Check the 2023 data and incorporate it into gfiphc

Andrew Edwards/Sean Anderson

Last compiled on 23 May, 2024

First read the survey section of the latest IPHC report (here) for background if it's published (2022 report published 28th March 2023). In 2022 (as for 2020 and 2021) 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 2022'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 2022 survey are shown in this IPHC sampling manual (click here); page 4 again has Vancouver [Island] Outside, showing that not all stations were intended to be fished there. The 2022 annual report notes issues with the survey (such as ship availability due to higher Sablefish quota, crew availability) that may have reduced the number of stations, though this was referring to the full survey so need to do the maps here to understand. After doing these analyses there are 171 usable stations in 2022, though only 113 are standard; some being up in inlets, but for spatiotemporal analyses we can include the useful non-standard ones.

For comparison first look at 2013 data included in gfiphc:

```
library(devtools)
> Loading required package: usethis
```

```
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
> # i 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
> # i 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:

2023: https://www.iphc.int/uploads/2024/03/IPHC-2024-VSM01.pdf

https://www.iphc.int/data/fiss-survey-raw-survey-data/ as of 2023

1. Year Range – 2022 to 2022.
2. Area 2B
3. Purpose Codes – All
4. IPHC Charter Regions – All
5. Maps – Nothing

6. Select non-Pacific halibut species – deselect All (yes, deselect).

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-2023.xlsx`. Open in Excel and Export as .csv, `set-and-halibut-data-2023.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 choose non-halibut in the CrossTab pop up)), and save as `non-halibut-data-2023.xlsx` and export as .csv in Excel, `non-halibut-data-2023.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(here::here("data-raw/set-and-halibut-data-2023.csv")) %>%
  dplyr::mutate_if(is.character, factor)
> Rows: 269 Columns: 45
> -- Column specification --------------------------------------------------------
> Delimiter: ","
> chr  (8): Vessel code, Gear, IPHC Reg Area, IPHC Charter Region, Purpose Cod...
> dbl (23): Row number, Year, Stlkey, Station, Setno, IPHC Stat Area, BeginLat...
> num  (3): O32 Pacific halibut weight, U32 Pacific halibut weight, Soak time ...
> lgl (11): Profiler Lat, Profiler Lon, Profiler Bottom Depth (m), Temp C, Max...
>
> i Use `spec()` to retrieve the full column specification for this data.
> i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

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(here::here("data-raw/set-and-halibut-data-2020.csv")) %
  dplyr::mutate_if(is.character, factor)
> Rows: 198 Columns: 33
> -- Column specification --------------------------------------------------------
> Delimiter: ","
> chr  (6): Vessel code, IPHC Reg Area, IPHC Charter Region, Purpose, Date, Eff
> dbl (24): Row number, Year, Stlkey, Station, Setno, IPHC Stat Area, BeginLat...
> num  (2): O32 Pacific halibut weight, U32 Pacific halibut weight
> lgl  (1): Ineffcde
>
> i Use `spec()` to retrieve the full column specification for this data.
> i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

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

```
# setdiff commands give the columns

# 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] "Gear"                      "Purpose Code"
>  [3] "Profiler Lat"              "Profiler Lon"
>  [5] "Profiler Bottom Depth (m)" "Temp C"
>  [7] "Max Pressure (db)"         "pH"
>  [9] "Salinity PSU"              "Sigma-t"
> [11] "Oxygen_ml"                 "Oxygen_umol"
> [13] "Oxygen_sat"
```

```
# For 2021 and 2022 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
>           269
```

```
testthat::expect_equal(typeof(sets_raw_2020$Purpose),
                       typeof(sets_raw$"Purpose Code"))
```

Those extra columns in 2021 and 2022 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"
# Get above error in 2021 and 2022, so for 2022 use the same wrangling as 2021
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
> # i 188 more rows
```

```
dplyr::select(sets_raw,
              overlap_col_names[12])
> # A tibble: 269 x 1
>    Ineffcde
>    <fct>
>  1 <NA>
>  2 <NA>
>  3 <NA>
>  4 <NA>
>  5 <NA>
>  6 <NA>
>  7 <NA>
>  8 <NA>
>  9 <NA>
> 10 <NA>
> # i 259 more rows
```

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

```
sets_raw
> # A tibble: 269 x 45
>    `Row number`  Year   Stlkey `Vessel code` Station Setno Gear  `IPHC Reg Area`
>           <dbl> <dbl>    <dbl> <fct>           <dbl> <dbl> <fct> <fct>
>  1            1  2023 20230001 PEN              2331     1 Fixe~ 2B
>  2            2  2023 20230002 PEN              2152     2 Fixe~ 2B
```

```
>  3            3  2023 20230003 PEN              2324     3 Fixe~ 2B
>  4            4  2023 20230004 PEN              2318     4 Fixe~ 2B
>  5            5  2023 20230005 PEN              2322     5 Fixe~ 2B
>  6            6  2023 20230006 PEN              2343     6 Fixe~ 2B
>  7            7  2023 20230007 PEN              2316     7 Fixe~ 2B
>  8            8  2023 20230008 PEN              2320     8 Fixe~ 2B
>  9            9  2023 20230009 PEN              2314     9 Fixe~ 2B
> 10           10  2023 20230010 PEN              2312    10 Fixe~ 2B
> # i 259 more rows
> # i 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>, ...
```

```
summary(sets_raw)
>    Row number       Year          Stlkey         Vessel code    Station
>  Min.   :  1   Min.   :2023   Min.   :20230001   PEN: 87     Min.   :2001
>  1st Qu.: 68   1st Qu.:2023   1st Qu.:20230149   STW:117     1st Qu.:2070
>  Median :135   Median :2023   Median :20230332   VNI: 65     Median :2139
>  Mean   :135   Mean   :2023   Mean   :20230368               Mean   :2194
>  3rd Qu.:202   3rd Qu.:2023   3rd Qu.:20230588               3rd Qu.:2275
>  Max.   :269   Max.   :2023   Max.   :20230782               Max.   :3210
>
>      Setno                Gear     IPHC Reg Area IPHC Stat Area
>  Min.   :  1.00   Fixed Hook:269   2B:269        Min.   : 60.0
>  1st Qu.: 23.00                                  1st Qu.: 91.0
>  Median : 46.00                                  Median :102.0
>  Mean   : 49.12                                  Mean   :104.5
>  3rd Qu.: 73.00                                  3rd Qu.:121.0
>  Max.   :119.00                                  Max.   :142.0
>
>         IPHC Charter Region        Purpose Code         Date      Eff
>  Charlotte         :87       Standard Grid:269   25-Jun-23: 10   N: 16
>  Goose Is.         :58                           10-Jun-23:  9   Y:253
>  Ketchikan         : 8                           16-Jun-23:  9
>  St. James         :59                           20-Jun-23:  9
>  Vancouver Outside:57                            26-Jun-23:  9
>                                                  30-May-23:  9
>                                                  (Other)  :214
>  Ineffcde      BeginLat        BeginLon      BeginDepth (fm)      EndLat
>  DO  :  3   Min.   :48.33   Min.   :-133.7   Min.   :  8.00   Min.   :48.33
>  DS  :  8   1st Qu.:51.17   1st Qu.:-131.1   1st Qu.: 44.00   1st Qu.:51.14
```

```
>  GI   :  4   Median :52.19   Median :-129.8   Median : 75.00   Median :52.18
>  MS   :  1   Mean   :52.07   Mean   :-129.6   Mean   : 90.76   Mean   :52.07
>  NA's:253   3rd Qu.:53.35   3rd Qu.:-128.4   3rd Qu.:122.00   3rd Qu.:53.36
>             Max.   :55.31   Max.   :-124.8   Max.   :335.00   Max.   :55.34
>
>      EndLon       EndDepth (fm)    MidLat fished   MidLon fished
>  Min.   :-133.7   Min.   :  9.00   Min.   :48.33   Min.   :-133.7
>  1st Qu.:-131.1   1st Qu.: 43.00   1st Qu.:51.16   1st Qu.:-131.1
>  Median :-129.8   Median : 78.50   Median :52.17   Median :-129.8
>  Mean   :-129.6   Mean   : 93.04   Mean   :52.07   Mean   :-129.6
>  3rd Qu.:-128.4   3rd Qu.:122.50   3rd Qu.:53.34   3rd Qu.:-128.4
>  Max.   :-124.8   Max.   :393.00   Max.   :55.33   Max.   :-124.8
>                   NA's   :1
>  AvgDepth (fm)    Lat - Grid target Lon - Grid target O32 Pacific halibut count
>  Min.   :  8.00   Min.   :48.33     Min.   :-133.7    Min.   :  0.00
>  1st Qu.: 45.00   1st Qu.:51.17     1st Qu.:-131.1    1st Qu.:  7.00
>  Median : 76.00   Median :52.17     Median :-129.8    Median : 14.00
>  Mean   : 90.35   Mean   :52.07     Mean   :-129.6    Mean   : 20.32
>  3rd Qu.:121.00   3rd Qu.:53.33     3rd Qu.:-128.4    3rd Qu.: 29.00
>  Max.   :357.00   Max.   :55.33     Max.   :-124.9    Max.   :119.00
>
>  U32 Pacific halibut count U32 Pacific halibut weight
>  Min.   :  0               Min.   :   0.0
>  1st Qu.:  4               1st Qu.: 150.0
>  Median : 13               Median : 310.0
>  Mean   : 23               Mean   : 479.3
>  3rd Qu.: 33               3rd Qu.: 709.0
>  Max.   :148               Max.   :2305.0
>
>  U32 Pacific halibut weight No. skates set  No. skates hauled
>  Min.   :   0.0             Min.   :4.000   Min.   :0.0
>  1st Qu.:  28.0             1st Qu.:8.000   1st Qu.:8.0
>  Median :  93.0             Median :8.000   Median :8.0
>  Mean   : 169.1             Mean   :7.974   Mean   :7.9
>  3rd Qu.: 249.0             3rd Qu.:8.000   3rd Qu.:8.0
>  Max.   :1044.0             Max.   :9.000   Max.   :9.0
>
>  Avg no. hook/skate Effective skates hauled Soak time (min.) Profiler Lat
>  Min.   : 70.00     Min.   :0.400           Min.   : 216.0   Mode:logical
>  1st Qu.: 98.00     1st Qu.:7.870           1st Qu.: 489.0   NA's:269
>  Median : 99.00     Median :7.950           Median : 588.0
>  Mean   : 98.76     Mean   :7.829           Mean   : 596.1
>  3rd Qu.: 99.00     3rd Qu.:7.950           3rd Qu.: 691.0
>  Max.   :102.00     Max.   :8.190           Max.   :1088.0
```

```
>
>  Profiler Lon   Profiler Bottom Depth (m)  Temp C        Max Pressure (db)
>  Mode:logical   Mode:logical              Mode:logical   Mode:logical
>  NA's:269       NA's:269                  NA's:269       NA's:269
>
>
>
>
>
>     pH          Salinity PSU   Sigma-t        Oxygen_ml      Oxygen_umol
>  Mode:logical   Mode:logical   Mode:logical   Mode:logical   Mode:logical
>  NA's:269       NA's:269       NA's:269       NA's:269       NA's:269
>
>
>
>
>
>  Oxygen_sat
>  Mode:logical
>  NA's:269
>
>
>
>
>
```

```
testthat::expect_equal(unique(sets_raw$"IPHC Reg Area"),
                       as.factor("2B")) # Check just BC
testthat::expect_equal(unique(sets_raw$Year), 2023)
try(testthat::expect_equal(length(unique(sets_raw$Station)),
                       length(sets_raw$Station)))
> Error : length(unique(sets_raw$Station)) not equal to length(sets_raw$Station).
> 1/1 mismatches
> [1] 268 - 269 == -1
```

