WGFP_Enc_Summaries_RShinyApp How-To

Location:
U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections\CodingDetections\WGFP_Enc_Summaries_RShinyApp

Table of Contents

Uses

- Combines, cleans, and displays data from Stationary OregonRFID readers, Biomark antennas, mobile runs, Recaptures, release data, pressure transducers, site visits, and USGS in one location.

- Shows fish movement by day in map, table, plot, and custom animation

- Shows dataset of physical "states"

- QAQC of marker tags, lengths and weights, unknown tags, crosstalk, ghost tag movements

General Notes

- All plots are interactive. Hover over plot for more info and toggle what is displayed by clicking and double clicking items in the legend. Zoom in by clicking and dragging within the plot.
- Datatables are also interactive. Click the column titles of any column to sort in ascending/descending order. You can use the Search bar on the right to search/filter for any specific values. Most of the individual filters work, some don't, depending on column type I believe.
- It's important when adding a new updated file to the app that it had the exact same column names and order as the older file
- Data displayed in tables is often downloadable
  - When the table is accompanied by this download button, all data within the table will be downloaded

  SAVE DATA
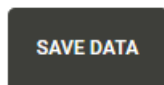
  - When the table is accompanied by this download button, only the data that is displayed on the screen will be downloaded. Display all rows of the table to download all data in this table.

  CSV

- UTM's for stationary and biomark stations are assigned in the metadata file
- DTY field for Stationary Antenna data refers to Detection Type in the raw .txt files but refers to date in the app

How to Open

OPENING FROM U: DRIVE
1. Navigate to App location and open the WGFP_dataclean_vis "R Project" file. Rstudio will open
2. If the "app.R" file isn't already open in the top left panel: in the bottom right panel, navigate to the "files" tab and open the app.R file.
3. Click "Run App" in the top right of the screen. You may be required to download some packages if this is the first time running the app.
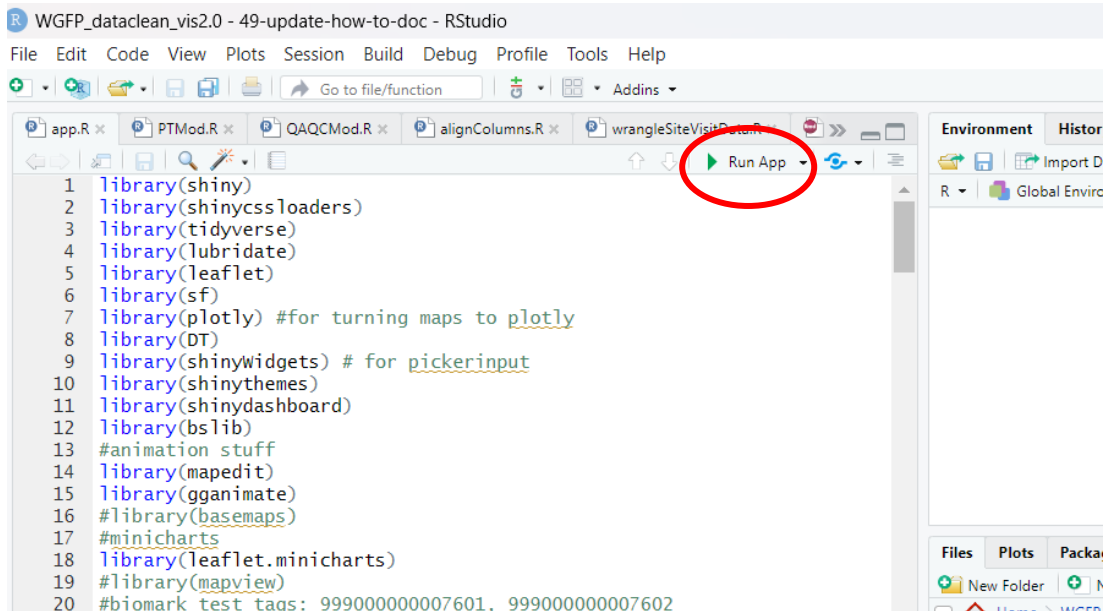
Note: It typically takes 20-40 seconds to read in all files from the app. If you are frequently opening and closing the app in one session, you can speed this process up by reading in the files to your local environment. These files are located in the "data read ins" section of the app.r file. To read them in, select all the code in this section, then use cntrl+enter to run this code. It takes 20-40 seconds to read in the files but once you do this, when you run the app each subsequent time, it will boot right up in a couple seconds.

Navigating the App

About/How to Use Tab

This tab is a concise version of this document. It doesn't have much there now but can be modified to include the most relevant pieces of the how-to doc.

Individual Datasets Tab

This Tab shows data incorporated into the app from different sources.

- Stationary Clean: A clean dataset of all Stationary antenna data with added UTMs, no marker tags, or duplicate rows. All timestamps and dates are in the same format. 900_ is taken off of the tags. See cleanStationary function in update data app for more info on cleaning.
- Biomark: All Biomark combined detections from Kaibab Park/Granby Diversion and both Windy Gap Sites including Marker Tags
- Mobile: All Detections from all Mobile Runs
- Recaptures: All Recaptured fish
- Release: All release data along with frequency plots for length and weight. Note: entries with no L/W listed won't show up in the plot
- Ghost Data: a list of ghost tags that also have a "ghost date" associated with them, or the date they turned into a ghost tag
- Aviation predation: list of tags with a "predation date" associated with them when they turned into a predated tag
- Pressure Transducers: a combined dataframe of all pressure transducer data in the app
- USGS 15 Minute Data: USGS Data from their website from the stream gauge at hitching Post. As of 5/22/2024, only water temp and discharge are read in
- Site Visit Data: Combined site visit data from all stationary and biomark sites.

Encounter Histories Tab

Encounter Release History Summary Wide: shows ENC_Release_wide_summary Dataframe from enc_hist_summary_wide_function.

- This tab is meant to be a summary file for all fish in "wide" data format. Each tag will have just one entry, along with release info and information about what sort of Events it has experienced over its life.
- Filters are pretty self-explanatory. Click Render Table/Data to display new tables with filters.
- Should be same amount of entries as release file
- Columns and filters to note:
  - SiteCode_n is the raw number of detections at one antenna, or recaptured, etc.
  - SiteCode/Recapture binary columns are just weather or not a fish was detected or captured by that method
  - TotalEncounters adds the number of unique Events (a recapture, stationary detection, mobile detection, etc). There is a filter for this column in the sidebar.
  - Through_dam tells weather a fish has gone through the dam in its history, based on release info, recaps, and detections. It does NOT tell if a fish went upstream or downstream through the dam. There is also a filter for this in the sidebar
  - Went_above_dam_no_channel and subsequent columns are derived from the States function, which tells if a fish went below/above the dam using the channel or not from upstream or downstream. See the Get-states_function section for more details
  - Sum_dist is the total number of meters travelled by a fish in its history. There's a filter for this in the sidebar.

- This is a good tab to see how many fish have swam through CRCC, fish that have swam a long way, or find fish with extensive encounter histories while being able to filter on tag number, species, length/weight, Release Site.

Examples
- Finding fish that have swam through the CRCC, specifically down through, that were less than 350mm.
  a. Adjust the filters pictured to desired categories and render table

  b. Scroll to the column went_below_dam_throughChannel and select "true" from the filters here

  c. Scroll back to the beginning of the table and view the row numbers. There have been 6 fish of length 350 or below that have swam through the connectivity channel from upstream to downstream (as of 5/14/2024).

	230000144362	3	River	Run
	230000143924	4	Colorado River	Lower R[...] Run
	230000143976	4	Colorado River	Lower R[...] Run

Showing 1 to 6 of 6 entries (filtered from 127 total entries)

    i. Note that the A fish registers TRUE in went_below_dam_throughChannel not solely if it is detected in the CRCC (see All Events and plot for that example), but only if the fish was actually detected in CRCC then subsequently detected (whether at RB, HP, Biomark, recapture, or mobile run) downstream of CD or CS.

- Identify Fish to explore as potentially avian predated
  - o Fish that have swam abnormally long distances may have been avian predated. Adjust the distance moved filter and render table

  - o Scroll to the "sum_dist" column and click it to sort by ascending

  - o Scroll back to the beginning of the table to view the tag names, then manually view the encounter history of each tag in the movements tab or "All Encounter Histories" to see if their encounter history makes spatial and chronological sense. Lots of these tags may already be in the avian predation file, so check that file in the "individual datasets" tab to see if that tag has already been accounted for.

All Encounter Histories: Shows and plots All_Events Dataframe from All_combined_events_function. These are all detections/events combined, including Statoinary, Biomark, and mobile antennas; recaptures and releases. 230000142723 has over 120k detections on its own at Hitching Post.

- Most filters are self-explanatory, but a the checkboxes are important to note:

- o Display USGS Discharge Filter allows you to filter data based off USGS discharge reading at Hitching Post within 13 hours of the detection (in winter, they record the data every 8-12 hours, 15 minutes in the summer). However, not every detection has an associated discharge reading because of this, so those rows get filtered out of the dataset immediately if you use this filter
- o Display Environmental Data Filter allows you to filter data based off Pressure Transducer readings at specific sites within one hour of the detection. However, only sites that have a transducer at them will get this data associated with their detections and only within one hour, so using this filter will automatically filter out rows of data that do not have any of these readings associated with the detections.
- o Remove Duplicate Days, TAGs, Events and UTMs condenses detections into the first and last detections of the day as well as any other antenna detections with unique UTMs. In this way, it filters out most of the redundant detections across days. For example, using this filter condenses TAG 230000142723 down to 996 entries. This equates to about 259 days of detections from May 6, 2021 to Feb 3, 2022. All at Hitching Post.
  - ▪ The df that results of this filter used on all detections is downloaded, brought into GIS, and used to make stations with, which is then subsequently used to make the "movements" df with.
  - ▪ This is one of the most "useful" files you can get from this app because of how it shaves off the dataset to the most relevant detections.
- o Remove Duplicate Tags Filter will remove instances where there are multiple detections of the same tag. This is helpful in answering questions involving "How many unique fish?". Example: on this tab you could answer the question "How many unique rainbow trout over 250 mm hit Windy Gap Biomark and Hitching Post antennas during the day in spring 2021?"
- The plot will plot anything made with the filtered data and the frequency table will display the raw number of detections from that filtered data for each antenna

Examples
- How many fish released in 2023 were detected in the CRCC in Fall 2023?
  - o In the filters tab, change the following filters:

Filter by Tag

Select a Date Range:

2023-10-01 to 2023-12-31

Hour of Day

0 — 23
0 3 6 9 12 15 18 21 23

Select Event

CD1, CD2, CS1, CS2, CU1, CU2 ▾

Select Fish Species:

LOC, MTS, NO INFO, RBT, RXN, ▾

Fish Release Length (mm)

63 — 663
63 123 183 303 423 543 663

Fish Release Weight (grams)

2 — 3,383
2 341 680 1,358 2,036 2,714 3,383

Release Date Range:

2023-01-01 to 2023-12-31

Select Release Site:

BELOW CONFLUENCE ANTENN ▾

And make sure the "Remove Duplicate Tags" filter is checked before clicking "render table"

☐ Display USGS Discharge Filter

☐ Display Environmental Data Filters

☐ Remove Duplicate Days, TAGs, Events and UTMs

☑ Remove Duplicate TAGs: doesn't work with TAG filter

RESET FILTERS

Note: If there are any NA entries in any of the applicable filters (especially the numeric ones), those rows are excluded from the table.

