Wrangling and Preparing PIT Tag Data using PITcleanr

Kevin See, Ryan Kinzer, Mike Ackerman

WDFW
Department of Fish Science

Nez Perce Tribe
Department of Fisheries Resources Management

February 1, 2024

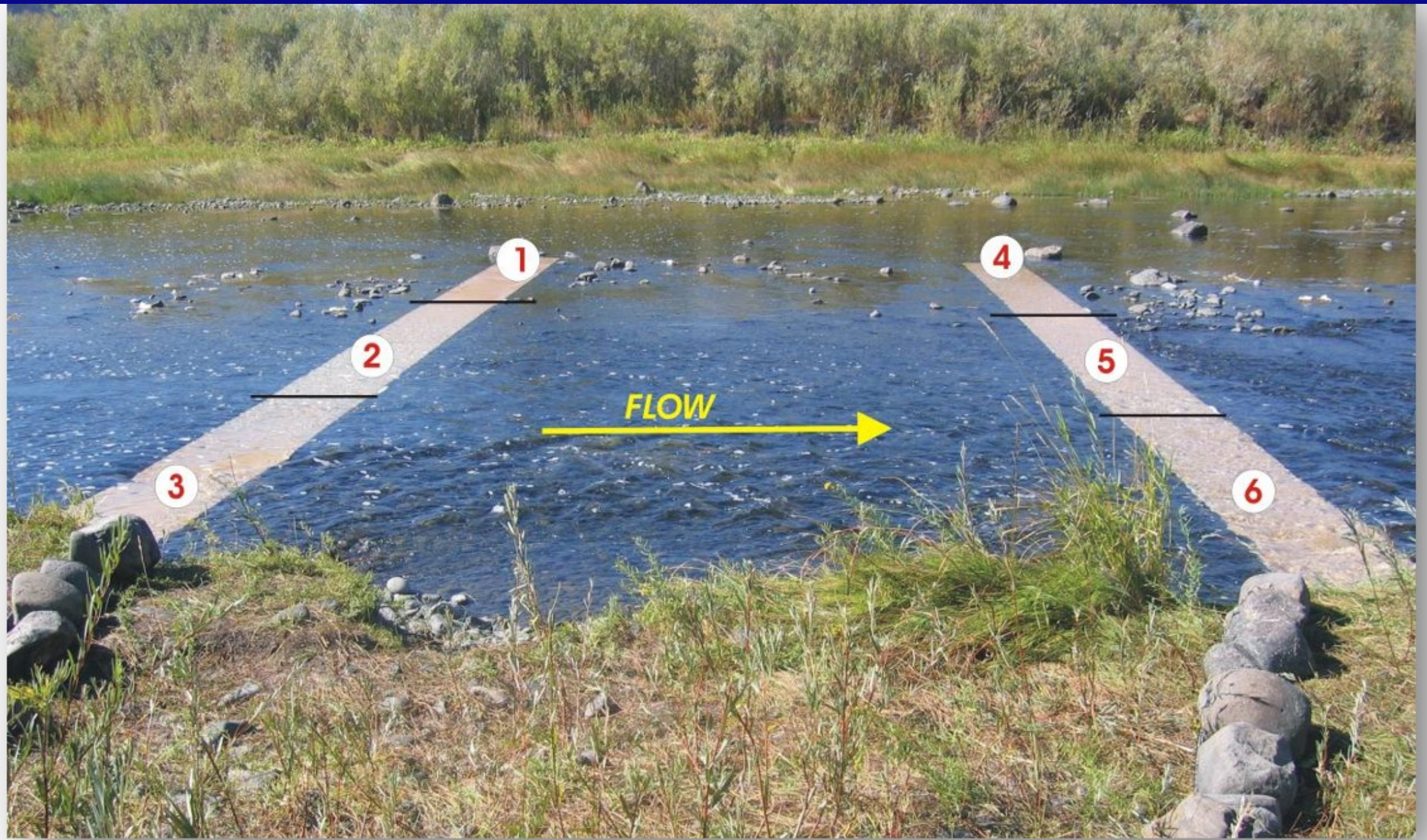
Today's Journey

- Benefits and challenges of PIT tag data
- PITcleanr:

  https://github.com/kevinsee/PITcleanr/

  - Purpose
  - Functionality
  - Package resources
- Summary

PIT Tag Data


1
2
3
4
5
6
FLOW

PIT Tag Data

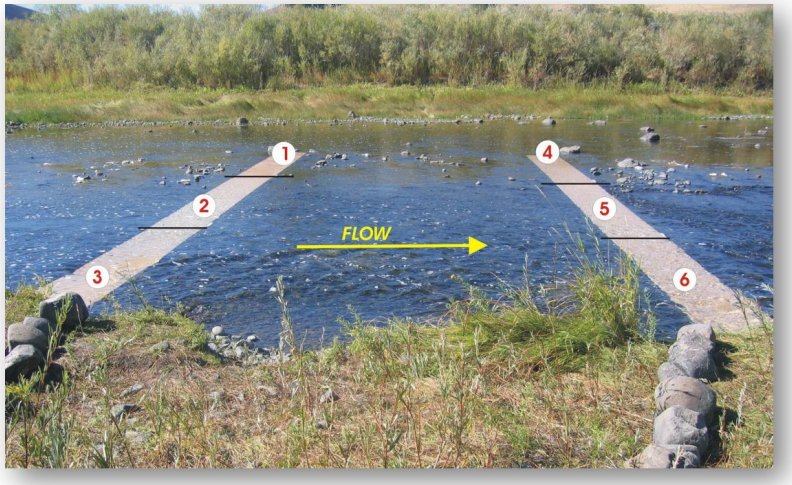
Benefits

Information about individual fish

- movement
- timing
- survival

Challenges

- **TONS** of data
- Can be overwhelming
- E.g. One juvenile fish detected ***681*** times at single site within 12 hour period on Lemhi River



