Check the 2021 data and incorporate it into gfiphc

Andrew Edwards

Last compiled on 10 November, 2021

In 2021 (as for 2020) only the first 20 hooks were evaluated, so those data are not easily imported into GFBio. Going to incorporate into gfiphc here. Likely need this as a template for future years: resave this file with new year, and change all 2021's to the subsequent year, go through the code somewhat manually to check the output as you go along (in Emacs do Alt-query-replace to change years but read carefully as going along), and then finally render the full document to make the .pdf. This code includes some manual checks to make sure the data look okay. The planned stations for the 2021 survey are shown in this IPHC sampling manual (click here); page 23 has Vancouver [Island] Outside, showing that not all stations were intended to be fished there.

For comparison first look at 2013 data included in gfiphc:

```
load_all()
> i Loading gfiphc
setData2013
> # A tibble: 170 x 8
>     year station   lat   lon avgDepth effSkateIPHC E_it20 usable
>    <int> <chr>   <dbl> <dbl>    <int>        <dbl>  <dbl> <chr>
>  1  2013 2001     48.3 -126.       76         5.96   1.19 Y
>  2  2013 2002     48.3 -126.       93         5.90   1.19 Y
>  3  2013 2003     48.5 -125.       79         5.90   1.19 Y
>  4  2013 2004     48.5 -126.       56         5.96   1.20 Y
>  5  2013 2005     48.5 -126.       58         6.02   1.20 Y
>  6  2013 2006     48.5 -126.      110         5.78   1.16 Y
>  7  2013 2007     48.7 -125.       35         5.96   1.20 Y
>  8  2013 2008     48.7 -125.       35         5.90   1.20 Y
>  9  2013 2009     48.7 -126.       67         5.90   1.19 Y
> 10  2013 2010     48.7 -126.       41         5.96   1.20 Y
> # ... with 160 more rows
countData2013
> # A tibble: 1,304 x 4
>     year station spNameIPHC           specCount
>    <int> <chr>   <chr>                    <int>
>  1  2013 2001    Spiny Dogfish               61
```

```
>  2  2013 2001    Empty Hook                57
>  3  2013 2001    Pacific Halibut            2
>  4  2013 2002    Spiny Dogfish             59
>  5  2013 2002    Empty Hook                56
>  6  2013 2002    Pacific Halibut            5
>  7  2013 2003    Sablefish (Blackcod)       1
>  8  2013 2003    Longnose Skate             4
>  9  2013 2003    Arrowtooth Flounder        7
> 10  2013 2003    Spiny Dogfish             13
> # ... with 1,294 more rows
```

We want to get the new data into the same format as those (columns with same names and classes, even though in retrospect some classes aren't ideally chosen, but also retaining retrieved and observed hooks for the set data). Two data sets are needed because later gfiphc code summarises catches of a particular species at the station level, and needs to create counts of zeros for the species of interest (and such zeros are not included in IPHC output).

Set-level information

For 2020, Maria was sent the file `2020 IPHCtoDFO_dataExtraction-Maria.xls` for set details, but this is multiple sheets and more complex than needed. So I tried extracting directly from the IPHC website (which they want us to do in the future anyway), using the following instructions, which worked for 2020 and 2021:

Go to https://www.iphc.int/data/fiss-data-query and select the following options:

1. Year Range – 2021 to 2021.
2. Area 2B
3. Purpose Codes – All
4. IPHC Charter Regions – All
5. Maps – Nothing
6. Select non-Pacific halibut species – deselect All.

Download tab on bottom right (see instructions above question 4), and select CrossTab. Select "Set and Pacific Halibut data" and .xlsx format (I tried .csv format but it didn't save with commas, strangely). Save in this folder as `set-and-halibut-data-2021.xlsx`. Open in Excel and Export as .csv, `set-and-halibut-data-2021.csv`, and when trying to quit Excel say no to save changes (not sure if that matters).

Repeat but with all non-halibut data (select All in number 6), and save as `non-halibut-data-2021.xlsx` and export as .csv in Excel, `non-halibut-data-2021.csv`. Importantly, this file (but not the first one) contains the numbers of observed hooks, needed in our calculations.

Load data for new year:

```
sets_raw <- readr::read_csv("set-and-halibut-data-2021.csv") %>%
  dplyr::mutate_if(is.character, factor)
>
> -- Column specification --------------------------------------------------------
> cols(
>   .default = col_double(),
>   `Vessel code` = col_character(),
>   `IPHC Reg Area` = col_character(),
>   `IPHC Charter Region` = col_character(),
>   `Purpose Code` = col_character(),
>   Date = col_character(),
>   Eff = col_character(),
>   Ineffcde = col_character(),
>   `032 Pacific halibut weight` = col_number(),
>   `U32 Pacific halibut weight` = col_number(),
>   `Sigma-t` = col_logical(),
>   Oxygen_umol = col_logical(),
>   Oxygen_sat = col_logical()
> )
> i Use `spec()` for the full column specifications.
```

Now load the original 2020 data (do not change the 2020 here) to then test that the column names and types do not change in future years, and then check columns match `sets_raw`:

```
sets_raw_2020 <- readr::read_csv("set-and-halibut-data-2020.csv") %>%
  dplyr::mutate_if(is.character, factor)
>
> -- Column specification --------------------------------------------------------
> cols(
>   .default = col_double(),
>   `Vessel code` = col_character(),
>   `IPHC Reg Area` = col_character(),
>   `IPHC Charter Region` = col_character(),
>   Purpose = col_character(),
>   Date = col_character(),
>   Eff = col_character(),
>   Ineffcde = col_logical(),
>   `032 Pacific halibut weight` = col_number(),
>   `U32 Pacific halibut weight` = col_number()
> )
> i Use `spec()` for the full column specifications.

# For 2021 these were different - uncomment for future for first test
# testthat::expect_equal(names(sets_raw_2020),
```

```
#                         names(sets_raw))

# testthat::expect_equal(sapply(sets_raw_2020, typeof),
#                         sapply(sets_raw, typeof))

# Columns in 2020 not in new data:
setdiff(names(sets_raw_2020), names(sets_raw))
> [1] "Purpose"

# Columns in new data not in 2020:
setdiff(names(sets_raw), names(sets_raw_2020))
>  [1] "Purpose Code"              "Profiler Lat"
>  [3] "Profiler Lon"              "Profiler Bottom Depth (m)"
>  [5] "Temp C"                    "Max Pressure (db)"
>  [7] "pH"                        "Salinity PSU"
>  [9] "Sigma-t"                   "Oxygen_ml"
> [11] "Oxygen_umol"               "Oxygen_sat"

# For 2021 looks like Purpose became Purpose Code, but are the same type:
summary(sets_raw_2020$Purpose)
>    Deep expansion Shallow expansion     Standard grid
>                 3                30               165
summary(sets_raw$"Purpose Code")
> Standard Grid
>           232
testthat::expect_equal(typeof(sets_raw_2020$Purpose),
                       typeof(sets_raw$"Purpose Code"))
```

Those extra columns in 2021 look related to oceanographic data, beyond the scope of gfiphc, so can just ignore shortly.

Want to check the overlapping columns have the same type:

```
overlap_col_names <- intersect(names(sets_raw_2020),
                               names(sets_raw))
# testthat::expect_equal(sapply(dplyr::select(sets_raw_2020,
#                                             overlap_col_names),
#                               typeof),
#                        sapply(dplyr::select(sets_raw,
#                                             overlap_col_names),
#                               typeof))
# Error: sapply(dplyr::select(sets_raw_2020, overlap_col_names), typeof) not equal to
# 1/32 mismatches
# x[12]: "logical"
# y[12]: "integer"
```

```
dplyr::select(sets_raw_2020,
              overlap_col_names[12])
> # A tibble: 198 x 1
>    Ineffcde
>    <lgl>
>  1 NA
>  2 NA
>  3 NA
>  4 NA
>  5 NA
>  6 NA
>  7 NA
>  8 NA
>  9 NA
> 10 NA
> # ... with 188 more rows
dplyr::select(sets_raw,
              overlap_col_names[12])
> # A tibble: 232 x 1
>    Ineffcde
>    <fct>
>  1 <NA>
>  2 <NA>
>  3 <NA>
>  4 <NA>
>  5 <NA>
>  6 <NA>
>  7 <NA>
>  8 <NA>
>  9 <NA>
> 10 <NA>
> # ... with 222 more rows
```

