GAP Production Data Documentation

Bering Sea Survey Team

Gulf of Alaska and Aleutian Island Survey Team

Table of contents

# 1. Survey Background

## 1.1 What we do

## 1.2 Who is conducting the research?

Scientists from the Alaska Fisheries Science Center’s Groundfish Assessment Program (GAP) conduct these bottom trawl surveys with participation from the Alaska Department of Fish & Game (ADF&G), the International Pacific Halibut Commission (IPHC), universities, and other organizations. This research is conducted primarily on chartered fishing vessels.

## 1.3 What is the research objective?

Learn more about the [program](https://www.fisheries.noaa.gov/alaska/science-data/groundfish-assessment-program-bottom-trawl-surveys). The objectives of these surveys are to:

* monitor the population and environmental trends in the marine ecosystem of the Bering Sea, Aleutian Islands, and Gulf of Alaska,
* produce fishery-independent biomass (weight) and abundance (number) estimates for commercially important fish and crab species, and
* collect other biological and environmental data for use in ecosystem-based fishery management.

## 1.4 Who is conducting the research?

Scientists from the Alaska Fisheries Science Center conduct these bottom trawl surveys with participation from the Alaska Department of Fish & Game (ADF&G), the International Pacific Halibut Commission (IPHC), and universities. This research is conducted on chartered fishing vessels.

## 1.5 Bottom trawl surveys and regions

|  |
| --- |
| Strata used in the all surveys. |

Each survey conducted by the [Groundfish Assessment Program](https://www.fisheries.noaa.gov/alaska/population-assessments/north-pacific-groundfish-stock-assessments-and-fishery-evaluation) are multispecies bottom trawl surveys. We collect environmental and biological data to assess how climate variability and [loss of sea](https://www.fisheries.noaa.gov/alaska/ecosystems/habitat-and-ecological-processes-research-regarding-loss-sea-ice) ice are affecting bottom-dwelling marine life on the Bering Sea shelf. We monitor trends in the distribution (location and movement patterns) and abundance of groundfish and crab species as well as oceanographic data (e.g., water temperature, depth). We collect biological information such as organism weight, length, stomachs to learn about diets, and [otoliths](https://www.fisheries.noaa.gov/alaska/science-data/alaska-age-and-growth-procedures-otolith-examination) to [determine fish ages](https://www.fisheries.noaa.gov/alaska/science-data/fish-otolith-chronologies). We use this information in [annual stock assessments](https://www.fisheries.noaa.gov/alaska/population-assessments/north-pacific-groundfish-stock-assessments-and-fishery-evaluation) and to assess the state of the ecosystem. This research is conducted on fishing industry contract vessels.

| **Survey** | **Survey Definition ID** | **Years** | **Depth (m)** | **Area (km2)** | **# Statistical Areas** | **# Possible Stations** |
| --- | --- | --- | --- | --- | --- | --- |
| Aleutian Islands Bottom Trawl Survey | 52 | 2024 - 1991 (14) | 1 - 500 | 64,415.0 | 80 | 1,312 |
| Eastern Bering Sea Slope Bottom Trawl Survey | 78 | 2016 - 2002 (6) | 201 - 1,200 | 32,861.3 | 37 |  |
| Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 2024 - 1982 (42) | 1 - 200 | 492,989.9 | 28 | 515 |
| Gulf of Alaska Bottom Trawl Survey | 47 | 2023 - 1990 (16) | 1 - 1,000 | 313,784.9 | 37 | 6,939 |
| Northern Bering Sea Crab/Groundfish Survey - Eastern Bering Sea Shelf Survey Extension | 143 | 2023 - 2010 (6) | 1 - 100 | 198,866.8 | 4 | 144 |

### 1.5.1 **Aleutian Islands**

Most recent data report: (Von Szalay et al., 2023)

* Upper Continental Slope of the Aleutian Islands from Unimak Pass to Stalemate Bank
* Triennial (1990s)/Biennial since 2000 in even years, since 1992
* Modified Index-Stratified Random of Successful Stations Survey Design
* Important commercial fish species include Atka mackerel, [Pacific ocean perch](https://www.fisheries.noaa.gov/species/pacific-ocean-perch), [walleye pollock](https://www.fisheries.noaa.gov/species/alaska-pollock), [Pacific cod](https://www.fisheries.noaa.gov/species/pacific-cod), [sablefish](https://www.fisheries.noaa.gov/species/sablefish), and other rockfish species.

|  |
| --- |
| Strata used in the Aleutian Islands bottom trawl survey. |

### 1.5.2 **Gulf of Alaska**

Most recent data report: (**GOA2023?**)

* Continental Shelf and Upper Slope of the Gulf of Alaska extending from the Islands of Four Mountains 2,300 km east to Dixon Entrance
* Triennial (1990s)/Biennial since 2001 in odd years, since 1991
* Stratified Random Survey Design
* Important commercial species in the Gulf of Alaska include [Pacific ocean perch](https://www.fisheries.noaa.gov/species/pacific-ocean-perch), [walleye pollock](https://www.fisheries.noaa.gov/species/alaska-pollock), [Pacific cod](https://www.fisheries.noaa.gov/species/pacific-cod), flatfish, and other rockfish species.

|  |
| --- |
| Strata used in the Gulf of Alaska bottom trawl survey. |

### 1.5.3 **Eastern Bering Sea Shelf**

Most recent data report: (Markowitz et al., In prep)

* The continental shelf of the eastern Bering Sea from the Aleutian Islands to the Bering Strait
* Conducted annually since 1982.
* Uses a stratified systematic sampling survey design with fixed stations at center of 20 x 20 nm grid.
* Similar in design to the northern Bering Sea shelf bottom trawl survey.
* Focus species for the Bering Sea include [walleye pollock](https://www.fisheries.noaa.gov/species/alaska-pollock), [Pacific cod](https://www.fisheries.noaa.gov/species/pacific-cod), [Greenland turbot](https://www.fisheries.noaa.gov/species/greenland-turbot), [yellowfin sole](https://www.fisheries.noaa.gov/species/yellowfin-sole), [northern rock sole](https://www.fisheries.noaa.gov/species/rock-sole), [red king crab](https://www.fisheries.noaa.gov/species/red-king-crab), and [snow](https://www.fisheries.noaa.gov/species/alaska-snow-crab) and Tanner crabs.

|  |
| --- |
| Strata used in the Eastern Bering Sea bottom trawl survey. |

### 1.5.4 **Northern Bering Sea**

Most recent data report: (Markowitz et al., In prep)

* The continental shelf of the northern Bering Sea, including the area north of St. Lawrence Island and Norton Sound
* Biennial/Annual; conducted intermittently since 2010
* Uses a stratified systematic sampling survey design with fixed stations at center of 20 x 20 nm grid.
* Similar in design to the eastern Bering Sea shelf bottom trawl survey.

|  |
| --- |
| Strata used in the Northern Bering Sea bottom trawl survey. |

### 1.5.5 **Eastern Bering Sea Upper Continental Slope**

Most recent data report: (Hoff, 2016)

* The eastern Bering Sea upper continental slope survey area extends from Unalaska and Akutan Islands to the U.S.-Russian Maritime Boundary at 61° N near the International Date Line (166° E to 180° W) at depths from 200 to 1,200 m
* Conducted intermittently since 2002 (funding dependent)
* Modified Index-Stratified Random of Successful Stations Survey Design
* Focus species for the Bering Sea slope include giant grenadier, [Pacific ocean perch](https://www.fisheries.noaa.gov/species/pacific-ocean-perch), popeye grenadier, [walleye pollock](https://www.fisheries.noaa.gov/species/alaska-pollock), and [arrowtooth flounder](https://www.fisheries.noaa.gov/species/arrowtooth-flounder).