So much data...

```
# the complete tag history file from PTAGIS
ptagis_file ←
  system.file("extdata",
              "TUM_chnk_cth_2018.csv",
              package = "PITcleanr")

# read the complete tag history file into R
observations ←
  readCTH(ptagis_file,
          file_type = "PTAGIS")

n_distinct(observations$tag_code) # unique tag codes
```

```
## [1] 1406
```

```
n_distinct(observations$event_site_code_value) # unique event sites
```

```
## [1] 50
```

```
nrow(observations) # total observations
```

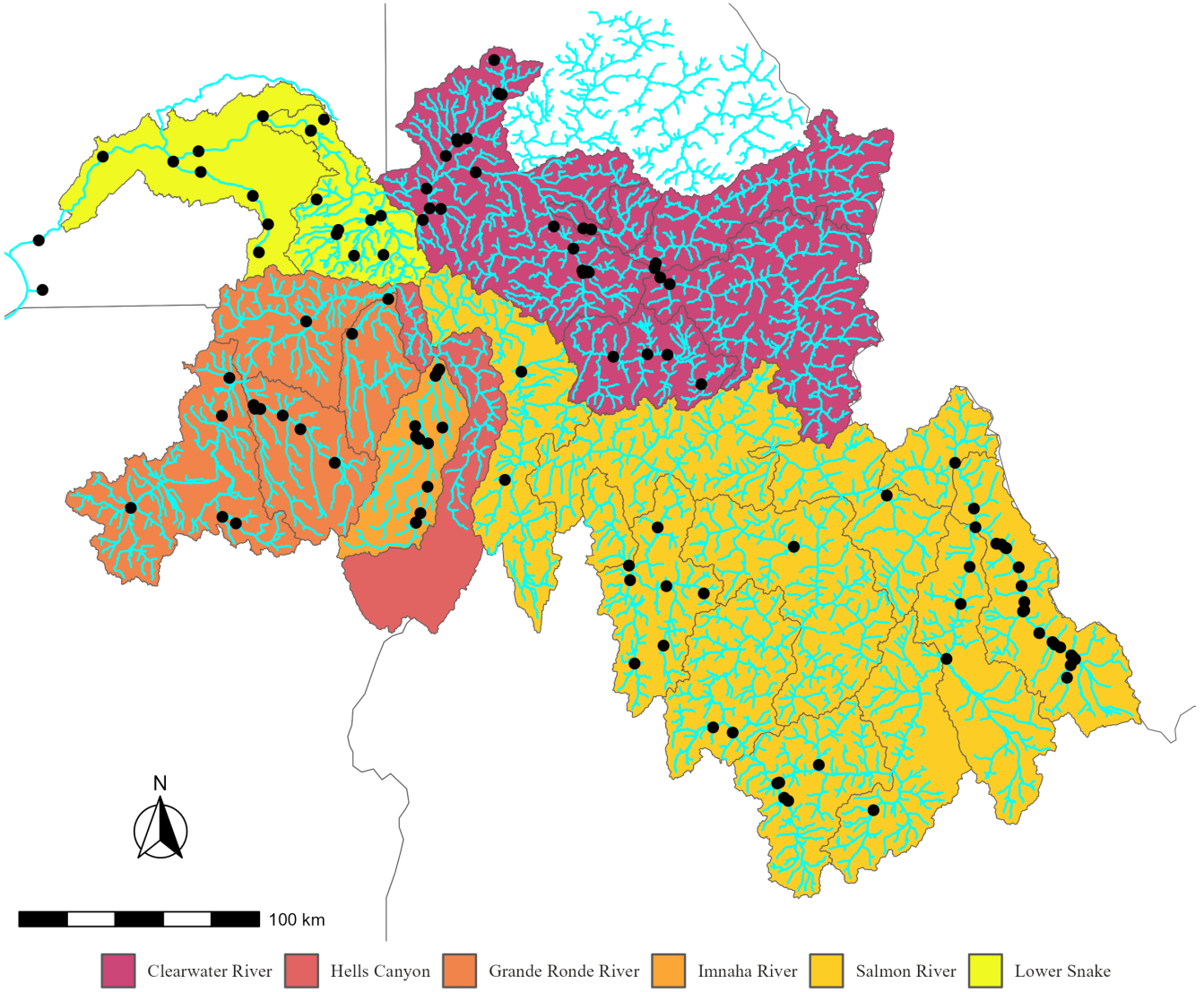
```
## [1] 13501
```

So much data…

Search:

	Tag Code	Event Site Code Value	Event Date Time Value	Antenna Id		Antenna Group Configuration Value	Mark Species Name	Mark Rear Type Name	Event Type Name	Event Site Descript
	All	All	All	All	All	All	All	All	All	All
1	384.3B239AC47B	NASONC	2016-03-01T10:34:00Z		0	Chinook	Wild Fish or Natural Production	Mark	River	NASONC	2016-03-04T17:00:00Z	1
2	384.3B239AC47B	BO3	2018-05-18T09:36:59Z	18	110	Chinook	Wild Fish or Natural Production	Observation	Adult Fishway			1
3	384.3B239AC47B	BO3	2018-05-18T09:37:11Z	16	110	Chinook	Wild Fish or Natural Production	Observation	Adult Fishway			1
4	384.3B239AC47B	BO3	2018-05-18T09:37:12Z	16	110	Chinook	Wild Fish or Natural Production	Observation	Adult Fishway			1
5	384.3B239AC47B	BO3	2018-05-18T09:42:16Z	14	110	Chinook	Wild Fish or Natural Production	Observation	Adult Fishway			1
							Wild Fish or					