RENDER TABLE

o The code filters the data based on the date range and events selected, then removes instances where the same tag shows up multiple times under these filters. The result is a df of unique tags that qualify for those filters. Find the number of fish by viewing the row number: 21

- Which antenna had the most raw detections in summer 2022?
  - Select a date range, render the table, and navigate to the "Plot and Table" tab
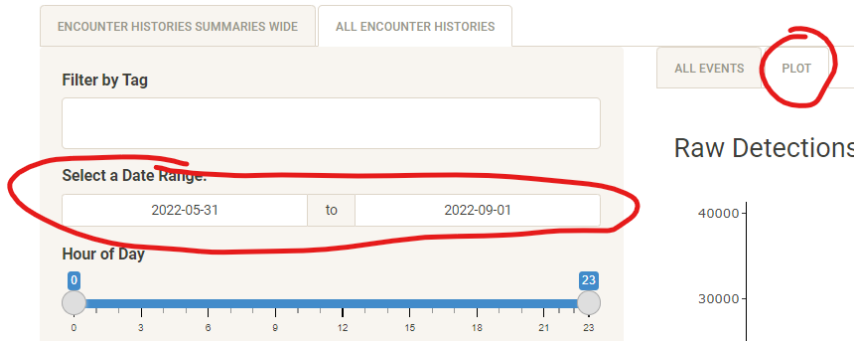
  - Scroll to the "Raw Detection Frequencies by Event" table to see the distribution of events during that time frame: Looks like HP3 had the most raw detections.

Raw Detection Frequencies by Event

Copy CSV Print Show 10 ∨ entries   Search: [   ]

Event	Raw Detections
All	All
HP3	840487
WG1	606174
CF6	399632
HP4	300309
GD1	168594
WG2	98010
RB1	97001
RB2	72061
CF5	19243
Release	4837

Showing 1 to 10 of 22 entries   Previous 1 2 3 Next

  - These numbers can shoot up if a fish is sitting on an antenna for a long amount of time. If we want to get a better picture of how much activity actually occurred on a daily level between antennas, we can use the "Remove duplicate days, tags, events and UTM's" filter described above.

o The result shows a much better representation of what is going on at the sites across this timeframe, showing instances where fish had their first and last detections of the day and all detections in between. Hitching Post saw a lot of action.

Event	Raw Detections
All	All
HP3	1398
HP4	1139
RB2	402
RB1	337
CF6	246
Release	218
GD1	132
Recapture	65
WG1	52
CF5	49
Recapture and Release	10

Showing 1 to 11 of 11 entries   Previous | 1 | Next

o Why might CF5 have so much less action than CF6? To answer this, there's a few things to check. My first thought would be that there's a sculpin hanging out on just CF. We can go back to the "All Events" tab, select just those antennas in the "Event" filter, and scroll through the data so see if this is true

o We see that there are a lot of different tags on CF6 during this timeframe, so that theory doesn't work. Next thought is if the detection distance was very low, or the antenna was just off during this time. To check that data, we go to the QAQC tab.

o Under "Marker Tags", can check if the antenna was running. Select CF5 and CF6 from the options, select all marker tags, change the timeframe to what we were looking at before, and render the data. Then toggle the plot displays by clicking in the legend.

- o What we learn is that CF5 actually didn't seem to be working for much of this time period. If you were to turn on CF6 on the map, you'd see marker tag detections pretty consistently across the same timeframe. This explains the discrepancy between CF5 and CF6 during this timeframe.
- Seeing a Fish's Full Encounter History/Investigate Individual Tag for Avian Predation
    - o Using one of the tags found (could be from by sorting on distance moved on the Encounter Histories Summaries Wide tab, seeing something in the states tab, seeing quick movement in the movements tab), we copy and paste that tag into the "All Encounter Histories" Tab. For example, I found 230000143683 (a big rainbow trout) by filtering for fish on Encounter Histories Summaries Wide tab that had 11 or more "events".
    - o Once the table is rendered, you can scroll to look at the fish's history (it default sorts by datetime). This fish has 630 encounters.

    Scrolling through this data, you find yourself seeing a lot of detections on the same antenna across a bunch of days that doesn't tell you much. However, using the "Remove duplicate days, tags, events and UTM's" filter here is helpful again, preserving useful detections and removing ones that are unnecessary. In this case, it takes the total encounters down to 104.

    - o Now this is easier to view. The Encounter history of the fish started with its release 2022-10-5, hit WG1 until the antenna was taken out of the river 2022-11-02. It hit hitching post and red barn next spring intermittently, including missing hp3 in late May, a time when the marker tag on HP3 wasn't being registered consistently (seen from the Marker tag QAQC tab). It was recaptured in June at lower Red Barn Fry Site and was registered as having grown 9mm in length. In December 2023, it made its way up the CRCC, hitting CD in December then resuming its journey in March, hitting CU March 12, then hit CF a week later where its more or less stayed since then (as of 5/14/2024). Based on this encounter history, I'd say the fish is not a tag succumbed to avian predation, just well-travelled.

Weekly States Tab

This is where "States" are displayed. The states are defined on a weekly basis as A (below the dam), B (above the Dam), C (in the Connectivity channel), G (ghost tag detection), or P (predated by bird). A weekly "Unique event" is a filter in the sidebar, and is defined as a detection on a new antenna for that week, recapture, or release. A fish could have 6 different events in a week, but if they are all below the dam, it will just get assigned state "A" for that week.

States Dataframe

- The States filter in the sidebar can help determine "magic" fish that may have succumbed to avian predation, since they hit multiple weird states in a week. For example, 230000228444 has one week where it transitions at confluence, then red barn, then confluence again.
- Det_type column is useful when there is only 1 state of the week, but doesn't accurately capture everything if there are multiple states on a week
- Date is the day that the week starts, not when the actual detection(s) happened.
- C_number_of_detections is the total number of raw detections for the week. Might be 22k, might be 1.
- Weekly_unique_events is the total number of events that happened in a week. This can also be helpful in finding "magic" fish.
- These states are used to determine the columns "went_below_dam_throughChannel" etc in "All Encounter Histories Summary Wide" tab, because if a fish started upstream then used the channel, it will have B then a C somewhere in its history

Examples

- Here is the fish 230000228709, a sizable brown trout.

Date	weeks_since	TAG	State	det_type	ReleaseSite	Species	Release_Length	Release_Weight	c_number_of_detections	weekly_unique_events
2020-10-06	5	230000228709	A	Release	Hitching Post	LOC	360	505	1	3
2020-10-13	6	230000228709	A	Hitching Post Stationary Antenna	Hitching Post	LOC	360	505	4	2
2020-10-21	7	230000228709	A	Mobile Run	Hitching Post	LOC	360	505	1	1
2021-01-27	21	230000228709	A	Hitching Post Stationary Antenna	Hitching Post	LOC	360	505	4	2
2021-02-24	25	230000228709	A	Hitching Post Stationary Antenna	Hitching Post	LOC	360	505	4	2

Showing 1 to 5 of 5 entries   Previous 1 Next

- Finding Avian Predated Fish to Investigate
  - Use the "States" filter to view all the different states that fish have been in in a week. Select the most dubious like ABABA, BAB, etc.

States:

A, AB, ABA, ABABA, ACB, AP, B, BA, BAB, BABA, BC, BCA, BCBC, C, CA, CB, CBC, CBCB, G, I ▼

SELECT ALL		DESELECT ALL
A		✓
AB		✓
ABA		✓
ABABA		✓
ACB		✓
AP		✓
B		✓
BA		✓
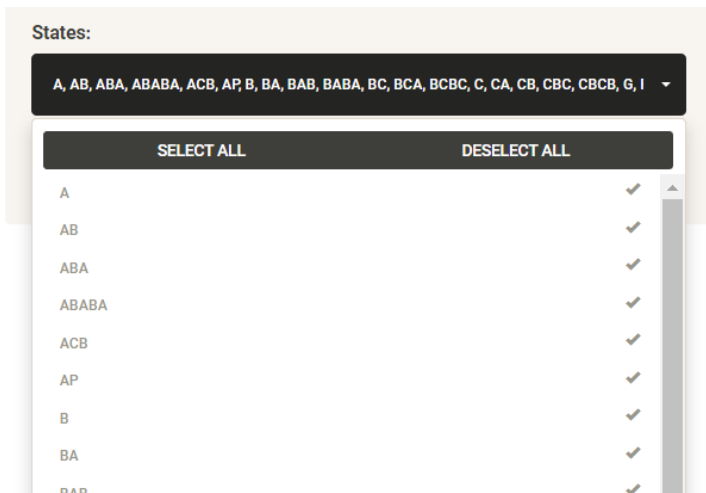
- o Copy Tag number and investigate in the "All Encounters" or "Movements" tab

Show 10 ˅ entries

Date	weeks_since	TAG	State	det_type	ReleaseSite	Species
	All	All			Al	
2021-05-14	36	230000272182	ABABA	Hitching Post Stationary Antenna	Chimney Rock Below Island	LOC

Showing 1 to 1 of 1 entries

States and Days Wide

This is the same data displayed in States_dataframe, but in wide format where each unique tag gets a row and each column is a day since the beginning of the study. This may have a bit more rows than the release file because there are some fish in there without release info, but should typically have the same

Unknown States

This is where events that were unable to be captured by states are displayed. This hopefully should be pretty small…it's filled with tags with detections before official 'Release' such as in in May 2021 and tags without release info. Mainly tags without an idea where they came from. But if a Tag shows up where release info is known, might have to go into the get_states_function.R code to make another case_when entry to account for the new state. All in all, this is a check to see how well the get_states_function is working, and also to see if fish without release info have gotten by the other filters.

Daily Movements Map, Plots, and Data

Displays data of "movements" (Movement_table_no_transitions from Get_movements_function), where a fish has changed position along the river.

"Movements" Defined

Movements are inferred by encounters of the same tag on different antennas (including mobile runs) or recaptures throughout the river. Each detection along the river is assigned a "station", breaking up the centerline of the stream into 10 m increments. The stations start at 0 m, starting about 150 m below the Sherriff Ranch Fry Site. They extend to 13480 m on the Upper Colorado River, where the mobile run begins, and to 22850 m (10120 m at confluence + 12730 m from confluence to Fraser River canyon) on the Fraser River (5/23/2024).

These stations, which are made in GIS from stream centerlines, are joined by coordinates to a dataset of all relevant detections for each unique UTM, using the Spatial_Join_function. The resulting dataset is passed back to a larger dataset of all detections to assign stations to all UTMs. From there, the data is condensed down to a dataset of daily fish detections, keeping the first and last detections of the day and one detection in between. Wacko fish/avian predation that hit multiple different antenna in between the first and last detection of the day won't display all of these detections. Example: tag 230000272182. These CAN be seen in the states and All Events dataframes though.

Once there is a dataset of daily movements, the distance moved between days is calculated for each fish when a new detection is recorded, calculated by taking the difference between stations. An upstream movement is a positive difference (IE new detection station = 8000, previous detection station 6000, so 8000-6000 > 0; upstream movement). A downstream movement is a negative difference between current station and previous station (new station = 6000 m, old station is 8000 m, -2000 <0; negative movement). No movement is if there is no difference.

The absolute value of total distance moved is summed for each fish and displayed in the movements tab under "sum_dist". It also seen in the Encounter Histories created in the Ind_tag_enc_hist_summary_wide function.