|  |
| --- |
| Strata used in the Bering Sea Slope bottom trawl survey. |

# 2. Workflow

## 2.1 Operational Product Development Timeline

Over the course of the year, the survey team is developing a variety of different data products. Planning and preparation for surveys happens in the late winter and spring, surveys occur in the summer, data validation takes place over the course of the survey and after the survey, and data products are produced through fall and late winter.

|  | January | February | March | April | May | June | July | August | September | October | November | December |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Surveys |  |  |  |  | 1 | 1 | 1 | 1 |  |  |  |  |
| Planning | 1 | 1 | 1 |  |  |  |  |  | 1 | 1 | 1 | 1 |
| Development | 1 | 1 | 1 | 1 | 1 |  |  |  |  | 1 | 1 | 1 |
| Deployment (survey deliverables) |  |  |  |  |  |  |  |  | 1 | 1 | 1 | 1 |
| Deployment (survey operations) |  |  |  |  | 1 | 1 | 1 | 1 |  |  |  |  |
| Triage (fixing bugs and errors) | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| User feedback and brainstorming | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

## 2.2 Data workflow from boat to production

Organisms first need to be collected aboard the vessel before data can be entered into tablets.

|  |
| --- |
| Figure 2.1: Simplified boat deck processing workflow. |

The objective of this process is to take raw data, QA/QC and clean these data, curate standard data products for these survey. Please note, through this process we are not providing “data” (what we consider lower level data material; see the data levels section below) but “data products”, which is intended to facilitate the most fool-proof standard interpretation of the data. These data products only use data from standard and validated hauls, and has undergone careful review.

**Once survey data collected on the vessel has been checked and validated**, the [gap\_products/code/run.R](https://github.com/afsc-gap-products/gap_products/blob/main/code/run.R) script is used to orchestrate a sequence of programs that calculate the standard data products resulting from the NOAA AFSC GAP bottom trawl surveys. Standard data products are the CPUE, BIOMASS, SIZECOMP, and AGECOMP tables in the GAP\_PRODUCTS Oracle schema. The tables are slated to be updated twice a year: once after the survey season following finalization of that summer’s bottom trawl survey data to incorporate the new catch, size, and effort data and once prior to an upcoming survey to incorporate new age data that were processed after the prior summer’s survey season ended. This second pre-survey production run will also incorporate changes in the data due to the specimen voucher process as well as other post-hoc changes in the survey data.

The data from these surveys constitute a **living data set** so we can continue to **provide the best available data to all partners, stakeholders, and fellow scientists**.

|  |
| --- |
| Figure 2.2: Simplified data workflow from boat to production. |

During each data product run cycle:

1. Versions of the tables in GAP\_PRODUCTS are locally imported within the gap\_products repository to compare with the updated production tables. Any changes to a production table will be compared and checked to make sure those changes are intentional and documented.
2. Use the gapindex R package to calculate the four major standard data products: CPUE, BIOMASS, SIZECOMP, AGECOMP. These tables are compared and checked to their respective locally saved copies and any changes to the tables are vetted and documented. These tables are then uploaded to the GAP\_PRODUCTS Oracle schema.
3. Calculate the various materialized views for AKFIN and FOSS purposes. Since these are derivative of the tables in GAP\_PRODUCTS as well as other base tables in RACEBASE and RACE\_DATA, it is not necessary to check these views in addition to the data checks done in the previous steps.

|  |
| --- |
| Figure 2.3: Major end-users of the GAP data product tables. |

## 2.3 Data levels

GAP produces numerous data products that are subjected to different levels of processing, ranging from raw to highly-derived. The suitability of these data products for analysis varies and there is ambiguity about which data products can be used for which purpose. This ambiguity can create challenges in communicating about data products and potentially lead to misunderstanding and misuse of data. One approach to communicating about the level of processing applied to data products and their suitability for analysis is to describe data products using a Data Processing Level system. Data Processing Level systems are widely used in earth system sciences to characterize the extent of processing that has been applied to data products. For example, the NOAA National Centers for Environmental Information (NCEI) Satellite Program uses a Data Processing Level system to describe data on a scale of 0-4, where Level 0 is raw data and Level 4 is model output or results from analysis. Example of how [NASA remote sensing data products](https://ladsweb.modaps.eosdis.nasa.gov/search/) are shared through a public data portal with levels of data processing and documentation.

For more information, see [Sean Rohan’s October 2022 SCRUGS presentation](https://docs.google.com/presentation/d/1rWSZpeghWJqzWMIa5oBc4BCoy-zy1Yue86RoTw58u6M/edit?usp=sharing) on the topic.

* **Level 0**: Raw and unprocessed data. Ex: Data on the G drive, some tables in RACE\_DATA
* **Level 1**: Data products with QA/QC applied that may or may not be expanded to analysis units, but either not georeferenced or does not include full metadata. Ex: Some tables in RACE\_DATA and RACEBASE
* **Level 2**: Analysis-ready data products that are derived for a standardized extent and account for zeros and missing/bad data. Ex: CPUE tables, some data products in public-facing archives and repositories
* **Level 3**: Data products that are synthesized across a standardized extent, often inputs in a higher-level analytical product. Ex: Abundance indices, some data products in public-facing archives and repositories
* **Level 4**: Analytically generated data products that are derived from lower-level data, often to inform management. Ex: Biological reference points from stock assessments, Essential Fish Habitat layers, indicators in Ecosystem Status Reports and Ecosystem and Socioeconomic Profiles

# 3. News

## 3.1 News/change logs

– [GAP\_PRODUCTS ChangeLog (last produced on 2024-09-03) using gapindex v2.2.0](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2024-09-03.txt): Run completed by: Ned Laman, Zack Oyafuso

– [GAP\_PRODUCTS ChangeLog (last produced on 2024-08-29) using gapindex v2.2.0](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2024-08-29.txt): The additions of previous years’ age data and 2024 EBS catch, effort, and size data

– [GAP\_PRODUCTS ChangeLog (last produced on 2024-08-20) using gapindex v2.2.0](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2024-08-20.txt): Initial 2024 post-survey run with new ages since last run and all of EBS Shelf 2024 survey data but none of AI 2024 survey data. While trying to update the records in the GAP\_PRODUCTS table, the connection was terminated, partially uploading records in the agecomp tables and outputting NA to the N\_HAUL and N\_LENGTH fields in the biomass tables. At this point, the GAP\_PRODUCTS tables are incomplete. The AKFIN and FOSS tables were NOT updated in this run.

– [GAP\_PRODUCTS ChangeLog (last produced on 2024-05-04) using gapindex v2.2.0](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2024-05-04.txt): A development branch version of gapindex called [using\_datatable](https://github.com/afsc-gap-products/gap_products/tree/using_datatable) uses the data.table package for many dataframe manipulations, which greatly decreased the computation time of many of the functions. There were no major changes in the calculations in this version of the gapindex package and thus the major changes listed below are not related to the gapindex package. The only major change from this run was the addition of GOA 2023 Pacific Ocean perch read otolith data.

– [GAP\_PRODUCTS ChangeLog (last produced on 2024-04-09) using gapindex v2.2.0](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2024-04-09.txt): A development branch version of gapindex called [using\_datatable](https://github.com/afsc-gap-products/gap_products/tree/using_datatable) uses the data.table package for many dataframe manipulations, which greatly decreased the computation time of many of the functions. There were no major changes in the calculations in this version of the gapindex package and thus the major changes listed below are not related to the gapindex package.