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.

For 2022 got same results as 2021.

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 (in 2022 not quite sure what that's referring to), but `twice_fished` gets used later, so do evaluate here:

```
length(unique(sets_raw$Station))
> [1] 268
```

```
length(sets_raw$Station)
> [1] 269
```

```
dplyr::count(sets_raw, Station) %>% dplyr::filter(n > 1)
> # A tibble: 1 x 2
>   Station     n
>     <dbl> <int>
> 1    2258     2
```

```
twice_fished <- dplyr::count(sets_raw, Station) %>%
  dplyr::filter(n > 1) %>%
  dplyr::select(Station) %>%
  as.numeric()
twice_fished
> [1] 2258
```

```
# If there's more than a single station then adapt later code
as.data.frame(dplyr::filter(sets_raw,
                            Station == twice_fished))
>   Row number Year   Stlkey Vessel code Station Setno       Gear IPHC Reg Area
> 1        154 2023 20230351         STW    2258    49 Fixed Hook            2B
> 2        155 2023 20230352         STW    2258    50 Fixed Hook            2B
>   IPHC Stat Area IPHC Charter Region  Purpose Code      Date Eff Ineffcde
> 1             90           Goose Is. Standard Grid 25-Jun-23   N       MS
> 2             90           Goose Is. Standard Grid 25-Jun-23   Y     <NA>
>   BeginLat  BeginLon BeginDepth (fm)  EndLat    EndLon EndDepth (fm)
> 1  50.9077 -129.5333             117 50.8697 -129.5332           180
> 2  50.9000 -129.5333             105 50.8623 -129.5328           124
>   MidLat fished MidLon fished AvgDepth (fm) Lat - Grid target Lon - Grid target
> 1       50.8887     -129.5333           112            50.834          -129.533
> 2       50.8812     -129.5330           109            50.834          -129.533
>   O32 Pacific halibut count U32 Pacific halibut count
> 1                         3                         2
> 2                        27                         7
>   O32 Pacific halibut weight U32 Pacific halibut weight No. skates set
> 1                         53                         19              8
> 2                        793                         64              8
>   No. skates hauled Avg no. hook/skate Effective skates hauled Soak time (min.)
> 1                 8                  99                   7.95              216
> 2                 8                  99                   7.95              672
>   Profiler Lat Profiler Lon Profiler Bottom Depth (m) Temp C Max Pressure (db)
> 1           NA           NA                       NA     NA               NA
> 2           NA           NA                       NA     NA               NA
```

```
>   pH Salinity PSU Sigma-t Oxygen_ml Oxygen_umol Oxygen_sat
> 1 NA           NA       NA         NA           NA          NA
> 2 NA           NA       NA         NA           NA          NA
```

Not needed for 2021 or 2022: So Station 2258 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 and 2022, just noting that two vessels were used, and these are different to those in 2020 (for which HAN then got excluded anyway); 2020 used BDP, HAN, VNI and 2022 used BDP and PEN:

```
summary(sets_raw$"Vessel code")
> PEN STW VNI
>  87 117  65
```

```
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: 269 x 12
>     year station   lat   lon avgDepth skatesHauled effSkateIPHC soakTimeMinutes
>    <int> <chr>   <dbl> <dbl>    <int>        <dbl>        <dbl>            <dbl>
>  1  2023 2001     48.3 -126.       76            8         8.03              542
>  2  2023 2002     48.3 -126.      170            8         7.95              435
>  3  2023 2003     48.5 -125.       74            8         7.95              523
>  4  2023 2004     48.5 -126.       57            8         8.03              663
>  5  2023 2005     48.5 -126.       57            8         7.95              562
>  6  2023 2006     48.5 -126.      109            8         7.87              431
>  7  2023 2007     48.7 -125.       33            8         7.85              492
>  8  2023 2008     48.7 -125.       33            8         7.95              621
>  9  2023 2009     48.7 -126.       58            8         7.95              720
> 10  2023 2010     48.7 -126.       40            8         7.95              433
> # i 259 more rows
> # i 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 and 2022 and 2023 we have all as Standard Grid, which gets corrected (some stations are non-standard) in the next section.

```
summary(sets_simp$purpose)
> Standard Grid
>           269
```

```
sets_simp_std <- dplyr::mutate(sets_simp,
                               standard_tmp = (purpose == "Standard Grid"))
                               # was grid in 2020, Grid in 2021, "Standard Grid"
                               # in 2022

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] 269
```

```
sets_simp_std <- cbind(sets_simp_std,
                       standard) %>%
  as_tibble() %>%
  dplyr::select(-c("standard_tmp"))
summary(sets_simp_std)
>       year        station               lat             lon
>  Min.   :2023   Length:269         Min.   :48.33   Min.   :-133.7
>  1st Qu.:2023   Class :character   1st Qu.:51.16   1st Qu.:-131.1
>  Median :2023   Mode  :character   Median :52.17   Median :-129.8
>  Mean   :2023                      Mean   :52.07   Mean   :-129.6
>  3rd Qu.:2023                      3rd Qu.:53.34   3rd Qu.:-128.4
>  Max.   :2023                      Max.   :55.33   Max.   :-124.8
>     avgDepth       skatesHauled  effSkateIPHC   soakTimeMinutes
>  Min.   :  8.00   Min.   :0.0   Min.   :0.400   Min.   : 216.0
>  1st Qu.: 45.00   1st Qu.:8.0   1st Qu.:7.870   1st Qu.: 489.0
>  Median : 76.00   Median :8.0   Median :7.950   Median : 588.0
>  Mean   : 90.35   Mean   :7.9   Mean   :7.829   Mean   : 596.1
>  3rd Qu.:121.00   3rd Qu.:8.0   3rd Qu.:7.950   3rd Qu.: 691.0
>  Max.   :357.00   Max.   :9.0   Max.   :8.190   Max.   :1088.0
>     usable                   purpose      U32halibut    O32halibut
>  Length:269         Standard Grid:269   Min.   :  0   Min.   :  0.00
>  Class :character                       1st Qu.:  4   1st Qu.:  7.00
>  Mode  :character                       Median : 13   Median : 14.00
>                                         Mean   : 23   Mean   : 20.32
>                                         3rd Qu.: 33   3rd Qu.: 29.00
>                                         Max.   :148   Max.   :119.00
>    standard
>  Length:269
>  Class :character
>  Mode  :character
>
>
>
```

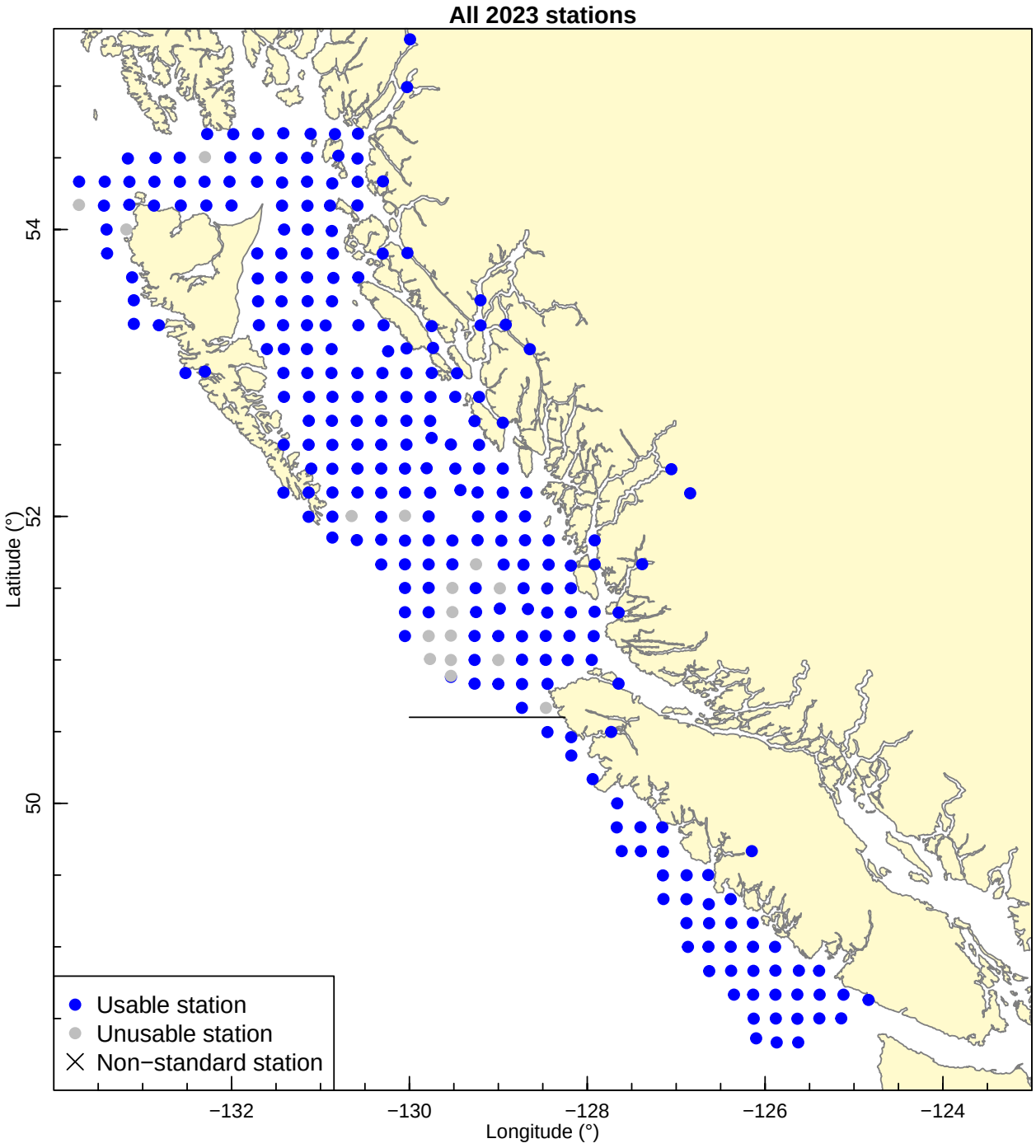
```
unique(sets_simp_std$standard)
> [1] "Y"
```

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 2023 we so far have this:

```
plot_iphc_map(sets_simp_std,
              sp = NULL,
              years = 2023,
              indicate_standard = TRUE)
```



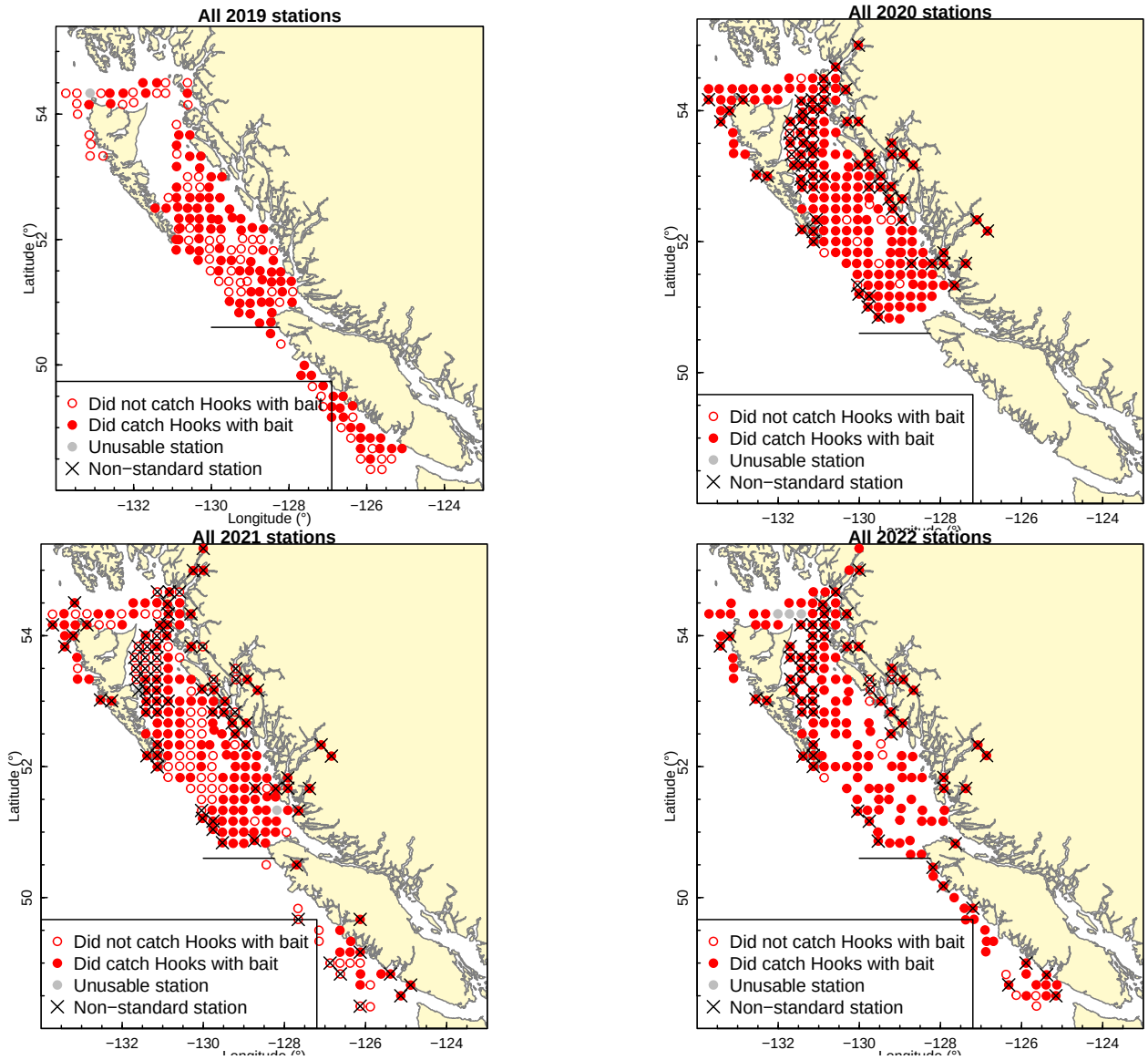
For 2021: So no stations are marked as being outside the standard grid, even though some are clearly new – the ones in the north have never been fished before (see the one-species vignette, though I'll investigate that here). 2022 the northern inlets ones are there again, as are some other inlets, only 1 in Strait of Georgia.

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:

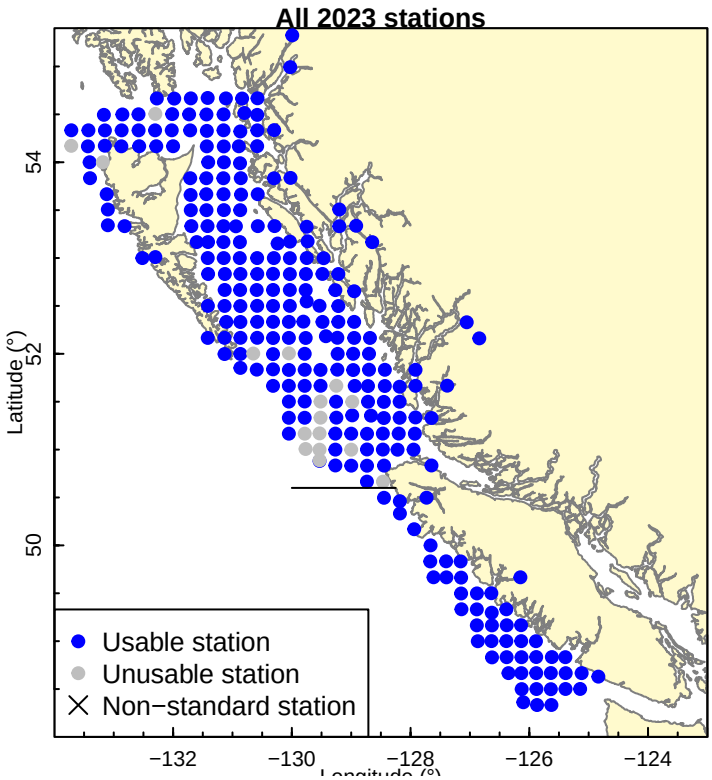
```
hooks_with_bait_revert <- hooks_with_bait

# 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'. (2023 is coloured different since no hooks with bait data yet, but the important bit is the crosses).

```
sets_2023 <- 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 = 2019:2022,    # the latest three years except
                                          # current data
                    indicate_standard = TRUE)
```




```
plot_iphc_map(sets_2023,
              sp = NULL,
              years = 2023,
              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 and 2022 have kind of done a few of those. The 2021 and 2022 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 or 2022 as non-standard. And the other main issue is that 2021 and 2022 is doing a random sample of WCVI stations (some of which will become non-standard). AND for 2021 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_2023 <- filter(sets_2023,
                       standard == "N")$station

# Not standard in both:
not_std_2018_and_2023 <- intersect(not_std_2018, not_std_2023)
not_std_2018_and_2023   # Empty for 2023
> character(0)
```

```
length(not_std_2018)
> [1] 131
```

```
length(not_std_2023)
> [1] 0
```

```
length(not_std_2018_and_2023)
> [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_2023 <- intersect(filter(sets_2018,
                                                    standard == "Y")$station,
                                             not_std_2023)
std_in_2018_but_not_std_in_2023
> character(0)
```