These are all NA's anyway (see below) and don't get saved, so no worries.

```
sets_raw
> # A tibble: 232 x 44
>    `Row number`  Year   Stlkey `Vessel code` Station Setno `IPHC Reg Area`
>           <dbl> <dbl>    <dbl> <fct>           <dbl> <dbl> <fct>
>  1            1  2021 20210014 VNI              2266     1 2B
>  2            2  2021 20210015 VNI              2267     2 2B
>  3            3  2021 20210016 VNI              2270     3 2B
>  4            4  2021 20210017 VNI              2272     4 2B
>  5            5  2021 20210018 VNI              2275     5 2B
>  6            6  2021 20210019 VNI              2268     6 2B
```

```
>  7           7  2021 20210020 VNI             2073     7 2B
>  8           8  2021 20210021 VNI             2078     8 2B
>  9           9  2021 20210022 VNI             2066     9 2B
> 10          10  2021 20210023 VNI             2065    10 2B
> # ... with 222 more rows, and 37 more variables: IPHC Stat Area <dbl>,
> #   IPHC Charter Region <fct>, Purpose Code <fct>, Date <fct>, Eff <fct>,
> #   Ineffcde <fct>, BeginLat <dbl>, BeginLon <dbl>, BeginDepth (fm) <dbl>,
> #   EndLat <dbl>, EndLon <dbl>, EndDepth (fm) <dbl>, MidLat fished <dbl>,
> #   MidLon fished <dbl>, AvgDepth (fm) <dbl>, Lat - Grid target <dbl>,
> #   Lon - Grid target <dbl>, 032 Pacific halibut count <dbl>,
> #   U32 Pacific halibut count <dbl>, 032 Pacific halibut weight <dbl>,
> #   U32 Pacific halibut weight <dbl>, No. skates set <dbl>,
> #   No. skates hauled <dbl>, Avg no. hook/skate <dbl>,
> #   Effective skates hauled <dbl>, Soak time (min.) <dbl>, Profiler Lat <dbl>,
> #   Profiler Lon <dbl>, Profiler Bottom Depth (m) <dbl>, Temp C <dbl>,
> #   Max Pressure (db) <dbl>, pH <dbl>, Salinity PSU <dbl>, Sigma-t <lgl>,
> #   Oxygen_ml <dbl>, Oxygen_umol <lgl>, Oxygen_sat <lgl>
summary(sets_raw)
>    Row number          Year          Stlkey         Vessel code    Station
>  Min.   :  1.00   Min.   :2021   Min.   :20210014   PEN: 88     Min.   :2002
>  1st Qu.: 58.75   1st Qu.:2021   1st Qu.:20210175   VNI:144     1st Qu.:2083
>  Median :116.50   Median :2021   Median :20210508               Median :2142
>  Mean   :116.50   Mean   :2021   Mean   :20210557               Mean   :2185
>  3rd Qu.:174.25   3rd Qu.:2021   3rd Qu.:20211033               3rd Qu.:2276
>  Max.   :232.00   Max.   :2021   Max.   :20211134               Max.   :3210
>
>     Setno        IPHC Reg Area IPHC Stat Area        IPHC Charter Region
>  Min.   :  1.00   2B:232        Min.   : 60.0   Charlotte         :87
>  1st Qu.: 29.75                 1st Qu.: 92.0   Goose Is.         :57
>  Median : 58.50                 Median :112.0   St. James         :59
>  Mean   : 61.88                 Mean   :107.3   Vancouver Outside:29
>  3rd Qu.: 87.25                 3rd Qu.:121.0
>  Max.   :144.00                 Max.   :142.0
>
>         Purpose Code        Date      Eff      Ineffcde      BeginLat
>  Standard Grid:232   02-Jun-21:  7   N:  2   DS  :  1   Min.   :48.34
>                      01-Jun-21:  6   Y:230   MS  :  1   1st Qu.:51.49
>                      04-Jun-21:  6           NA's:230   Median :52.34
>                      09-Jul-21:  6                      Mean   :52.31
>                      10-Jul-21:  6                      3rd Qu.:53.48
>                      10-Jun-21:  6                      Max.   :55.31
>                      (Other)  :195
>     BeginLon      BeginDepth (fm)      EndLat          EndLon
>  Min.   :-133.7   Min.   :  8.00   Min.   :48.32   Min.   :-133.7
```

```
>  1st Qu.:-131.1   1st Qu.: 40.00   1st Qu.:51.50   1st Qu.:-131.1
>  Median :-130.0   Median : 78.00   Median :52.33   Median :-130.0
>  Mean   :-129.9   Mean   : 91.45   Mean   :52.31   Mean   :-129.9
>  3rd Qu.:-128.9   3rd Qu.:122.25   3rd Qu.:53.48   3rd Qu.:-129.0
>  Max.   :-124.9   Max.   :336.00   Max.   :55.35   Max.   :-124.9
>
>  EndDepth (fm)    MidLat fished   MidLon fished    AvgDepth (fm)
>  Min.   :  8.00   Min.   :48.33   Min.   :-133.7   Min.   : 10.0
>  1st Qu.: 42.00   1st Qu.:51.50   1st Qu.:-131.1   1st Qu.: 44.0
>  Median : 76.50   Median :52.33   Median :-130.0   Median : 76.0
>  Mean   : 89.98   Mean   :52.31   Mean   :-129.9   Mean   : 89.6
>  3rd Qu.:121.25   3rd Qu.:53.50   3rd Qu.:-128.9   3rd Qu.:119.2
>  Max.   :339.00   Max.   :55.33   Max.   :-124.9   Max.   :334.0
>
>  Lat - Grid target Lon - Grid target O32 Pacific halibut count
>  Min.   :48.33     Min.   :-133.7    Min.   :  0.00
>  1st Qu.:51.50     1st Qu.:-131.1    1st Qu.:  5.75
>  Median :52.33     Median :-130.0    Median : 16.00
>  Mean   :52.31     Mean   :-129.9    Mean   : 24.66
>  3rd Qu.:53.50     3rd Qu.:-128.9    3rd Qu.: 35.25
>  Max.   :55.33     Max.   :-124.9    Max.   :126.00
>
>  U32 Pacific halibut count O32 Pacific halibut weight
>  Min.   :  0.00            Min.   :   0.0
>  1st Qu.:  1.00            1st Qu.: 123.8
>  Median : 10.00            Median : 404.5
>  Mean   : 22.94            Mean   : 612.6
>  3rd Qu.: 31.00            3rd Qu.: 819.5
>  Max.   :175.00            Max.   :4015.0
>
>  U32 Pacific halibut weight No. skates set No. skates hauled Avg no. hook/skate
>  Min.   :   0.00           Min.   :4.0    Min.   :3.000     Min.   : 96
>  1st Qu.:  10.75           1st Qu.:8.0    1st Qu.:8.000     1st Qu.: 99
>  Median :  80.50           Median :8.0    Median :8.000     Median : 99
>  Mean   : 180.33           Mean   :7.5    Mean   :7.478     Mean   : 99
>  3rd Qu.: 240.00           3rd Qu.:8.0    3rd Qu.:8.000     3rd Qu.: 99
>  Max.   :1355.00           Max.   :8.0    Max.   :8.000     Max.   :101
>
>  Effective skates hauled Soak time (min.)  Profiler Lat    Profiler Lon
>  Min.   :2.460           Min.   :361.0    Min.   :50.81   Min.   :-133.4
>  1st Qu.:7.950           1st Qu.:461.2    1st Qu.:51.84   1st Qu.:-131.1
>  Median :7.950           Median :564.5    Median :52.69   Median :-130.3
>  Mean   :7.431           Mean   :567.8    Mean   :52.79   Mean   :-130.3
>  3rd Qu.:7.950           3rd Qu.:647.5    3rd Qu.:53.67   3rd Qu.:-129.2
```

```
>  Max.   :8.110           Max.   :929.0    Max.   :55.35   Max.   :-126.8
>                                           NA's   :55      NA's   :55
>  Profiler Bottom Depth (m)     Temp C       Max Pressure (db)       pH
>  Min.   : 18.0             Min.   : 5.146   Min.   :  3.0     Min.   : 7.369
>  1st Qu.: 73.0             1st Qu.: 6.022   1st Qu.: 62.0     1st Qu.: 7.652
>  Median :135.0             Median : 6.637   Median :122.0     Median : 7.801
>  Mean   :145.7             Mean   : 7.168   Mean   :130.8     Mean   : 8.314
>  3rd Qu.:210.0             3rd Qu.: 7.713   3rd Qu.:197.0     3rd Qu.: 8.060
>  Max.   :435.0             Max.   :13.920   Max.   :407.0     Max.   :14.671
>  NA's   :55                NA's   :55       NA's   :55        NA's   :55
>   Salinity PSU   Sigma-t          Oxygen_ml     Oxygen_umol    Oxygen_sat
>  Min.   :30.54   Mode:logical   Min.   :1.352   Mode:logical   Mode:logical
>  1st Qu.:32.37   NA's:232       1st Qu.:2.204   NA's:232       NA's:232
>  Median :33.18                  Median :2.868
>  Mean   :32.98                  Mean   :3.475
>  3rd Qu.:33.72                  3rd Qu.:4.424
>  Max.   :33.98                  Max.   :8.404
>  NA's   :55                     NA's   :55
testthat::expect_equal(unique(sets_raw$"IPHC Reg Area"),
                       as.factor("2B")) # Check just BC
testthat::expect_equal(unique(sets_raw$Year), 2021)
testthat::expect_equal(length(unique(sets_raw$Station)),
                       length(sets_raw$Station))
```

Understand any issues raised above

Uncomment those three `testthat` commands when looking at new data each year. If any of fail then have to comment it out and figure out what it means here.

This is for 2020 (check for future years), to look for station(s) that was fished twice. Not really needed for 2021 since that third test passed, but `twice_fished` gets used later, so do evaluate here:

```
length(unique(sets_raw$Station))
> [1] 232
length(sets_raw$Station)
> [1] 232
dplyr::count(sets_raw, Station) %>% dplyr::filter(n > 1)
> # A tibble: 0 x 2
> # ... with 2 variables: Station <dbl>, n <int>
twice_fished <- dplyr::count(sets_raw, Station) %>%
  dplyr::filter(n > 1) %>%
  dplyr::select(Station) %>%
  as.numeric()
```

```
twice_fished
> [1] NA
# If there's more than a single station then adapt later code
#as.data.frame(dplyr::filter(sets_raw,
#                            Station == twice_fished))
```

Not needed for 2021: So Station NA had two vessels fishing the same station (which the code below originally caused a total of four rows for that station, explaining the 200 rows I had in original `setData2020` before fixing the issue). Interestingly the halibut catches were almost double for one vessel than the other (but were 6 days apart):

2020: Note that one of those entries has 'Vessel code' HAN, but HAN only appears once in the whole data set (as seen in `summary(sets_raw)` above.

For 2021, just noting that two vessels were used, and these are different to those in 2020 (for which HAN then got excluded anyway):

```
summary(sets_raw$"Vessel code")
> PEN VNI
>  88 144
summary(sets_raw_2020$"Vessel code")
> BDP HAN VNI
> 139   1  58
```

2020: So given we want to exclude one of the duplicates, makes sense to exclude HAN. (Also, Dana mentioned some gear comparison studies for 2020).

Simplify down to what's needed and rename, based on `iphc2013data.Rnw` (need to include the 'purpose' column, unlike 2013):

```
# sets_simp <- dplyr::filter(sets_raw, `Vessel code` != "HAN") %>%
sets_simp <- dplyr::select(sets_raw,
                           year = Year,
                           station = Station,
                           lat = "MidLat fished",
                           lon = "MidLon fished",
                           avgDepth = "AvgDepth (fm)",
                           skatesHauled = "No. skates hauled",
                           effSkateIPHC = "Effective skates hauled",
                           soakTimeMinutes = "Soak time (min.)",  # Joe might want
                           usable = Eff,
                           purpose = "Purpose Code",
                           U32halibut = "U32 Pacific halibut count",
                           O32halibut = "O32 Pacific halibut count") %>%
  arrange(station) %>%
  dplyr::mutate(year = as.integer(year),
                station = as.character(station),
```

```
                avgDepth = as.integer(avgDepth),
                usable = as.character(usable))
sets_simp
> # A tibble: 232 x 12
>     year station   lat   lon avgDepth skatesHauled effSkateIPHC soakTimeMinutes
>    <int> <chr>   <dbl> <dbl>    <int>        <dbl>        <dbl>            <dbl>
>  1  2021 2002     48.3 -126.      195            4         3.98              630
>  2  2021 2010     48.7 -126.       40            4         3.98              537
>  3  2021 2011     48.7 -126.       77            4         3.98              782
>  4  2021 2012     48.8 -126.       24            4         3.98              480
>  5  2021 2014     48.8 -126.       56            4         3.98              599
>  6  2021 2016     49.0 -126.       37            4         3.93              533
>  7  2021 2017     49.0 -126.       75            4         3.98              489
>  8  2021 2018     49.0 -127.      127            4         3.93              540
>  9  2021 2019     49.2 -126.       47            4         3.98              533
> 10  2021 2020     49.2 -127.       68            4         3.98              464
> # ... with 222 more rows, and 4 more variables: usable <chr>, purpose <fct>,
> #   U32halibut <dbl>, O32halibut <dbl>
```

Standard grid or not

Need to change purpose to `standard` (`Y/N`) to match 2018 data (`Y` for the standard grid). In the raw 2020 data, `Purpose` took three values that we converted to `standard` to save in the package:

```
summary(sets_raw_2020$Purpose)
>    Deep expansion Shallow expansion     Standard grid
>                 3                30               165
summary(setData2020$standard)
>   N   Y
>  71 126
```

For 2021 we have all as Standard Grid, which gets corrected (some stations are non-standard) in the next section.

```
summary(sets_simp$purpose)
> Standard Grid
>           232

sets_simp_std <- dplyr::mutate(sets_simp,
                               standard_tmp = (purpose == "Standard Grid"))
                               # was grid in 2020, Grid in 2021

standard <- as.character(sets_simp_std$standard_tmp)  # to get the right length
```