– [GAP\_PRODUCTS ChangeLog (last produced on 2024-02-29) using gapindex v2.2.0](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2024-02-29.txt): A new version of gapindex [2.2.0](https://github.com/afsc-gap-products/gapindex/releases/tag/v2.2.0) was used for this production run and now accesses taxonomic information from RACEBASE.SPECIES instead of GAP\_PRODUCTS.TAXONOMIC\_CLASSIFICATION. As a result, there will be some SPECIES\_CODE values that are supported due to slight differences between the two tables. Discussion in this [github issue #54](https://github.com/afsc-gap-products/gapindex/issues/54). As a result there are new cpue records for SPECIES\_CODE values 22290 and 22292 and removed cpue records for SPECIES\_CODE values 21345, 22200 and 69326.

– [GAP\_PRODUCTS ChangeLog (last produced on 2024-01-09) using gapindex v2.1.3](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2024-01-09.txt): A new version of gapindex ([v2.1.3](https://github.com/afsc-gap-products/gapindex/releases/tag/v2.1.3)) was used to produced these data. Data for SPECIES\_CODE 68590 (Chionoecetes hybrids) are now removed, per this issue (https://github.com/afsc-gap-products/gap\_products/issues/3). New read otolith data were incorporated into the age compositions. GOA depth subareas are now included in the size comps, and there were some modifications with EBS skate length data that are now incorporated into the length compositions.

– [GAP\_PRODUCTS ChangeLog (last produced on 2023-11-17) using gapindex v2.1.2](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2023-11-17.txt): A new version of gapindex ([v2.1.2](https://github.com/afsc-gap-products/gapindex/releases/tag/v2.1.2)) was used to produced these data. There was a slight change to how subarea biomass totals are calculated that was not fully addressed in v2.1.1. The modified biomass records reflect this change.

– [GAP\_PRODUCTS ChangeLog (last produced on 2023-11-14) using gapindex v2.1.1](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2023-11-14_v2-1-1.txt): A new version of gapindex (v2.1.1) was used to produced these data. There was a slight change to how subarea biomass totals are calculated. The modified biomass records reflect this change. New 2022 otolith data were available since the last iteration of the GAP\_PRODUCTS for Aleutian Island Pacific ocean perch and northern rockifsh and Eastern Bering Sea northern rock sole. Zero-filled CPUE records for four GOA species codes (SPECIES\_CODE: 21210, 30010, 30360, 77102, 98101) were added due to how the 1990 data were integrated in the last production run of GAP\_PRODUCTS. Two Arctic cod (SPECIES\_CODE: 21725) and one plain sculpin (SPECIES\_CODE: 21371) count records were modified in the NBS data, which changes the numerical CPUE estimates for those hauls which changes the estimated population abundance and size composition for those species.

– [Groundfish Assessment Program Survey Data Serving and Data Improvements](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2023-06-20_datachangesbreif): Initial data changes brief distributed to SSMA and other partners by Ned Laman, Zack Oyafuso, and Emily Markowitz

– [Run 2023-06-01 gapindex v2.1.0](https://raw.githubusercontent.com/afsc-gap-products/gap_products/main/content/intro-news/2023-06-01_v2-1-0.txt): Initial compiling and planning notes

# 4. Code of Conduct

## 4.1 What are Codes of Conduct?

Codes of Conduct are voluntary sets of rules that assist creators, developers, and users of code and data with data protection compliance and accountability in specific sectors or relating to particular processing operations.

Codes can help organizations to ensure all participants follow best practices and rules designed specifically for their sector or processing operations, thus enhancing compliance and collaboration. They are developed and managed by an association or other body (the ‘Code Owner’) which is representative of a sector (or category of data controllers or processors), with the expert and sectoral knowledge of how to enhance data protection in their area.

### 4.1.1 [Code of Conduct](https://github.com/nmfs-opensci/.github/blob/main/CODE_OF_CONDUCT.md) from the [nmfs-opensci GitHub](https://nmfs-opensci.github.io/).

# 5. NOAA Fisheries Open Science Code of Conduct

This code of conduct was developed and adapted from the Atom code of conduct in October 2021.

## 5.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and expression, level of experience, nationality, personal appearance, race, religion, or sexual identity and orientation.

## 5.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

* Using welcoming and inclusive language
* Being respectful of differing viewpoints and experiences
* Gracefully accepting constructive criticism
* Focusing on what is best for the community
* Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

* The use of sexualized language or imagery and unwelcome sexual attention or advances
* Trolling, insulting/derogatory comments, and personal or political attacks
* Public or private harassment
* Publishing others’ private information, such as a physical or electronic address, without explicit permission
* Other conduct which could reasonably be considered inappropriate in a professional setting

## 5.3 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

## 5.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

## 5.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. Further details of specific enforcement policies may be posted separately.

## 5.6 Attribution

This Code of Conduct is adapted from the [Contributor Covenant](https://contributor-covenant.org), version 1.4, available at [https://contributor-covenant.org/version/1/4](https://contributor-covenant.org/version/1/4/)

# 6. Data description

## 6.1 Data tables

### 6.1.1 AGECOMP

Stratum- and region-level age compositions by sex.

Number of rows: 672,956

Number of columns: 10

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AGE

Taxon age bin (yrs)

integer

NUMBER(38,0)

Age bin of taxon. Age bin of a taxon in years estimated by the age comp estimate. Age -9 indicates unaged lengths for a particular sex because no otoliths were collected for that sex/length combination. Age -99 indicates a case where no lengths were collected within a stratum for a species/year even though catch numbers were recorded.

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

AREA\_ID\_FOOTPRINT

Survey Footprint

text

VARCHAR2(4000 BYTE)

Survey footprint, usually equivalent to the SURVEY\_DEFINITION\_ID with the exception of the Standard and Standard +NW survey footprints in the Eastern Bering Sea shelf bottom trawl survey

LENGTH\_MM\_MEAN

Mean length at age weighted by numbers at length

numeric

NUMBER(38,3)

Mean length (millimeters)

LENGTH\_MM\_SD

Standard deviation of length at age weighted by numbers at length

numeric

NUMBER(38,3)

Variance of mean length.

POPULATION\_COUNT

Estimated population

numeric

NUMBER(38,0)

The estimated population caught in the survey for a species, group, or total for a given survey.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

Sex of a specimen where “1” = “Male”, “2” = “Female”, “3” = Unsexed.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 6.1.2 AREA

Information related to the various strata, subareas, INPFC and NMFS management areas, and regions for the Aleutian Islands, Gulf of Alaska, and Bering Sea shelf and slope bottom trawl surveys.

Number of rows: 395

Number of columns: 9

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

AREA\_KM2

Area (km2)

kilometers squared

NUMBER(38,3)

Area in square kilometers.

AREA\_NAME

Area ID name

text

VARCHAR2(4000 BYTE)

Descriptive name of each AREA\_ID. These names often identify the region, depth ranges, or other regional information for the area ID.

AREA\_TYPE

Area ID type description

category

VARCHAR2(255 BYTE)

The type of stratum that AREA\_ID represents. Types include: STRATUM (the smallest building-block unit of area in these surveys), REGION, DEPTH, SUBAREA, INPFC BY DEPTH, INPFC, SUBAREA BY DEPTH, REGULATORY AREA, NMFS STATISTICAL AREA.