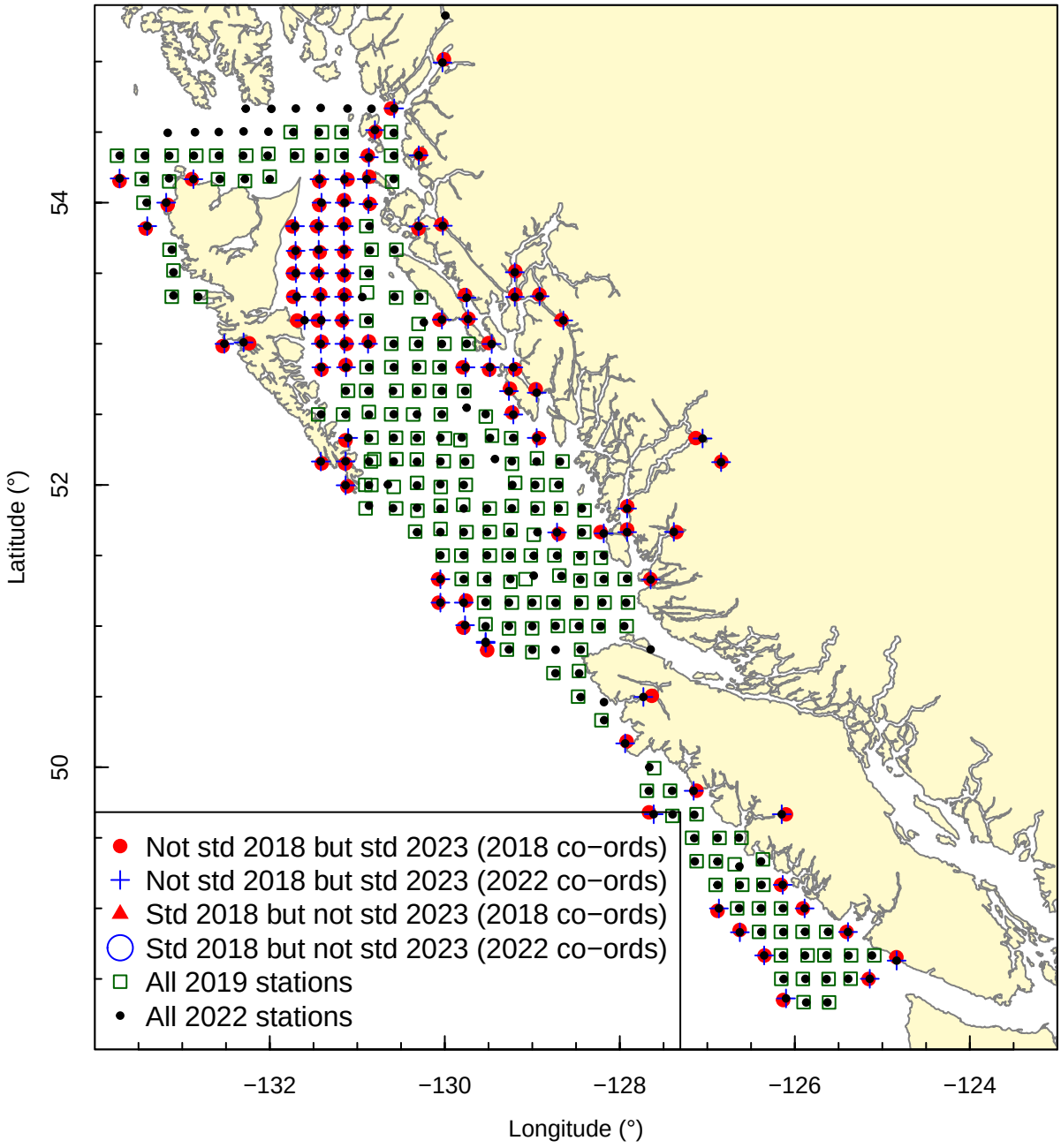
```
not_std_in_2018_but_std_in_2023 <- intersect(not_std_2018,
                                             filter(sets_2023,
                                                    standard == "Y")$station)
not_std_in_2018_but_std_in_2023
>  [1] "2258" "2261" "2263" "2262" "2265" "2266" "2264" "2269" "2272" "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" "2329"
> [41] "2328" "2327" "2324" "2322" "2318" "2305" "2287" "2285" "2288" "2311"
> [51] "2313" "2292" "2289" "2247" "2242" "2233" "2237" "2232" "2208" "2209"
> [61] "2213" "2205" "2214" "2210" "2218" "2221" "2217" "2276" "2278" "2273"
> [71] "2271" "2274" "2277" "2279" "2283" "2284" "2280" "2307" "2303" "2301"
> [81] "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 2023, 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 2023. Also (for 2021 and then 2023) 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_2023),
       col="red",
       pch = 19)

# Do the same but using 2023 station co-ordinates - should overlap:
points(lat~lon,
       data = filter(sets_2023,
                     station %in% not_std_in_2018_but_std_in_2023),
       col="blue",
       pch = 3)

# And for 2020 showed the single station std in 2018 but not 2020, for 2021 and 2022
# there are none:
points(lat~lon,
       data = filter(sets_2018,
                     station %in% std_in_2018_but_not_std_in_2023),
       col="red",
       pch = 17)
points(lat~lon,
       data = filter(sets_2023,
                     station %in% std_in_2018_but_not_std_in_2023),
       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 2023 stations as a small black dot
points(lat~lon,
       data = sets_2023,
       col="black",
       pch = 20,
       cex = 0.8)

legend("bottomleft",
```

```
       legend = c("Not std 2018 but std 2023 (2018 co-ords)",
                  "Not std 2018 but std 2023 (2022 co-ords)",
                  "Std 2018 but not std 2023 (2018 co-ords)",
                  "Std 2018 but not std 2023 (2022 co-ords)",
                  "All 2019 stations",
                  "All 2022 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 2022 (or 2021 before) 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. Empty green squares with no black dots (2022 stations) off WCVI clearly shows the reduced coverage there (similar for 2021 and 2022).

2020 only (there were no non-standard stations defined in raw data for 2021 or 2022): 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 2022, and this is corrected
dplyr::filter(hooks_with_bait_revert$set_counts,
              station == std_in_2018_but_not_std_in_2023) %>%
  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. For 2022 this is same as we did for 2021.

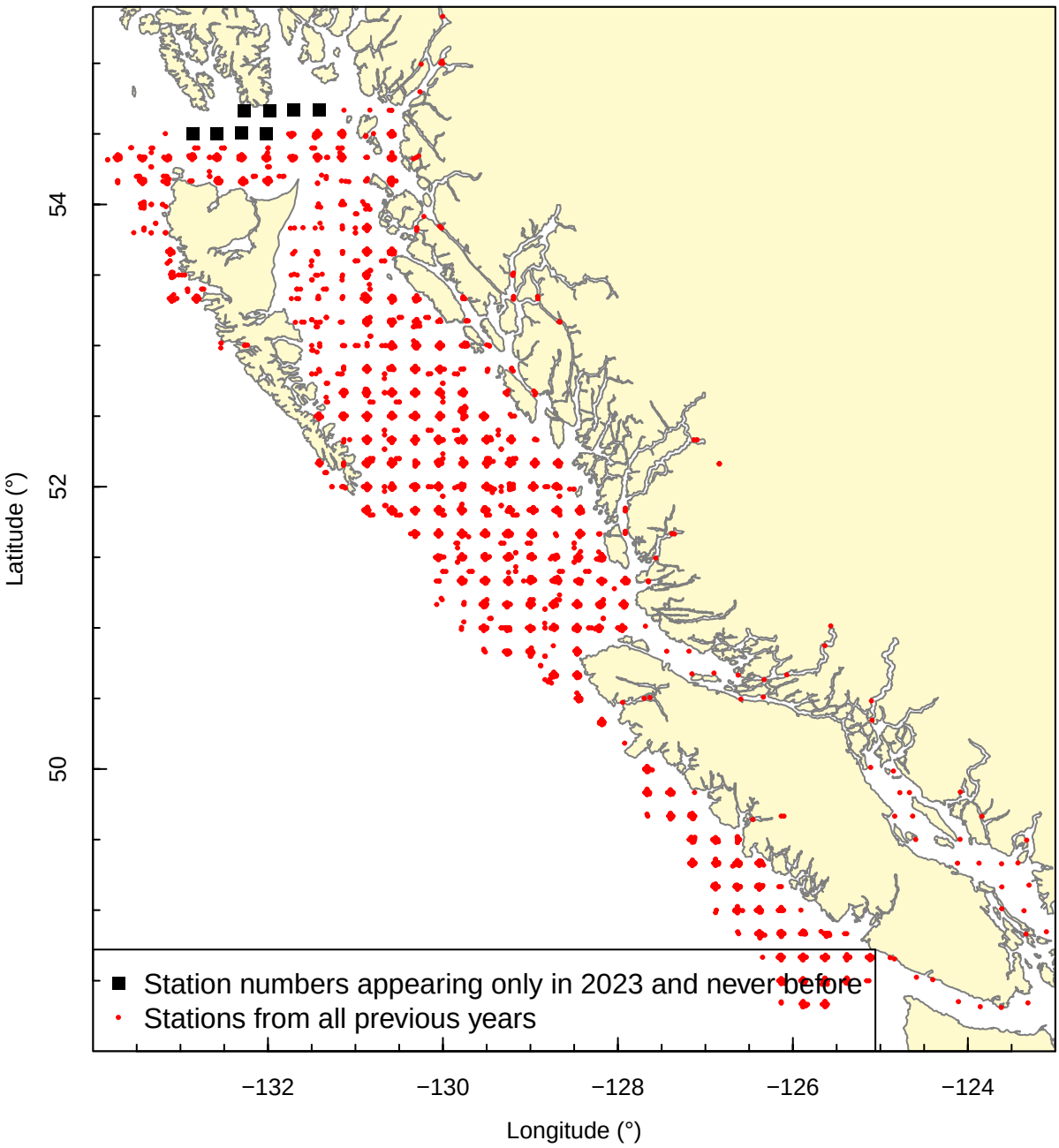
Doing that shortly (in `sets_simp_std_corrected`), but first also look for any new 2022 stations. I hadn't expected any in 2021 but saw them when doing the one-species vignette, so had to come back to redo this.