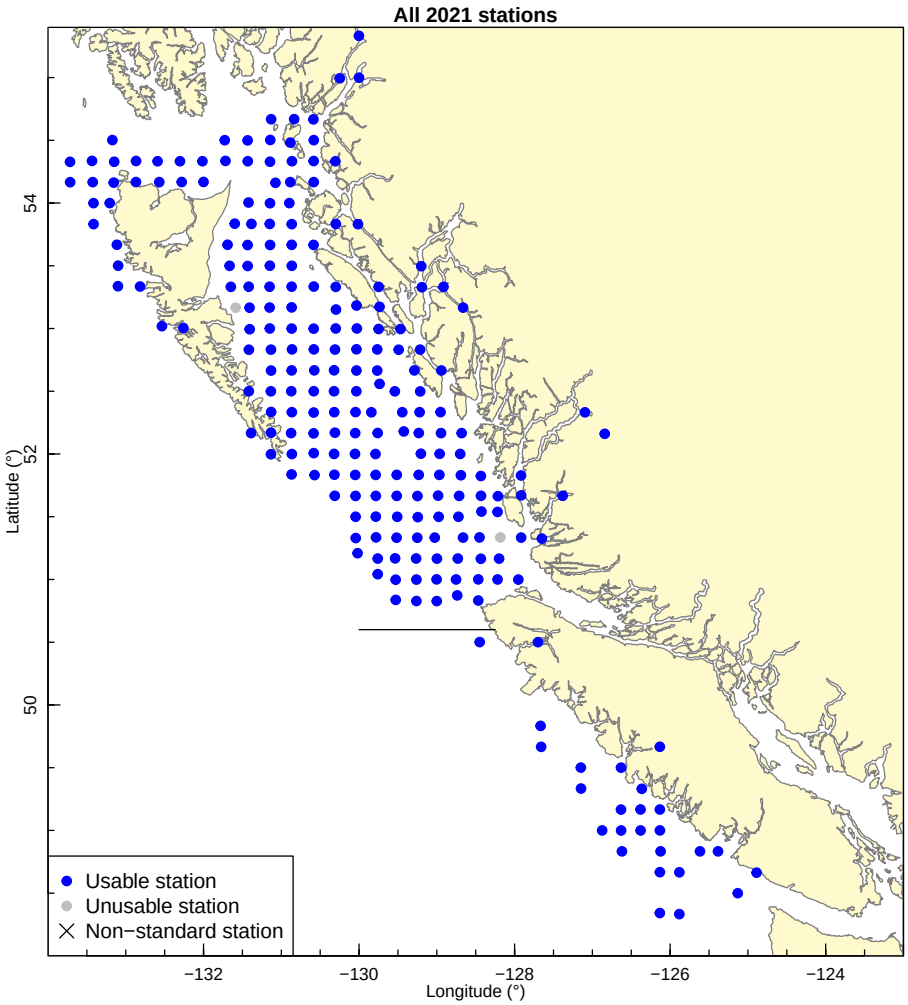
```
standard[sets_simp_std$standard_tmp] = "Y"
standard[!sets_simp_std$standard_tmp] = "N"
length(standard)
> [1] 232

sets_simp_std <- cbind(sets_simp_std,
                       standard) %>%
  as_tibble() %>%
  dplyr::select(-c("standard_tmp"))
summary(sets_simp_std)
>       year        station               lat             lon
>  Min.   :2021   Length:232         Min.   :48.33   Min.   :-133.7
>  1st Qu.:2021   Class :character   1st Qu.:51.50   1st Qu.:-131.1
>  Median :2021   Mode  :character   Median :52.33   Median :-130.0
>  Mean   :2021                      Mean   :52.31   Mean   :-129.9
>  3rd Qu.:2021                      3rd Qu.:53.50   3rd Qu.:-128.9
>  Max.   :2021                      Max.   :55.33   Max.   :-124.9
>     avgDepth      skatesHauled    effSkateIPHC   soakTimeMinutes
>  Min.   : 10.0   Min.   :3.000   Min.   :2.460   Min.   :361.0
>  1st Qu.: 44.0   1st Qu.:8.000   1st Qu.:7.950   1st Qu.:461.2
>  Median : 76.0   Median :8.000   Median :7.950   Median :564.5
>  Mean   : 89.6   Mean   :7.478   Mean   :7.431   Mean   :567.8
>  3rd Qu.:119.2   3rd Qu.:8.000   3rd Qu.:7.950   3rd Qu.:647.5
>  Max.   :334.0   Max.   :8.000   Max.   :8.110   Max.   :929.0
>    usable                   purpose      U32halibut       O32halibut
>  Length:232         Standard Grid:232   Min.   :  0.00   Min.   :  0.00
>  Class :character                       1st Qu.:  1.00   1st Qu.:  5.75
>  Mode  :character                       Median : 10.00   Median : 16.00
>                                         Mean   : 22.94   Mean   : 24.66
>                                         3rd Qu.: 31.00   3rd Qu.: 35.25
>                                         Max.   :175.00   Max.   :126.00
>    standard
>  Length:232
>  Class :character
>  Mode  :character
>
>
>
unique(sets_simp_std$standard)
> [1] "Y"
```

So they are all classified as standard. For 2020 we stuck with the 2018 definitions of standard, so doing that next.

Look at data and show map to understand changing definition of standard station from 2018 to 2020.

The definition of 'standard grid' changed from 2018 (when first needed due to the expanded grid) to 2020 (and 2021). Simply equating them as above is not sufficient. For 2021 we so far have this:

```
plot_iphc_map(sets_simp_std,
              sp = NULL,
              years = 2021,
              indicate_standard = TRUE)
```



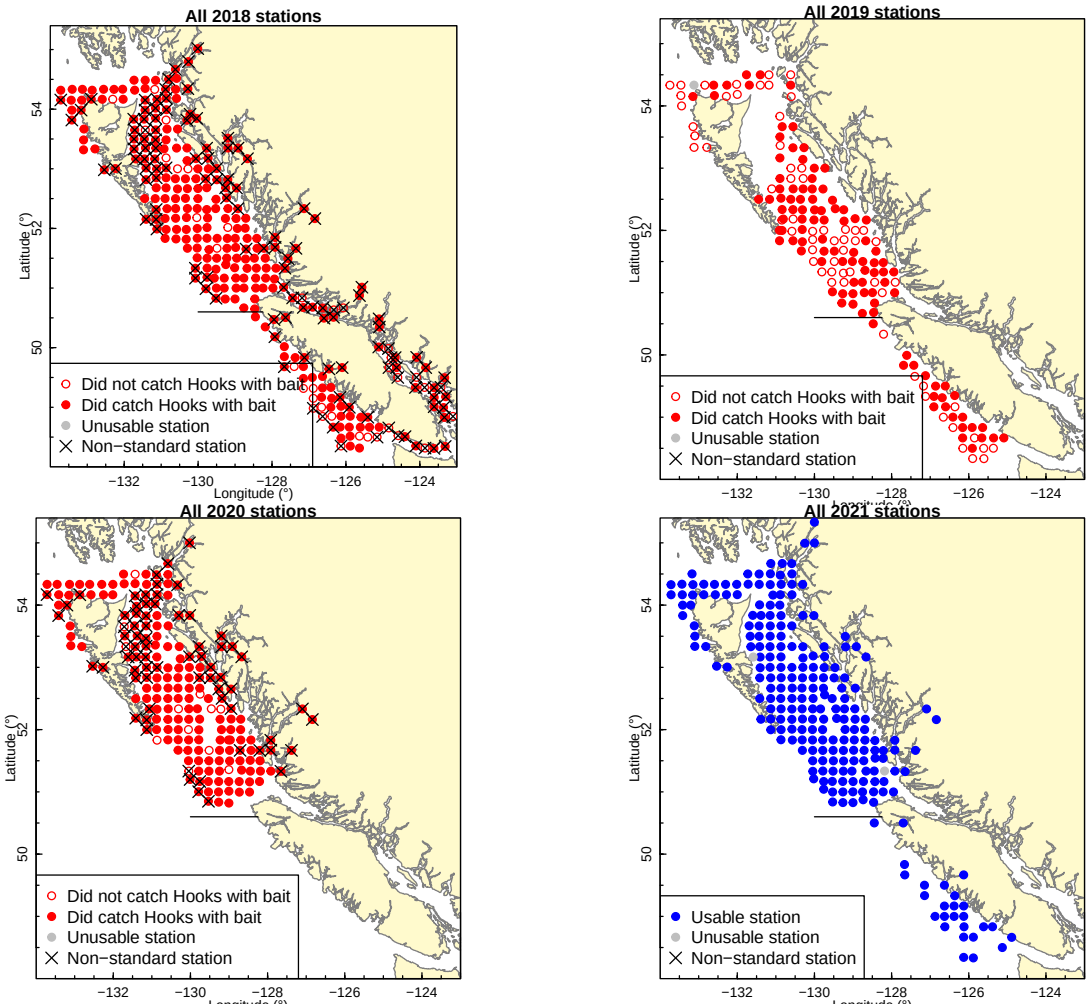
So no stations are marked as being outside the standard grid, even though some are clearly new – the ones in the north have never been fished before (see the one-species vignette, though I'll investigate that here).

This next section was to first figure out the twice-fished station 2343 in 2020, and to replicate that original analysis (station ends up being non-standard later), so mostly commented out except first bit which is used later so keeping in case need in future years:

```
hooks_with_bait_revert <- hooks_with_bait

# This should be commented out for 2021 survey analysis in iphc-2021-data.Rmd,
#  since the problem is presumably fixed. This is to revert back to the original
#  problem, for which 2343 was called standard in 2018 but we changed it. Map on
#  page 10 of iphc-2020-data.pdf has this station (second one down off
#  north-east tip of Haida Gwaii) as non-standard in 2018 but not 2020.
# hooks_with_bait_revert$set_counts[hooks_with_bait_revert$set_counts$year == 2018 &
#                             hooks_with_bait_revert$set_counts$station == 2343,
#                             ]$standard = "Y"

#filter(hooks_with_bait$set_counts, year == 2018, station == 2343) %>%
#  as.data.frame()      # saved version
#filter(hooks_with_bait_revert$set_counts, year == 2018, station == 2343) %>%
#  as.data.frame()      # reverted version
```

Now to figure out standard/non-standard stations. Plotting four years, with crosses showing 'non-standard'. (2021 is coloured different since no hooks with bait data yet, but the important bit is the crosses).

```
sets_2021 <- dplyr::select(sets_simp_std,
                           -c(U32halibut, O32halibut))
                           # else not the same structure as sets_2018, below

plot_iphc_map_panel(hooks_with_bait$set_counts,
                    sp = "Hooks with bait",
                    years = 2018:2020,
                    indicate_standard = TRUE)


plot_iphc_map(sets_2021,
              sp = NULL,
              years = 2021,
              indicate_standard = TRUE)
```



Can see that 2020 has a few less stations just north of Vancouver Island, but not enough to worry about greatly, and 2021 has kind of done a few of those. The 2021 ones way in in the inlets are not currently flagged as non-standard but will be below (using the 2018 definitions). In fact no stations are flagged for 2021 as non-standard. And the other main issue is that 2021 is doing a random sample of WCVI stations (some of which will become non-standard). AND that there are new stations in the north (and maybe elsewhere) that have never been fished before (as I discovered when updating the one-species vignette and redefining the default axes limits for `plot_BC()`; the version before updating that isaved as iphc-2021-data-all-2021-stations.pdf‘). Will examine those shortly.

Need to look and plot values:

```
sets_2018 <- filter(hooks_with_bait_revert$set_counts,
                    year == 2018)
not_std_2018 <- filter(sets_2018,
                       standard == "N")$station

not_std_2021 <- filter(sets_2021,
                       standard == "N")$station
```