DEPTH\_MAX\_M

Area ID maximum depth (m)

meters

NUMBER(38,3)

Maximum depth (meters).

DEPTH\_MIN\_M

Area ID minimum depth (m)

meters

NUMBER(38,3)

Minimum depth (meters).

DESCRIPTION

Description

text

VARCHAR2(4000 BYTE)

Description of row observation.

DESIGN\_YEAR

Design year

year

NUMBER(10,0)

Year ID associated with a given value AREA\_ID. This field describes the changes in the survey design over time.

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

### 6.1.3 BIOMASS

Stratum/subarea/region-level mean CPUE (weight and numbers), total biomass, and total abundance with associated variances.

Number of rows: 2,580,501

Number of columns: 16

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

BIOMASS\_MT

Estimated biomass

numeric

NUMBER(38,6)

The estimated total biomass.

BIOMASS\_VAR

Estimated biomass variance

numeric

NUMBER(38,6)

The estimated variance associated with the total biomass.

CPUE\_KGKM2\_MEAN

Mean weight CPUE

kilograms per kilometers squared

NUMBER(38,6)

The mean catch weight (kilograms) per unit effort (area swept by the net, units squared kilometers).

CPUE\_KGKM2\_VAR

Variance of the mean weight CPUE

kilograms per kilometers squared

NUMBER(38,6)

The variance of mean catch weight (kilograms) per unit effort (area swept by the net, units squared kilometers).

CPUE\_NOKM2\_MEAN

Mean numeric CPUE

count per kilometers squared

NUMBER(38,6)

The mean of numerical catch per unit effort (area swept by the net, units square kilometers).

CPUE\_NOKM2\_VAR

Variance of the mean numeric CPUE

count per kilometers squared

NUMBER(38,6)

The variance of mean numerical catch per unit effort (area swept by the net, units square kilometers).

N\_COUNT

Hauls with taxon counts

numeric

NUMBER(38,0)

Total number of hauls with positive count data.

N\_HAUL

Valid hauls

count

NUMBER(38,0)

Total number of hauls.

N\_LENGTH

Hauls with taxon lengths

count

NUMBER(38,0)

Total number of hauls with length data.

N\_WEIGHT

Hauls with catch

count

NUMBER(38,0)

Total number of hauls with positive catch biomass.

POPULATION\_COUNT

Estimated population

numeric

NUMBER(38,0)

The estimated population caught in the survey for a species, group, or total for a given survey.

POPULATION\_VAR

Estimated population variance

numeric

NUMBER(38,6)

The estimated population variance caught in the survey for a species, group, or total for a given survey.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 6.1.4 CPUE

Haul-level zero-filled weight and numerical catch-per-unit-effort.

Number of rows: 21,416,058

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_SWEPT\_KM2

Area swept (km)

kilometers

NUMBER(38,6)

The area the net covered while the net was fishing (kilometers squared), defined as the distance fished times the net width.

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

Total whole number of individuals caught in haul or samples collected.

CPUE\_KGKM2

Weight CPUE (kg/km2)

kilograms per kilometers squared

NUMBER(38,6)

Catch weight (kilograms) per unit effort (area swept by the net, units square kilometers).

CPUE\_NOKM2

Number CPUE (no/km2)

count per kilometers squared

NUMBER(38,6)

Numerical catch per unit effort (area swept by the net, units square kilometers).

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

WEIGHT\_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

Weight (thousandths of a kilogram) of individuals in a haul by taxon.

### 6.1.5 SURVEY\_DESIGN

This table contains for a given survey (via SURVEY\_DEFINITION\_ID) and survey year (YEAR), which version (DESIGN\_YEAR) of the AREA\_IDs that were used to calculate the various standard data products.

Number of rows: 87

Number of columns: 3

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

DESIGN\_YEAR

Design year

year

NUMBER(10,0)

Year ID associated with a given value AREA\_ID. This field describes the changes in the survey design over time.

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 6.1.6 METADATA\_TABLE

This table is used to string together the various table comments for the tables in GAP\_PRODUCTS.

Number of rows: 21

Number of columns: 3

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

METADATA\_SENTENCE

Sentence

text

VARCHAR2(4000 BYTE)

Table metadata sentence.

METADATA\_SENTENCE\_NAME

Metadata sentence name

text

VARCHAR2(4000 BYTE)

Name of table metadata sentence.

METADATA\_SENTENCE\_TYPE

Sentence type

text

VARCHAR2(4000 BYTE)

Type of sentence to have in table metadata.

### 6.1.7 STRATUM\_GROUPS

Lookup table for which strata are contained within a given subarea, INPFC or NMFS management area, or region for the Aleutian Islands, Gulf of Alaska, and Bering Sea shelf and slope bottom trawl surveys.

Number of rows: 768

Number of columns: 4

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

DESIGN\_YEAR

Design year

year

NUMBER(10,0)

Year ID associated with a given value AREA\_ID. This field describes the changes in the survey design over time.

STRATUM

Stratum ID

ID key code

NUMBER(10,0)

RACE database statistical area for analyzing data. Strata were designed using bathymetry and other geographic and habitat-related elements. The strata are unique to each survey region. Stratum of value 0 indicates experimental tows.

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

### 6.1.8 SIZECOMP

Stratum/subarea/region-level size compositions by sex.

Number of rows: 3,216,121

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

LENGTH\_MM

Length of a specimen

millimeters

NUMBER(10,0)

Length bin in millimeters. A length of -9 indicates cases where no lengths were collected within a stratum for a species/year, even though catch numbers were recorded.

POPULATION\_COUNT

Estimated population

numeric

NUMBER(38,0)

The estimated population caught in the survey for a species, group, or total for a given survey.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

Sex of a specimen where “1” = “Male”, “2” = “Female”, “3” = Unsexed.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

# 7. Data description

[AKFIN Answers](https://akfin.psmfc.org/akfin-answers/) is an Oracle BI tool used for distributing data to stock assessors and other users. Usernames and passwords are distinct from direct AKFIN database credentials.

## 7.1 Data tables

### 7.1.1 AKFIN\_AGECOMP

Mirror of GAP\_PRODUCTS.AGECOMP.

Number of rows: 680,094

Number of columns: 10

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AGE

Taxon age bin (yrs)

integer

NUMBER(38,0)

Age bin of taxon. Age bin of a taxon in years estimated by the age comp estimate. Age -9 indicates unaged lengths for a particular sex because no otoliths were collected for that sex/length combination. Age -99 indicates a case where no lengths were collected within a stratum for a species/year even though catch numbers were recorded.

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

AREA\_ID\_FOOTPRINT

Survey Footprint

text

VARCHAR2(4000 BYTE)

Survey footprint, usually equivalent to the SURVEY\_DEFINITION\_ID with the exception of the Standard and Standard +NW survey footprints in the Eastern Bering Sea shelf bottom trawl survey

LENGTH\_MM\_MEAN

Mean length at age weighted by numbers at length

numeric

NUMBER(38,3)

Mean length (millimeters)

LENGTH\_MM\_SD

Standard deviation of length at age weighted by numbers at length

numeric

NUMBER(38,3)

Variance of mean length.

POPULATION\_COUNT

Estimated population

numeric

NUMBER(38,0)

The estimated population caught in the survey for a species, group, or total for a given survey.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

Sex of a specimen where “1” = “Male”, “2” = “Female”, “3” = Unsexed.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 7.1.2 AKFIN\_AREA

Mirror of GAP\_PRODUCTS.AREA.