```
# For 2021: yelloweye_rockfish$set_counts is saved in gfiphc, already has 2021 data
#  because I had to come back to redo this .pdf after updating the data, hence
#  need the <2021 here; station codes do change over time, but I think are
#  recently consistent
# For 2022 checking before saving any data into gfiphc. Then rerunning (step 8
# in README) so need the < 2022 here as package contains 2022 data
previous_stations <- dplyr::filter(yelloweye_rockfish$set_counts,
                                   year < 2023)$station %>%
                     unique()

stations_in_2023_only <- dplyr::filter(sets_2023,
                                       !(station %in% previous_stations))
stations_in_2023_only
> # A tibble: 8 x 11
>    year station   lat   lon avgDepth skatesHauled effSkateIPHC soakTimeMinutes
>   <int> <chr>   <dbl> <dbl>    <int>        <dbl>        <dbl>            <dbl>
> 1  2023 3001     54.5 -132.      146            8         7.95              471
> 2  2023 3002     54.5 -132.      197            0         0.4               628
> 3  2023 3003     54.5 -133.      203            4         3.93              439
> 4  2023 3004     54.5 -133.      208            4         3.98              373
> 5  2023 3010     54.7 -131.      107            8         8.03              688
> 6  2023 3011     54.7 -132.      221            8         7.95              696
```

```
> 7  2023 3012     54.7 -132.      102            8         7.95             889
> 8  2023 3013     54.7 -132.       59            8         7.95             804
> # i 3 more variables: usable <chr>, purpose <fct>, standard <chr>
```

and plot those stations:

```
plot_BC()
points(lat~lon,
       data = stations_in_2023_only,
       col = "black",
       pch = 15)

points(lat~lon,
       data = dplyr::filter(yelloweye_rockfish$set_counts,
                            year < 2022),
       col = "red",
       pch = 20,
       cex = 0.4)

legend("bottomleft",
       legend = c("Station numbers appearing only in 2023 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 two 2022 (six in 2021) stations that have never been fished before!

For 2021: That map suggests that we should call the five northern ones non-standard also, to exclude from the standard Series A-F analyses.

2021 (for reference): 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

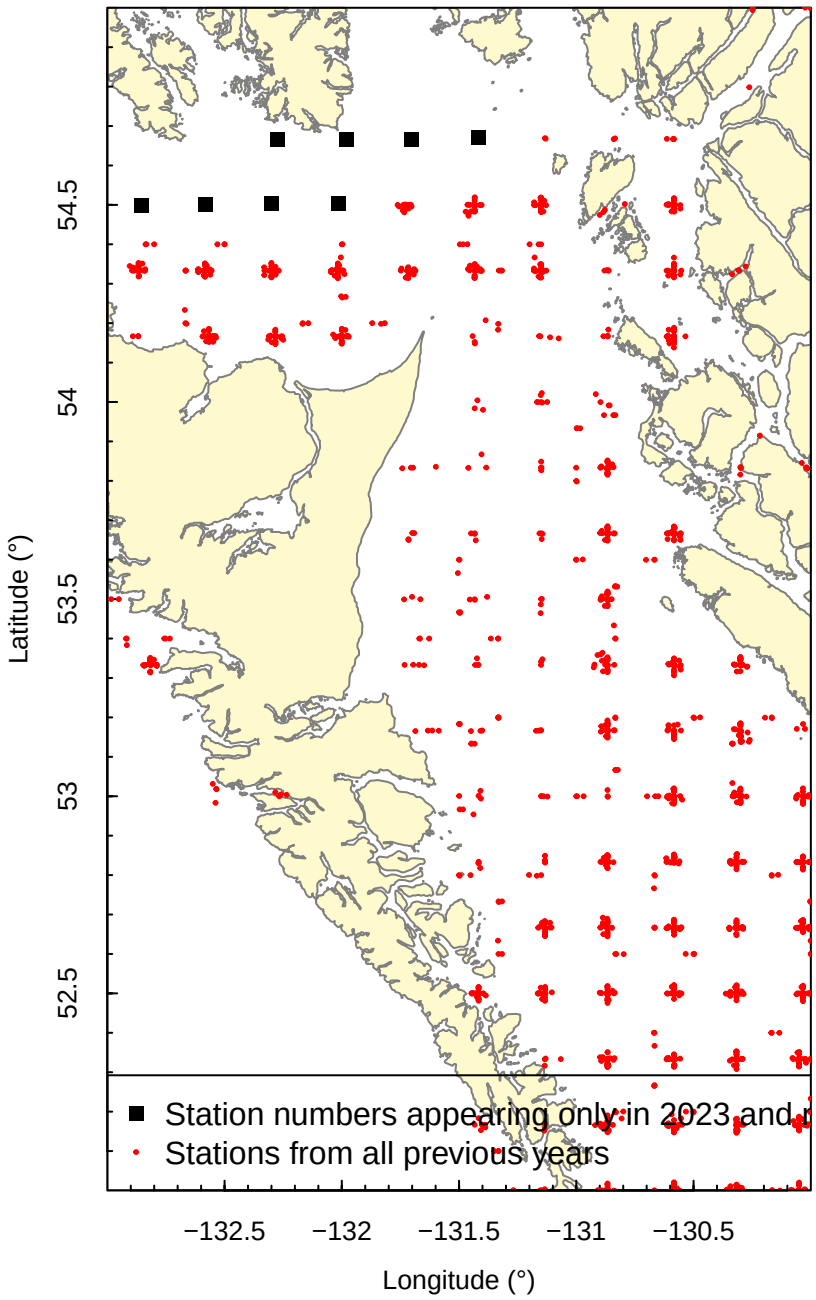
data in the past (and which I used here for recent years to extract from their website). So there may be data available, and if needed it will have to be obtained. Here we will call those five northern newly-fished stations `non-standard`.

2021: For the sixth station off the northwest of Vancouver Island, zooming in and including the Scott Islands Rockfish Conservation Area (clickable) as a blue rectangle shows (see saved 2021 .pdf, and rectangle in next map).

2022: The two new ones are also off the northwest of Vancouver Island, so zoom in:

2023: 8 new ones at northern tip

```
plot_BC(xlim = c(-133, -130),
        ylim = c(52, 55))

# 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_2023_only,
       col = "black",
       pch = 15)

points(lat~lon,
       data = yelloweye_rockfish$set_counts,
       col = "red",
       pch = 20,
       cex = 0.4)

legend("bottomleft",
       legend = c("Station numbers appearing only in 2023 and never before",
                  "Stations from all previous years"),
       pch = c(15, 20),
       col = c("black", "red"),
       pt.cex = c(1, 0.4))
```



2021: 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.

2021: 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):

2022: The two new stations are also not outliers in terms of depth, though are very close to shore:

```
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
```

2021: 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.

2022: Given so close to shore and not close to previous stations, will call these two new 2022 stations non-standard also.

2023: ?? further north, but aligned with others... don't include? revisit in future years perhaps