- Filters in the sidebar control what is displayed on the map or plot
  - USGS Data is combined with movements on a *daily* level, contrasted with events in All Encounter Histories where PTdata and USGS data is associated with an event within 13 hours for USGS data and 1 hour for PTdata
  - Like the Environmental Filters and USGS data filters in Encounter Histories Tab, using the filters for Release length, USGS WaterTemp, and USGS Discharge will automatically filter out rows that don't have that data associated with the movements. All the Tiger Musky in the release file for example don't have release length/weight data so they don't show up on the map/table/plots if the release length filter is on

Map and Table

- This tab can be used to answer questions like "what time of year are fish moving?" and is the easiest/funnest way to examine individual tag histories. A method I frequently use is to find fish that have moved a long distance or have weird daily states recorded, copy and paste the tag, and plug it into this tab. Or find an event on the map, click on it, copy the tag number, and plug that in.
- Map is interactive, where you can click and hover to get more info about a movement. Clicking on an event in the map will navigate to the event column in the table, and vice versa.
- Can be used to help find suspect avian predation tags by clicking/sorting on the dist_moved and MetersPerSecondBetweenDetections columns

	MAP AND TABLE	MINICHARTS MAP	MOVEMENT GRAPHS

	dist_moved	MPerSecondBetweenDetections	sum_dist
	A	All	
Antenna	-1030	-0.006507126250884464	1030
Antenna	0	0	1030
ry Antenna	-4790	-0.02201701607380067	4790

- Layer control is available in top right, controlling detections, antennas, Release Sites, Stream centerlines, Stations that were used to make these movements, and mobile reaches. Note that loading the Stations takes about 10 seconds and can also slow map performance.
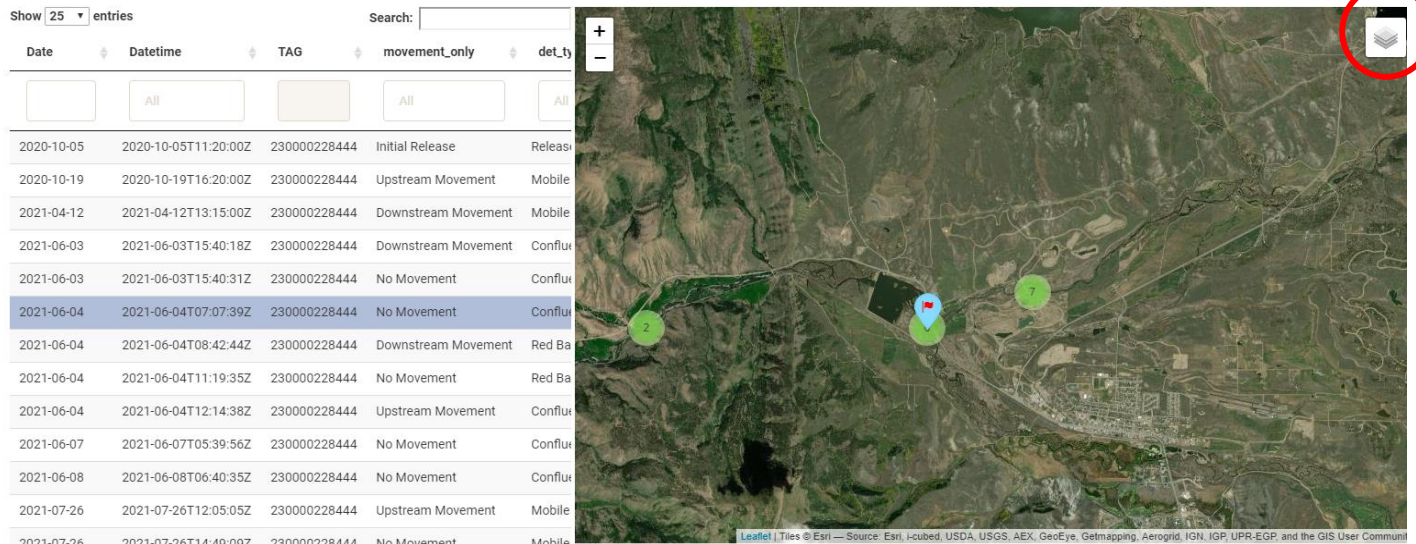
Plots

- The plots are also controlled by the sidebar filters
- Fish Movement by Day: simple histogram representation of movements across time, colored by movement type
  - Note: data can be misleading if mobile detections aren't filtered out, as these cause big spikes in movement activity the day of the mobile run. Activity is also generally higher after release events as well.
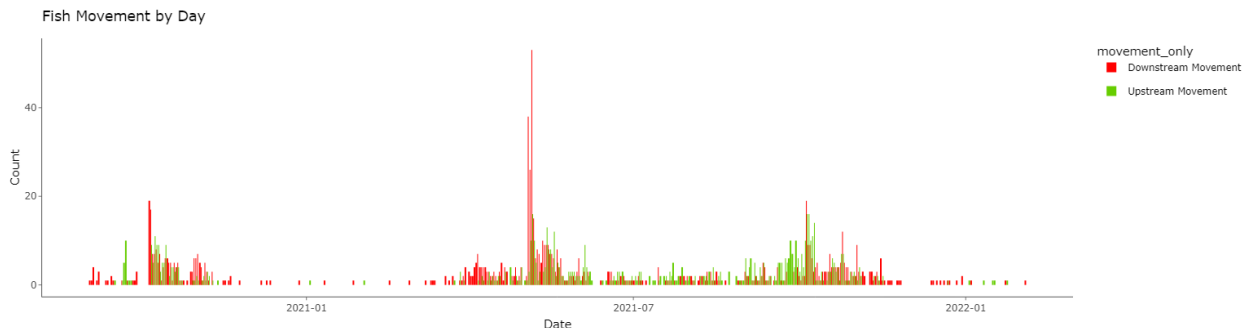
- Seasonal Daily Movements: Same plot as above but grouped by day across the years, which is helpful in seeing seasonal movement trends (big spikes here are probably mobile runs)
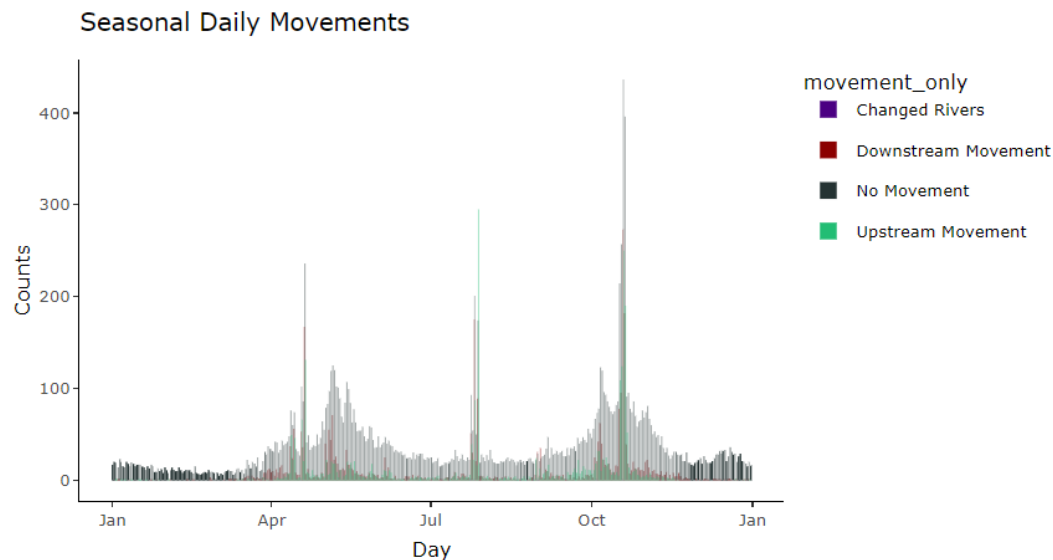
- Individual Movements: showing all the times a fish has moved (upstream, downstream or changed rivers) upstream and changed rivers are positive integers, downstream is negative. Note the spikes around 2200m due to fish travelling between red barn and hitching post, a distance of 2190m according to the stations.
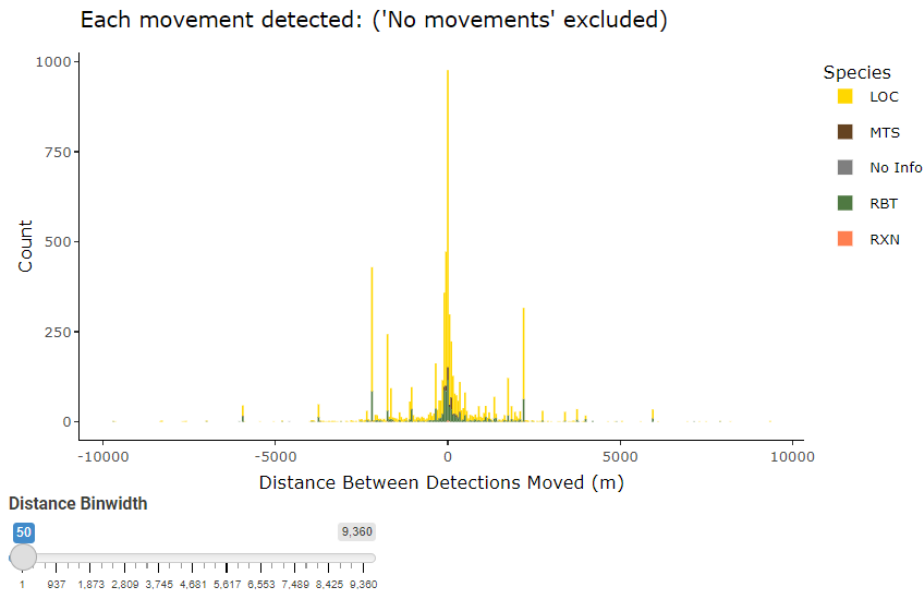
- Cumulative Movement: the amount each fish has moved in absolute distance, in meters, in total over the course of its life.
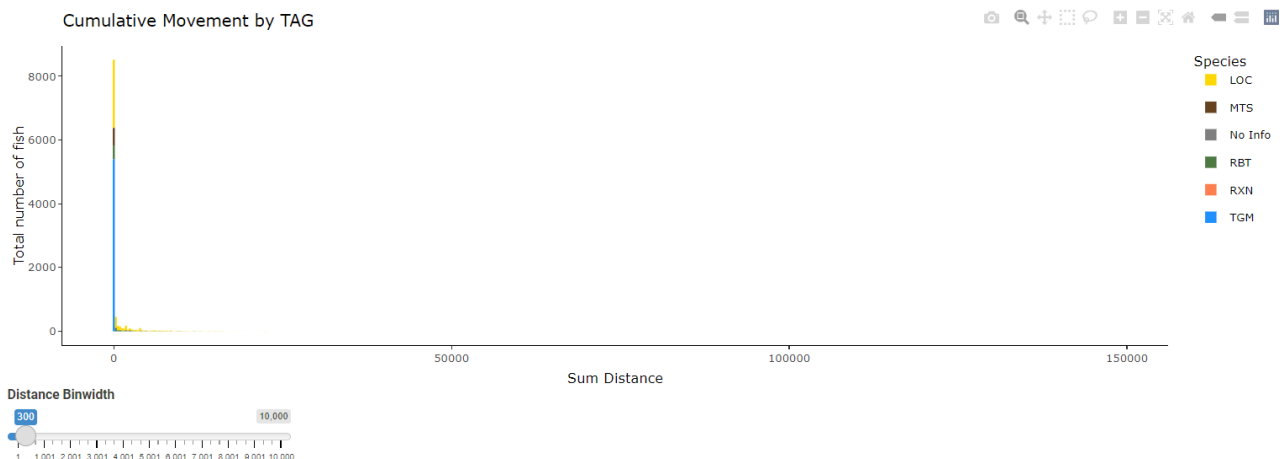
- Movement by hour: displays how many movements have been recorded during each hour of the day. Can be skewed towards hour 0 (12:00 AM) especially if a fish is hanging out on an antenna, since the movements df take the first and last activity of a fish during the day as well as unique events in between.
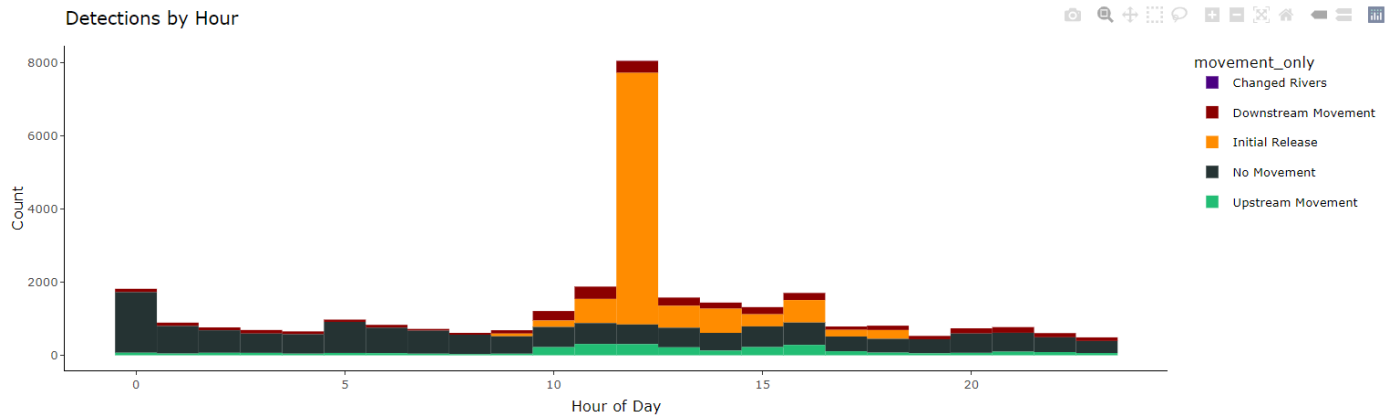
Detections by Hour

Data may be misleading; Movements are calculated by the day and if a fish has multiple detections on the same day at the same antenna, the first detection will be used which can fall commonly at hour 0

- o If "No Movements" and "initial release" are filtered out from movement type, and "mobile detections", "release" and "recaptures" are filtered from movement types, you get movements registered by antennas in the river from fish that are not hanging out on antennas for weeks at a time.