Number of rows: 395

Number of columns: 9

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

AREA\_KM2

Area (km2)

kilometers squared

NUMBER(38,3)

Area in square kilometers.

AREA\_NAME

Area ID name

text

VARCHAR2(4000 BYTE)

Descriptive name of each AREA\_ID. These names often identify the region, depth ranges, or other regional information for the area ID.

AREA\_TYPE

Area ID type description

category

VARCHAR2(255 BYTE)

The type of stratum that AREA\_ID represents. Types include: STRATUM (the smallest building-block unit of area in these surveys), REGION, DEPTH, SUBAREA, INPFC BY DEPTH, INPFC, SUBAREA BY DEPTH, REGULATORY AREA, NMFS STATISTICAL AREA.

DEPTH\_MAX\_M

Area ID maximum depth (m)

meters

NUMBER(38,3)

Maximum depth (meters).

DEPTH\_MIN\_M

Area ID minimum depth (m)

meters

NUMBER(38,3)

Minimum depth (meters).

DESCRIPTION

Description

text

VARCHAR2(4000 BYTE)

Description of row observation.

DESIGN\_YEAR

Design year

year

NUMBER(10,0)

Year ID associated with a given value AREA\_ID. This field describes the changes in the survey design over time.

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

### 7.1.3 AKFIN\_BIOMASS

Mirror of GAP\_PRODUCTS.BIOMASS.

Number of rows: 2,656,482

Number of columns: 16

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

BIOMASS\_MT

Estimated biomass

numeric

NUMBER(38,6)

The estimated total biomass.

BIOMASS\_VAR

Estimated biomass variance

numeric

NUMBER(38,6)

The estimated variance associated with the total biomass.

CPUE\_KGKM2\_MEAN

Mean weight CPUE

kilograms per kilometers squared

NUMBER(38,6)

The mean catch weight (kilograms) per unit effort (area swept by the net, units squared kilometers).

CPUE\_KGKM2\_VAR

Variance of the mean weight CPUE

kilograms per kilometers squared

NUMBER(38,6)

The variance of mean catch weight (kilograms) per unit effort (area swept by the net, units squared kilometers).

CPUE\_NOKM2\_MEAN

Mean numeric CPUE

count per kilometers squared

NUMBER(38,6)

The mean of numerical catch per unit effort (area swept by the net, units square kilometers).

CPUE\_NOKM2\_VAR

Variance of the mean numeric CPUE

count per kilometers squared

NUMBER(38,6)

The variance of mean numerical catch per unit effort (area swept by the net, units square kilometers).

N\_COUNT

Hauls with taxon counts

numeric

NUMBER(38,0)

Total number of hauls with positive count data.

N\_HAUL

Valid hauls

count

NUMBER(38,0)

Total number of hauls.

N\_LENGTH

Hauls with taxon lengths

count

NUMBER(38,0)

Total number of hauls with length data.

N\_WEIGHT

Hauls with catch

count

NUMBER(38,0)

Total number of hauls with positive catch biomass.

POPULATION\_COUNT

Estimated population

numeric

NUMBER(38,0)

The estimated population caught in the survey for a species, group, or total for a given survey.

POPULATION\_VAR

Estimated population variance

numeric

NUMBER(38,6)

The estimated population variance caught in the survey for a species, group, or total for a given survey.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 7.1.4 AKFIN\_CATCH

Catch subsetted for standard GAP bottom trawl stations (i.e., HAULJOIN values associated with ABUNDANCE\_HAUL = Y in RACEBASE.HAUL).

Number of rows: 973,540

Number of columns: 6

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

CATCHJOIN

Catch observation ID

ID key code

NUMBER(38,0)

Unique integer ID assigned to each survey, vessel, year, and catch observation combination.

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

Total whole number of individuals caught in haul or samples collected.

CRUISEJOIN

Cruise ID

ID key code

NUMBER(38,0)

Unique integer ID assigned to each survey, vessel, and year combination.

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

WEIGHT\_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

Weight (thousandths of a kilogram) of individuals in a haul by taxon.

### 7.1.5 AKFIN\_CPUE

Mirror of GAP\_PRODUCTS.CPUE

Number of rows: 21,848,430

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_SWEPT\_KM2

Area swept (km)

kilometers

NUMBER(38,6)

The area the net covered while the net was fishing (kilometers squared), defined as the distance fished times the net width.

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

Total whole number of individuals caught in haul or samples collected.

CPUE\_KGKM2

Weight CPUE (kg/km2)

kilograms per kilometers squared

NUMBER(38,6)

Catch weight (kilograms) per unit effort (area swept by the net, units square kilometers).

CPUE\_NOKM2

Number CPUE (no/km2)

count per kilometers squared

NUMBER(38,6)

Numerical catch per unit effort (area swept by the net, units square kilometers).

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

WEIGHT\_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

Weight (thousandths of a kilogram) of individuals in a haul by taxon.

### 7.1.6 AKFIN\_CRUISE

Cruise data for the following five standard GAP bottom trawl regions.

Number of rows: 174

Number of columns: 10

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

CRUISE

Cruise Name

ID key code

NUMBER(38,0)

This is a six-digit integer identifying the cruise number of the form: YYYY99 (where YYYY = year of the cruise; 99 = 2-digit number and is sequential; 01 denotes the first cruise that vessel made in this year, 02 is the second, etc.).

CRUISEJOIN

Cruise ID

ID key code

NUMBER(38,0)

Unique integer ID assigned to each survey, vessel, and year combination.

DATE\_END

End date

YYYY-MM-DD

DATE

The date (YYYY-MM-DD) of the end of the event (e.g., cruise).

DATE\_START

Start date

YYYY-MM-DD

DATE

The date (YYYY-MM-DD) of the beginning of the event (e.g., cruise).

SPONSOR\_ACRONYM

NA

NA

NA

NA

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_NAME

Survey name official

text

VARCHAR2(255 BYTE)

Long name of the survey conducted

VESSEL\_ID

Vessel ID

ID key code

NUMBER(38,0)

ID number of the vessel used to collect data for that haul. The column vessel\_id is associated with the vessel\_name column. Note that it is possible for a vessel to have a new name but the same vessel id number. For a complete list of vessel ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

VESSEL\_NAME

Vessel name

text

VARCHAR2(255 BYTE)

Name of the vessel used to collect data for that haul. The column vessel\_name is associated with the vessel\_id column. Note that it is possible for a vessel to have a new name but the same vessel id number. For a complete list of vessel ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 7.1.7 AKFIN\_HAUL

Standard GAP bottom trawl haul (station) data.

Number of rows: 34,263

Number of columns: 25

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

ACCESSORIES

Type of gear accessories used on the net

ID key code

NUMBER(38,0)

Type of accessories used on net. For a complete list of accessories ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

BOTTOM\_TYPE

Seafloor bottom type code

ID key code

NUMBER(38,0)

Bottom type on sea floor at haul location. For a complete list of bottom type ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

CRUISEJOIN

Cruise ID

ID key code

NUMBER(38,0)

Unique integer ID assigned to each survey, vessel, and year combination.

DATE\_TIME\_START

Start date and time

MM/DD/YYYY HH::MM

TIMESTAMP

The date (MM/DD/YYYY) and time (HH:MM) of the beginning of the haul. All dates and times are in Alaska time (AKDT) of Anchorage, AK, USA (UTC/GMT -8 hours).

DEPTH\_GEAR\_M

Depth of gear (m)

degrees Celsius

NUMBER(38,1)

Depth of gear (meters).

DEPTH\_M

Depth (m)

degrees Celsius

NUMBER(38,1)

Bottom depth (meters).