Showing 1 to 100 of 100 entries


N
100 km
Clearwater River
Hells Canyon
Grande Ronde River
Imnaha River
Salmon River
Lower Snake

PITcleanr Package

PITcleanr Purpose

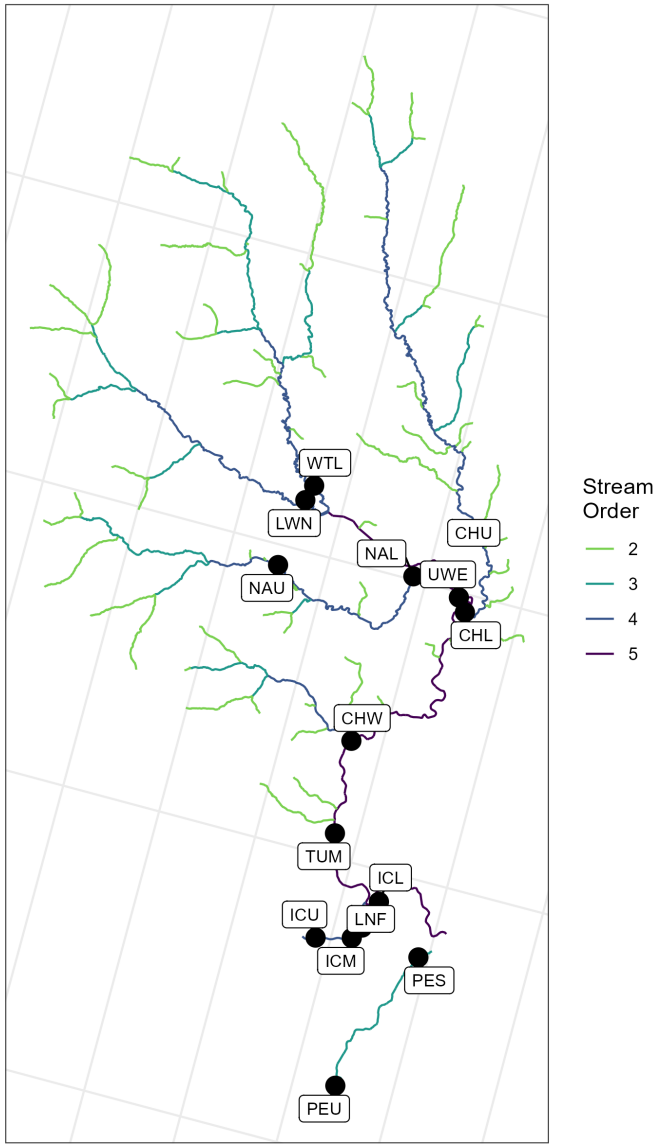
Make your life easier

- Query some data from PTAGIS
  - Metadata about sites
  - Configuration files
  - MRR files
- Read in data from PTAGIS or non-PTAGIS sources
  - Consistent format
  - Identify orphan or disowned tags
- Compress observations
  - Reduce redundant detections
- Determine movement direction
  - Based on parent-child table
  - Includes tools to build parent-child table
- Estimate array/site efficiency
- Build capture history matrix

Example Data

Adult Spring Chinook

- Tagged at Tumwater Dam on Wenatchee River as adults
- Goal is to estimate movement or survival past each site



Read in Data

Complete tag histories from PTAGIS

- Start with a list of tags
- Need certain columns

```
print(cth_file_nm)
```

```
## [1] "TUM_chnk_cth_2018.csv"
```

Read in Data

Complete tag histories from PTAGIS

- Start with a list of tags
- Need certain columns

```
print(cth_file_nm)
```

```
## [1] "TUM_chnk_cth_2018.csv"
```

Use `readCTH()` function:

```
# read the complete tag history file into R
cth_df ←
  system.file("extdata",
              cth_file_nm,
              package = "PITcleanr") ▷
  readCTH(file_type = "PTAGIS")
```

- PTAGIS not the only format
  - Biologic
  - Raw .log, .xlsx, or .txt files
- Change `file_type`

Compress data

Key feature of PITcleanr

Map each site / antenna to a *node*

Start with:

- 13,501 observations of
- 1,406 unique tags from
- 50 different sites.

```
comp_obs ← compress(cth_df)
```

End with 4,435 rows of data.

- 32.8% of original data

Compress data

Search:

	Tag Code	Node	Slot	Event Type Name	N Dets	Min Det	Max D
	All	All	All	All	All	All	All
1	384.3B239AC47B	NASONC	1	Mark	1	2016-03-04T17:00:00Z	2016-03-04T17:00:00Z	0	
2	384.3B239AC47B	BO3	2	Observation	18	2018-05-18T09:36:59Z	2018-05-18T12:00:05Z	143.1	1158757
3	384.3B239AC47B	BO4	3	Observation	7	2018-05-18T13:03:11Z	2018-05-18T13:09:29Z	6.3	63.1
4	384.3B239AC47B	TD1	4	Observation	4	2018-05-20T17:20:39Z	2018-05-20T17:20:44Z	0.1	3131.2
5	384.3B239AC47B	JO1	5	Observation	2	2018-05-21T17:49:21Z	2018-05-21T17:51:20Z	2	1468.6
6	384.3B239AC47B	MC2	6	Observation	12	2018-05-24T09:04:38Z	2018-05-24T09:54:53Z	50.2	3793.3

Showing 1 to 100 of 100 entries

Compress data

1. Shrink size of data
2. Assign observations to node of your choice
3. First, last and number of detections on each node
4. Total duration between first and last detections
5. Travel time from previous node

Configuration File

Configuration file

Map each detection to a user-defined node

Node could be:

- Individual antenna
- Array (group of antennas)
- Site
- Group of sites

Configuration file

Map each detection to a user-defined node

Node could be:

- Individual antenna
- Array (group of antennas)
- Site
- Group of sites

Can use PTAGIS site metadata to construct

```
my_config ←
  buildConfig(node_assign = "site")
```

Configuration file

Essential columns

Search:

	Site Code	Antenna Id	Config	
	All	All	All	
1	0HR	01	100	0HR
2	0HR	02	100	0HR
3	0HR	03	100	0HR
4	158	D1	100	158
5	158	D2	100	158
6	158	D3	100	158
7	158	D4	100	158
8	158	D5	100	158
9	158	D6	100	158

Showing 1 to 200 of 200 entries

Configuration file

But PTAGIS columns include:

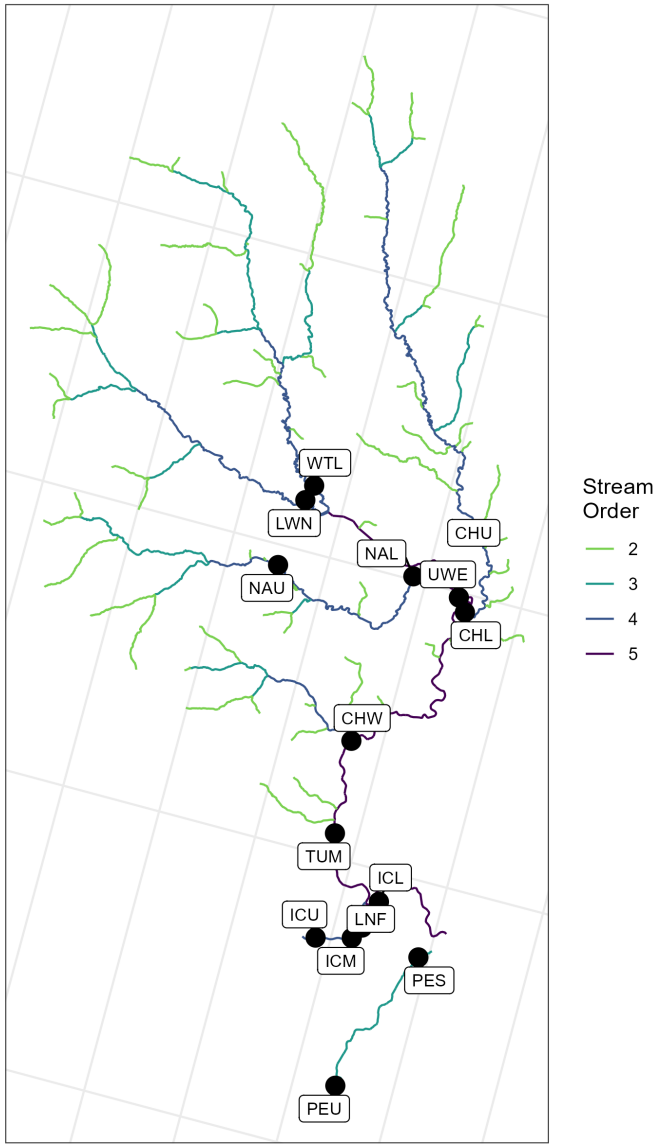
```
names(my_config) ▷
  janitor::make_clean_names("title")
```

```
##  [1] "Site Code"        "Config Id"        "Antenna Id"       "Node"
##  [5] "Start Date"       "End Date"         "Site Type"        "Site Name"
##  [9] "Antenna Group"    "Site Description" "Site Type Name"   "Rkm"
## [13] "Rkm Total"        "Latitude"         "Longitude"
```

Directionality

Map of Sites

- Which sites are upstream / downstream of one another?
- What are possible paths a fish might take?



Parent-Child Table

- Moving upstream, tags move from parent sites to child sites
- Details which sites are upstream of a particular site
- Can be reversed for downstream movement
- Can be made by hand...
- `PITcleanr` has tools to help

```
parent_child ▷
  select(parent, child)
```

```
## # A tibble: 14 x 2
##    parent child
##    <chr>  <chr>
##  1 ICL    LNF
##  2 ICL    ICM
##  3 TUM    ICL
##  4 TUM    UWE
##  5 TUM    CHL
##  6 TUM    PES
##  7 TUM    CHW
##  8 ICM    ICU
##  9 UWE    LWN
## 10 UWE    WTL
## 11 UWE    NAL
## 12 CHL    CHU
## 13 PES    PEU
## 14 NAL    NAU
```

Parent-Child Table

- PITcleanr has a `buildParentChild()` function.
- Requires an `sf` point object of sites
  - Build yourself or use `extractSites()`
- Requires an `sf` line object of the NHDPlus flowlines, downloaded from USGS.
  - Use `queryFlowlines()` to download it

Build `sf` object of sites

One way

```
# extract sites from detection file
sites_sf ←
  extractSites(cth_file = cth_df,
               as_sf = T,
               # drop juvenile detection/marking sites
               min_date = "20180501")

# focus on sites within Wenatchee subbasin, and drop a few sites we don't care about
sites_sf ←
  sites_sf %>%
  # all sites in the Wenatchee have a river kilometer that starts with 754
  filter(str_detect(rkm, "^754."),
         type ≠ "MRR",
         site_code ≠ "LWE") ▷
  mutate(across(site_code,
                ~ recode(.,
                         "TUF" = "TUM")))
```