Detections by Hour
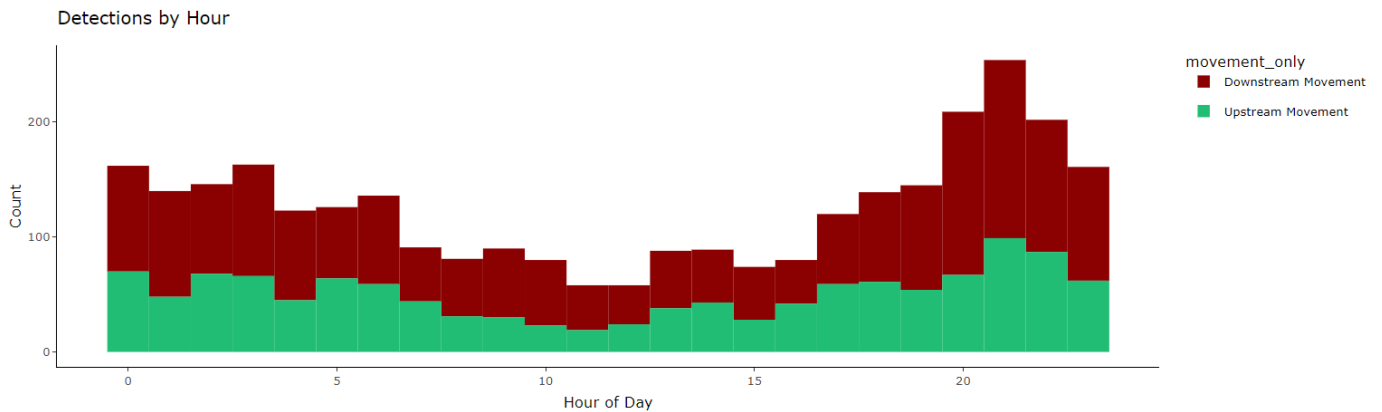
- The animation tab lets you render an amination of filtered data from the left sidebar. Click "render" in the left sidebar before clicking the "render animation" button. There is a progress bar in the Rstudio terminal that shows if you minimize the map. It automatically saves a .gif in the parent directory called WindyGapFishMovements.gif

Pressure Transducer, USGS and Detection Distance

This tab shows graphical representation and overlays of data from Pressure Transducers, USGS and Detection Distance from Site Visits.

- Water_level_NoIce is the best variable for water level. Ice readings/observations are displayed as 0.
- USGS data is all from the gauge at Hitching Post. It can be misleading if you select a site and see discharge, but it's all from the same gauge.

Time Series

PT data with the option to overlay USGS data. You can select variables to plot and which Y axis to display it on
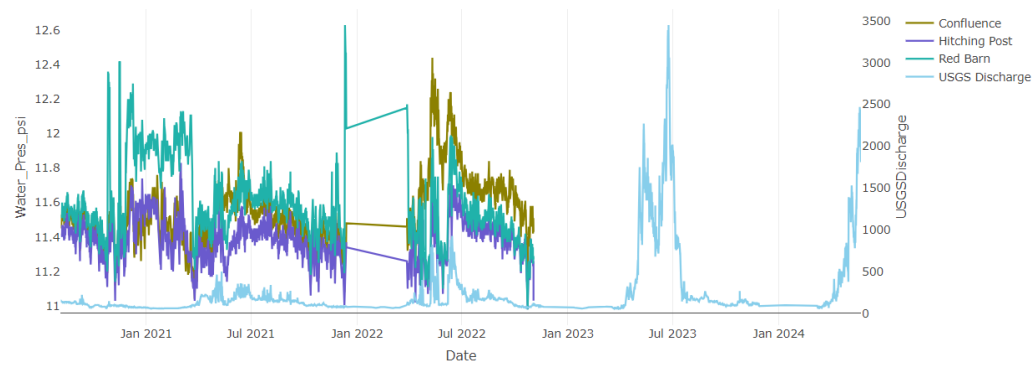
Movements Overlay

PT/USGS data overlaid with the same movements df displayed and used in the movements tab. All the same filters are available for movements, and the graph used is the Fish Movement by Day histogram of movements across time. Use the tabs in the sidebar to get to different data filters.
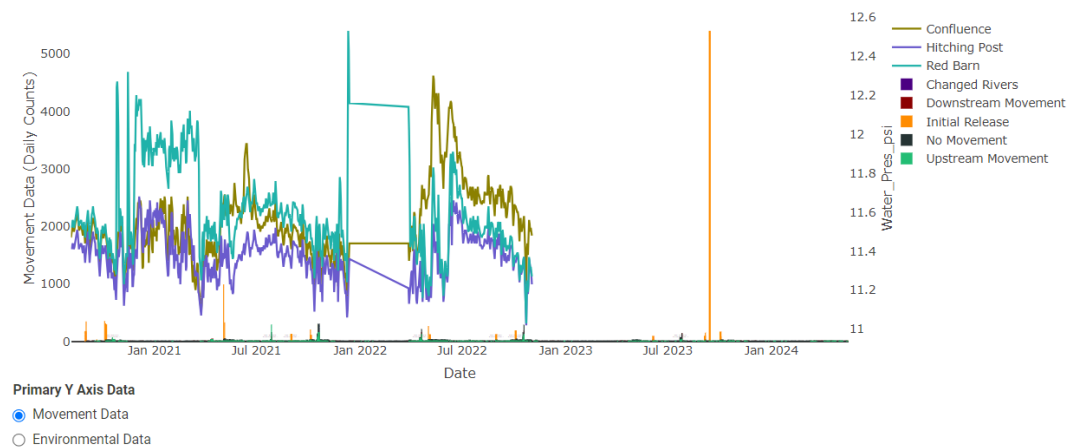
Variable Correlations

Plot PT/USGS variables against each other. "Summarize by timeframe" is an option used to group data by specified timeframe and can get rid of daily fluctuations. R-squared value in the left graph corner is calculated and found with this function in R: rquared = summary(lm(variableX ~ variableY))$r.squared
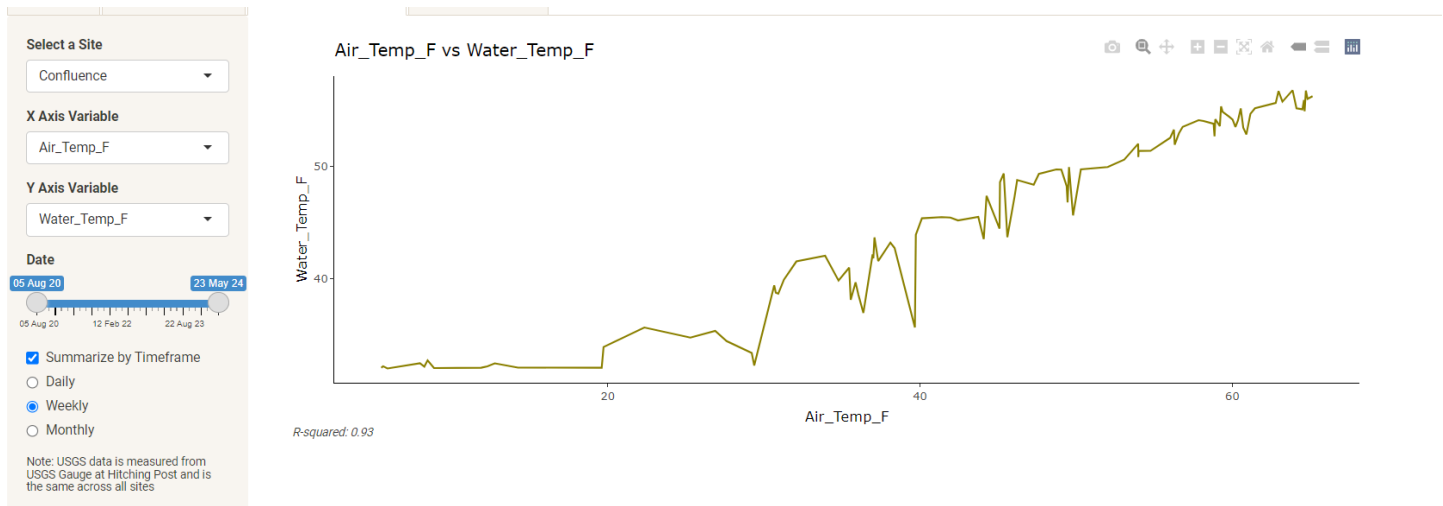
Detection Distance

Detection distance from site visits of Biomark and Staionary Sites with PT data overlaid. You can select which reading you'd like to plot from the sidebar. You can change the opacity of the lines for preferred representation as well.
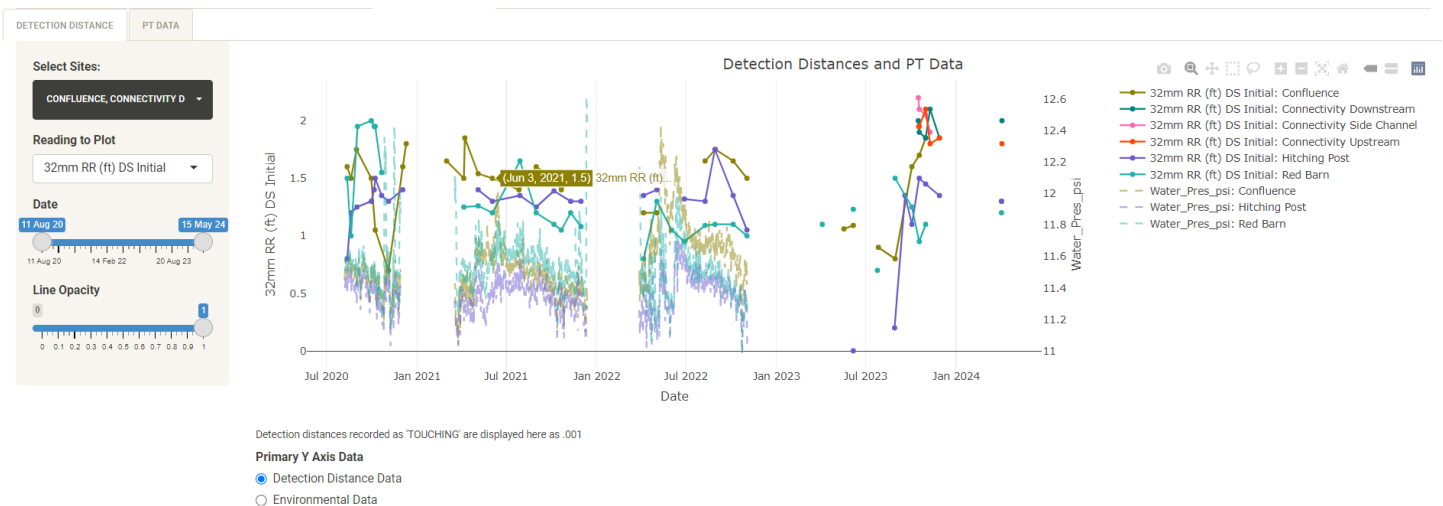
QAQC Tab

Used to help verify data.