DISTANCE\_FISHED\_KM

Distance fished (km)

degrees Celsius

NUMBER(38,3)

Distance the net fished (thousands of kilometers).

DURATION\_HR

Tow duration (decimal hr)

hours

NUMBER(38,1)

This is the elapsed time between start and end of a haul (decimal hours).

GEAR

Type of gear used on the net

ID key code

NUMBER(38,0)

Type of gear used on net. For a complete list of gear ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

GEAR\_TEMPERATURE\_C

Gear temperature (degrees Celsius)

degrees Celsius

NUMBER(38,1)

Temperature recorded by net gear (tenths of a degree Celsius); NA indicates removed or missing values.

HAUL

Haul number

ID key code

NUMBER(38,0)

This number uniquely identifies a sampling event (haul) within a cruise. It is a sequential number, in chronological order of occurrence.

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

HAUL\_TYPE

Haul sampling type

ID key code

NUMBER(38,0)

Type of haul sampling method. For a complete list of haul type ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

LATITUDE\_DD\_END

End latitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Latitude (one hundred thousandth of a decimal degree) of the end of the haul.

LATITUDE\_DD\_START

Start latitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Latitude (one hundred thousandth of a decimal degree) of the start of the haul.

LONGITUDE\_DD\_END

End longitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Longitude (one hundred thousandth of a decimal degree) of the end of the haul.

LONGITUDE\_DD\_START

Start longitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Longitude (one hundred thousandth of a decimal degree) of the start of the haul.

NET\_HEIGHT\_M

Net height (m)

meters

NUMBER(38,1)

Measured or estimated distance (meters) between footrope and headrope of the trawl.

NET\_MEASURED

Net measured during haul

logical

BINARY\_DOUBLE

Logical, describing if the net was measured (TRUE) or not (FALSE) by wheelhouse and marport programs during the haul.

NET\_WIDTH\_M

Net width (m)

meters

NUMBER(38,1)

Measured or estimated distance (meters) between wingtips of the trawl.

PERFORMANCE

Haul performance code

category

NUMBER(38,0)

This denotes what, if any, issues arose during the haul. For more information, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

STATION

Station ID

ID key code

VARCHAR2(255 BYTE)

Alpha-numeric designation for the station established in the design of a survey.

STRATUM

Stratum ID

ID key code

NUMBER(10,0)

RACE database statistical area for analyzing data. Strata were designed using bathymetry and other geographic and habitat-related elements. The strata are unique to each survey region. Stratum of value 0 indicates experimental tows.

SURFACE\_TEMPERATURE\_C

Surface temperature (degrees Celsius)

degrees Celsius

NUMBER(38,1)

Surface temperature (tenths of a degree Celsius); NA indicates removed or missing values.

WIRE\_LENGTH\_M

Trawl wire length

meters

NUMBER(38,0)

Length of wire deployed during a given haul in meters.

### 7.1.8 AKFIN\_LENGTH

Length data subsetted for standard GAP bottom trawl stations (i.e., HAULJOIN values associated with ABUNDANCE\_HAUL = Y in RACEBASE.HAUL).

Number of rows: 4,456,403

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

FREQUENCY

Count of observation

count

NUMBER(38,0)

Frequency, or count, of an observation.

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

LENGTH\_MM

Length of a specimen

millimeters

NUMBER(10,0)

Length bin in millimeters. A length of -9 indicates cases where no lengths were collected within a stratum for a species/year, even though catch numbers were recorded.

LENGTH\_TYPE

Length type

ID key code

NUMBER(38,0)

How the taxon was measured (e.g., fork length, carapace width). For a complete list of length\_type ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SAMPLE\_TYPE

Sample type

ID key code

NUMBER(38,0)

Sampling information on how the taxon was sampled. For a complete list of length\_type ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

Sex of a specimen where “1” = “Male”, “2” = “Female”, “3” = Unsexed.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

### 7.1.9 AKFIN\_METADATA\_COLUMN

Mirror of GAP\_PRODUCTS.METADATA\_COLUMN.

Number of rows: 170

Number of columns: 5

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

METADATA\_COLNAME

Column name

text

VARCHAR2(4000 BYTE)

Name of the column in a table.

METADATA\_COLNAME\_DESC

Column description

text

VARCHAR2(4000 BYTE)

Description of the column.

METADATA\_COLNAME\_LONG

Column name spelled out

text

VARCHAR2(4000 BYTE)

Long name for the column.

METADATA\_DATATYPE

Oracle datatype code

text

VARCHAR2(4000 BYTE)

Oracle data type of data column.

METADATA\_UNITS

Units

category

VARCHAR2(4000 BYTE)

Units of the column.

### 7.1.10 AKFIN\_SIZECOMP

Mirror of GAP\_PRODUCTS.SIZECOMP.

Number of rows: 3,284,606

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

LENGTH\_MM

Length of a specimen

millimeters

NUMBER(10,0)

Length bin in millimeters. A length of -9 indicates cases where no lengths were collected within a stratum for a species/year, even though catch numbers were recorded.

POPULATION\_COUNT

Estimated population

numeric

NUMBER(38,0)

The estimated population caught in the survey for a species, group, or total for a given survey.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

Sex of a specimen where “1” = “Male”, “2” = “Female”, “3” = Unsexed.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 7.1.11 AKFIN\_SPECIMEN

Specimen data subsetted for standard GAP bottom trawl stations (i.e., HAULJOIN values associated with ABUNDANCE\_HAUL = Y in RACEBASE.HAUL).

Number of rows: 589,317

Number of columns: 12

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AGE

Taxon age bin (yrs)

integer

NUMBER(38,0)

Age bin of taxon. Age bin of a taxon in years estimated by the age comp estimate. Age -9 indicates unaged lengths for a particular sex because no otoliths were collected for that sex/length combination. Age -99 indicates a case where no lengths were collected within a stratum for a species/year even though catch numbers were recorded.

AGE\_DETERMINATION\_METHOD

Aging method

ID key code

NUMBER(10,0)

Numeric code corresponding to the method of age determination. For a complete list of age determination codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

GONAD\_G

Weight of gonads (g)

grams

NUMBER(38,1)

Weight of specimen gonads (grams).

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

LENGTH\_MM

Length of a specimen

millimeters

NUMBER(10,0)

Length bin in millimeters. A length of -9 indicates cases where no lengths were collected within a stratum for a species/year, even though catch numbers were recorded.

MATURITY

Specimen maturity code

ID key code

NUMBER(38,0)

The maturity code or the condition identified by the maturity code.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

Sex of a specimen where “1” = “Male”, “2” = “Female”, “3” = Unsexed.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SPECIMEN\_ID

Specimen unique ID

ID key code

NUMBER(38,0)

Each individual examined must have a number assigned to it that is unique within each haul (0001 to 9999), though specimen numbers may be repeated between hauls

SPECIMEN\_SAMPLE\_TYPE

Specimen sample type

ID key code

NUMBER(38,0)

The specimen sample type ID key code as defined in the RACE\_DATA.SPECIMEN\_SAMPLE\_TYPES table. For a complete list of Specimen sample type ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SPECIMEN\_SUBSAMPLE\_METHOD

Specimen subsample method

ID key code

NUMBER(38,0)

For a complete list of specimen subsample method ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

WEIGHT\_G

Specimen weight (g)

grams

NUMBER(38,1)

Weight of specimen (grams).

### 7.1.12 AKFIN\_STRATUM\_GROUPS

Mirror of GAP\_PRODUCTS.STRATUM\_GROUPS.