Build `sf` object of sites

Another way

```
# type site codes by hand
sites_sf ←
  tibble(site_code =
           c("ICL", "TUM", "ICM", "UWE", "CHL", "PES", "NAL",
             "LNF", "CHW", "ICU", "LWN", "WTL", "CHU", "PEU",
             "NAU")) ▷
  left_join(my_config ▷
              select(site_code,
                     site_name,
                     site_type_name,
                     site_type,
                     rkm,
                     site_description,
                     latitude,
                     longitude) ▷
              distinct(),
            by = join_by(site_code)) ▷
  st_as_sf(coords = c("longitude",
                      "latitude"),
           crs = 4326) ▷
  arrange(rkm)
```

Query Flowlines

```
# query the flowlines
nhd_list ←
  queryFlowlines(sites_sf = sites_sf,
                 root_site_code = "TUM",
                 min_strm_order = 2,
                 dwnstrm_sites = T,
                 dwn_min_stream_order_diff = 2)

# join the upstream and downstream flowlines
flowlines ←
  nhd_list$flowlines %>%
  rbind(nhd_list$dwn_flowlines)
```

```
flowlines ▷
  st_drop_geometry() ▷
  select(COMID,
         Hydroseq,
         UpHydroseq,
         DnHydroseq)
```

```
## # A tibble: 386 x 4
##       COMID Hydroseq UpHydroseq DnHydroseq
##       <int>    <dbl>      <dbl>      <dbl>
##  1 23080784 50063945   50070238   50058967
##  2 23080786 50070238   50078399   50063945
##  3 23080788 50078399   50089588   50070238
##  4 23080790 50089588   50106483   50078399
##  5 23080792 50106483   50137781   50089588
##  6 23080794 50137781   50220575   50106483
##  7 23081176 50089577   50106469   50078388
##  8 23080910 50106479   50137746   50033323
##  9 23081092 50031394   50032319   50030525
## 10 23081090 50030525   50031394   50029716
## # i 376 more rows
```

Build Parent-Child Table

```
parent_child ←
  suppressWarnings(
    buildParentChild(sites_sf,
                     flowlines))
```

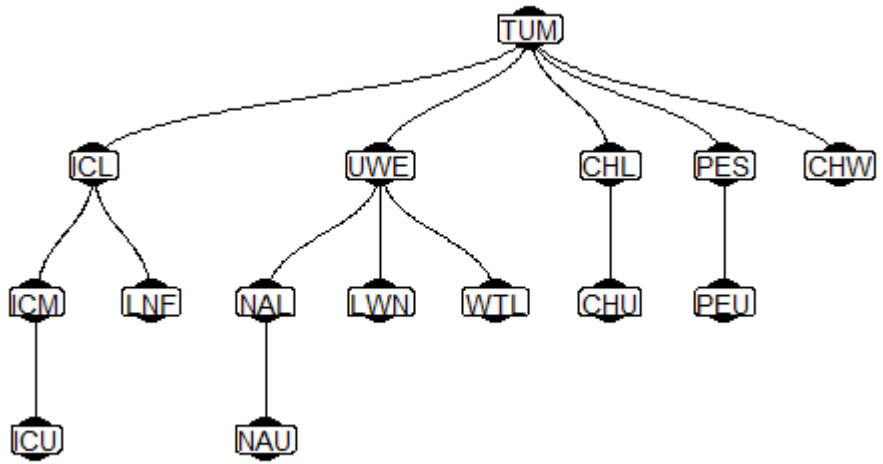
```
parent_child ←
  editParentChild(parent_child,
                  fix_list =
                    list(c(NA, "PES", "TUM"),
                         c(NA, "LNF", "ICL"),
                         c(NA, "ICL", "TUM"),
                         c("PES", "ICL", "TUM")),
                  switch_parent_child =
                    list(c("ICL", 'TUM')))
```

```
# view corrected parent_child table
parent_child ▷
  select(parent, child)
```

```
## # A tibble: 14 x 2
##    parent child
##    <chr>  <chr>
##  1 ICL    LNF
##  2 ICL    ICM
##  3 TUM    ICL
##  4 TUM    UWE
##  5 TUM    CHL
##  6 TUM    PES
##  7 TUM    CHW
##  8 ICM    ICU
##  9 UWE    LWN
## 10 UWE    WTL
## 11 UWE    NAL
## 12 CHL    CHU
## 13 PES    PEU
## 14 NAL    NAU
```

Plot Parent-Child Table

```
plotNodes(parent_child,
          point_size = 10,
          label_size = 5)
```



All possible fish pathways

```
buildNodeOrder(parent_child)
```

```
## # A tibble: 15 x 3
##    node  node_order path
##    <chr>      <dbl> <chr>
##  1 TUM            1 TUM
##  2 CHL            2 TUM CHL
##  3 CHU            3 TUM CHL CHU
##  4 CHW            2 TUM CHW
##  5 ICL            2 TUM ICL
##  6 ICM            3 TUM ICL ICM
##  7 ICU            4 TUM ICL ICM ICU
##  8 LNF            3 TUM ICL LNF
##  9 PES            2 TUM PES
## 10 PEU            3 TUM PES PEU
## 11 UWE            2 TUM UWE
## 12 LWN            3 TUM UWE LWN
## 13 NAL            3 TUM UWE NAL
## 14 NAU            4 TUM UWE NAL NAU
## 15 WTL            3 TUM UWE WTL
```