Marker Tags

Stationary and Biomark Marker Tag Data set from All_Combined_events_function

- Helpful for finding periods of inconsistent marker tag detections, indicating antenna was not functioning properly during these time.
- In this case, there was a gap: closer examination shows there were not detections between May 21 2022 and August 9 2022 at CF5

- These found gaps are manually recorded in the "Marker Tag Issues" tab in the metadata and are displayed in the "Downtime Periods" tab here

- This is a large dataset to plot with a scatterplot, so it will run quickest when plotting a smaller date range with only one marker tag or site selected.
  - A smaller date range will get better resolution in the plot, making it easier to see downtime periods with the tag.
  - What I do to find these periods is examine a small date range of about 2-8 months, select just one site, then select all marker tags and render the plot.
- If the ARR column is brought in incorrectly from stationary data (which can sometimes happen if the stationary ARR column gets corrupted, rarely happens) then there will be an abundance of detections in the 12:00-13:00 range.

Summarized Marker Tag data

- Raw counts of marker tag detections across selected sites for the duration of the study.
- Helpful for seeing which tags have been deployed at which sites and which sites may have had more downtime periods than others

Summarized Marker tag Data for Selected Tags and Sites

Show 10 ⌄ entries

Date filter does not apply to this table

Site_Code	TAG	totalDetectionsSinceProjectInception
All	All	All
CF5	00000000000000001570	100927
CF5	00000000000000004948	81603

Showing 1 to 2 of 2 entries

Previous 1 Next

[SAVE DATA]

Release and Recap Lengths/Weights

- Plot to help ensure that there were no length/weight typos in the release and recapture files.
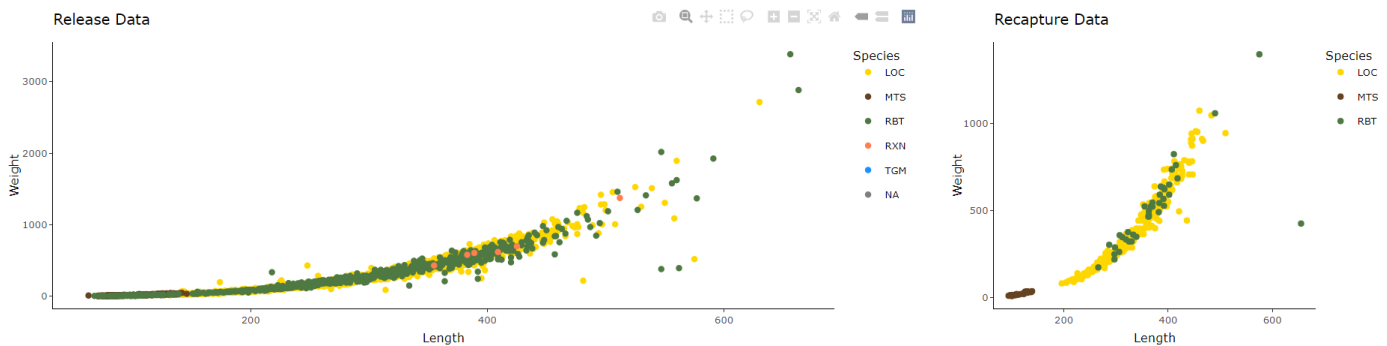
Unknown Tags

- List of Tags without release info but started with 900_ initially and have detections on some sort of antenna
- Display of Enc_hist_wide_summary_function output "Unknown Tags"

Show 200 ⌄ entries

Tags that initially started with 900_ but are not marker tags, test tags, or part of the release file.

TAG	Length	Weight	RB1_n	RB2_n	HF
All					
00000495240232787684					
10240123810058196921					
450115000072106					
00000247620116478391			1		

Ghost Tag Movements

A df made from the "Ghost Tags" csv input. This shows registered ghost tags that have moved >0m after their assigned ghost date. This is helpful in revising the ghost tags file if one erroneously turns out to be a fish.

- Table is automatically sorted by "total_distmovedAfterGhostDate" so the most problematic-looking fish are first displayed

- antennasDetectedAfterGhostDate is a vector of all antennas that have detected the tag after the ghost date. If the tag was identified as a ghost tag somewhere near upstream of a stationary antenna, it isn't uncommon to see the tag be detected on that antenna during a high flow event
  - To get more context, you can check USGS discharge in the "Pressure Transducer, USGS, and Detection Distance" tab, find the discharge associated with that specific detection within 13 hours in "Encounter Histories – All Encounter Histories, or find the daily discharge associated with each daily movement in "Daily Movements" tab – Map and Table – "USGSDischargeDaily" column
- If a fish is moving upstream significantly after its ghost date, it might be worth investigating. This is seen by sorting/clicking the "maxUpstreamDistMovedAfterGhost" column.
- Movements of 10-50m ish between mobile runs can typically be attributed to GPS variation

Ghost Tags To Double Check: each of these tags has moved > 0m since its assigned Ghost Date.

TagID	GhostDate	antennasDetectedAfterGhostDate	total_distmovedAfterGhostDate	maxUpstreamDistMovedAfterGhost	Notes
All	All	All	All	All	All
230000272949	2021-07-28	Mobile Run	230	0	Likely Ghost; Big fish location and probably date of 7-28-21); ghos 26-23
230000142602	2021-07-28	Mobile Run	220	20	confirmed using enco

Crosstalk QAQC

Marker Tag crosstalk, where the tag goes off on one antenna but also registers on the other antenna as well, has been occurring frequently on CD since its installation. This tab is meant to monitor crosstalk instances of *Fish*, where a non-marker tag is registered at both antennas of a site at the exact same time.

Is created from combinedData_df_list$All_Events, originally stemming from All_combined_events_function().

- Summary Table shows the percentage of fish detections across the selected timeframe have been recorded at the exact same time
- Individual site tabs show when those simultaneous detections occurred
- There are instances where the simultaneous detections occurs naturally. In the app, milliseconds are not used, but the data coming off the readers do use milliseconds. Sometimes there are instances where a fish is recorded milliseconds apart on different antennas, but these will show up as the exact same timestamp in the app. However, when CD is doing it far more frequently than the other sites, that probably means something more is going on.

Date

31 Aug 20   14 Jan 21   30 May 21   13 Oct 21   26 Feb 22   12 Jul 22   25 Nov 22   10 Apr 23   24 Aug 23   07 Jan 24   17 May 24

RENDER TABLE

SUMMARY TABLE	RB	HP	CF	CD	CS	CU

Crosstalk Occurrence Percentage

[Copy] [CSV] [Print]

% of FISH detections on each antenna with the exact same timestamp. Detections in raw data may differ by milliseconds, but milliseconds are not used in the app data.

AntennaCodes	PercentageOfDetectionsWithSameTimestamp
RB1, RB2	0.13%
HP3, HP4	0.00%
CF5, CF6	0.00%
CD1, CD2	1.26%
CS1, CS2	
CU1, CU2	0.00%

Showing 1 to 6 of 6 entries
May take a few seconds to load

Detection Distance/Water Level

Shows a color coded table of SiteVisitAndPTData, a joined df of PTdata and Site Visit Data by time to the nearest 13 hours.

- Using the variable "Water_Level_NoIce_ft", this table colors red the 32mm readings that are *below* the observed water level and green *above or at* the observed water level
- Helpful for identifying periods where detections may have been missed because of high flows.
- Readings that were entered as "TOUCHING" are changed here to .001

[Copy] [CSV] [Print] Show [25 ∨] entries   Search: [ ]

Green cells show where 32mm detection distance is greater than or equal to water level, red is where 32mm detection distance is less than water level. Water level readings are within 13 hours of site visit.

Date	Time	Site	Water_Level_NoIce_ft	32mm RR (ft) DS Initial	32mm RR (ft) US Initial	32mm RR (ft) DS Final	32mm RR (ft) US Final	32mm Center (ft) DS Initial	32mm Center (ft) US Initia
All	All	/	All	All	All	All	All	All	All
2023-06-06T00:00:00Z	09:00:00	Hitching Post		0.001	0.7				
2023-08-29T00:00:00Z	14:28:00	Hitching Post		0.2	1.2			0.25	1
2023-07-25T00:00:00Z	16:58:00	Red Barn		0.7	1.2			0.65	1
2020-11-04T00:00:00Z	13:20:00	Confluence	0.88	0.7	1.3	0.9	1.3	0.6	
2023-08-30T00:00:00Z	15:34:00	Confluence		0.8	1.3	0.8		0.75	
2020-08-12T00:00:00Z	11:10:00	Hitching Post	0.95	0.8	1.3	1	1.3	0.9	
2022-04-06T00:00:00Z	10:30:00	Red Barn	1.28	0.8	0.2	1	1.55	0.5	
2023-07-27T00:00:00Z	09:07:00	Confluence		0.9	1.5			1.1	

Adding/Removing Dummy rows in the data

Dummy rows were added to the stationary, biomark, and release data using the dummy_rows.R script to ensure the framework for new antennas was in place.

```
54
55  #### This part was for checking if new antennas to be put in will work
56  source("functions/dummy_rows.R")
57  dummy_rows_list <- add_dummy_rows(stationary = Stationary, biomark = Bio
58  Stationary <- dummy_rows_list$Stationary
59  Biomark <- dummy_rows_list$Biomark
60  Release <- dummy_rows_list$Release
61  #ghost_tag_df <- dummy_rows_list$Ghost_tags #date column is named "Ghost
62
```

The rows remain while the function is ran to set the framework then the rows are removed in the data so this dummy data isn't used. The rows are removed....

- In the get_movements_function

```
 8     movement_table_notrans <- combined_events_stations %>%
 9       #### removing dummy tag
10       filter(!TAG %in% c("230000999999")) %>%
11     select(Date, Datetime, TAG, det_type, Event, FT_STATION, S
```

- In the reactive for all events in app.r:

```
919
920 ▴         }
921         #### filter dummy row
922         all_events_filtered <- all_events_filtered %>%
923           filter(!TAG %in% c("230000999999"))|
924
925
926           return(all_events_filtered)
927 ▴     }) #end of ENC data list eventReactive
928
```

```
30      #daily_unique_events = length(unique(Event))
31      states1 <- wghost_av %>%
32        filter(!TAG %in% c('230000999999')) %>%
33      mutate(
```

- In get_states_function:
- After the functions are ran in the following individual datasets in app.r:

```
151  ### taking dummy tag out
152
153  Biomark <- Biomark %>%
154    filter(!DEC.Tag.ID %in% c("900.230000999999"))
155  Release_05 <- Release_05 %>%
156    filter(!TagID %in% c("230000999999"))
157  Stationary <- Stationary %>%
158    filter(!TAG %in% c("900_230000999999"))
159
160  # Define UI for application that draws a histogram
```

- In Ind_tag_enc_hist_wide_summary function:

```
170
171    #### dummy rows removal: 1/14/23
172    ENC_Release6 <- ENC_Release6 %>%
173      filter(!TAG %in% c("230000999999"))
174
175
```

To put the dummy rows back in, comment out the code that filters out the dummy rows above.

Data Used in the App

The following files are used in the app and are manually derived.