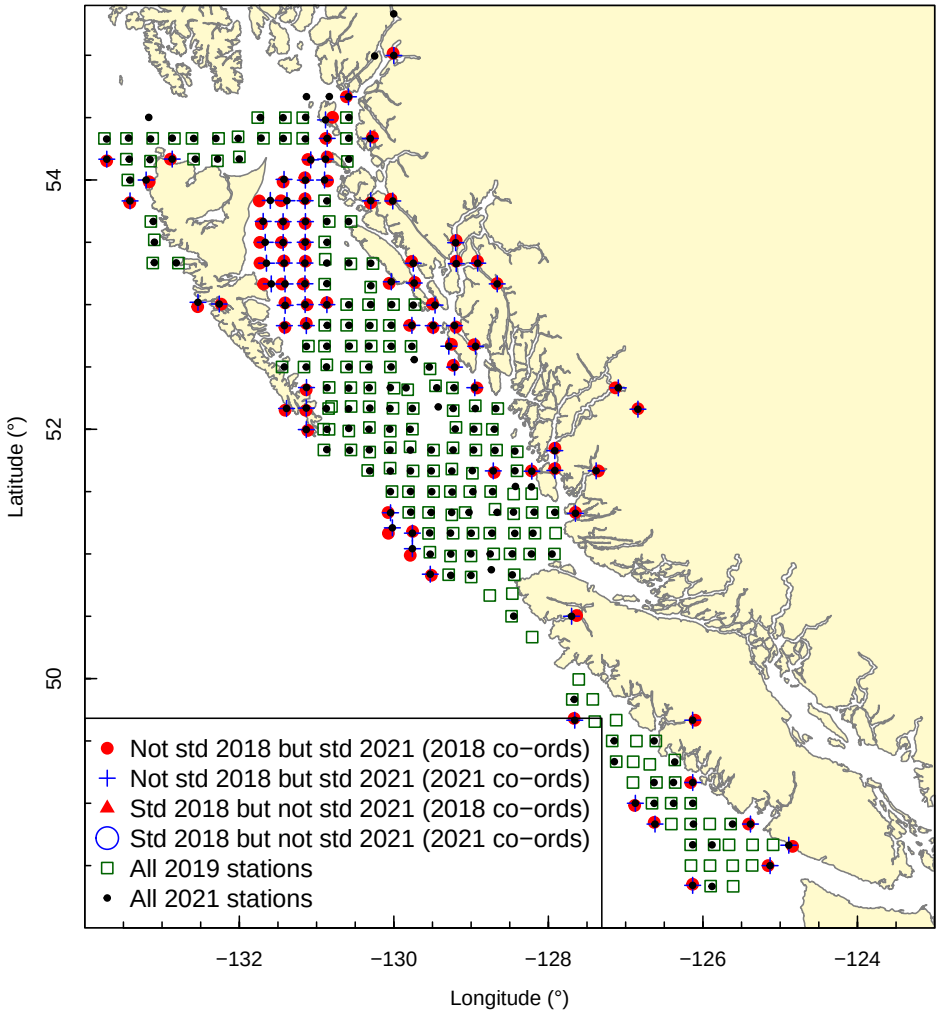
```
# Not standard in both:
not_std_2018_and_2021 <- intersect(not_std_2018, not_std_2021)
not_std_2018_and_2021
> character(0)

length(not_std_2018)
> [1] 131
length(not_std_2021)
> [1] 0
length(not_std_2018_and_2021)
> [1] 0

# 2018 has some east of the map, all non-standard:
filter(hooks_with_bait_revert$set_counts, year == 2018, lon > -124)$standard
>  [1] N N N N N N N N N N N N N N
> Levels: Y N
nrow(filter(hooks_with_bait_revert$set_counts, year == 2018, lon > -124))
> [1] 14

std_in_2018_but_not_std_in_2021 <- intersect(filter(sets_2018,
                                                    standard == "Y")$station,
                                             not_std_2021)
std_in_2018_but_not_std_in_2021
> character(0)

not_std_in_2018_but_std_in_2021 <- intersect(not_std_2018,
                                             filter(sets_2021,
                                                    standard == "Y")$station)
not_std_in_2018_but_std_in_2021
>  [1] "2258" "2261" "2263" "2262" "2265" "2266" "2264" "2269" "2272" "2275"
> [11] "2270" "2267" "2268" "2290" "2293" "2321" "2323" "2326" "2330" "2331"
> [21] "2320" "2316" "2312" "2314" "2308" "2309" "2304" "2302" "2295" "2296"
> [31] "2297" "2299" "2317" "2315" "2334" "2335" "2333" "2332" "2343" "2328"
> [41] "2327" "2324" "2322" "2318" "2305" "2287" "2285" "2288" "2311" "2313"
> [51] "2292" "2289" "2247" "2233" "2232" "2208" "2209" "2213" "2205" "2214"
> [61] "2218" "2221" "2276" "2278" "2273" "2271" "2274" "2277" "2279" "2283"
> [71] "2284" "2280" "2307" "2303" "2301" "2294" "2306" "2298" "2291" "2286"

# setdiff(x, y) - elements in x but not in y
# setdiff(not_std_2018, not_std_2020) - but 2020 fewer coverage so misleading
```

Plot stations not standard in 2018 but standard in 2021, and vice versa, using each years' lats and lons (to verify that they all still agree – i.e., that station numbers have consistent lats and lons), and show 2019 data to check no 'usual' stations are non-standard in 2018 or

2021. Also (for 2021) adding all stations, since this will clearly show the random sampling off WCVI:

```
plot_BC()
points(lat~lon,
       data = filter(sets_2018,
                     station %in% not_std_in_2018_but_std_in_2021),
       col="red",
       pch = 19)

# Do the same but using 2021 station co-ordinates - should overlap:
points(lat~lon,
       data = filter(sets_2021,
                     station %in% not_std_in_2018_but_std_in_2021),
       col="blue",
       pch = 3)

# And for 2020 showed the single station std in 2018 but not 2020, for 2021
# there are none:
points(lat~lon,
       data = filter(sets_2018,
                     station %in% std_in_2018_but_not_std_in_2021),
       col="red",
       pch = 17)
points(lat~lon,
       data = filter(sets_2021,
                     station %in% std_in_2018_but_not_std_in_2021),
       col="blue",
       pch = 1,
       cex = 2)


# Now show all 2019 stations:
points(lat~lon,
       data = filter(hooks_with_bait_revert$set_counts,
                     year == 2019),
       col="darkgreen",
       pch = 0)

# Add all 2021 stations as a small black dot
points(lat~lon,
       data = sets_2021,
       col="black",
       pch = 20,
       cex = 0.8)
```

```
legend("bottomleft",
       legend = c("Not std 2018 but std 2021 (2018 co-ords)",
                  "Not std 2018 but std 2021 (2021 co-ords)",
                  "Std 2018 but not std 2021 (2018 co-ords)",
                  "Std 2018 but not std 2021 (2021 co-ords)",
                  "All 2019 stations",
                  "All 2021 stations"),
              pch = c(19, 3, 17, 1, 0, 20),
              pt.cex = c(1, 1, 1, 2, 1, 0.8),
              col = c("red", "blue", "red", "blue", "darkgreen", "black"))
```



So the co-ordinates look close enough (red circles and blue crosses overlap), none were defined as non-standard in 2021 so there are no red triangles or blue circles, and the green squares for 2019 stations correctly do not overlap with the non-standard 2018 stations. Black dots (2021 stations) with no green squares off WCVI clearly shows the reduced coverage there.

2020 only (there were no non-standard stations defined in raw data for 2021): Check if the one standard station in 2018 but not in 2020 (not fished at all in 2019) appears in any earlier years:

```
# Fails (so not evaluated here) since empty in 2021, and this is corrected
dplyr::filter(hooks_with_bait_revert$set_counts,
              station == std_in_2018_but_not_std_in_2021) %>%
  as.data.frame()
```

For 2020 I worked out it was only fished in 2018 and 2020 so we defined it as non-standard.

So, the conclusion from this section so far is that we should retain the 2018 definitions of standard stations, not the new ones defined in 2021, as we did for 2020.

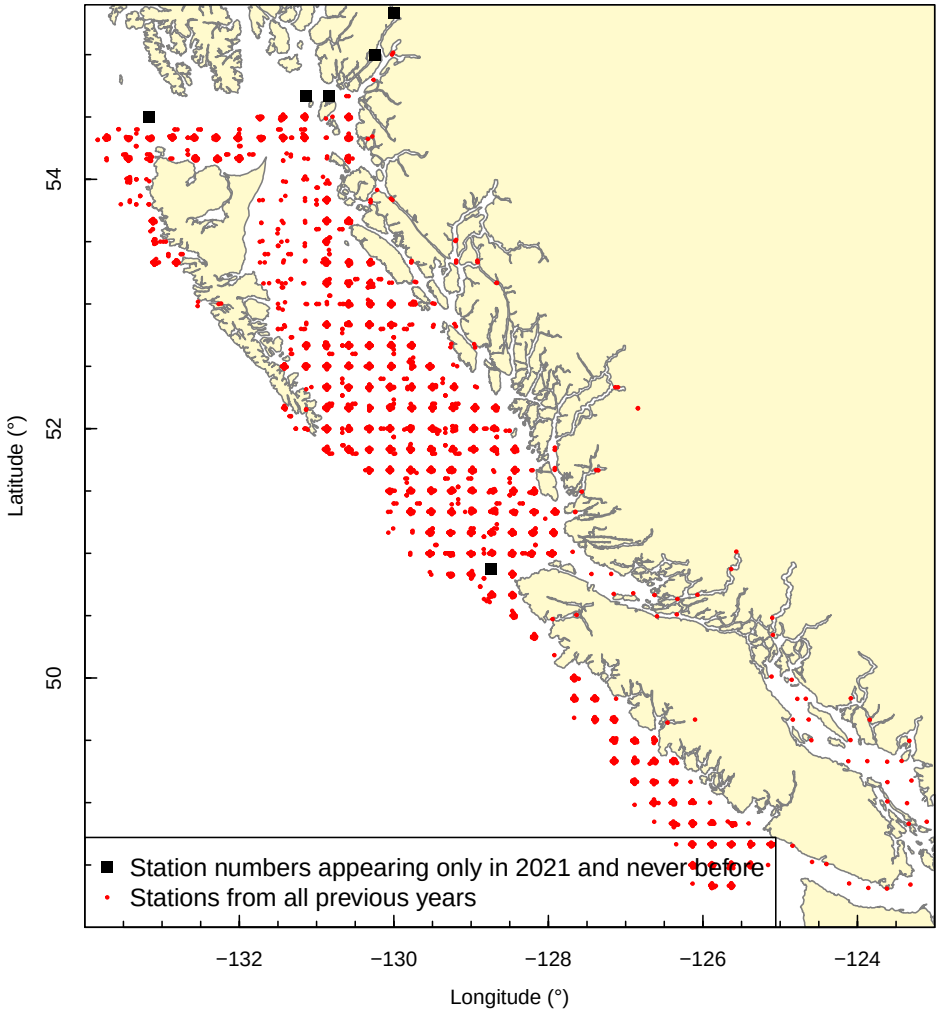
Doing that shortly (in `sets_simp_std_corrected`), but first also look for any new 2021 stations. I hadn't expected any but saw them when doing the one-species vignette, so had to come back to redo this.

```
# yelloweye_rockfish$set_counts is saved in gfiphc, already has 2021 data
#  because I had to come back to redo this .pdf after updating the data, hence
#  need the <2021 here; station codes do change over time, but I think are
#  recently consistent
previous_stations <- dplyr::filter(yelloweye_rockfish$set_counts,
                                   year < 2021)$station %>%
                     unique()
stations_in_2021_only <- dplyr::filter(sets_2021,
                                       !(station %in% previous_stations))
stations_in_2021_only
> # A tibble: 6 x 11
>    year station   lat   lon avgDepth skatesHauled effSkateIPHC soakTimeMinutes
>   <int> <chr>   <dbl> <dbl>    <int>        <dbl>        <dbl>           <dbl>
> 1  2021 2257     50.9 -129.       40            8         7.95             442
> 2  2021 3005     54.5 -133.      118            8         7.95             573
> 3  2021 3008     54.7 -131.      153            8         7.95             424
> 4  2021 3009     54.7 -131.      119            8         7.95             542
> 5  2021 3204     55.0 -130.       67            8         7.95             483
> 6  2021 3210     55.3 -130.      146            8         7.95             622
> # ... with 3 more variables: usable <chr>, purpose <fct>, standard <chr>
```

and plot those stations:

```
plot_BC()
points(lat~lon,
```

```
       data = stations_in_2021_only,
       col = "black",
       pch = 15)

points(lat~lon,
       data = dplyr::filter(yelloweye_rockfish$set_counts,
                            year < 2021),
       col = "red",
       pch = 20,
       cex = 0.4)

legend("bottomleft",
       legend = c("Station numbers appearing only in 2021 and never before",
                  "Stations from all previous years"),
       pch = c(15, 20),
       col = c("black", "red"),
       pt.cex = c(1, 0.4))
```