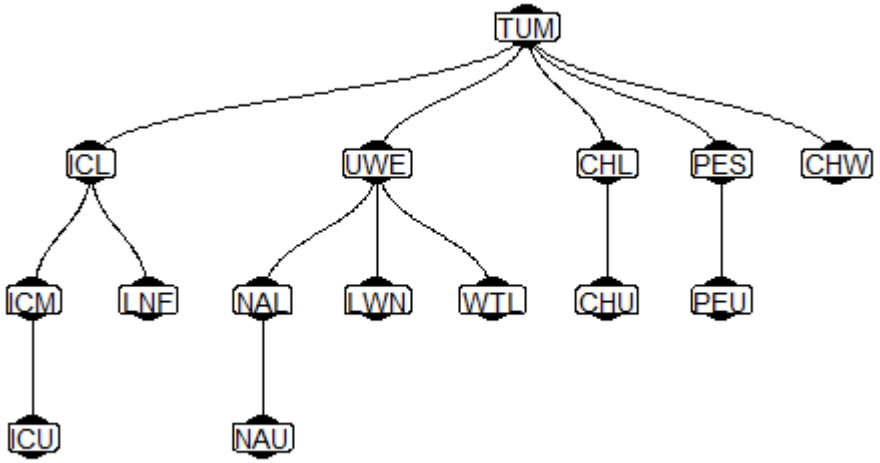
Add Direction

```
comp_dir ←
  addDirection(compress_obs = comp_obs,
               parent_child = parent_child)

comp_dir ▷
  select(tag_code,
         node,
         min_det,
         node_order,
         path,
         direction)
```

```
## # A tibble: 3,257 x 6
##    tag_code       node  min_det             node_order path        direction
##    <chr>          <chr> <dttm>                   <dbl> <chr>       <chr>
##  1 384.3B239AC47B TUM   2018-06-16 15:26:31          1 TUM         start
##  2 384.3B239AC47B UWE   2018-06-23 22:03:13          2 TUM UWE     forward
##  3 384.3B239AC47B NAL   2018-07-01 00:06:16          3 TUM UWE NAL forward
##  4 3DD.003B9DD720 TUM   2018-06-17 09:39:26          1 TUM         start
##  5 3DD.003B9DD725 TUM   2018-06-19 09:33:08          1 TUM         start
##  6 3DD.003B9DD725 CHU   2018-06-28 21:34:58          3 TUM CHL CHU forward
##  7 3DD.003B9DDAB0 TUM   2018-07-17 11:51:44          1 TUM         start
##  8 3DD.003B9F123C TUM   2018-06-21 09:45:14          1 TUM         start
##  9 3DD.003B9F123C UWE   2018-07-01 12:55:31          2 TUM UWE     forward
## 10 3DD.003BC7A654 TUM   2018-06-09 22:36:03          1 TUM         start
## # i 3,247 more rows
```

Strange Directions




```
## # A tibble: 8 x 6
##   tag_code       node  min_det             node_order path            direction
##   <chr>          <chr> <dttm>                   <dbl> <chr>           <chr>
## 1 3DD.003BC80D81 TUM   2018-06-21 16:14:26          1 TUM             start
## 2 3DD.003BC80D81 UWE   2018-06-29 15:29:19          2 TUM UWE         forward
## 3 3DD.003BC80D81 NAL   2018-06-30 01:35:27          3 TUM UWE NAL     forward
## 4 3DD.003BC80D81 LWN   2018-07-05 23:05:02          3 TUM UWE LWN     unknown
## 5 3DD.003BC80D81 NAL   2018-07-08 03:06:42          3 TUM UWE NAL     unknown
## 6 3DD.003BC80D81 UWE   2018-07-17 21:52:42          2 TUM UWE         backward
## 7 3DD.003BC80D81 NAL   2018-07-20 21:50:13          3 TUM UWE NAL     forward
## 8 3DD.003BC80D81 NAU   2018-07-30 22:40:06          4 TUM UWE NAL NAU forward
```

Filter Observations

```
comp_filter ←
  filterDetections(compress_obs = comp_obs,
                   parent_child = parent_child,
                   max_obs_date = "20180930")

comp_filter ▷
  filter(tag_code %in% examp_tag) ▷
  select(tag_code, node, min_det, direction,
         path,
         ends_with("keep_obs"))
```

```
## # A tibble: 8 x 7
##   tag_code node  min_det             direction path  auto_keep_obs user_keep_obs
##   <chr>    <chr> <dttm>              <chr>     <chr> <lgl>         <lgl>
## 1 3DD.003~ TUM   2018-06-21 16:14:26 start     TUM   TRUE          NA
## 2 3DD.003~ UWE   2018-06-29 15:29:19 forward   TUM ~ FALSE         NA
## 3 3DD.003~ NAL   2018-06-30 01:35:27 forward   TUM ~ FALSE         NA
## 4 3DD.003~ LWN   2018-07-05 23:05:02 unknown   TUM ~ FALSE         NA
## 5 3DD.003~ NAL   2018-07-08 03:06:42 unknown   TUM ~ FALSE         NA
## 6 3DD.003~ UWE   2018-07-17 21:52:42 backward  TUM ~ TRUE          NA
## 7 3DD.003~ NAL   2018-07-20 21:50:13 forward   TUM ~ TRUE          NA
## 8 3DD.003~ NAU   2018-07-30 22:40:06 forward   TUM ~ TRUE          NA
```

Other Functionality

Array Efficiency

Based on tags detected at an array, and any tags detected upstream of that array

Array Efficiency

Based on tags detected at an array, and any tags detected upstream of that array

Start by building a description of how each node is connected to others

```
node_order ←
  buildNodeOrder(parent_child)

node_order
```

```
## # A tibble: 15 x 3
##    node  node_order path
##    <chr>      <dbl> <chr>
##  1 TUM            1 TUM
##  2 CHL            2 TUM CHL
##  3 CHU            3 TUM CHL CHU
##  4 CHW            2 TUM CHW
##  5 ICL            2 TUM ICL
##  6 ICM            3 TUM ICL ICM
##  7 ICU            4 TUM ICL ICM ICU
##  8 LNF            3 TUM ICL LNF
##  9 PES            2 TUM PES
## 10 PEU            3 TUM PES PEU
## 11 UWE            2 TUM UWE
## 12 LWN            3 TUM UWE LWN
## 13 NAL            3 TUM UWE NAL
## 14 NAU            4 TUM UWE NAL NAU
## 15 WTL            3 TUM UWE WTL
```