So - retain the 2018 definitions of standard stations (as we did for 2020 and 2021), and call both new 2023 stations non-standard (like we did for the six new 2021 stations and 2 in 2022):

```
sets_simp_std_corrected <- sets_simp_std
summary(as.factor(sets_simp_std_corrected$standard))
>   Y
> 269
```

```
sets_simp_std_corrected$standard[sets_simp_std_corrected$station %in%
                                 not_std_in_2018_but_std_in_2023] <- "N"
sets_simp_std_corrected$standard[sets_simp_std_corrected$station %in%
                                 stations_in_2023_only$station] <- "N"
summary(as.factor(sets_simp_std_corrected$standard))
>   N   Y
>  94 175
```

```
# 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:

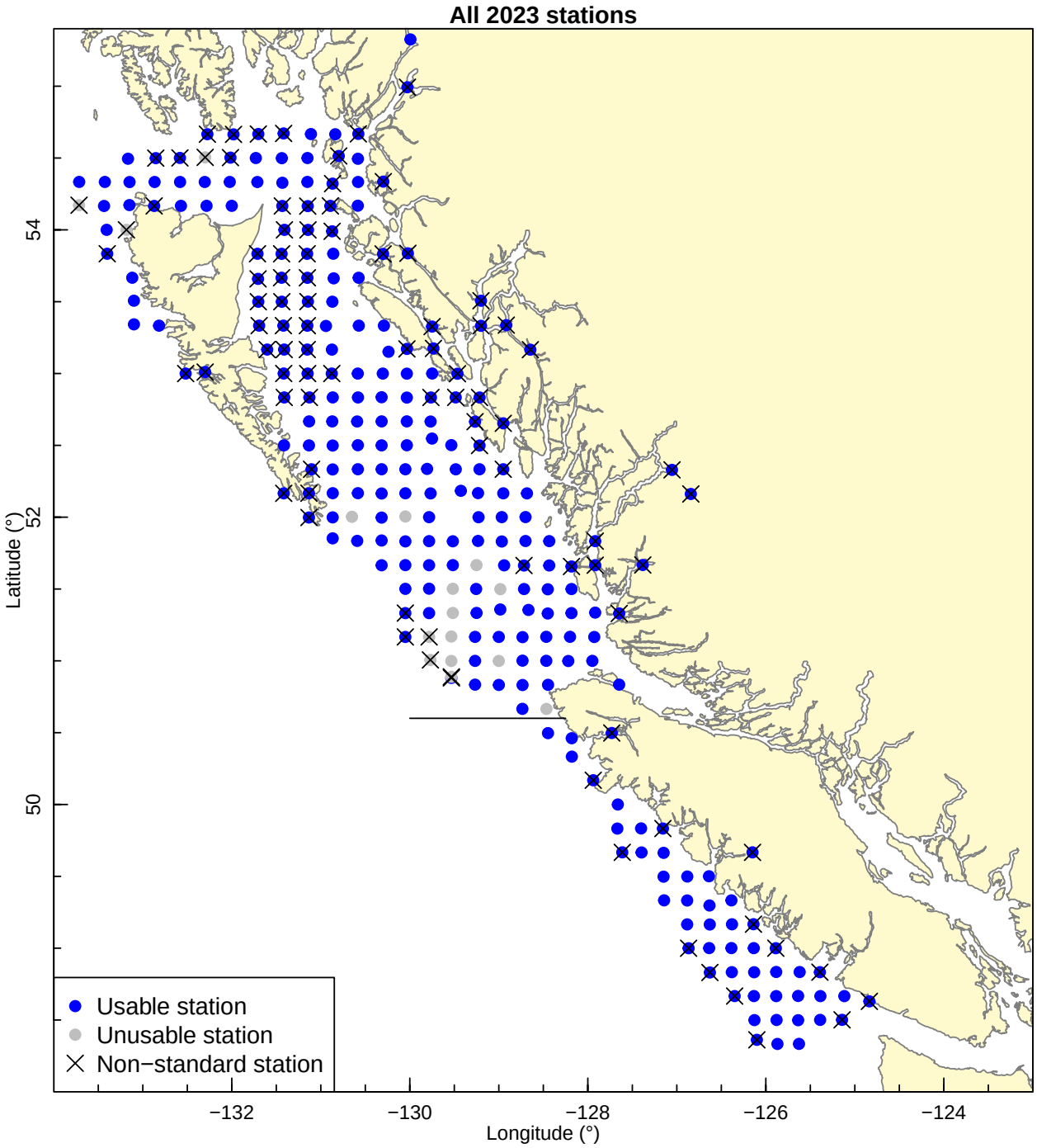
```
dplyr::filter(hooks_with_bait$set_counts,
              station %in% not_std_2018) %>%
```

```
  dplyr::select(year) %>%
  unique()
> # A tibble: 4 x 1
>    year
>   <dbl>
> 1  2018
> 2  2020
> 3  2021
> 4  2022
```

Note 2023 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 had in 2021).

So here are the final station designations for 2023:

```
plot_iphc_map(sets_simp_std_corrected,
              sp = NULL,
              years = 2023,
              indicate_standard = TRUE)
```



Can see that they did 6 random stations off WCVI that we're calling non-standard (because they were never fished before 2018; was 10 for 2021). Which is a bit of a shame as there are only 19 stations left off WCVI for 2022 (16 for 2022).

2020 (no need to change for 2021 or 2022): 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; no change here for 2022):

```
counts_raw_2020 <- readr::read_csv(here::here("data-raw/non-halibut-data-2020.csv")) %>%
  dplyr::mutate_if(is.character, factor)
> Rows: 1441 Columns: 13
> -- Column specification --------------------------------------------------------
> Delimiter: ","
> chr (3): Scientific Name, Species Name, SampleType
> dbl (9): Year, Stlkey, Station, Setno, IPHC Species Code, HooksFished, Hooks...
> num (1): Row number
>
> i Use `spec()` to retrieve the full column specification for this data.
> i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
counts_raw <- readr::read_csv(here::here("data-raw/non-halibut-data-2023.csv")) %>%
  dplyr::mutate_if(is.character, factor)
> Rows: 2098 Columns: 13
> -- Column specification --------------------------------------------------------
> Delimiter: ","
> chr (3): Scientific Name, Species Name, SampleType
> dbl (9): Year, Stlkey, Station, Setno, IPHC Species Code, HooksFished, Hooks...
> num (1): Row number
>
> i Use `spec()` to retrieve the full column specification for this data.
> i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
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 for 2023.

```
counts_raw
> # A tibble: 2,098 x 13
>    `Row number`  Year Stlkey Station Setno `IPHC Species Code` `Scientific Name`
>           <dbl> <dbl>  <dbl>   <dbl> <dbl>                <dbl> <fct>
>  1            1  2023 2.02e7    2331     1                   54 Squalus suckleyi
```

```
>  2           2  2023 2.02e7    2331     1                100 Junkis nonspecif~
>  3           3  2023 2.02e7    2331     1                141 Raja binoculata
>  4           4  2023 2.02e7    2331     1                143 Raja rhina
>  5           5  2023 2.02e7    2331     1                148 Myoxocephalus po~
>  6           6  2023 2.02e7    2331     1                295 <NA>
>  7           7  2023 2.02e7    2331     1                303 <NA>
>  8           8  2023 2.02e7    2331     1                304 <NA>
>  9           9  2023 2.02e7    2331     1                305 <NA>
> 10          10  2023 2.02e7    2152     2                 26 Gadus macrocepha~
> # i 2,088 more rows
> # i 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.   :2023   Min.   :20230001   Min.   :2001
>  1st Qu.: 525.2   1st Qu.:2023   1st Qu.:20230151   1st Qu.:2072
>  Median :1049.5   Median :2023   Median :20230336   Median :2137
>  Mean   :1049.5   Mean   :2023   Mean   :20230372   Mean   :2195
>  3rd Qu.:1573.8   3rd Qu.:2023   3rd Qu.:20230589   3rd Qu.:2275
>  Max.   :2098.0   Max.   :2023   Max.   :20230782   Max.   :3210
>
>      Setno       IPHC Species Code            Scientific Name
>  Min.   :  1.0   Min.   :  2.0     Squalus suckleyi   :196
>  1st Qu.: 23.0   1st Qu.: 54.0     Raja rhina         :147
>  Median : 46.0   Median :143.0     Anoplopoma fimbria :123
>  Mean   : 49.7   Mean   :178.9     Sebastes ruberrimus: 79
>  3rd Qu.: 75.0   3rd Qu.:304.0     Atheresthes stomias: 62
>  Max.   :119.0   Max.   :601.0     (Other)            :550
>                                    NA's               :941
>               Species Name  SampleType    HooksFished    HooksRetrieved
>  Empty Hook         :268   20Hook:2098   Min.   :392.0   Min.   : 39.0
>  Hook with Skin     :255                 1st Qu.:784.0   1st Qu.:784.0
>  Hook with Bait     :230                 Median :792.0   Median :792.0
>  Spiny Dogfish      :196                 Mean   :787.8   Mean   :781.8
>  Bent/Broken/Missing:162                 3rd Qu.:792.0   3rd Qu.:792.0
>  Longnose Skate     :147                 Max.   :816.0   Max.   :816.0
>  (Other)            :840
>  HooksObserved   Number Observed
>  Min.   : 20.0   Min.   :  1.00
>  1st Qu.:160.0   1st Qu.:  1.00
>  Median :160.0   Median :  4.00
>  Mean   :158.1   Mean   : 18.81
```

```
>  3rd Qu.:160.0   3rd Qu.: 19.00
>  Max.   :161.0   Max.   :143.00
>
```

```
testthat::expect_equal(unique(counts_raw$Year), 2023)  # All 2023
testthat::expect_equal(unique(counts_raw$SampleType), as.factor("20Hook")) # All 20Hook

# This mismatches for 2020, not for 2021 or 2022:
try(testthat::expect_equal(length(unique(counts_raw$Station)),
                           length(sets_raw$Station)))
> Error : length(unique(counts_raw$Station)) not equal to length(sets_raw$Station).
> 1/1 mismatches
> [1] 268 - 269 == -1
```

```
unique(counts_raw$"Species Name")
>  [1] Spiny Dogfish                Inanimate Object
>  [3] Big Skate                    Longnose Skate
>  [5] Great Sculpin                unident. algae
>  [7] Hook with Skin               Empty Hook
>  [9] Hook with Bait               Pacific Cod
> [11] Sea Whip                     Scallop
> [13] Quillback Rockfish           Sea Anemone
> [15] unident. Starfish            unident. organic matter
> [17] Dungeness Crab               Bivalve
> [19] Sea Cucumber                 Shells
> [21] Soupfin Shark                Sunflower Sea Star
> [23] Giant Pacific Octopus        Sea Pen
> [25] Silvergray Rockfish          Bent/Broken/Missing
> [27] Copper Rockfish              Gorgonian coral
> [29] Sand Dab                     Lingcod
> [31] Sea Urchin                   Yellow Irish Lord
> [33] Petrale Sole                 Yelloweye Rockfish
> [35] Sablefish (Blackcod)         Redbanded Rockfish
> [37] Bocaccio                     Arrowtooth Flounder
> [39] Greenstriped Rockfish        unident. Sculpin
> [41] Fish-eating Star             Sixgill Shark
> [43] Flathead Sole                Canary Rockfish
> [45] Rougheye Rockfish            Shortraker Rockfish
> [47] Sleeper Shark                Yellowmouth Rockfish
> [49] Shortspine Thornyhead        Unident. Flatfish
> [51] Solaster sp (starfish)       Sun Sea Star
> [53] Cabezon                      Unident. Salmon
> [55] Stylaster campylecus (coral) Tiger Rockfish
> [57] Spotted Ratfish              Basketstar
```

```
> [59] unident. Invertebrate        unident. Coral
> [61] unident. Hagfish             Brittle Star
> [63] China Rockfish               Blue Shark
> [65] Octopus                      unident. Skate
> [67] Giant Wrymouth               Aleutian Skate
> [69] Red Irish Lord               Worms
> [71] Black-footed Albatross       unident. Sponge
> 72 Levels: Aleutian Skate Arrowtooth Flounder ... Yellowmouth Rockfish
```