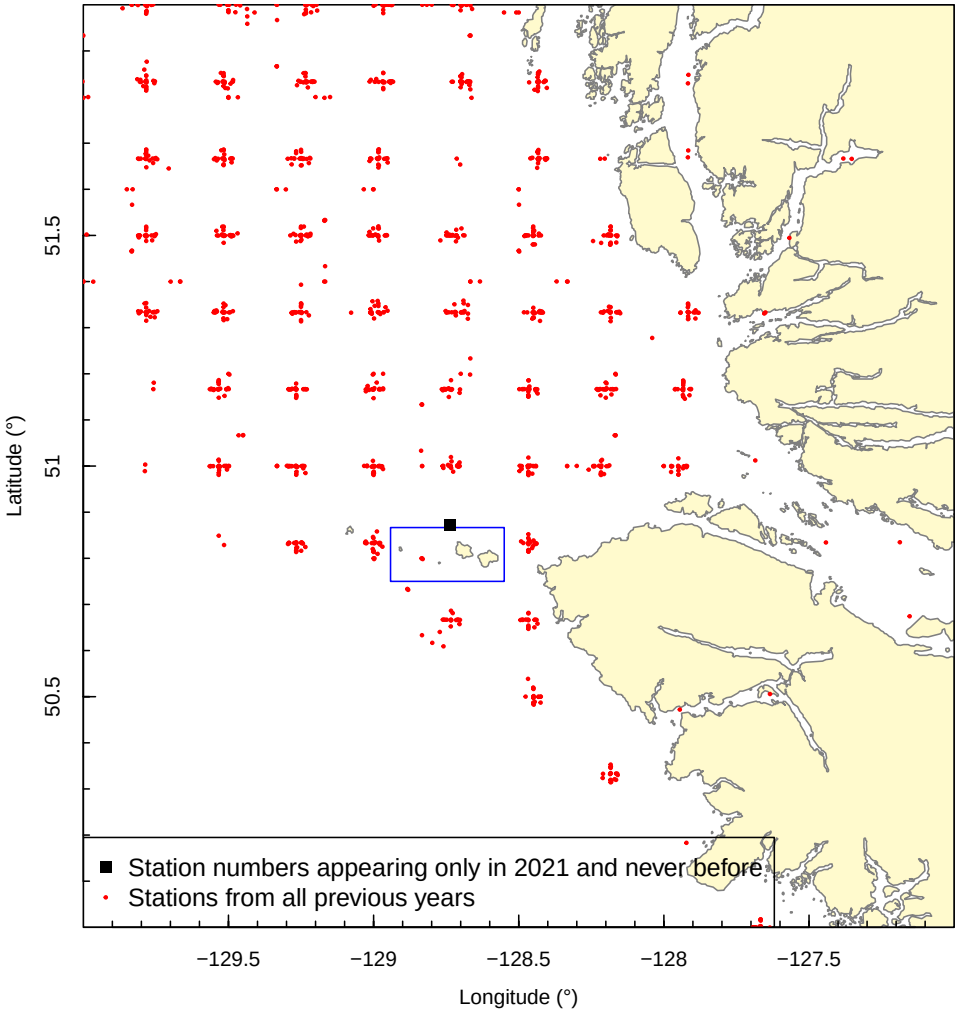


So there are six 2021 stations that have never been fished before! That map suggests that we should call the five northern ones non-standard also, to exclude from the standard Series A-F analyses.

However, Ann-Marie Huang thinks that these stations may have been fished before but considered as part of Area 2C (Alaskan waters). Some waters around there are claimed by both Canada and the US; there's a clear map and explanation in Canada's Unresolved Maritime Boundaries (clickable), which is linked from this Wikipedia article on Dixon Entrance. So there may be earlier data, which are not in gfiphc because such stations would not have been considered Area 2B, which is the area for which the IPHC sent DFO data in the past (and which I used here for recent years to extract from their website). So there may be data available, and if needed it will have to be obtained. Here we will call those five northern newly-fished stations `non-standard`.

For the sixth station off the northwest of Vancouer Island, zooming in and including the Scott Islands Rockfish Conservation Area (clickable) as a blue rectangle shows:

```
plot_BC(xlim = c(-130, -127),
        ylim = c(50, 52))

scott_island_RCA_lon <- -c(128 + 56.5/60, 128 + 33/60)
scott_island_RCA_lat <- c(50 + 45/60, 50 + 52/60)

# rect(xleft, ybottom, xright, ytop, density = NULL, angle = 45,
rect(scott_island_RCA_lon[1],
     scott_island_RCA_lat[1],
     scott_island_RCA_lon[2],
     scott_island_RCA_lat[2],
     border = "blue")

points(lat~lon,
       data = stations_in_2021_only,
       col = "black",
       pch = 15)

points(lat~lon,
       data = dplyr::filter(yelloweye_rockfish$set_counts,
                            year < 2021),
       col = "red",
       pch = 20,
       cex = 0.4)

legend("bottomleft",
       legend = c("Station numbers appearing only in 2021 and never before",
                  "Stations from all previous years"),
       pch = c(15, 20),
       col = c("black", "red"),
       pt.cex = c(1, 0.4))
```



So the new station is just outside the RCA. Presumably in previous years the RCA was avoided as the grid would have put a station in the RCA, close to (or even on) Lanz Island.

It is station 2257 (see above), with a depth of only 40 fathoms, which is not an outlier. For example, for 2013 (depth data for all years is not in gfiphc I don't think):

```
sort(setData2013$avgDepth)
>   [1]  18  21  22  24  25  25  26  27  29  32  32  32  35  35  35  36  36  37
>  [19]  39  39  40  41  41  42  44  44  44  45  45  46  46  46  47  48  48  48
>  [37]  48  48  50  50  51  51  52  52  54  54  54  55  56  56  56  58  58  58
>  [55]  58  58  59  61  62  62  62  63  63  64  66  67  67  67  67  67  71  73
>  [73]  74  74  74  75  75  75  76  76  76  77  78  78  78  79  79  81  81  81
>  [91]  82  82  87  88  88  88  90  91  92  92  93  93  95  96  96  97  97  98
> [109]  98  98  99 101 101 102 102 102 102 103 103 104 105 105 110 111 112 112
```

```
> [127] 113 113 113 114 115 115 116 118 119 120 122 123 123 123 123 124 128 129
> [145] 130 132 135 136 137 139 139 139 140 142 142 144 145 145 150 156 161 183
> [163] 189 190 190 209 215 217 219 256
```

However, since it is a new station and not been used before, we will flag it as `non-standard` (as used for the Series A-F analyses). Also Dana Haggarty says that there is good habitat right close to those islands, but not great further away, and she has used Remotely Operated Vehicles there – it's all sand/gravel/cobble with massive sand waves from the crazy exposure, but perhaps there are pockets of good habitat. So either way (close to an RCA so may be expected to be good for rockfish at least, or not great rockfish habitat) it shouldn't really be included for rockfish species, and in general should be excluded since a new station.

So - retain the 2018 definitions of standard stations (as we did for 2020), and call all six 2021 stations non-standard:

```
sets_simp_std_corrected <- sets_simp_std
summary(as.factor(sets_simp_std_corrected$standard))
>   Y
> 232

sets_simp_std_corrected$standard[sets_simp_std_corrected$station %in%
                                 not_std_in_2018_but_std_in_2021] <- "N"
sets_simp_std_corrected$standard[sets_simp_std_corrected$station %in%
                                 stations_in_2021_only$station] <- "N"
summary(as.factor(sets_simp_std_corrected$standard))
>   N   Y
>  86 146
# cbind(sets_simp_std$standard, sets_simp_std_corrected$standard) # to check them
```

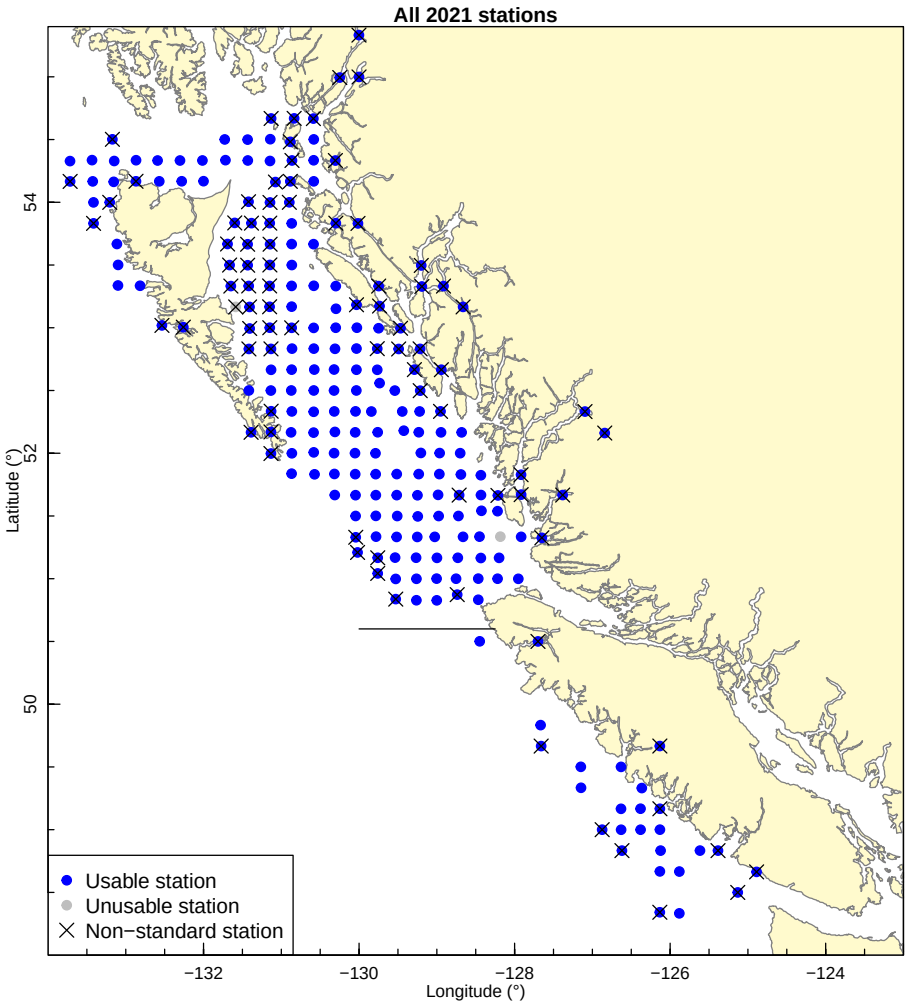
Think I hadn't originally defined them as factors in early code, so keeping them as characters now. Just to vefify that none of the 2018 non-standard stations were fished before 2018:

```
dplyr::filter(hooks_with_bait$set_counts,
              station %in% not_std_2018) %>%
  dplyr::select(year) %>%
  unique()
> # A tibble: 3 x 1
>    year
>   <dbl>
> 1  2018
> 2  2020
> 3  2021
```

Note 2021 won't show up here until .rda objects are resaved in package, at the end of this .pdf (so it will if this .Rmd has already been run, as it has in 2021).

So here are the final station designations for 2021:

```
plot_iphc_map(sets_simp_std_corrected,
              sp = NULL,
              years = 2021,
              indicate_standard = TRUE)
```



Can see that they did 10 random stations off WCVI that we're calling non-standard (because they were never fished before 2018). Which is a bit of a shame as there are only 16 stations left off WCVI for 2021.

2020 (no need to change for 2021): So check which functions need changing, since they create a 'standard' column. These do not need changing: `get_iphc_hooks()` and `get_iphc_skates_info`.

2020: Then `get_iphc_sets_info()` does return `standard`, but the `standard` designation is not saved in GFBio it is saved in `setDataExpansion` in `gfiphc`. So just need to add a line in `IPHC-stations-expanded.R` and then re-save all .rda files. Fixed that, now recreating all .rda files, as per the README.