Array Efficiency

Based on tags detected at an array, and any tags detected upstream of that array

```
estNodeEff(comp_obs,
           node_order)
```

```
## If no node is supplied, calculated for all nodes
```

Search:

	node	tags_at_node	tags_above_node	tags_resighted	est_tags_at_node		
	All	All	All	All	All		All
1	ICL	6	20	1	72	34.1	0.083	0.039
2	CHL	441	454	165	1211	59	0.364	0.018
3	NAL	168	133	83	269	12.7	0.625	0.029
4	UWE	361	256	175	528	16	0.684	0.021
5	TUM	1406	1039	1039	1406	0	1	0
6	CHU	454	0	0	454	0	1	0

Capture Histories

May need matrix of 1s and 0s

```
buildCapHist(comp_filter,
             parent_child = parent_child,
             configuration = my_config)
```

```
## # A tibble: 1,406 x 2
##    tag_code       cap_hist
##    <chr>          <chr>
##  1 384.3B239AC47B 100000000010100
##  2 3DD.003B9DD720 100000000000000
##  3 3DD.003B9DD725 101000000000000
##  4 3DD.003B9DDAB0 100000000000000
##  5 3DD.003B9F123C 100000000010000
##  6 3DD.003BC7A654 100000000010110
##  7 3DD.003BC7F0E3 100000000010110
##  8 3DD.003BC80A3C 100000000010100
##  9 3DD.003BC80D81 100000000010110
## 10 3DD.003BC80F20 100000000000100
## # i 1,396 more rows
```

Capture Histories

May need matrix of 1s and 0s

```
buildCapHist(comp_filter,
             parent_child = parent_child,
             configuration = my_config)
```

```
## # A tibble: 1,406 x 2
##    tag_code       cap_hist
##    <chr>          <chr>
##  1 384.3B239AC47B 100000000010100
##  2 3DD.003B9DD720 100000000000000
##  3 3DD.003B9DD725 101000000000000
##  4 3DD.003B9DDAB0 100000000000000
##  5 3DD.003B9F123C 100000000010000
##  6 3DD.003BC7A654 100000000010110
##  7 3DD.003BC7F0E3 100000000010110
##  8 3DD.003BC80A3C 100000000010100
##  9 3DD.003BC80D81 100000000010110
## 10 3DD.003BC80F20 100000000000100
## # i 1,396 more rows
```

What do columns stand for?

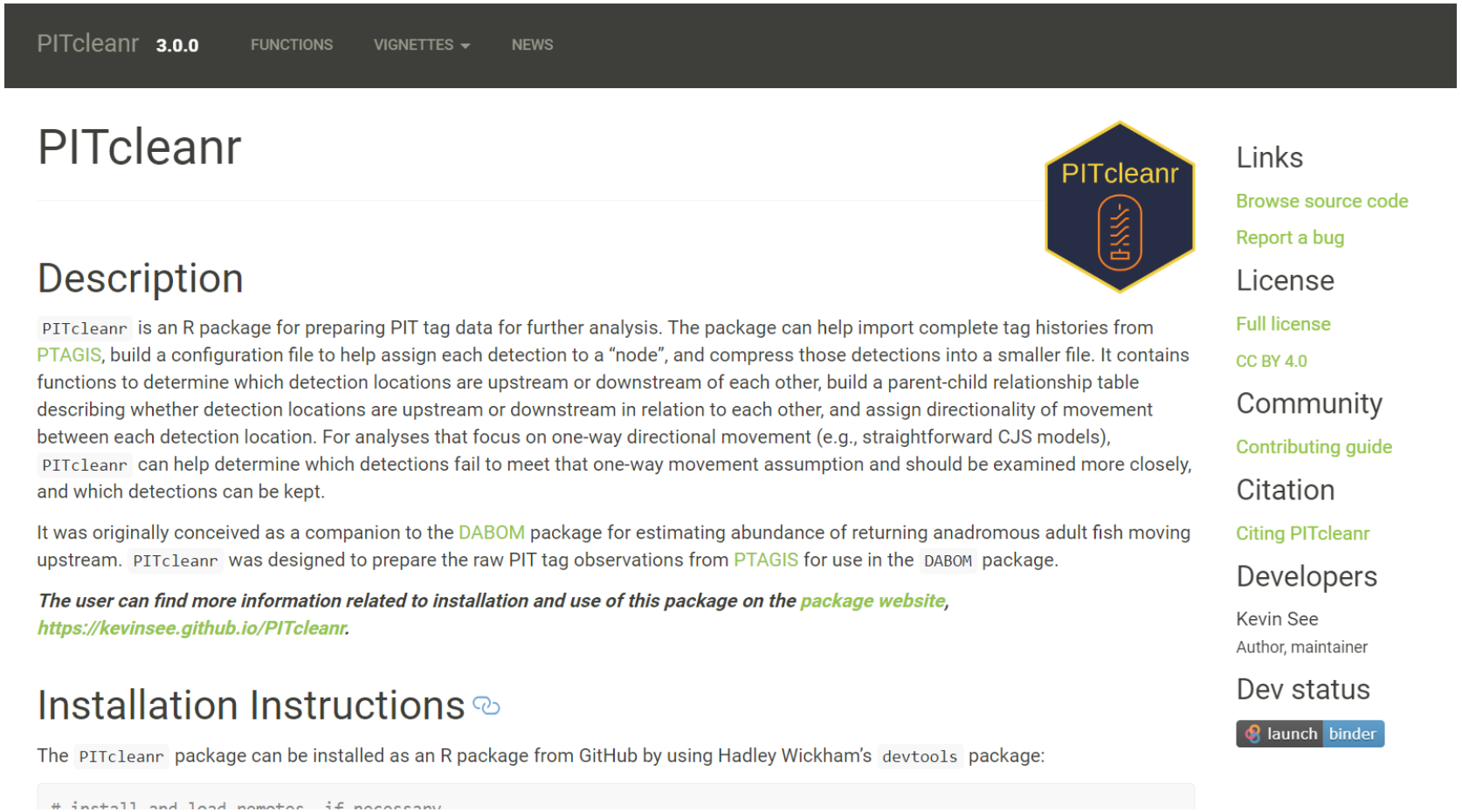
```
defineCapHistCols(parent_child = parent_child,
                  configuration = my_config) ▷
  as_tibble()
```

```
## # A tibble: 15 x 1
##    value
##    <chr>
##  1 TUM
##  2 CHL
##  3 CHU
##  4 CHW
##  5 ICL
##  6 ICM
##  7 ICU
##  8 LNF
##  9 PES
## 10 PEU
## 11 UWE
## 12 LWN
## 13 NAL
## 14 NAU
## 15 WTL
```