Number of rows: 768

Number of columns: 4

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_ID

Area ID

ID key code

NUMBER(38,0)

Area ID key code for each statistical area used to produce production estimates (e.g., biomass, population, age comps, length comps). Each area ID is unique within each survey.

DESIGN\_YEAR

Design year

year

NUMBER(10,0)

Year ID associated with a given value AREA\_ID. This field describes the changes in the survey design over time.

STRATUM

Stratum ID

ID key code

NUMBER(10,0)

RACE database statistical area for analyzing data. Strata were designed using bathymetry and other geographic and habitat-related elements. The strata are unique to each survey region. Stratum of value 0 indicates experimental tows.

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

### 7.1.13 AKFIN\_SURVEY\_DESIGN

snapshot table for snapshot GAP\_PRODUCTS.AKFIN\_SURVEY\_DESIGN

Number of rows: 87

Number of columns: 3

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

DESIGN\_YEAR

Design year

year

NUMBER(10,0)

Year ID associated with a given value AREA\_ID. This field describes the changes in the survey design over time.

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 7.1.14 AKFIN\_TAXONOMIC\_CLASSIFICATION

Mirror of GAP\_PRODUCTS.TAXONOMIC\_CLASSIFICATION, subsetting for SURVEY\_SPECIES = 1 includes taxonomic classification information accepted by the AFSC GAP bottom trawl survey.

Number of rows: 2,718

Number of columns: 19

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

CLASS\_TAXON

Class phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of class of a given species.

COMMON\_NAME

Taxon common name

text

VARCHAR2(255 BYTE)

The common name of the marine organism associated with the scientific\_name and species\_code columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

DATABASE

Database source

category

VARCHAR2(255 BYTE)

Taxonomic database source, either ITIS or WoRMS.

DATABASE\_ID

Species ID in database

ID key code

VARCHAR2(255 BYTE)

Species ID key code of a species in the taxonomic “DATABASE” source.

FAMILY\_TAXON

Family phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of family of a given species.

GENUS\_TAXON

Genus phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of genus of a given species.

ID\_RANK

Lowest taxonomic rank

text

VARCHAR2(255 BYTE)

Lowest taxonomic rank of a given species entry.

KINGDOM\_TAXON

Kingdom phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of kingdom of a given species.

ORDER\_TAXON

Order phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of order of a given species.

PHYLUM\_TAXON

Phylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of phylum of a given species.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SPECIES\_NAME

Scientific name of species

text

VARCHAR2(255 BYTE)

Scientific name of species.

SUBCLASS\_TAXON

Subclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subclass of a given species.

SUBFAMILY\_TAXON

Subfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subfamily of a given species.

SUBORDER\_TAXON

Suborder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of suborder of a given species.

SUBPHYLUM\_TAXON

Subphylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subphylum of a given species.

SUPERCLASS\_TAXON

Superclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superclass of a given species.

SUPERFAMILY\_TAXON

Superfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superfamily of a given species.

SUPERORDER\_TAXON

Superorder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superorder of a given species.

### 7.1.15 AKFIN\_TAXONOMIC\_GROUPS

Mirror of GAP\_PRODUCTS.TAXON\_GROUPS.

Number of rows: 2,758

Number of columns: 21

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

CLASS\_TAXON

Class phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of class of a given species.

COMMON\_NAME

Taxon common name

text

VARCHAR2(255 BYTE)

The common name of the marine organism associated with the scientific\_name and species\_code columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

DATABASE

Database source

category

VARCHAR2(255 BYTE)

Taxonomic database source, either ITIS or WoRMS.

DATABASE\_ID

Species ID in database

ID key code

VARCHAR2(255 BYTE)

Species ID key code of a species in the taxonomic “DATABASE” source.

FAMILY\_TAXON

Family phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of family of a given species.

GENUS\_TAXON

Genus phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of genus of a given species.

GROUP\_CODE

NA

NA

NA

NA

ID\_RANK

Lowest taxonomic rank

text

VARCHAR2(255 BYTE)

Lowest taxonomic rank of a given species entry.

INFRAORDER\_TAXON

Infraorder phylogenetic rank

category

VARCHAR2(255 BYTE)

Infraorder phylogenetic rank. Phylogenetic latin rank of infraorder of a given speices.

KINGDOM\_TAXON

Kingdom phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of kingdom of a given species.

ORDER\_TAXON

Order phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of order of a given species.

PHYLUM\_TAXON

Phylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of phylum of a given species.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SPECIES\_NAME

Scientific name of species

text

VARCHAR2(255 BYTE)

Scientific name of species.

SUBCLASS\_TAXON

Subclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subclass of a given species.

SUBFAMILY\_TAXON

Subfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subfamily of a given species.

SUBORDER\_TAXON

Suborder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of suborder of a given species.

SUBPHYLUM\_TAXON

Subphylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subphylum of a given species.

SUPERCLASS\_TAXON

Superclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superclass of a given species.

SUPERFAMILY\_TAXON

Superfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superfamily of a given species.

SUPERORDER\_TAXON

Superorder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superorder of a given species.

# 8. Access data via Oracle and R

## Access data via Oracle (AFSC only)

AFSC Oracle users can access the database via SQL developer to view and pull the production data directly from the GAP\_PRODUCTS Oracle schema. The user can also use SQL developer to view and pull the GAP Products data directly from the GAP\_PRODUCTS Oracle schema.

### 8.0.1 Connect to Oracle from R

Many users will want to access the data from Oracle using R. The user will need to install the RODBC R package and ask OFIS (IT) connect R to Oracle. Then, use the following code in R to establish a connection from R to Oracle:

Here, the user can establish the oracle connection by entering their username and password in the channel <- gapindex::oracle\_connect() function. Never save usernames or passwords in scripts that may be intentionally or unintentionally shared with others. If no username and password is entered in the function, pop-ups will appear on the screen asking for the username and password.

After you connect to VPN, you’ll be able to log into Oracle.

library(RODBC)  
channel <- gapindex::get\_connected()

## Data SQL Query Examples:

library(gapindex)  
library(RODBC)  
library(flextable)  
library(ggplot2)  
library(magrittr)  
library(dplyr)

### 8.0.2 Ex. Select all data from tables

You can download all of the tables locally using a variation of the code below. Once connected, pull and save the tables of interest into the R environment.