Species counts

Now get the species counts into the desired format (to match `countData2013` shown earlier). First check that the column names and types haven't changed (they did for set data from 2020 to 2021):

```
counts_raw_2020 <- readr::read_csv("non-halibut-data-2020.csv") %>%
  dplyr::mutate_if(is.character, factor)
>
> -- Column specification --------------------------------------------------------
> cols(
>   `Row number` = col_number(),
>   Year = col_double(),
>   Stlkey = col_double(),
>   Station = col_double(),
>   Setno = col_double(),
>   `IPHC Species Code` = col_double(),
>   `Scientific Name` = col_character(),
>   `Species Name` = col_character(),
>   SampleType = col_character(),
>   HooksFished = col_double(),
>   HooksRetrieved = col_double(),
>   HooksObserved = col_double(),
>   `Number Observed` = col_double()
> )

counts_raw <- readr::read_csv("non-halibut-data-2021.csv") %>%
  dplyr::mutate_if(is.character, factor)
>
> -- Column specification --------------------------------------------------------
> cols(
>   `Row number` = col_number(),
>   Year = col_double(),
>   Stlkey = col_double(),
>   Station = col_double(),
>   Setno = col_double(),
>   `IPHC Species Code` = col_double(),
>   `Scientific Name` = col_character(),
>   `Species Name` = col_character(),
>   SampleType = col_character(),
>   HooksFished = col_double(),
>   HooksRetrieved = col_double(),
>   HooksObserved = col_double(),
>   `Number Observed` = col_double()
> )
```

```
testthat::expect_equal(names(counts_raw_2020),
                       names(counts_raw))

testthat::expect_equal(sapply(counts_raw_2020, typeof),
                       sapply(counts_raw, typeof))
```

Great, nothing changed in the structure.

```
counts_raw
> # A tibble: 1,684 x 13
>    `Row number`  Year  Stlkey Station Setno `IPHC Species Cod~ `Scientific Name`
>           <dbl> <dbl>   <dbl>   <dbl> <dbl>              <dbl> <fct>
>  1            1  2021  2.02e7    2266     1                 31 Sebastes aleutia~
>  2            2  2021  2.02e7    2266     1                 54 Squalus suckleyi
>  3            3  2021  2.02e7    2266     1                143 Raja rhina
>  4            4  2021  2.02e7    2266     1                303 <NA>
>  5            5  2021  2.02e7    2266     1                304 <NA>
>  6            6  2021  2.02e7    2266     1                305 <NA>
>  7            7  2021  2.02e7    2266     1                307 <NA>
>  8            8  2021  2.02e7    2267     2                  2 Atheresthes stom~
>  9            9  2021  2.02e7    2267     2                 27 Anoplopoma fimbr~
> 10           10  2021  2.02e7    2267     2                 54 Squalus suckleyi
> # ... with 1,674 more rows, and 6 more variables: Species Name <fct>,
> #   SampleType <fct>, HooksFished <dbl>, HooksRetrieved <dbl>,
> #   HooksObserved <dbl>, Number Observed <dbl>
summary(counts_raw)
>    Row number          Year          Stlkey            Station
>  Min.   :   1.0   Min.   :2021   Min.   :20210014   Min.   :2002
>  1st Qu.: 421.8   1st Qu.:2021   1st Qu.:20210184   1st Qu.:2090
>  Median : 842.5   Median :2021   Median :20210518   Median :2143
>  Mean   : 842.5   Mean   :2021   Mean   :20210591   Mean   :2188
>  3rd Qu.:1263.2   3rd Qu.:2021   3rd Qu.:20211036   3rd Qu.:2276
>  Max.   :1684.0   Max.   :2021   Max.   :20211134   Max.   :3210
>
>      Setno       IPHC Species Code            Scientific Name
>  Min.   :  1.0   Min.   :  2.0     Squalus suckleyi   :180
>  1st Qu.: 34.0   1st Qu.: 54.0     Anoplopoma fimbria :111
>  Median : 63.0   Median :143.0     Raja rhina         :111
>  Mean   : 64.6   Mean   :169.9     Sebastes ruberrimus: 61
>  3rd Qu.: 89.0   3rd Qu.:304.0     Sebastes babcocki  : 58
>  Max.   :144.0   Max.   :307.0     (Other)            :451
>                                    NA's               :712
>                Species Name  SampleType    HooksFished    HooksRetrieved
>  Empty Hook          :232   20Hook:1684   Min.   :388.0   Min.   :245.0
```

```
>  Hook with Skin      :215                1st Qu.:792.0   1st Qu.:792.0
>  Spiny Dogfish       :180                Median :792.0   Median :792.0
>  Hook with Bait      :161                Mean   :759.2   Mean   :756.9
>  Longnose Skate      :111                3rd Qu.:792.0   3rd Qu.:792.0
>  Sablefish (Blackcod):111                Max.   :808.0   Max.   :808.0
>  (Other)             :674
>  HooksObserved   Number Observed
>  Min.   : 77.0   Min.   :  1.00
>  1st Qu.:160.0   1st Qu.:  1.00
>  Median :160.0   Median :  3.00
>  Mean   :152.9   Mean   : 19.15
>  3rd Qu.:160.0   3rd Qu.: 16.00
>  Max.   :160.0   Max.   :154.00
>
testthat::expect_equal(unique(counts_raw$Year), 2021)  # All 2021
testthat::expect_equal(unique(counts_raw$SampleType), as.factor("20Hook")) # All 20Hook

# This mismatches for 2020, not for 2021:
testthat::expect_equal(length(unique(counts_raw$Station)),
                       length(sets_raw$Station))

unique(counts_raw$"Species Name")
>  [1] Rougheye Rockfish            Spiny Dogfish
>  [3] Longnose Skate               Hook with Skin
>  [5] Empty Hook                   Hook with Bait
>  [7] Bent/Broken/Missing          Arrowtooth Flounder
>  [9] Sablefish (Blackcod)         Inanimate Object
> [11] Solaster sp (starfish)       Lingcod
> [13] Quillback Rockfish           Yelloweye Rockfish
> [15] Big Skate                    Brittle Star
> [17] Pacific Cod                  Walleye Pollock
> [19] Redbanded Rockfish           Glass Sponge
> [21] unident. thornyhead (Idiot)  Silvergray Rockfish
> [23] unident. Sculpin             Copper Rockfish
> [25] Soupfin Shark                Cabezon
> [27] Spotted Ratfish              unident. Crab
> [29] unident. Starfish            Sea Anemone
> [31] Great Sculpin                Petrale Sole
> [33] Shortspine Thornyhead        Wolf-Eel
> [35] Shortraker Rockfish          Fish-eating Star
> [37] Bocaccio                     Canary Rockfish
> [39] Red Tree Coral               Sea Pen
> [41] Jellyfish                    Dungeness Crab
> [43] Stylaster campylecus (coral) Basketstar
```

```
> [45] Sandpaper Skate              unident. Sponge
> [47] Unident. Salmon              Sleeper Shark
> [49] Unident. Rockfish            unident. organic matter
> [51] unident. Coral               Blackspotted Rockfish
> [53] Tiger Rockfish               Gorgonian coral
> [55] Blue Shark                   Yellowmouth Rockfish
> [57] Sun Sea Star                 China Rockfish
> [59] Aleutian Skate               Salmon Shark
> [61] Sea Cucumber                 Flathead Sole
> [63] Giant Pacific Octopus        Sea Whip
> [65] Rock Sole
> 65 Levels: Aleutian Skate Arrowtooth Flounder ... Yellowmouth Rockfish
```

Here's what was seen in 2020 but not 2021, and vice versa:

```
# Seen in 2020 not 2021
setdiff(unique(counts_raw_2020$"Species Name"),
        unique(counts_raw$"Species Name"))
> [1] "Sand Dab"           "Oregon Rock Crab"   "Sea Urchin"
> [4] "Octopus"            "Gastropod"          "Sunflower Sea Star"
# Seen in 2021 not 2020
setdiff(unique(counts_raw$"Species Name"),
        unique(counts_raw_2020$"Species Name"))
>  [1] "Inanimate Object"             "Solaster sp (starfish)"
>  [3] "Walleye Pollock"              "Cabezon"
>  [5] "unident. Crab"                "Great Sculpin"
>  [7] "Jellyfish"                    "Dungeness Crab"
>  [9] "Stylaster campylecus (coral)" "Sandpaper Skate"
> [11] "Unident. Salmon"              "Unident. Rockfish"
> [13] "unident. organic matter"      "Gorgonian coral"
> [15] "Sun Sea Star"                 "China Rockfish"
> [17] "Salmon Shark"                 "Sea Cucumber"
> [19] "Flathead Sole"                "Sea Whip"
> [21] "Rock Sole"
```

Presumably Sun Sea Star and Sunflower Sea Star are the same. Will mention this later on.

Note that halibut are not included in these counts:

```
dplyr::filter(counts_raw, "Species Name" == "Pacific Halibut")
> # A tibble: 0 x 13
> # ... with 13 variables: Row number <dbl>, Year <dbl>, Stlkey <dbl>,
> #   Station <dbl>, Setno <dbl>, IPHC Species Code <dbl>, Scientific Name <fct>,
> #   Species Name <fct>, SampleType <fct>, HooksFished <dbl>,
> #   HooksRetrieved <dbl>, HooksObserved <dbl>, Number Observed <dbl>
# Should be: dplyr::filter(counts_raw, `Species Name` == as.character("Pacific
```

```
#                               Halibut")) %>% as.data.frame()
# Still 0 in 2021
```

which I presume explains why total number of counts for a station does not add up to `HooksObserved`. See later for halibut calculations.

2020 only: Need to remove the HAN records for the twice-fished station, which turns out to be set number 4 for station 2104:

```
dplyr::filter(counts_raw, Station == twice_fished) %>%
  dplyr::select(c("Station", "Setno", "Species Name",
                  "Number Observed")) %>%
    as.data.frame()
> [1] Station         Setno           Species Name    Number Observed
> <0 rows> (or 0-length row.names)

dplyr::filter(sets_raw, Station == twice_fished)
> # A tibble: 0 x 44
> # ... with 44 variables: Row number <dbl>, Year <dbl>, Stlkey <dbl>,
> #   Vessel code <fct>, Station <dbl>, Setno <dbl>, IPHC Reg Area <fct>,
> #   IPHC Stat Area <dbl>, IPHC Charter Region <fct>, Purpose Code <fct>,
> #   Date <fct>, Eff <fct>, Ineffcde <fct>, BeginLat <dbl>, BeginLon <dbl>,
> #   BeginDepth (fm) <dbl>, EndLat <dbl>, EndLon <dbl>, EndDepth (fm) <dbl>,
> #   MidLat fished <dbl>, MidLon fished <dbl>, AvgDepth (fm) <dbl>,
> #   Lat - Grid target <dbl>, Lon - Grid target <dbl>,
> #   O32 Pacific halibut count <dbl>, U32 Pacific halibut count <dbl>,
> #   O32 Pacific halibut weight <dbl>, U32 Pacific halibut weight <dbl>,
> #   No. skates set <dbl>, No. skates hauled <dbl>, Avg no. hook/skate <dbl>,
> #   Effective skates hauled <dbl>, Soak time (min.) <dbl>, Profiler Lat <dbl>,
> #   Profiler Lon <dbl>, Profiler Bottom Depth (m) <dbl>, Temp C <dbl>,
> #   Max Pressure (db) <dbl>, pH <dbl>, Salinity PSU <dbl>, Sigma-t <lgl>,
> #   Oxygen_ml <dbl>, Oxygen_umol <lgl>, Oxygen_sat <lgl>
```

