]<-"Pesticide" # rename first column that was created
  if (i == 1) EECs.df<-temp
  else EECs.df<-rbind(EECs.df,temp)
}
names(EECs.df)[6]<-"Crop"

# ___________________  overlaps.ar, cdl.names
# import CDL data into a 3D array of overlaps (year,huc12,use with HUC12
and acres first)
# list of CDL files
cdl.files<-
c("CDL_L48_2010.csv","CDL_L48_2011.csv","CDL_L48_2012.csv","CDL_L48_2013.c
sv","CDL_L48_2014.csv","CDL_L48_2015.csv")
# get dir name for CDL files assuming they are in the same dir
cdl.dir<-file.path(wd,mdir) # or use dirname(file.choose())

for (i in 1:length(cdl.files)) {
  temp<-read.csv(file.path(cdl.dir,cdl.files[i])) # read CDL csv
  temp<-temp[order(temp$HUC12),] # make sure to order overlap data by
HUC12
```

```
  if (i == 1) overlaps.ar<-array(-1,dim=c(6,dim(as.matrix(temp)))) #
create initial array
  overlaps.ar[i,,]<-as.matrix(temp)
}

cdl.names<-gsub("\\."," ",names(temp)) # list of use names with spaces

# ____________________  huc_acres.df huc_convert.df
# create dataframe with huc12, acres, huc2

huc_convert.df<-read.csv(file.path(cdl.dir,"huc_convert.csv")) # get huc12
to huc2 conversion
huc_acres.df<-data.frame(overlaps.ar[1,,1:2])
names(huc_acres.df)<-c("HUC12","Acres")

# ____________________  speciesHUC.df
# create dataframe with species id and comma-separated range huc12 list
speciesHUC.df<-read.csv(file.path(cdl.dir,"species_huc12.csv"),as.is=TRUE)

# ____________________  habitatHUC.df
# create dataframe with species id and comma-separated critical habitat
huc12 list
habitatHUC.df<-
read.csv(file.path(cdl.dir,"critical_habitat_huc12.csv"),as.is=TRUE)

# ____________________  species_info.df
# create dataframe from Species Summary with HUC, Bin, EntityID and
Species
species_info.df<-
read.delim(file.path(cdl.dir,"species_info.txt"),as.is=TRUE)

# ____________________ save data objects to same dir
save(EECs.df,file=file.path(wd,rdir,"EECs.df"))
save(overlaps.ar,file=file.path(wd,rdir,"overlaps.ar"))
save(cdl.names,file=file.path(wd,rdir,"cdl.names"))
save(huc_convert.df,file=file.path(wd,rdir,"huc_convert.df"))
save(huc_acres.df,file=file.path(wd,rdir,"huc_acres.df"))
save(speciesHUC.df,file=file.path(wd,rdir,"speciesHUC.df"))
save(habitatHUC.df,file=file.path(wd,rdir,"habitatHUC.df"))
save(species_info.df,file=file.path(wd,rdir,"species_info.df"))
```

Functions R Code

R code with various functions used by other R files. Needs to be loaded into R before other files are run. Not all functions are used by *AqEECsSummariesWithOverlapZg.R* or *TerrEECsSummariesWithOverlapE.R.*

AqEECsFunctionsB.R

```
# *************************
# set of functions used in other places
parsename <- function(fn) {
  fn<-basename(fn)
  t1<-unlist(strsplit(fn,"_"))
  esaindex<-grep("ESA",t1)
  t2<-unlist(strsplit(t1[esaindex],"ESA"))
  tmp<-paste(t1[1:(esaindex-2)],collapse="_")
  output<-list(huc=t2[2],scenario=t2[1],bin=t1[esaindex-1],run=tmp)
  return(output)
}

meanci <- function(X1, conf = 95) {
  lenX1 <- length(X1)
  X195 <- qt(1 - (100 - conf)/200, df = lenX1 - 1) *
sqrt(var(X1))/sqrt(lenX1)
  output <- c(lower.ci = mean(X1) - X195, upper.ci = mean(X1) +X195)
  return(output)
}

dist95 <- function(X1) {
  lower05 <- qnorm(0.05,mean = mean(X1), sd = sd(X1))
  upper95 <- qnorm(0.95,mean = mean(X1), sd = sd(X1))
  output <- c(lower05, upper95)
  return(output)
}

quant95 <- function(X1) {
  output<-quantile(X1,c(0.05,0.95))
  return(output)
}

f1in15 <- function(X1) {
  output <- qnorm(14/15,mean = mean(X1), sd = sd(X1))
  return(output)
}

# extract daily eecs from the *daily.csv file into a 3D array
dayeecs <- function(fn) {
      foo<-read.csv(fn,header=FALSE,skip=5) # read SWCC daily
      foo<-foo[,2:4]*10^6 # get peaks and convert to ppb
      foo<-foo[-seq(from=1155, length=7, by=1461),] # omit leap days
      output<-array(dim=c(365,30,3)) # create 3D array for daily eecs
(day, year, value)
      for (i in 1:3){
```

```
            output[,,i]<-array(foo[,i], dim=c(365,30)) # build array from
daily values (avg aqueous, avg benthic, peak aqueous)
      }
      return(output)
}

# extract yearly summaries from a *15_Parent.text
yreecs <- function(fn) {
      tmp<-scan(fn,skip=19) # read yearly summary data from *15_Parent.txt
summary file
      tmparr<-array(tmp, dim=c(9,30))
      output<-t(tmparr) # 2D array (30,9) of year on rows and different
summaries in columns
}

# extract overall summaries from a *15_Parent.txt file
runeecs <- function(fn) {
      tmpfile<-readLines(fn,18) # get overall summary lines
      tmpfile<-tmpfile[6:15] # overall summary with each line having the
summary value

      # extract data from each summary line
      measlist <- vector("list",length(tmpfile))
      for (i in 1:length(tmpfile)) {
            l1<-unlist(strsplit(tmpfile[i],"="))
            names(measlist)[i]<-paste0("x",gsub("[ ,-
]",".",trimws(l1[1]))) # measurement description w/o spaces, hyphens
            measlist[[i]]<-as.numeric(unlist(strsplit(l1[2]," ppb"))[1]) #
measurement value
      }
      output<-measlist
}

# get array of HUC12s from comma delimited list of range
getHUC <- function(id) {
      tempHUC<-speciesHUC.df[speciesHUC.df$EntityID==id,2]
      output<-as.numeric(gsub("\\D","",unlist(strsplit(tempHUC,","))))
      return(output)
}

# get array of HUC12s from comma delimited list of critical habitat
getHabitatHUC <- function(id) {
      tempHUC<-habitatHUC.df[habitatHUC.df$EntityID==id,2]
      output<-as.numeric(gsub("\\D","",unlist(strsplit(tempHUC,","))))
      return(output)
}
```