Tabular data: WGFP_dataclean_vis2.0\data
Stationary detections: WGFP_Raw_yyyymmdd.rds

- Combined and cleaned file of all stationary detections. Obtained from the "Combine data RShiny app" found at U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections\CodingDetections\WGFP _CombiningData_RShinyApp.
- Most recent file is found at U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections\All_Stationary

Mobile: WGFP_Mobile_Detect_AllData.csv

- A combined file of all mobile detections found at
  U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections\MobileRaftAntenna\Mobile_Detections
- Usually Eric R prepares this
- Column names and order: "Num", "River" , "MobileSite", "Date", "Time" , "T109_C", "UTM_X", "UTM_Y" ,"TagType" , "TagID", "Event", "Ant", "Pass", "Species" , "Length" , "Weight" , "TagSize" , "RS_Num", "ReleaseSite", "Survey", "Notes"
- Of these, TagID, Date, Time, UTM_X, UTM_Y, and Ant are the most important columns

Biomark: Biomark_Raw_yyyymmdd.csv

- Combined file of all Biomark detections, found at
  U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections\Biomark
- Made from the "Combine New Data RShiny App"
- Column names and order: "Scan.Date" , "Scan.Time" , "Download.Date", "Download.Time", "Reader.ID" , "Antenna.ID",   "HEX.Tag.ID" , "DEC.Tag.ID",   "Temperature.C", "Signal.mV" ,   "Is.Duplicate", "Latitude", "Longitude", "File.Name"
- Most important columns are Reader.ID, Scan.Date, Scan.Time, DEC.Tag.ID

Release: WGFP_ReleaseData_Master_ yyyymmdd.csv

- Master Release file of all tagged fish. Found at
  U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Tagging
- Column names and order: "RS_Num", "River", "ReleaseSite", "Date" , "Time" , "Year", "UTM_X",   "UTM_Y", "Species", "Length" , "Weight", "TagType", "TagID", "QAQC", "TagSize", "Ant" , "Event", "FinClip", "Mortality", "Comments"

Recaptures: WGFP_RecaptureData_Master_ yyyymmdd.csv

- File of all fish that were recaptured found at
  U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Recaptures
- Column names and order: "RS_Num",   "River" , "RecaptureSite" ,"Date", "Time", "UTM_X" , "UTM_Y", "Species" , "Length", "Weight", "TagType", "TagID", "QAQC" ,   "TagSize" , "Ant", "Event", "FinClip", "Mortality" ,   "Comments"

Avian Predation: WGFP_AvianPredation.csv

- csv of tags succumbed to avian predation with a date of predation found at
  "U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections\AvianPredation\WGFP_AvianPredation.csv"
- used in the "States" function to assign a predated state

Ghost Tags: WGFP_GhostTags.csv

- "U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections\GhostTags\WGFP_Ghost Tags.xlsx" – Need to export/save the "Ghost tags" tab to csv
- csv of ghost tags with date of ghost tag
- used in the "States" function to assign a ghost state
- Column names: RS_Num, River, ReleaseSite, ReleaseDate, Species, Length, Weight, TagID, TagSize, Event, GhostDate, UTM_X, UTM_Y, Comments

Metadata: WGFP Metadata.xlsx

- Found in U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\Detections
- Used to keep track of antenna names, UTMs and what they're called in the data, as well as test tags, marker tag downtimes, and important stationing variables.
- Includes tabs:
  - Antenna Metadata: Used to help assign names and UTMs to the antennas
    - AntennaSite: Name of Antenna (ie Red Barn (Upstream))
    - SiteName: Name of Site (ie Red Barn Stationary Antenna). **DO NOT CHANGE THIS FIELD** for an antenna
    - FrontendSiteCode: How we want the shorthand code to display in the app. Needs to be just one entry but can be changed as you wish.
    - BackendSiteCode: code(s) that have been used in the data to denote this antenna. Make sure this code is the same ones that come off the readers: SCD field for Stationary, Reader ID for Biomark. It's ok to have multiple names for the Biomark Antennas separated by comma and a space, but not for Stationary ones
    - PressureTransducerSiteName: Name of the Site as it appears in the Pressure Transducer Data. Must be updated if new sites are added
    - DetectionDistanceSiteName: Name of the site as it appears in Site Visit/Detection Distance Data
    - UTM_X and UTM_Y: UTMs of the site
    - River: River it's deployed on. **DO NOT CHANGE THIS FIELD** for an antenna
    - Deployment Duration: Dates and times where they have been deployed
    - Notes: any notes on the antenna
  - MarkerTagIssues: used to keep track of downtime periods on the antennas. Displayed in the QAQC tab of Marker Tags under "Downtimes"
  - ImportantStationingVariables: Used to keep track of important sites, used for movement calculations within the app
  - TestTags: Keeps track of all test tags and removes these tag detections from the data
  - Notes: Keeps track of general metadata in the worksheet. Current notes: (5/24/2024)
    - Assumes that for AntennaMetadata, SiteName and River will not be changing. If they do change, you'll have to go into the runscript and ac couple functions including "PrepareforMovementsStatesand Summaries" to change how those variables are located
    - For the biomark antennas, I don't think you can actually code the readers to detect as the frontend codes; only codes like A4,B2 etc…hence the need for frontend/backend codes
    - For antenna metadatea codes, the frontend code should only have 1 entry. It's ok for the biomark backend codes to have multiple entries, but not the stationary ones
    - For marker color to be assigned correctly, need to have "Biomark Antenna", "Stationary Antenna", and "Mobile Run" in SiteName
    - First data date: 2020-08-06. detections before this date are removed
    - PressureTransducerSiteName needs to line up with the sites that are in the Pressuretransducer data

Pressure Transducer: WGFP_PressureTransducer_sitename.csv

- U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\Pressure_Transducer
- Cleaned Pressure Transducer files from each site in csv form are all put into \WGFP_dataclean_vis2.0\data\PressureTransducer
- **ALL FILES MUST HAVE SAME COLUMN NAMES/ORDER**

Site Visits: WGFP_SiteVisits_FieldData.xlsx

- U:\Projects\Colorado_River\Windy_Gap_FishMovementStudy\Data\RFID\SiteVisits
- 2 relevant sheets, Stationary and Biomark used for detection distances.

Antenna_sites1.shp: point file of antenna sites locations

Stream_Centerline_Post.shp: centerline of the Fraser, Upper Colorado, and Colorado River below windy gap

ReleaseSites2021.shp: point file of release site locations

mobile_reaches.shp: mobile reaches extents

simpleStations.rds: stream centerlines broken into 10 meter sections. Originally from Stations_10m_Post.shp, but simplified to 10% of original resolution to improve map loading efficiency (see

Updating the App

The app data is updated by CreateFlatFiles_Runscript.R, an RMarkdown (.rmd) file that takes files (see Data Used in the App), wrangles the data with a variety of functions, and saves them as .rds files in \WGFP_dataclean_vis2.0\data\flatFilesforApp. This wrangling is done outside the app to reduce load time and computations within the actual app as much as possible. Anytime there is new data for any of these files, the runscript must be ran in order to update the data in the app.

The typical workflow for updating the app for new data is below, following the use of UpdateDataApp to combine new detections with the previous detection files.

Raw stationary data output saved from updateData app in .rds format goes in the data directory of the data vis app

Open R project file within the data vis app directory

Navigate to, open, and Run or Knit CreateFlatFiles_ Runscript.rmd

Navigate to, open, and run Data Visualization app

- The resulting dataframe is a cleaned file of stationary data
- Biomark process still uses .csv
- 2023 example Stationary file size of .rds file vs .csv: ~13 mb vs ~157 mb

- Uses tabular files outlined in the "Data used in the app" section (File Readins)
- Creates static files used in the data visualization app by cleaning and combining with other files in "data" directory (Creating New Flat Files/Running the Script)
- Outputs .rds files into the flatFilesforApp folder
- Typically takes a little less than 5 minutes
- Needs to be run anytime there is a change to a data file to get the most up to date data in the app; this includes files in the "data" directory and updated spatial .shp files in the "gis" directory

- App loading time cut down by at least 3 minutes
- Output should be consistent without the need to clean the files within the app
- Check to make sure the expected up to date data is included

File Readins

Anytime there is new data of any of the files ending in yyyymmdd listed above, the name of the file must be updated in the Runscript.

Column names and order matter. When updating the csv or excel files, make sure the new file has the same column name/order as the old one.*

Column names outlined below are how they appear when brought into R at the beginning of the app. In Excel, names will appear slightly differently.

If you do need to change the column names or order, you'll have to go into the app and follow where that column is used in the app and change the name

Make sure tag is read in correctly. The way to do this is by saving the Tag column as numeric, instead of general (default). If not, the runscript won't run all the way through (see Common Errors)

Updating file names

To update the file name, change these dates/names for the appropriate file.

```
Stationary <- readRDS("./data/WGFP_Raw_20240514.rds")

Mobile <- read.csv("./data/WGFP_Mobile_Detect_AllData.csv" , colClasses= c(rep
rep("character", 3)))

Biomark <- read.csv("./data/Biomark_Raw_20221102.csv", dec = ",")

# need to have tagID as a numeric field in the .csv file in order to be read i

Release <- read.csv("./data/WGFP_ReleaseData_Master_20240520.csv", na.strings
colClasses=c(rep("character",8), "numeric", "numeric",rep("character",8)))
```

Other Data Updates

- Pressure Transducer: overwrite the current file with the new one. If a new site is added, just add that to the "data/PressureTransducer" directory. No need to change anything in the app, it will automatically get read in and combined with the other site data **as long as it has the same columns/order as the other data**
- USGS data: scraped from the USGS server and is updated when the runscript runs.
- All other files: overwrite the previous file, making sure column names/order were the same as the previous one

Creating New Flat Files/Running the Script

To run the script and update the data in the app, click "Knit"

```
IkMod.R* ×    CreateFlatFiles_Runscript.Rmd* ×    standardizeDates.R ×    All_Combined_even

          ☐ Knit on Save   ABC ⌕       Knit    ⚙ ▾

Source   Visual

14   whole_doc_startTime <- Sys.time()
15   start_time <- Sys.time()
16   print("Reading in Stationary, Mobile, Biomark, Release, an
17
18   #functions
19 ▾ for (i in list.files("./functions/")) {
20 ▾   if (grepl(".R", i)) {
```
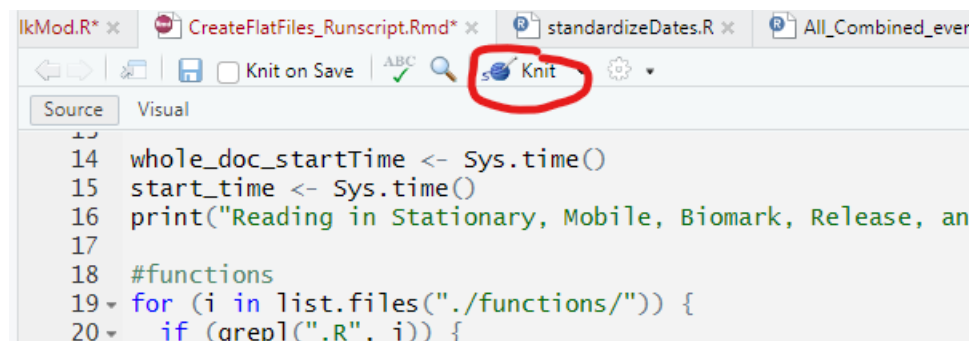
Progress will show in the "Render" window

```
     .../WGFP_dataclean_vis2.0/CreateFlatFiles_Runscript.Rmd

     |......                                    |  13% [readins]

processing file: CreateFlatFiles_Runscript.Rmd
     |............                              |  27% [USGSData]
```
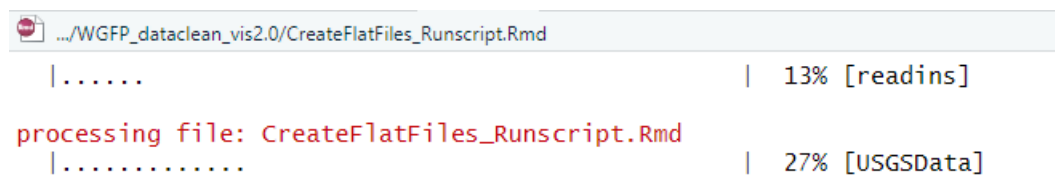
At the end, the document will output a HTML document just to confirm that the data was saved. On a local machine this usually takes 3-4 minutes, sometimes up to 10 or more when using a vpn or directly on the U drive. Saving the files takes particularly long on the U drive.