So for 2020 had to use that here to remove the species counts for that vessel (note that vessel code is not in `counts_raw`), just commenting that part out for 2021:

```
dplyr::filter(counts_raw,
              Station == twice_fished & Setno == 4)
> # A tibble: 0 x 13
> # ... with 13 variables: Row number <dbl>, Year <dbl>, Stlkey <dbl>,
> #   Station <dbl>, Setno <dbl>, IPHC Species Code <dbl>, Scientific Name <fct>,
> #   Species Name <fct>, SampleType <fct>, HooksFished <dbl>,
> #   HooksRetrieved <dbl>, HooksObserved <dbl>, Number Observed <dbl>

# So just keep these:
# dplyr::filter(counts_raw,
```

```
#               !(Station == twice_fished & Setno == 4))

#countData2020_no_halibut <- dplyr::filter(counts_raw,
#                                  !(Station == twice_fished & Setno == 4)) %>%
# Seems that can't just keep using that even if twice_fished = NA
countData2021_no_halibut <- counts_raw %>%
  dplyr::select(year = Year,
                station = Station,
                spNameIPHC = "Species Name",
                specCount = "Number Observed") %>%
  arrange(station) %>%
  dplyr::mutate(year = as.integer(year),
                station = as.character(station),
                spNameIPHC = as.character(spNameIPHC),
                specCount = as.integer(specCount))

testthat::expect_equal(names(countData2013), names(countData2021_no_halibut))
countData2021_no_halibut
> # A tibble: 1,684 x 4
>     year station spNameIPHC          specCount
>    <int> <chr>   <chr>                   <int>
>  1  2021 2002    Spiny Dogfish              29
>  2  2021 2002    Empty Hook                 50
>  3  2021 2002    Bent/Broken/Missing         1
>  4  2021 2010    Spiny Dogfish               3
>  5  2021 2010    Longnose Skate              2
>  6  2021 2010    Hook with Skin              2
>  7  2021 2010    Empty Hook                 61
>  8  2021 2011    Spiny Dogfish              26
>  9  2021 2011    Empty Hook                 52
> 10  2021 2011    Hook with Bait              1
> # ... with 1,674 more rows
summary(countData2021_no_halibut)
>       year        station           spNameIPHC          specCount
>  Min.   :2021   Length:1684        Length:1684        Min.   :  1.00
>  1st Qu.:2021   Class :character   Class :character   1st Qu.:  1.00
>  Median :2021   Mode  :character   Mode  :character   Median :  3.00
>  Mean   :2021                                         Mean   : 19.15
>  3rd Qu.:2021                                         3rd Qu.: 16.00
>  Max.   :2021                                         Max.   :154.00
```

Hooks observed and retrieved

Now, obtain the numbers of hooks observed and retrieved from `counts_raw`, to then merge into the set details:

```
# hook_details <- dplyr::filter(counts_raw,
#                               !(Station == twice_fished & Setno == 4)) %>%
hook_details <- counts_raw %>%
  dplyr::group_by(Station) %>%
  dplyr::summarise(year = unique(Year),
                   hooksRetr = unique(HooksRetrieved),
                   hooksObs = unique(HooksObserved)) %>%
  dplyr::rename(station = Station) %>%
  dplyr::ungroup() %>%
  arrange(station) %>%
  dplyr::mutate(year = as.integer(year),
                station = as.character(station))

hook_details
> # A tibble: 232 x 4
>    station  year hooksRetr hooksObs
>    <chr>   <int>     <dbl>    <dbl>
>  1 2002     2021       396       80
>  2 2010     2021       396       80
>  3 2011     2021       396       80
>  4 2012     2021       396       80
>  5 2014     2021       396       80
>  6 2016     2021       392       79
>  7 2017     2021       396       80
>  8 2018     2021       392       80
>  9 2019     2021       396       80
> 10 2020     2021       396       80
> # ... with 222 more rows

testthat::expect_equal(sets_simp_std_corrected$station, hook_details$station)
```

So now need to get the hook details into the set details, and keep columns as for `setData2013` but also with `standard`, and may as well keep `hooksRetr` and `hooksObs`:

```
setData2021 <- dplyr::left_join(sets_simp_std_corrected,
                                hook_details,
                                by = c("year", "station")) %>%
  dplyr::mutate(E_it20 = effSkateIPHC * hooksObs / hooksRetr) %>%
  dplyr::select(year,
                station,
                lat,
```

```
                lon,
                avgDepth,
                effSkateIPHC,
                E_it20,
                usable,
                standard,
                hooksRetr,
                hooksObs) %>%
  dplyr::mutate(year = as.integer(year),
                station = as.character(station),
                avgDepth = as.integer(avgDepth),
                usable = as.character(usable),
                standard = as.factor(standard))
setData2021
> # A tibble: 232 x 11
>     year station   lat   lon avgDepth effSkateIPHC E_it20 usable standard
>    <int> <chr>   <dbl> <dbl>    <int>        <dbl>  <dbl> <chr>  <fct>
>  1  2021 2002     48.3 -126.      195         3.98  0.804 Y      Y
>  2  2021 2010     48.7 -126.       40         3.98  0.804 Y      Y
>  3  2021 2011     48.7 -126.       77         3.98  0.804 Y      Y
>  4  2021 2012     48.8 -126.       24         3.98  0.804 Y      Y
>  5  2021 2014     48.8 -126.       56         3.98  0.804 Y      Y
>  6  2021 2016     49.0 -126.       37         3.93  0.792 Y      Y
>  7  2021 2017     49.0 -126.       75         3.98  0.804 Y      Y
>  8  2021 2018     49.0 -127.      127         3.93  0.802 Y      Y
>  9  2021 2019     49.2 -126.       47         3.98  0.804 Y      Y
> 10  2021 2020     49.2 -127.       68         3.98  0.804 Y      Y
> # ... with 222 more rows, and 2 more variables: hooksRetr <dbl>, hooksObs <dbl>
testthat::expect_equal(names(setData2013), names(setData2021)[1:ncol(setData2013)])
summary(setData2021)
>       year        station              lat             lon
>  Min.   :2021   Length:232         Min.   :48.33   Min.   :-133.7
>  1st Qu.:2021   Class :character   1st Qu.:51.50   1st Qu.:-131.1
>  Median :2021   Mode  :character   Median :52.33   Median :-130.0
>  Mean   :2021                      Mean   :52.31   Mean   :-129.9
>  3rd Qu.:2021                      3rd Qu.:53.50   3rd Qu.:-128.9
>  Max.   :2021                      Max.   :55.33   Max.   :-124.9
>     avgDepth      effSkateIPHC       E_it20          usable          standard
>  Min.   : 10.0   Min.   :2.460   Min.   :0.7731   Length:232         N: 86
>  1st Qu.: 44.0   1st Qu.:7.950   1st Qu.:1.6060   Class :character   Y:146
>  Median : 76.0   Median :7.950   Median :1.6061   Mode  :character
>  Mean   : 89.6   Mean   :7.431   Mean   :1.5011
>  3rd Qu.:119.2   3rd Qu.:7.950   3rd Qu.:1.6061
>  Max.   :334.0   Max.   :8.110   Max.   :1.6062
```

```
>    hooksRetr        hooksObs
>  Min.   :245.0   Min.   : 77.0
>  1st Qu.:792.0   1st Qu.:160.0
>  Median :792.0   Median :160.0
>  Mean   :740.2   Mean   :149.5
>  3rd Qu.:792.0   3rd Qu.:160.0
>  Max.   :808.0   Max.   :160.0
```

Pacific Halibut counts

As noted above, the data extraction for the counts is for all non-halibut species. We still want the halibut counts for just the first 20 hooks – the `data_for_all_species` vignette (for data up to 2019) shows that the 20-hook and full hook counts (Series A and B) are very similar when rescaled, and the rescaling is miniscule with $G_A/G_B = 1.005$. So this justifies sticking with 20-hook counts for halibut, even though the full data are available for all sets, given it is a halibut survey. (Using all hooks for all years could be done, but would be a lot of new code).

There are two options for getting halibut counts for the first 20 hooks (given we don't have hook-by-hook data, though it could probably be obtained just maybe not from the IPHC website).

Option 1.

Take the halibut counts for all the hooks (which we have in `sets_raw` and subsequent objects) and create `N_it20_halibut_est = E_it20 / E_it * N_it`, or equivalently just `N_it20_halibut_est = hooksObs / hooksRetr * N_it`. Note that observed refers to observed for non-halibut species (presumably `hooksRetr` works for halibut). Not strictly the first 20 hooks, but is a rescaling. But will not guarantee integer values.

```
setData2021_and_halibut <-
  dplyr::left_join(setData2021,
                   dplyr::select(sets_simp_std_corrected,
                                 c(station,
                                   U32halibut,
                                   O32halibut)),
                   by = "station") %>%
  dplyr::mutate(N_it_halibut = U32halibut + O32halibut,
                N_it20_halibut_opt_1 = hooksObs / hooksRetr * N_it_halibut)
setData2021_and_halibut %>% dplyr::select(station,
                                          N_it_halibut,
                                          N_it20_halibut_opt_1)
> # A tibble: 232 x 3
```

```
>    station N_it_halibut N_it20_halibut_opt_1
>    <chr>          <dbl>                <dbl>
>  1 2002               0                0
>  2 2010              37                7.47
>  3 2011               1                0.202
>  4 2012              43                8.69
>  5 2014              21                4.24
>  6 2016              10                2.02
>  7 2017               5                1.01
>  8 2018              14                2.86
>  9 2019               6                1.21
> 10 2020              13                2.63
> # ... with 222 more rows
```

Option 2.

Add all the 20-hook counts for a set (which include `Hook with Skin` etc.) and compare with `hooksObs`. The latter is higher (or equal), and the difference is halibut (as the only **non** non-halibut' species). Compare with the results from option 1. If close then use option 2, since it will be just be halibut counts and gives an integer number, and is based on the first 20 hooks.

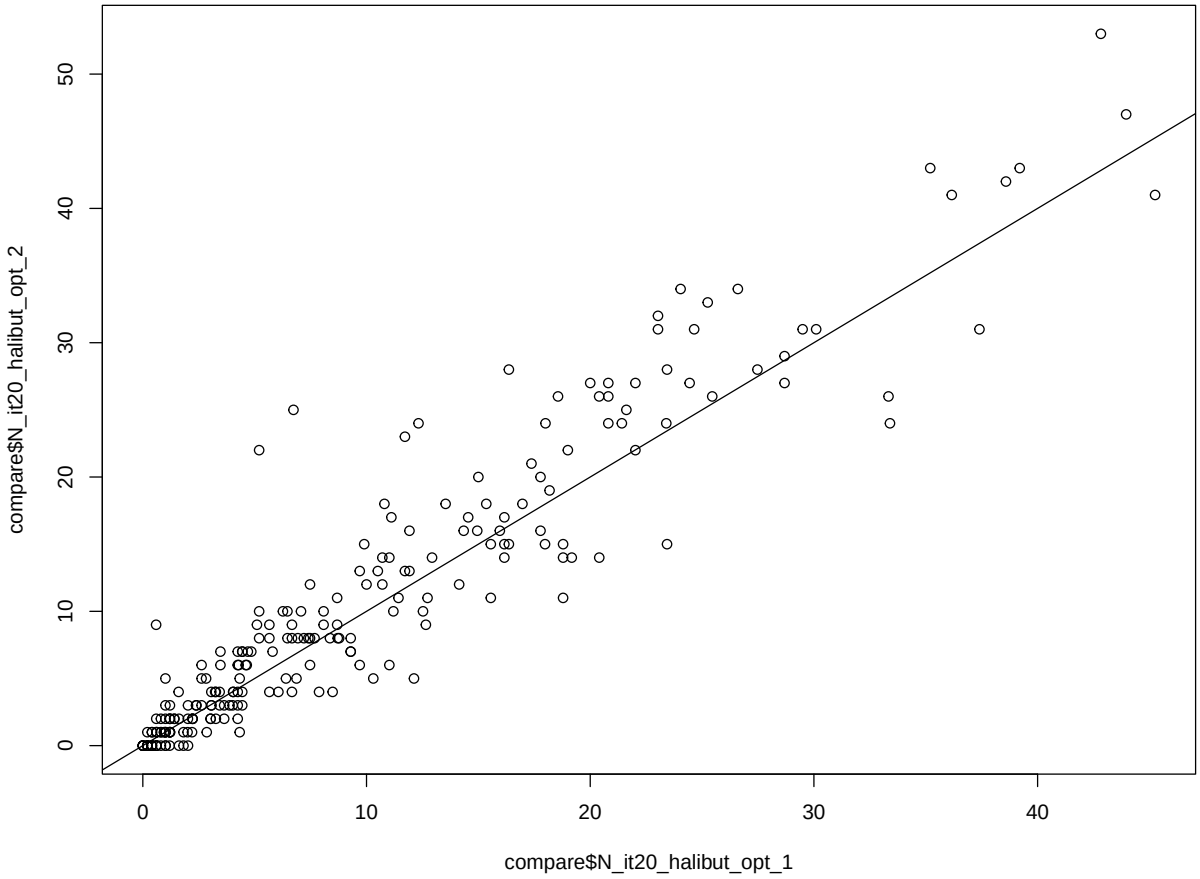
Add counts for each set:

```
counts_20 <- countData2021_no_halibut %>%
  dplyr::group_by(station) %>%
  dplyr::summarise(non_halibut = sum(specCount)) %>%
  dplyr::ungroup()
counts_20
> # A tibble: 232 x 2
>    station non_halibut
>    <chr>         <int>
>  1 2002             80
>  2 2010             68
>  3 2011             80
>  4 2012             69
>  5 2014             77
>  6 2016             77
>  7 2017             79
>  8 2018             79
>  9 2019             79
> 10 2020             77
> # ... with 222 more rows
```