Package Resources

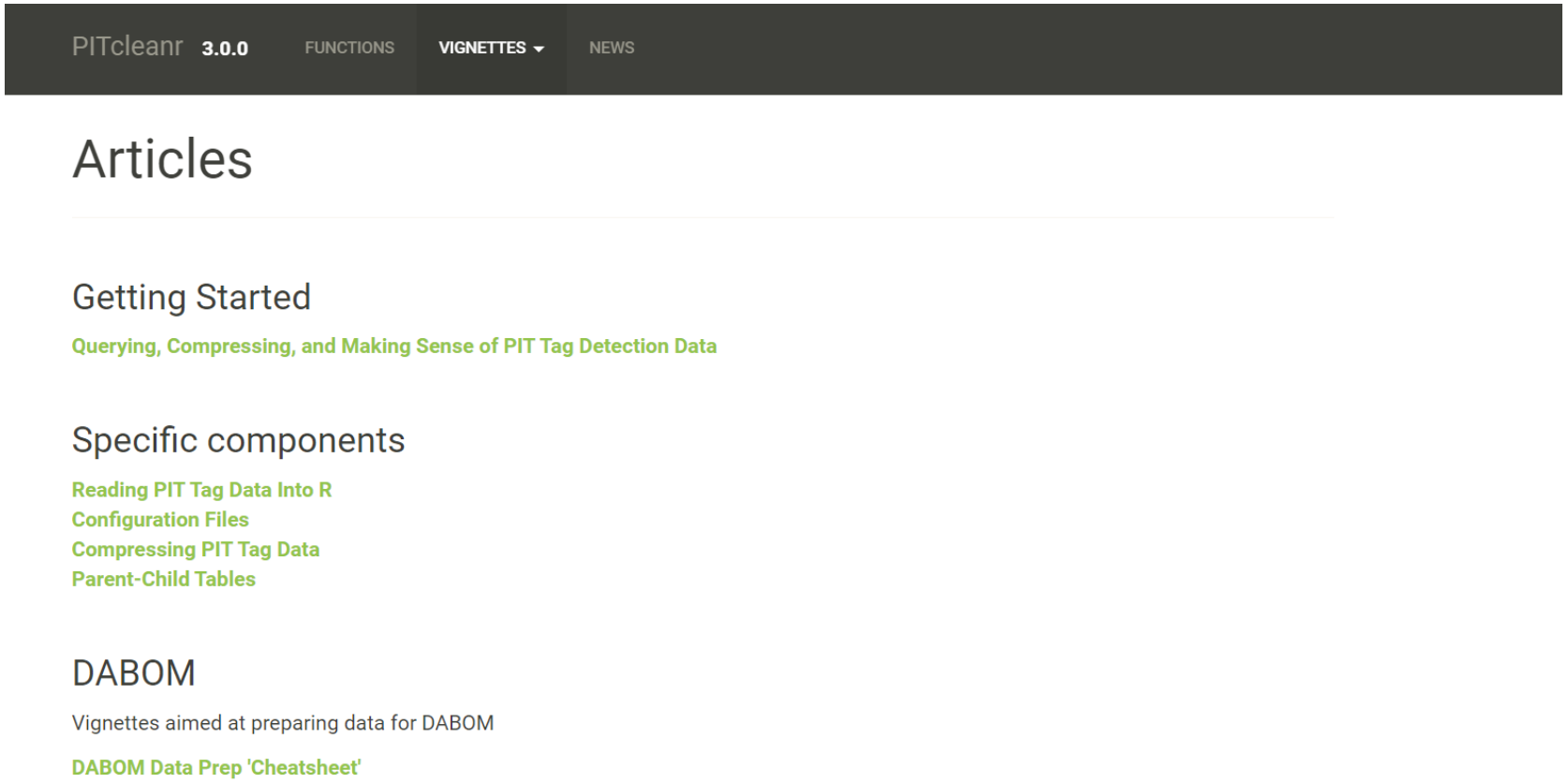
Package Website

- **https://kevinsee.github.io/PITcleanr/**



Package Website

- **https://kevinsee.github.io/PITcleanr/**



Contact Developers

- **Kevin See**: Kevin.See@dfw.wa.gov
- **Mike Ackerman**: MikeA@nezperce.org
- **Ryan Kinzer**: RyanK@nezperce.org

Contact Developers

- **Kevin See**: Kevin.See@dfw.wa.gov
- **Mike Ackerman**: MikeA@nezperce.org
- **Ryan Kinzer**: RyanK@nezperce.org

Come to workshop this afternoon


Questions?