Create Files for App Runscript

Sam Graf

2023-07-06

```
## [1] "Reading in USGS Data took 13.65"
```

```
## [1] "Running All_combined_events_function: Combining and cleaning Stationary, Mobile, Biomark, Release, and Rec
apture csv inputs......"
## [1] "All_combined_events_function took 1.13 minutes"
```

```
## [1] "Running spatial_join_stations_detections function: Joining detections and events to stations shapefile."
## [1] "Spatial_join_stations_detections took 0.03 minutes"
```

```
## [1] "Running States Function: Assigns letters A, B, C, or G based on position relative to dam, or Ghost/predate
d tag."
## [1] "States Function took 0.26 minutes"
```

```
## [1] "Running Ind_tag_enc_hist_wide_summary_function: Summarizes detection and movement data from each released
Tag."
## [1] "Encounter Histories Summary Wide Function took 0.06 minutes"
```

```
## [1] "Running get_movements_function: Calculates movements of fish based off a change in station."
## [1] "Movements Function took 0.34 minutes"
```

```
## [1] "Saving Files took 0.76 minutes"
```

The whole document took 3.27 minutes to run. Flat Files are saved to the flat files directory and the data vis app is ready to run with the most updated data.

There is some QAQC that happens in this runscript as well, including checking if Tags have multiple entries in the Avian Predation, ghost tag, and release files. If they do, it will show up in the final HTML doc. This also catches if tags were saved incorrectly (see Common Errors).

The resulting flat/static files are saved to data/flatFilesforApp and are automatically used in the app, it's ready to use (see How to Open)!

Common errors:

Warning: cannot open file './data/Biomar_Raw_20221102.csv': No such file or directoryError in file(file, "rt") : cannot open the connection.

- This usually means the filename is spelled wrong or is in the wrong directory

"Error in scan(file = file, what = what, sep = sep, quote = quote, dec = dec, : scan() expected 'a real', got a ___"

- This typically means that there is an incorrect number of columns in the data compared to what the code was expecting. Check the code where the problem file is being read in to see which column types it expects to find in each place, and/or modify the excel file. If you aren't sure what a chunk of code does, it's helpful to copy and paste it into chatgpt and it will explain it to you.

Runscript is not able to run all the way through and the console spits out that there are multiple tags of the same name (2.3E11, 2.26E11)

- Annoyingly, the tags in one of the csvs didn't save correctly after being opened and closed in excel. In order to get them to save correctly, the best way is to open the problem file in excel, select the TAG column, change the type from General to Numeric, and resave it. You may have to go grab the file from the original location on the U drive again is the tag numbers have already been saved as 230000000000 instead of 230000678123

USGS not being read in

- I can't remember exactly what error this throws but it's something related to server connection. This happens sometimes when the USGS server is not working well and so we're unable to scrape the most recent metrics from their site. Sometimes it helps to not use a VPN, but mainly it's not something we can fix on our end and typically this resolves within a few hours. Otherwise, try to run the script tomorrow.

This section of the app adds dummy rows with fake data from CD, CU, B5, and B6. It was added to make sure the functionality for these new antennas was working, and is not currently in use.

```
#### Tilis part was for Checking if new antennas to be put in
source("functions/dummy_rows.R")
dummy_rows_list <- add_dummy_rows(stationary = Stationary, bi
Stationary <- dummy_rows_list$Stationary
Biomark <- dummy_rows_list$Biomark
Release <- dummy_rows_list$Release
ghost_tag_df <- dummy_rows_list$Ghost_tags #date column is na
```

Modifying, Updating, Adding New Antenna/Stations

There are 2 parts to adding a new antenna to the app: updating the spatial files and updating the code that wrangles the tabular data. You can just update the tabular data code without updating the spatial file if you want.

Updating the code and data that wrangles the tabular data

Metadata

- Add new antenna and its data in with WGFP Antenna Metadata (See Data in app)

AntennaSite	SiteName	FrontendSiteCode	BackendSiteCode	PressureTransducerSiteName	DetectionDistance	UTM_X	UTM_Y	River	DeploymentDur
Biomark test Antenna in random spot	Biomark test Antenna	T1	A5			420727	4437229	Fraser River	

CreateFlatFilesRunscript

Metadata Variable names

- In CreateFlatFilesRunscript, add the variable names to the list in the "metadatavariables" code chunk
  - Do this for a backend variable name and a frontend one. "AntennaSite" column in the metadata is used to identify these codes, so change the code to match what is in AntennaSite

```
```{r metadatavariables, include=FALSE, echo = FALSE}
###Variables that are used throughout the funcitons
###Frontend (display) codes
TestBiomarkAntennaFrontendSiteCode <-
as.character(wgfpMetadata$AntennaMetadata[!is.na(wgfpMetadata$AntennaMetadata$AntennaSite) &
wgfpMetadata$AntennaMetadata$AntennaSite == "Biomark test Antenna in random spot", "FrontendSiteCode"])

WindyGapBypassAntennaFrontendSiteCode <-
as.character(wgfpMetadata$AntennaMetadata[!is.na(wgfpMetadata$AntennaMetadata$AntennaSite) &
wgfpMetadata$AntennaMetadata$AntennaSite == "Windy Gap Bypass Channel", "FrontendSiteCode"])
```

- Add these variables to metaDataVariableNames

```
metaDataVariableNames <- list(
  "TestBiomarkAntennaBackendSiteCode" = TestBiomarkAntennaBackendSiteCode,
  "TestBiomarkAntennaFrontendSiteCode" = TestBiomarkAntennaFrontendSiteCode,
  "WindyGapBypassAntennaFrontendSiteCode" = WindyGapBypassAntennaFrontendSiteCode,
  "WindyGapAuxiliaryAntennaFrontendSiteCode" = WindyGapAuxiliaryAntennaFrontendSiteCod
  "GranbyDiversionAntennaFrontendSiteCode" = GranbyDiversionAntennaFrontendSiteCode,
  "RiverRunAntennaFrontendSiteCode" = RiverRunAntennaFrontendSiteCode,
  "FraserRiverCanyonAntennaFrontendSiteCode" = FraserRiverCanyonAntennaFrontendSiteCod
```

Dummy Rows

If the antenna doesn't have any new data yet, you will need to set up the functions to account for that antenna. This is done by using the function dummy_rows.r

- Navigate to dummy_rows.r (in the functions directory) and add a row to the biomark or stationary data depending on which antenna type we want to add. The only columns that are important are adding the dumy tag

number (230000999999) and making sure we've got the right reader ID that lines up with what we put in the metadata

```
Biomark <- biomark1 %>%
  add_row(Scan.Date = "2022-12-07",Scan.Time = "12:52:10.810", Download.Date = "9/16/2022",Download.Time = "11:30:41",
          Reader.ID = "A3",Antenna.ID = 1,HEX.Tag.ID = "384.358D14F739",
          DEC.Tag.ID = "900.230000999999",Temperature.C = NA,Signal.mV= NA,Is.Duplicate= "Yes",Latitude = NA,Longitude= NA,
  add_row(Scan.Date = "2022-12-08",Scan.Time = "12:52:10.810", Download.Date = "9/16/2022",Download.Time = "11:30:41",
          Reader.ID = "A4",Antenna.ID = 1,HEX.Tag.ID = "384.358D14F739",
          DEC.Tag.ID = "900.230000999999",Temperature.C = NA,Signal.mV= NA,Is.Duplicate= "Yes",Latitude = NA,Longitude= NA,
  add_row(Scan.Date = "2022-12-09",Scan.Time = "12:52:10.810", Download.Date = "9/16/2022",Download.Time = "11:30:41",
          Reader.ID = "A5",Antenna.ID = 1,HEX.Tag.ID = "384.358D14F739",
          DEC.Tag.ID = "900.230000999999",Temperature.C = NA,Signal.mV= NA,Is.Duplicate= "Yes",Latitude = NA,Longitude= NA,
```

- Comment out the lines of code that remove the dummy tag from the data (add them back in when there is actual data for the antenna)
    - In Runscript:

```
355
356   ### taking dummy tag out
357
358   Biomark <- Biomark #%>%
359     #filter(!DEC.Tag.ID %in% c("900.230000999999"))
360   cleanedRelease <- cleanedRelease #%>%
361     #filter(!TagID %in% c("230000999999"))
362   Stationary <- Cleaned_Stationary_FishdetectionsOnly #%>%
363     #filter(!TAG %in% c("900230000999999"))
364
```

    - In "get_statesFunction2.r"

```
states <- eventsWithGhostDatesAndAvianPredation %>%
  #filter(!TAG %in% c('230000999999')) %>%
  mutate(
    #the case_whens apply with priority, so it's impo
    state = case_when(Date >= GhostDate ~ "G",
```

    - In "get_movements_function"

```
5
6       dailyMovementsTable <- combined_events_stations %>%
7         #### removing dummy tag
8         #filter(!TAG %in% c("230000999999")) %>%
9         select(Date, Datetime, TAG, det_type, Event, ET_STATION
```

    - In Ind_tag_enc_hist_wide_summary_function.r:

```
149
150    #### dummy rows removal: |
151    encountersAndRelease6 <- encountersAndRelease6# %>%
152      #filter(!TAG %in% c("230000999999"))
153
```

    - In AllEncountersMod, twice

```
420
421              #### filter dummy row
422              all_events_filtered <- all_events_filtered #%>%
423                #filter(!TAG %in% c("230000999999"))
424
```

```
464 ▾         output$alleventsfrequencies1 <- renderDataTable({
465           frequenciesSummarized <- all_events_data() %>%
466             #filter(!TAG %in% c(230000999999)) %>%
467             count(Event  name = "Raw Detections")
```

All_combined_events function
- In this function, add the following line to the case_when() function based off the Frontend and Backend variable codes

```
12    biomarkCleaned <- Biomark %>%
13      mutate(TAG = str_replace(DEC.Tag.ID, "\\.", ""),
14            # i wish this could be a join, but when there are 2 dif codes (A1, B1, etc) used for backend na
15            Reader.ID = case_when(
16              Reader.ID %in% TestBiomarkAntennaBackendSiteCode ~ TestBiomarkAntennaFrontendSiteCode,
17              Reader.ID %in% WindyGapBypassAntennaBackendSiteCode ~ WindyGapBypassAntennaFrontendSiteCode,
```

Ind_tag_hist_summary_wide_function:

- Add in your antennaFrontend Variable in column order

```
18    #column order is just nice to have for the user
19    columnOrder <- c(RedBarnFrontendCodes, HitchingPostFront
20                MobileRunFrontendCodes, WindyGapBypassA
21                GranbyDiversionAntennaFrontendSiteCode,
22                TestBiomarkAntennaFrontendSiteCode)
```

- Add the frontend site code to these lines of code. If it's a biomark antenna, add it to "TotalBiomark", and if its a Stationary antenna, add to TotalStatiionary