Now join the two options together to calculate `N_it20_halibut_opt_2` and then compare

the two estimates of N_it20_halibut:

```
compare <-
  dplyr::left_join(setData2021_and_halibut,
                   counts_20,
                   by = "station") %>%
  dplyr::mutate(N_it20_halibut_opt_2 = hooksObs - non_halibut,
                N_it20_opt_1_over_opt_2 = N_it20_halibut_opt_1 / N_it20_halibut_opt_2) %
  dplyr::select(year,
                station,
                usable,
                N_it20_halibut_opt_1,
                N_it20_halibut_opt_2,
                N_it20_opt_1_over_opt_2)
compare$spNameIPHC <- "Pacific Halibut"
compare
> # A tibble: 232 x 7
>     year station usable N_it20_halibut_op~ N_it20_halibut_op~ N_it20_opt_1_over~
>    <int> <chr>   <chr>               <dbl>              <dbl>              <dbl>
>  1  2021 2002    Y                   0                      0            NaN
>  2  2021 2010    Y                   7.47                  12              0.623
>  3  2021 2011    Y                   0.202                  0            Inf
>  4  2021 2012    Y                   8.69                  11              0.790
>  5  2021 2014    Y                   4.24                   3              1.41
>  6  2021 2016    Y                   2.02                   2              1.01
>  7  2021 2017    Y                   1.01                   1              1.01
>  8  2021 2018    Y                   2.86                   1              2.86
>  9  2021 2019    Y                   1.21                   1              1.21
> 10  2021 2020    Y                   2.63                   3              0.875
> # ... with 222 more rows, and 1 more variable: spNameIPHC <chr>

plot(compare$N_it20_halibut_opt_1, compare$N_it20_halibut_opt_2)
abline(a = 0, b = 1)
```




```
cor(compare$N_it20_halibut_opt_1,
    compare$N_it20_halibut_opt_2)
> [1] 0.9428069
```

So this is the right approach and correlation coefficient is high, though numbers not quite as close as may have thought. But these data are used for aggregating across all stations in a year (and any further analyses on halibut for management purposes should be done using the full halibut data anyway – we wouldn't really need that). And the means aren't too bad:

```
mean(compare$N_it20_halibut_opt_1)
> [1] 9.604416
mean(compare$N_it20_halibut_opt_2)
> [1] 10.55603
```

So either of these would work. So use option 2 since gives an integer count:

```
compare$N_it20_halibut_opt_2
>   [1]  0 12  0 11  3  2  1  1  1  3  2  2  2  3  5  1  8  0  0 16 26  5 17  4  1
>  [26]  3 10  2 41 14  3  0  2  4 16 15  8  9  2 10 12 10 20 10 11 42 13  7  7 22
```

```
>  [51] 14  6  4  8  7 53 22  0 31 41  8  5  9 25  3  8  8 22  8  8 27  0  8  9 24
>  [76]  3 34  1 12  0  8  7 43 33 24  3  0  7 12 32 47  6  2 24 26  8  3  2 18  0
> [101]  2 18  5 27  9 28  5 17 31  0 23  1 10 43 16 27 18 11 15  4  6 14 27  2 34
> [126]  6 13 27 31 24  7  9 15  6 15  4 13  2  6  4 25 21 14  7 31  1  0  1  1  0
> [151]  0  0  5  0  0  0 15  1  0  0  1  0  2  2 14 11  4  3 28  1  4  4  1 26 10
> [176]  5  4 18 15  9  7 14  8  9  5 19 15 31 11  2  0  2  6  7 26  0  1  3  8 20
> [201]  1  1  0  3 16  0  6  1  0  4  4 14  1 29 13 24 17 28  6 16 26  2  4  3  0
> [226] 10  3 24  2  1  1  4
countData2021_halibut <- dplyr::select(compare,
                                       year,
                                       station,
                                       spNameIPHC,
                                       specCount = N_it20_halibut_opt_2) %>%
  dplyr::mutate(specCount = as.integer(specCount))
countData2021 <- rbind(countData2021_no_halibut,
                       countData2021_halibut) %>%
  dplyr::arrange(station)
# First time running, called the above countData2020_NEW to check remaining data didn'
# expect_equal(countData2020, filter(countData2020_NEW, spNameIPHC !=
#                                                       "Pacific Halibut"))
```

Note that for 2021 this does give zeros for Pacific Halibut (the only species that will have a zero, because we have a value for each station because zero counts are in the original `sets_raw`):

```
summary(dplyr::filter(countData2021,
                      spNameIPHC == "Pacific Halibut"))
>       year        station           spNameIPHC          specCount
>  Min.   :2021   Length:232         Length:232         Min.   : 0.00
>  1st Qu.:2021   Class :character   Class :character   1st Qu.: 2.00
>  Median :2021   Mode  :character   Mode  :character   Median : 7.00
>  Mean   :2021                                         Mean   :10.56
>  3rd Qu.:2021                                         3rd Qu.:15.25
>  Max.   :2021                                         Max.   :53.00
unique(dplyr::filter(countData2021, specCount == 0)$spNameIPHC)
> [1] "Pacific Halibut"
```

Check species names

The file `inst/extdata/iphc-spp-names.csv` contains species common names (as used for gfsynopsis, and a few extra like `unidentified skate`) and the IPHC common name. The function `check_iphc_spp_name()` has a list of non-groundfish species that are automatically ignored. These first results are from running these functions *before* updating anything, so the results are hardwired here (chunks are not evaluated). Then we update the species list and

re-run the functions.

These are IPHC names that are not given in `iphc-spp-names.csv` (automatically ignoring obvious ones that are listed in the function), for years up to 2020 (since not updated code yet):

```
check_iphc_spp_name()
##  [1] "Unidentified Shark"          "Unident. Rockfish"
##  [3] "unident. thornyhead (Idiot)" "Grenadier (Rattails)"
##  [5] "Miscellaneous Shark"         "Eelpout"
##  [7] "unident. Roundfish"          "unident. Sculpin"
##  [9] "Unident. Flatfish"           "Greenland Turbot"
## [11] "unident. Hagfish"            "Starry Skate"
## [13] "Black Skate"                 "Brittle Star"
## [15] "Glass Sponge"                "Basketstar"
## [17] "Blackspotted Rockfish"
```

These are the ones just for the new 2021 data:

```
check_iphc_spp_name(countData2021)
##  [1] "Basketstar"                   "unident. thornyhead (Idiot)"
##  [3] "Sandpaper Skate"              "Sea Whip"
##  [5] "Stylaster campylecus (coral)" "Brittle Star"
##  [7] "unident. Sculpin"             "Glass Sponge"
##  [9] "Sun Sea Star"                 "Salmon Shark"
## [11] "Jellyfish"                    "Great Sculpin"
## [13] "Cabezon"                      "Unident. Salmon"
## [15] "Unident. Rockfish"            "unident. organic matter"
## [17] "Dungeness Crab"               "Blackspotted Rockfish"
```

There were only six for 2020 though (a lot more for 2021):

```
check_iphc_spp_name(countData2020)
## [1] "unident. thornyhead (Idiot)" "Brittle Star"
## [3] "Glass Sponge"                "Basketstar"
## [5] "Blackspotted Rockfish"       "unident. Sculpin"
```

For 2020 I said that only the Thornyhead and Blackspotted Rockfish are likely of interest (Issues #17 and #18). And the sharks from the earlir list. So look at just the new ones in 2021 that aren't in 2020 or any previous year:

```
setdiff(check_iphc_spp_name(countData2021),
        check_iphc_spp_name())
# Before updating anything this gives:
# [1] "Sandpaper Skate"              "Sea Whip"
# [3] "Stylaster campylecus (coral)" "Sun Sea Star"
# [5] "Salmon Shark"                 "Jellyfish"
# [7] "Great Sculpin"                "Cabezon"
```

```
# [9] "Unident. Salmon"                "unident. organic matter"
# [11] "Dungeness Crab"
```

Of these, Sandpaper Skate, Salmon Shark, Great Sculpin, and Cabezon are in gfsynopsis but have not been designated an `iphc_common_name` in `iphc-spp-names.csv` (have to do that manually). Though Sandpaper Skate, Salmon Shark, and Great Sculpin do show up has having IPHC data in 2019 gfsynopsis report, but looks like only data from GFBio, looking carefully at the `data_for_all_species` vignette for 2020: http://htmlpreview.github.io/?https://github.com/pbs-assess/gfiphc/blob/master/vignettes/data_for_all_species.html They did not have 2020 IPHC data, but do for 2021 (GS had 1995 and 1996 as zeros; don't think others did). Cabezon has no previous data.

So, need to add those species to `iphc-spp-names.csv`, which may discover some old data for those years when I redo the vignettes, as it seems strange that they never seem to show up in the 20-hook-only data, just in GFBio.

Also add these to the `ignore_obvious` list in `check_iphc_spp_name()`:

"Sea Whip", "Stylaster campylecus (coral)", "Sun Sea Star", "Jellyfish", "Unident. Salmon", "unident. organic matter", "Dungeness Crab"

That list already had Sunflower Sea Star in it, presumably the same as Sun Sea Star.

Then redoing those above commands with updated code gives this, where some species are returned because they are not non-groundfish ones (or Brittle Star or Glass Sponge which we also kept in the past) that we want to automatically ignore:

```
check_iphc_spp_name(countData2021)
> [1] "Basketstar"                  "unident. thornyhead (Idiot)"
> [3] "Brittle Star"                "unident. Sculpin"
> [5] "Glass Sponge"                "Unident. Rockfish"
> [7] "Blackspotted Rockfish"
# That still retains some we may want to think about further at some point, but
#  these are all in the overall list for all years:
setdiff(check_iphc_spp_name(countData2021),
        check_iphc_spp_name())
> character(0)
check_iphc_spp_name()
>  [1] "Unidentified Shark"          "Unident. Rockfish"
>  [3] "unident. thornyhead (Idiot)" "Grenadier (Rattails)"
>  [5] "Miscellaneous Shark"         "Eelpout"
>  [7] "unident. Roundfish"          "unident. Sculpin"
>  [9] "Unident. Flatfish"           "Greenland Turbot"
> [11] "unident. Hagfish"            "Starry Skate"
> [13] "Black Skate"                 "Brittle Star"
> [15] "Glass Sponge"                "Basketstar"
> [17] "Blackspotted Rockfish"
```

Save data sets

```
usethis::use_data(countData2021,
                  overwrite = TRUE)
> v Setting active project to 'C:/andy18/github/gfiphc'
> v Saving 'countData2021' to 'data/countData2021.rda'
> * Document your data (see 'https://r-pkgs.org/data.html')

usethis::use_data(setData2021,
                  overwrite = TRUE)
> v Saving 'setData2021' to 'data/setData2021.rda'
> * Document your data (see 'https://r-pkgs.org/data.html')
```

Add descriptions for new years in `R/data.R`.