Here's what was seen in 2020 but not 2023, and vice versa:

```
# Seen in 2020 not 2022
setdiff(unique(counts_raw_2020$"Species Name"),
        unique(counts_raw$"Species Name"))
> [1] "Blackspotted Rockfish"       "Oregon Rock Crab"
> [3] "Glass Sponge"                "unident. thornyhead (Idiot)"
> [5] "Wolf-Eel"                    "Gastropod"
> [7] "Red Tree Coral"
```

```
# Seen in 2022 not 2020
setdiff(unique(counts_raw$"Species Name"),
        unique(counts_raw_2020$"Species Name"))
>  [1] "Inanimate Object"             "Great Sculpin"
>  [3] "unident. algae"               "Sea Whip"
>  [5] "Scallop"                      "unident. organic matter"
>  [7] "Dungeness Crab"               "Bivalve"
>  [9] "Sea Cucumber"                 "Shells"
> [11] "Gorgonian coral"              "Yellow Irish Lord"
> [13] "Greenstriped Rockfish"        "Sixgill Shark"
> [15] "Flathead Sole"                "Unident. Flatfish"
> [17] "Solaster sp (starfish)"       "Sun Sea Star"
> [19] "Cabezon"                      "Unident. Salmon"
> [21] "Stylaster campylecus (coral)" "unident. Invertebrate"
> [23] "unident. Hagfish"             "China Rockfish"
> [25] "unident. Skate"               "Giant Wrymouth"
> [27] "Red Irish Lord"               "Worms"
> [29] "Black-footed Albatross"
```

2021: Presumably Sun Sea Star and Sunflower Sea Star are the same. Will mention this later on.

2022: Updated the csv file to include Rosethorn and Red Irish Lord, the rest are all dealt with.

2023: done nothing

Note that halibut are not included in these counts:

```
dplyr::filter(counts_raw, "Species Name" == "Pacific Halibut")
> # A tibble: 0 x 13
> # i 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 and 2022
```

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()
>    Station Setno          Species Name Number Observed
> 1     2258    50              Lingcod               1
> 2     2258    50 Sablefish (Blackcod)               6
> 3     2258    50        Spiny Dogfish              21
> 4     2258    50  Silvergray Rockfish               1
> 5     2258    50   Yelloweye Rockfish              19
> 6     2258    50   Redbanded Rockfish               2
> 7     2258    50             Bocaccio               5
> 8     2258    50       Longnose Skate              11
> 9     2258    50       Hook with Skin               6
> 10    2258    50           Empty Hook              41
> 11    2258    50       Hook with Bait              34
> 12    2258    50  Bent/Broken/Missing               4
```

```
dplyr::filter(sets_raw, Station == twice_fished)
> # A tibble: 2 x 45
>   `Row number`  Year   Stlkey `Vessel code` Station Setno Gear   `IPHC Reg Area`
>          <dbl> <dbl>    <dbl> <fct>           <dbl> <dbl> <fct>  <fct>
> 1          154  2023 20230351 STW              2258    49 Fixed~ 2B
> 2          155  2023 20230352 STW              2258    50 Fixed~ 2B
> # i 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>, ...
```

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 and 2022:

```
dplyr::filter(counts_raw,
              Station == twice_fished & Setno == 4)
> # A tibble: 0 x 13
> # i 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
countData2023_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(countData2023_no_halibut))
countData2023_no_halibut
> # A tibble: 2,098 x 4
>     year station spNameIPHC           specCount
>    <int> <chr>   <chr>                    <int>
>  1  2023 2001    Spiny Dogfish               82
>  2  2023 2001    Empty Hook                  78
>  3  2023 2002    Arrowtooth Flounder          4
>  4  2023 2002    Sablefish (Blackcod)        32
>  5  2023 2002    Rougheye Rockfish            1
>  6  2023 2002    Spiny Dogfish                2
>  7  2023 2002    Yelloweye Rockfish           1
>  8  2023 2002    Soupfin Shark                1
```

```
>  9  2023 2002    Longnose Skate                1
> 10  2023 2002    Hook with Skin                1
> # i 2,088 more rows
```

```
summary(countData2023_no_halibut)
>       year        station            spNameIPHC          specCount
>  Min.   :2023   Length:2098        Length:2098        Min.   :  1.00
>  1st Qu.:2023   Class :character   Class :character   1st Qu.:  1.00
>  Median :2023   Mode  :character   Mode  :character   Median :  4.00
>  Mean   :2023                                         Mean   : 18.81
>  3rd Qu.:2023                                         3rd Qu.: 19.00
>  Max.   :2023                                         Max.   :143.00
```

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: 268 x 4
>    station  year hooksRetr hooksObs
>    <chr>   <int>     <dbl>    <dbl>
>  1 2001     2023       800      160
>  2 2002     2023       792      160
>  3 2003     2023       792      160
>  4 2004     2023       800      160
>  5 2005     2023       792      160
>  6 2006     2023       784      160
>  7 2007     2023       782      160
>  8 2008     2023       792      160
>  9 2009     2023       792      160
> 10 2010     2023       792      160
```

```
> # i 258 more rows
```

```
try(testthat::expect_equal(sets_simp_std_corrected$station, hook_details$station))
> Error : sets_simp_std_corrected$station not equal to hook_details$station.
> Lengths differ: 269 is not 268
```

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`:

```
setData2023 <- 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))
setData2023
> # A tibble: 269 x 11
>     year station   lat   lon avgDepth effSkateIPHC E_it20 usable standard
>    <int> <chr>   <dbl> <dbl>    <int>        <dbl>  <dbl> <chr>  <fct>
>  1  2023 2001     48.3 -126.       76         8.03   1.61 Y      Y
>  2  2023 2002     48.3 -126.      170         7.95   1.61 Y      Y
>  3  2023 2003     48.5 -125.       74         7.95   1.61 Y      Y
>  4  2023 2004     48.5 -126.       57         8.03   1.61 Y      Y
>  5  2023 2005     48.5 -126.       57         7.95   1.61 Y      Y
>  6  2023 2006     48.5 -126.      109         7.87   1.61 Y      Y
>  7  2023 2007     48.7 -125.       33         7.85   1.61 Y      Y
>  8  2023 2008     48.7 -125.       33         7.95   1.61 Y      Y
>  9  2023 2009     48.7 -126.       58         7.95   1.61 Y      Y
> 10  2023 2010     48.7 -126.       40         7.95   1.61 Y      Y
> # i 259 more rows
> # i 2 more variables: hooksRetr <dbl>, hooksObs <dbl>
```

```
testthat::expect_equal(names(setData2013), names(setData2023)[1:ncol(setData2013)])
summary(setData2023)
>       year        station               lat             lon
>  Min.   :2023   Length:269         Min.   :48.33   Min.   :-133.7
>  1st Qu.:2023   Class :character   1st Qu.:51.16   1st Qu.:-131.1
>  Median :2023   Mode  :character   Median :52.17   Median :-129.8
>  Mean   :2023                      Mean   :52.07   Mean   :-129.6
>  3rd Qu.:2023                      3rd Qu.:53.34   3rd Qu.:-128.4
>  Max.   :2023                      Max.   :55.33   Max.   :-124.8
>     avgDepth       effSkateIPHC       E_it20          usable          standard
>  Min.   :  8.00   Min.   :0.400   Min.   :0.2051   Length:269         N: 94
>  1st Qu.: 45.00   1st Qu.:7.870   1st Qu.:1.6060   Class :character   Y:175
>  Median : 76.00   Median :7.950   Median :1.6061   Mode  :character
>  Mean   : 90.35   Mean   :7.829   Mean   :1.5832
>  3rd Qu.:121.00   3rd Qu.:7.950   3rd Qu.:1.6061
>  Max.   :357.00   Max.   :8.190   Max.   :1.6161
>    hooksRetr        hooksObs
>  Min.   : 39.0   Min.   : 20.0
>  1st Qu.:784.0   1st Qu.:160.0
>  Median :792.0   Median :160.0
>  Mean   :779.9   Mean   :157.7
>  3rd Qu.:792.0   3rd Qu.:160.0
>  Max.   :816.0   Max.   :161.0
```

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.

```
setData2023_and_halibut <-
  dplyr::left_join(setData2023,
                   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)
> Warning in dplyr::left_join(setData2023, dplyr::select(sets_simp_std_corrected, : Dete
> i Row 186 of `x` matches multiple rows in `y`.
> i Row 186 of `y` matches multiple rows in `x`.
> i If a many-to-many relationship is expected, set `relationship =
>   "many-to-many"` to silence this warning.
```

```
setData2023_and_halibut %>% dplyr::select(station,
                                          N_it_halibut,
                                          N_it20_halibut_opt_1)
> # A tibble: 271 x 3
>    station N_it_halibut N_it20_halibut_opt_1
>    <chr>          <dbl>                <dbl>
>  1 2001               0                0
>  2 2002              98               19.8
>  3 2003              25                5.05
>  4 2004              25                5
>  5 2005               4                0.808
>  6 2006               8                1.63
>  7 2007              39                7.98
>  8 2008              96               19.4
>  9 2009              13                2.63
> 10 2010              83               16.8
> # i 261 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 just be halibut counts and gives an integer number, and is based on the first 20 hooks.

Add counts for each set:

```
counts_20 <- countData2023_no_halibut %>%
  dplyr::group_by(station) %>%
  dplyr::summarise(non_halibut = sum(specCount)) %>%
```

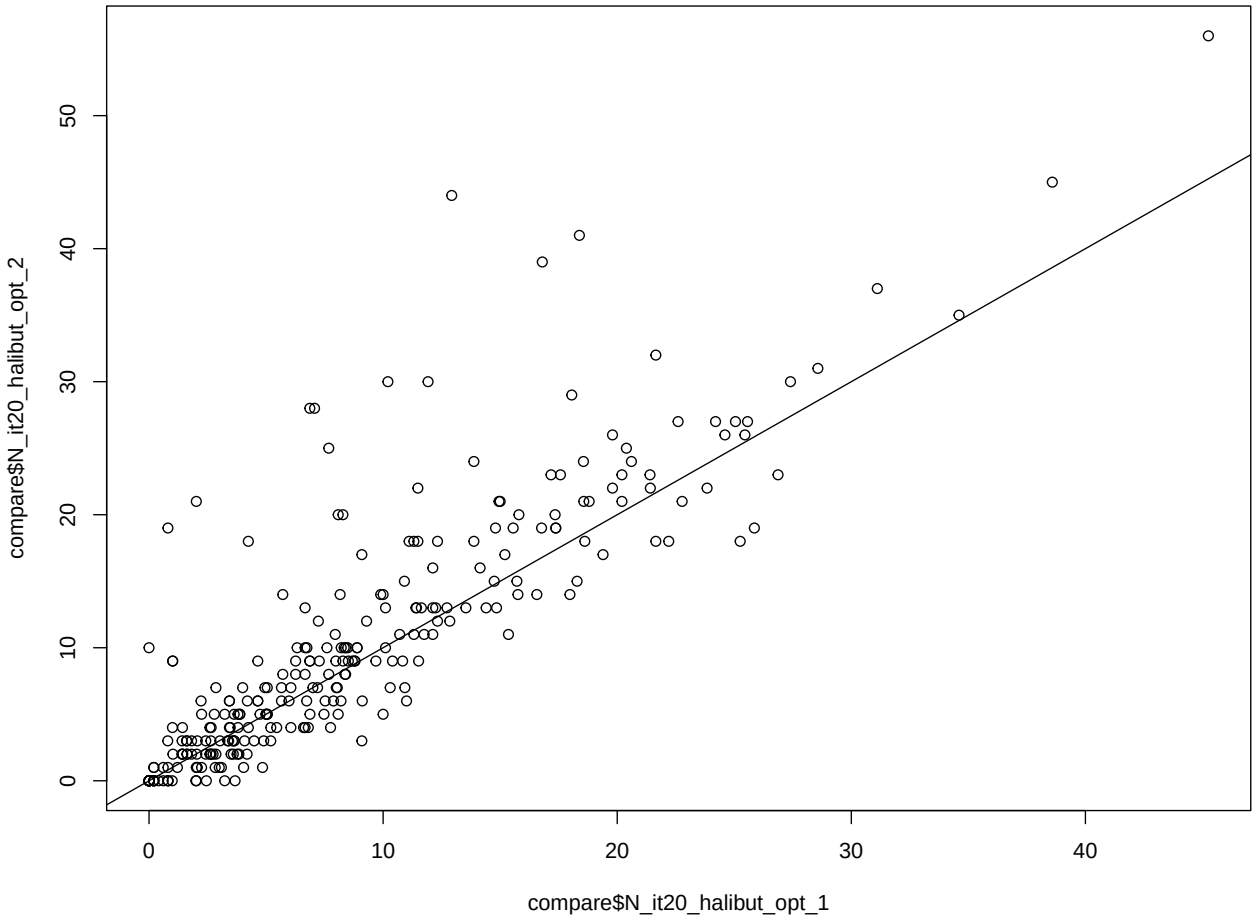
```
  dplyr::ungroup()
counts_20
> # A tibble: 268 x 2
>    station non_halibut
>    <chr>         <int>
>  1 2001            160
>  2 2002            134
>  3 2003            155
>  4 2004            155
>  5 2005            159
>  6 2006            158
>  7 2007            151
>  8 2008            143
>  9 2009            158
> 10 2010            141
> # i 258 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(setData2023_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: 271 x 7
>     year station usable N_it20_halibut_opt_1 N_it20_halibut_opt_2
>    <int> <chr>   <chr>                 <dbl>                 <dbl>
>  1  2023 2001    Y                     0                         0
>  2  2023 2002    Y                    19.8                      26
>  3  2023 2003    Y                     5.05                      5
>  4  2023 2004    Y                     5                         5
>  5  2023 2005    Y                     0.808                     1
>  6  2023 2006    Y                     1.63                      2
>  7  2023 2007    Y                     7.98                      9
>  8  2023 2008    Y                    19.4                      17
```

```
>  9  2023 2009    Y                    2.63                    2
> 10  2023 2010    Y                   16.8                    19
> # i 261 more rows
> # i 2 more variables: N_it20_opt_1_over_opt_2 <dbl>, 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] 0.8577569
```

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] 8.728464
```

```
mean(compare$N_it20_halibut_opt_2)
> [1] 10.41328
```

(9.6 and 10.5 in 2021; 10.8 and 11.5 in 2022; 8.7 and 10.4 in 2023).

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

```
compare$N_it20_halibut_opt_2
>   [1]  0 26  5  5  1  2  9 17  2 19  3  3  9  3  0  7  0  2  2  4  4  7  2 23  6
>  [26]  3 15 18  6 11 13  9 26 19  8 13  3 24  5  8  4  9  1  7 22 12  2  1  8  2
>  [51]  6 21  1  5  4  3 32  1  4  5  5  7  9  3 13 27 21 29 13 10  3  9 23 37  2
>  [76]  0 13 25 11  3 22 21  3  5  2  1  7 26 23 14  2 18 10  4  5 14 19  0  0  5
> [101] 24  0  7 10 22  6  7  4  0 10 18 24 35  6  3 23 18  9  2  6 14  0  2  2  9
> [126] 27  1 30 41  5 16  2 14  4  4 17 31 20 18 10  9  6 10 12 18 10 27 15  5  2
> [151] 20 45  3  8 19 11  8  5 14  3  6 21 14 19 12  3 27  2  3  3  0  0  2  0 13
> [176]  0  1  0  0 10 18  6 13  0 19  9  9  9  9  0  0 10  4 10  5  7  5  8  3 11
> [201]  1  3  1  1 11  7  7  3  0 15  7 28 56  9  7 13 17  6  5  5  4 25  4 20  4
> [226] 21  1  9  0 16  9  4  0 10  4 30  6 12  6  0 13 19 14  2 21 15 30 20 18 44
> [251] 13 18 21  0 11  6 28  6 22  1  0  0  0 13  2 39  9  1 10 23 18
```

```
countData2023_halibut <- dplyr::select(compare,
                                       year,
                                       station,
                                       spNameIPHC,
                                       specCount = N_it20_halibut_opt_2) %>%
  dplyr::mutate(specCount = as.integer(specCount))
countData2023 <- rbind(countData2023_no_halibut,
                       countData2023_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 and 2022 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(countData2023,
                      spNameIPHC == "Pacific Halibut"))
>       year        station           spNameIPHC          specCount
>  Min.   :2023   Length:271         Length:271         Min.   : 0.00
>  1st Qu.:2023   Class :character   Class :character   1st Qu.: 3.00
>  Median :2023   Mode  :character   Mode  :character   Median : 8.00
>  Mean   :2023                                         Mean   :10.41
>  3rd Qu.:2023                                         3rd Qu.:16.00
>  Max.   :2023                                         Max.   :56.00
```

```
unique(dplyr::filter(countData2023, 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); so set `eval=TRUE` then back to `eval=FALSE`; 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()
```

These are the ones just for the new 2023 data:

```
check_iphc_spp_name(countData2023)
```

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

```
check_iphc_spp_name(countData2020)
```

For 2020 I said that only the Thornyhead and Blackspotted Rockfish are likely of interest (Issues #17 and #18). And the sharks from the earlier list. So look at just the new ones in 2023 that aren't in 2020 or any previous year (switch this to `eval=TRUE`, paste results in, then set back to `eval=FALSE`, or maybe try leaving as it should automatically give `character(0)` once fixed (as it does for 2021):

```
setdiff(check_iphc_spp_name(countData2023),
        check_iphc_spp_name())
> [1] "Worms"             "unident. algae"    "Giant Wrymouth"
> [4] "Yellow Irish Lord"
```

2021: 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, in 2021 added 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.

2021: 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.

2022: The `setdiff()` just gave `"Rosethorn Rockfish" "Red Irish Lord"` which do appear in latest synopsis report with older IPHC data, which I presume is why I just need to update `iphc-spp-names.csv` for which we have `NA` and `*****` as their IPHC names. Doing that, and rerunning the `setdiff()` now correctly gives an empty result (above and then here also):

2023: none of the non-matching ones are of interest

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(countData2023)
> [1] "Unident. Flatfish" "Basketstar"        "Worms"
> [4] "unident. algae"    "Brittle Star"      "Giant Wrymouth"
> [7] "unident. Sculpin"  "Yellow Irish Lord" "unident. Hagfish"
```

```
# 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(countData2023),
        check_iphc_spp_name())
> [1] "Worms"             "unident. algae"    "Giant Wrymouth"
> [4] "Yellow Irish Lord"
```

```
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(countData2023,
                  overwrite = TRUE)
> v Setting active project to '/Users/seananderson/src/gfiphc'
```

```
> v Saving 'countData2023' to 'data/countData2023.rda'
> * Document your data (see 'https://r-pkgs.org/data.html')
```

```
usethis::use_data(setData2023,
                  overwrite = TRUE)
> v Saving 'setData2023' to 'data/setData2023.rda'
> * Document your data (see 'https://r-pkgs.org/data.html')
```

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