```
57        mutate(
58          TotalEncounters = rowSums(select(., all_of(c(RedBarnFrontendCodes, HitchingPostFronte
59                                            ConnectivityChannelDownstreamFrontendCod
60                                            ConnectivityChannelUpstreamFrontendCodes
61                                            WindyGapBypassAntennaFrontendSiteCode, W
62                                            RiverRunAntennaFrontendSiteCode, FraserR
63                                            TestBiomarkAntennaFrontendSiteCode))) ==
64          TotalAntennas = rowSums(select(., all_of(c(RedBarnFrontendCodes, HitchingPostFrontend
65                                            ConnectivityChannelDownstreamFrontendCodes
66                                            ConnectivityChannelUpstreamFrontendCodes,
67                                            WindyGapBypassAntennaFrontendSiteCode, Win
68                                            GranbyDiversionAntennaFrontendSiteCode,
69                                            RiverRunAntennaFrontendSiteCode, FraserRiv
70                                            TestBiomarkAntennaFrontendSiteCode))) == T
71          TotalStationary = rowSums(select(., all_of(c(RedBarnFrontendCodes, HitchingPostFronte
72                                            ConnectivityChannelDownstreamFrontendCod
73                                            ConnectivityChannelUpstreamFrontendCodes
74
75          TotalMobile = rowSums(select(., all_of(MobileRunFrontendCodes)) == TRUE),
76          TotalBiomark = rowSums(select(., all_of(c(WindyGapBypassAntennaFrontendSiteCode, Wind
77                                            GranbyDiversionAntennaFrontendSiteCode,
78                                            RiverRunAntennaFrontendSiteCode, FraserRive
79                                            TestBiomarkAntennaFrontendSiteCode))) == TR
80          TotalRedBarn =rowSums(select(., all_of(RedBarnFrontendCodes)) == TRUE),
81          TotalHitchingPost = rowSums(select(., all_of(HitchingPostFrontendCodes)) == TRUE),
```

With that, you are ready to Run the CreateFlatFilesRunscript. Once the data is updated, run the app. Check to make sure your antenna has been integrated correctly in the data by filtering for the dummy tag in the Ecnounters dataframes, movements, and States tabs where it will also have dummy Release Data.

TAG	Date	Time	Datetime	Event	Species	Release_Length	Release_Weight	ReleaseSite	Release_Date
All						All	All	Al	All
230000999999	2020-09-01	12:00:00	2020-09-01T12:00:00Z	Release	LOC	566	600	Fraser River Ranch	2020-09-01
230000999999	2022-12-07	12:52:10.810	2022-12-07T12:52:10Z	RR1	LOC	566	600	Fraser River Ranch	2020-09-01
230000999999	2022-12-08	12:52:10.810	2022-12-08T12:52:10Z	FC1	LOC	566	600	Fraser River Ranch	2020-09-01
230000999999	2024-05-24	12:52:10.810	2024-05-24T12:52:10Z	T1	LOC	566	600	Fraser River Ranch	2020-09-01

Showing 1 to 4 of 4 entries   Previous 1 Next

Figure 1all Encounters History with Dummy Data

	T1_n	Rec

		A
1	1	

Figure 2 colunm was automatically created in All EncountersHistory SummaryWide for that new antenna

Show 10 ⌄ entries

Date	weeks_since	TAG	State	det_type	ReleaseSite	Species
	All	All			Al	
2020-09-01	0	230000999999	B	Release	Fraser River Ranch	LOC
2022-12-07	118	230000999999	B	River Run Biomark Antenna	Fraser River Ranch	LOC
2024-05-24	194	230000999999	B	Biomark test Antenna	Fraser River Ranch	LOC

Showing 1 to 3 of 3 entries

Figure 3States df with dummy data

Filter by TAG

230000999999

Select Movement Type:

CHANGED RIVERS, DOWNSTREA ⌄

Select Detection Type

CONFLUENCE STATIONARY ANT ⌄

Select Species Type

LOC, MTS, NO INFO, RBT, RXN, T ⌄

Date

31 Aug 20   25 May 24

MAP AND TABLE   MINICHARTS MAP   MOVEMENT GRAPHS

nly	det_type	dist_moved	MPerSecondB
	All	A	All
e	Release		
rs	River Run Biomark Antenna	3820	0.0000
rs	Fraser River Canyon Biomark Antenna	14940	0.
Movement	Test Biomark Antenna	-6090	-0.000

Next

Figure 4 Movements table with dummy data

Adding or updating layers on the map

- In GIS, go to the GIS database and export desired layer as shapefile, saving it in "gis" folder within the app directory
    - Right click on layer in left column, scroll down to data -> export

- Using the sf package, read in the .shp file with the name of your layer

```
8
9   layerLocation <- file.path("./gis/")
0   latLongCRS <- st_crs("+proj=longlat +datum=WGS84 +no_defs") #should be same as +init=epsg:4326
1
2   antenna_sites <- st_transform(read_sf(file.path(layerLocation, "antenna_sites1.shp")), latLongCRS)
```

- The espg:4326 converts the projection to lat/longs able to be plotted with leaflet
- If the layer is very large, it might be good to convert the .shp file to a .rds file, which decreases resolution. For example, the stations file is a .rds file and was converted below. It keeps 10% of the original resolution

```
stations_10m <- readOGR(dsn = layer_location, layer = "stations_10m")
stations_10m <- sp::spTransform(stations_10m, CRS("+init=epsg:4326"))
simple_stations1 <- ms_simplify(stations_10m, keep = .1)
write_rds(simple_stations1, file = file.path(paste0(layer_location,"/"), paste0("simple_stations.rds")))
```

- It is read in like so and is much faster than when read in as a shapefile

```
simple_stations2 <- read_rds(file.path("./gis/simple_stations.rds"))
```

- If you make a new layer (not just updating an existing antenna, stations or stream centerline layer that is already included on the map) that you want to display, you'll have to add it to the code that makes the map, found in movementsMod.R.
- Add_polylines is used to bring in lines like stream_centerline, addMarkers is used for SpatialPoints, addPolygons is used for polygons.
- Add a Group argument to the data and add it in the addLayersControl function to make sure it shows up

```
addAwesomeMarkers(data = releasesites@coords,
                  icon = release_icons,
                  clusterOptions = markerClusterOptions(),
                  label = releasesites@data$ReleaseSit,
                  popup = paste("Release Date1:", releasesites@data$ReleaseDat, "<br>","Release Da
                  group = "Release Sites") %>%
addPolylines(data = simple_stations2,
             label = simple_stations2@data$ET_STATION,
             labelOptions = labelOptions(noHide = T, textOnly = TRUE, style = label_style),
             group = "Stations (m)") %>%
addLayersControl(overlayGroups = c("Antennas", "Detections", "Release Sites", "Stream Centerlines"
hideGroup(c("Stream Centerlines", "Stations (m)", "Antennas", "Release Sites", "Mobile Reaches"))
```

Adding New Antenna to map

The data will still be displayed spatially on the movements map even without an updated antenna file, but if you want to display the antenna on the map (and you should), then here a couple options to do this. The first option is to add it in GIS, then export the layer (see above) and read in that later.

The second option is to modify the already existing Antenna layer within R. This can be done with already-written code in map_polygon_readins.R starting on line 57.

- Modify the UTMs and other relevant fields to fit the antenna.

```
57  #####adding new antenna to data
58
59  utm_x <- 420727
60  utm_y <- 4437229
61  new_point <- st_sfc(st_point(c(utm_x, utm_y)), crs = 32613) # EPSG:32613 is UTM zone 13N
62
63  # Transform the new point to the desired CRS (latLongCRS)
64  new_point_transformed <- st_transform(new_point, st_crs(latLongCRS))
65
66  new_row <- tibble(
67    OBJECTID = NA,
68    SiteName = NA,
69    Objective = NA,
70    StudyPerio = NA,
71    SiteLabel = NA,
72    AntennaNam = NA,
73    WaterName = NA,
74    ChannelWid = NA,
75    UTM_X = utm_x,
76    UTM_Y = utm_y,
77    Notes = NA,
78    geometry = new_point_transformed
79  )
```

- Add this row to the data and save it in the correct location with st_write. This WILL overwrite the previous file.

```
1  antenna_sites1 <- antenna_sites %>%
2    add_row(new_row)
3
4  output_file <- file.path(layerLocation, "antenna_sites1.shp")
5
6  # Write the updated sf object to a new shapefile
7  st_write(antenna_sites, output_file)
```

- Once the new layer is saved, comment out those lines of code from lines 57 to 87 to make sure these lines aren't ran again.

Functions

Shapefile/Polygon Readins

The file "map_polygon_readins.R" reads in all shapefiles for the movements map. Also contains some graphics options for release sites, stationary antenna sites, and labels for stations.

```
#mapping
source("map_polygon_readins.R")
```

.Rds files

.rds files in R are a binary format used to save R objects, such as data frames, lists, or models, preserving their internal structure and attributes. Unlike CSV files, which store data in a plain text format, .rds files maintain the integrity of R-specific data types and structures, allowing for more complex objects to be saved and loaded efficiently. For example, if you have a data frame df that you want to save as an .rds file, you can use the saveRDS(df, "data.rds") function. To load this data back into R, you would use df <- readRDS("data.rds"). Editing an .rds file involves loading the object, making the desired changes, and then saving it again.

Advantages of .rds files in this app

- Easy to save data and reload it again across different R projects without losing any metadata or attributes
  - Helpful when saving Stationary data from CombineDataApp and moving over to this app

- Preserves data types and structures
  - Ensures that when the data is read into R, it preserves the same format, vs a csv which saves data in plaintext, sometimes necessitating type conversion to change the data back to its original structure when opened in Excel. This is currently seen when we open and save data in Excel that have tag numbers, and every time we save/modify it in Excel, we have to make sure the tag number is saved as a numeric type rather than general (see Creating New Flat Files/Running the Script: Common errors). This was also causing issues with the timestamps in the stationary data, leading to the decision to keep the Stationary Data in .rds form.
- Ability to save objects other than DFs
  - This enables us to save lists and spatial files as flat files created in the runscript, reducing computational load within the app for faster speed
- Efficiency
  - .rds files are smaller than csvs of the same data. This leads to quicker read-in times and easier file transfers across projects.

As of late 2023, WGFP_Raw (containing all Stationary Antenna data) has been saved as a .rds file in from the update Data app. The new resultant workflow is as follows as it affects this app.

Raw stationary data output saved from updateData app in .rds format goes in the data directory of the data vis app

Open R project file within the data vis app directory

Navigate to, open, and Run or Knit CreateFlatFiles_ Runscript.rmd

Navigate to, open, and run Data Visualization app

- The resulting dataframe is no different than the dataframe output from the previous process
- Biomark process is unaffected
- Current Stationary file size of .rds file vs .csv: ~13 mb vs ~157 mb

- Creates static files used in the data visualization app by cleaning and combining with other files in "data" directory
- Uses the same files as before in the "data" directory of the data visualization app except the stationary file is a .rds file
- Outputs .rds files into the flatFilesforApp folder
- Typically takes a little less than 5 minutes
- Needs to be run anytime there is a change to a data file to get the most up to date data in the app; this includes files in the "data" directory and updated spatial .shp files in the "gis" directory

- App loading time cut down by at least 3 minutes
- No change to functionality
- Output should be more consistent without the need to clean the files within the app anymore

Shiny Modules

Each tab is split up into its own module. This makes the code reusable (can plug the ui and server into any app easily with the right arguments) and organized. This is where the data processing occurs from the sidebar filters and how graphs and tables are rendered. Read more about Shiny Modules here.

Site, Movement and Species Colors

The colors of the Sites, movements and species used in the app are determined by this chunk of code in the beginning of the app.r file

```
#colors
#rainbow trout color pallete used to assign to Sites:
# "Confluence, Connectivity Downstream, Connectivity Side Channel, Connectivity Upstream, Hitching Post,
# Kaibab Park, Red Barn, Windy Gap Auxiliary, Windy Gap Bypass Channel"
rainbow_trout_colors <- c("#8B8000", "#008080", "#FF69B4", "#FF4500", "#6A5ACD","#32CD32", "#20B2AA", "#FF1C55", "#4682B4
#currently "Changed Rivers", "Downstream Movement" "Initial Release", "No Movement", "Upstream Movement" (5/17/24)
#note: these are a little diffeerent than what we see in the map because the map/marker color optoins are very limited.
movementColors <- c("#4B0082", "#8B0000", "#FF8C00", "#253333", "#22bd74") #, "#66FF00"
# currently "LOC" "MTS" "RBT" "RXN" "TGM" (5/17/24)
speciesColors <- c("#FFD700", "#654321", "#4F7942", "#FF7F50", "#1E90FF", "#008080", "#DAA520", "#D2691E", "#9A5ECD")
```

Rainbow_trout_colors is used to assign colors to site in succession of the colors: so Confluence gets assigned #8b8000, Connectivity Downstream is assigned #008080, etc. When a new site is added to the data, it will automatically get assigned a color based off the rainbow_trout_colors vector, provided that the number of sites is less than the length of rainbow_trout_colors. As of 5/24/2024, there are 9 sites and 15 colors in the vector, so we can add 6 more antenna sites that won't require adjustment to this code.

MovementColors and SpeciesColors are similar to rainbow_trout_colors. If we want to change the colors for a given species/movement etc, then change the colors in this code. Note: changing movementColors does not affect the color shown in the map, just the graphs. Go to get_movements_function.R to edit the movement colors on the map.

Let me know if there are other questions and concerns, I'm happy to add to this if needed. Let me know sfigraf@gmail.com