locations <- c(  
 "GAP\_PRODUCTS.AKFIN\_AGECOMP",   
 "GAP\_PRODUCTS.AKFIN\_AREA",   
 "GAP\_PRODUCTS.AKFIN\_BIOMASS",   
 "GAP\_PRODUCTS.AKFIN\_CATCH",   
 "GAP\_PRODUCTS.AKFIN\_CPUE",   
 "GAP\_PRODUCTS.AKFIN\_CRUISE",   
 "GAP\_PRODUCTS.AKFIN\_HAUL",   
 "GAP\_PRODUCTS.AKFIN\_LENGTH",   
 "GAP\_PRODUCTS.AKFIN\_METADATA\_COLUMN",   
 "GAP\_PRODUCTS.AKFIN\_SIZECOMP",   
 "GAP\_PRODUCTS.AKFIN\_SPECIMEN",   
 "GAP\_PRODUCTS.AKFIN\_STRATUM\_GROUPS",   
 "GAP\_PRODUCTS.AKFIN\_SURVEY\_DESIGN",   
 "GAP\_PRODUCTS.AKFIN\_TAXONOMIC\_CLASSIFICATION"  
)  
  
for (i in 1:length(locations)) {  
 print(locations[i])  
 a <- RODBC::sqlQuery(channel, paste0("SELECT \* FROM ", locations[i]))  
 write.csv(x = a, file = here::here("data", paste0(locations[i], ".csv")))  
}

library(odbc)  
library(RODBC)  
library(dbplyr)  
  
my\_spp\_codes <- c(  
 30010, # Sebastolobus sp. thornyhead unid.  
 30020, # Sebastolobus alascanus shortspine thornyhead  
 30025, # Sebastolobus macrochir broadfin thornyhead  
 30330, # Sebastes melanops black rockfish  
 30430, # Sebastes proriger redstripe rockfish  
 30470, # Sebastes ruberrimus yelloweye rockfish  
 30475, # Sebastes babcocki redbanded rockfish  
 30535, # Sebastes variegatus harlequin rockfish  
 30560, # Sebastes zacentrus sharpchin rockfish  
 30600, # Sebastes reedi yellowmouth rockfish  
 30030, # Sebastolobus altivelis longspine thornyhead  
 30040, # Sebastes sp. rockfish unid.  
 30100, # Sebastes brevispinis silvergray rockfish  
 30150, # NA dusky and dark rockfishes unid.  
 30152, # Sebastes variabilis dusky rockfish  
 30170, # Sebastes crameri darkblotched rockfish  
 30270) # Sebastes helvomaculatus rosethorn rockfish  
  
a <- dplyr::tbl(channel, dplyr::sql('gap\_products.akfin\_biomass')) %>%   
 dplyr::rename\_all(tolower) %>%   
 dplyr::select(survey\_definition\_id, area\_id, species\_code, year, biomass\_mt, biomass\_var) %>%   
 dplyr::filter(species\_code %in% my\_spp\_codes &   
 area\_id %in% 99904 &   
 year >= 1991) %>%   
 dplyr::collect()   
  
flextable::flextable(head(a)) %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra()

### 8.0.3 Ex. CPUE for all EBS and NBS stations with associated haul, cruise, and species information.

a <- RODBC::sqlQuery(channel = channel, # NOT RACEBASE.HAUL  
 query = paste0(  
 "  
-- Select columns for output data  
SELECT  
cr.CRUISEJOIN,  
cr.CRUISE,  
cr.YEAR,  
cr.SURVEY\_DEFINITION\_ID,  
cr.SURVEY\_NAME,  
cr.VESSEL\_ID,  
cr.VESSEL\_NAME,  
cp.HAULJOIN,  
cp.SPECIES\_CODE,  
tt.SPECIES\_NAME,  
tt.COMMON\_NAME,  
cp.WEIGHT\_KG,  
cp.COUNT,  
cp.AREA\_SWEPT\_KM2,  
cp.CPUE\_KGKM2,  
cp.CPUE\_NOKM2,  
hh.HAUL,  
hh.STATION  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_HAUL hh  
LEFT JOIN GAP\_PRODUCTS.AKFIN\_CRUISE cr  
ON hh.CRUISEJOIN = cr.CRUISEJOIN  
LEFT JOIN GAP\_PRODUCTS.AKFIN\_CPUE cp  
ON hh.HAULJOIN = cp.HAULJOIN  
LEFT JOIN GAP\_PRODUCTS.TAXONOMIC\_CLASSIFICATION tt  
ON cp.SPECIES\_CODE = tt.SPECIES\_CODE  
  
-- Filter for EBS and NBS observations  
WHERE SURVEY\_DEFINITION\_ID IN (143, 98) -- 143 NBS, 98 EBS  
AND tt.SURVEY\_SPECIES = 1  
  
-- Only return the first 3 rows because otherwise this would be a huge table!  
FETCH FIRST 3 ROWS ONLY;"))   
  
flextable::flextable(head(a[,2:8])) %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra()

| **CRUISE** | **YEAR** | **SURVEY\_DEFINITION\_ID** | **SURVEY\_NAME** | **VESSEL\_ID** | **VESSEL\_NAME** | **HAULJOIN** |
| --- | --- | --- | --- | --- | --- | --- |
| 198,203 | 1,982 | 98 | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 1 | CHAPMAN | 877 |
| 198,203 | 1,982 | 98 | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 1 | CHAPMAN | 878 |
| 198,203 | 1,982 | 98 | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 1 | CHAPMAN | 879 |

### 8.0.4 Ex. CPUE for all stations contained in the INPFC Shumagin region (AREA\_ID = 919) for Pacific cod.

dat <- RODBC::sqlQuery(channel = channel,  
 query =  
 "  
-- Select columns for output data  
SELECT   
HAULJOIN,   
SPECIES\_CODE,   
STRATUM,   
LATITUDE\_DD\_START,   
LONGITUDE\_DD\_START,  
CPUE\_KGKM2,   
GEAR\_TEMPERATURE\_C  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_CPUE cpue  
LEFT JOIN GAP\_PRODUCTS.AKFIN\_HAUL haul  
USING (HAULJOIN)   
  
-- Filter for P. Cod observations  
WHERE SPECIES\_CODE IN (21720)  
  
-- Select all stratum within the area\_id 919 (INPFC Shumagin region)  
AND haul.STRATUM IN  
(  
SELECT   
STRATUM  
FROM GAP\_PRODUCTS.AKFIN\_STRATUM\_GROUPS   
WHERE AREA\_ID = 919  
);")

dat <- dat %>%   
 dplyr::select(HAULJOIN, STRATUM, SPECIES\_CODE, LATITUDE\_DD\_START, LONGITUDE\_DD\_START, CPUE\_KGKM2, GEAR\_TEMPERATURE\_C) %>%   
 dplyr::mutate(SPECIES\_CODE = as.character(SPECIES\_CODE),   
 STRATUM = as.character(STRATUM)) %>%   
 dplyr::arrange(SPECIES\_CODE)  
  
flextable::flextable(head(dat)) %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra()

| **HAULJOIN** | **STRATUM** | **SPECIES\_CODE** | **LATITUDE\_DD\_START** | **LONGITUDE\_DD\_START** | **CPUE\_KGKM2** | **GEAR\_TEMPERATURE\_C** |
| --- | --- | --- | --- | --- | --- | --- |
| -12,880 | 210 | 21720 | 52.55793 | -169.7829 | 6,863.3672 |  |
| -12,881 | 10 | 21720 | 52.63840 | -169.7815 | 1,536.8594 | 4.9 |
| -12,882 | 111 | 21720 | 52.67131 | -169.4279 | 10,044.8409 | 4.7 |
| -12,883 | 10 | 21720 | 53.24099 | -168.0725 | 1,937.7294 | 5.2 |
| -12,884 | 10 | 21720 | 53.16771 | -167.9810 | 830.2039 | 5.1 |
| -12,885 | 111 | 21720 | 53.06838 | -167.6713 | 2,891.8092 | 4.9 |

### 8.0.5 Ex. EBS Pacific Ocean perch CPUE and [akgfmaps](https://github.com/afsc-gap-products/akgfmaps) map

Pacific Ocean perch catch-per-unit-effort estimates for EBS in 2021 from GAP\_PRODUCTS.AKFIN\_CPUE and map constructed using [akgfmaps](https://github.com/afsc-gap-products/akgfmaps). Here, we’ll use AKFIN HAUL and CRUISES data also included in this repo, for convenience, though they are very similar to their RACEBASE analogs.

dat <- RODBC::sqlQuery(channel = channel,   
 query =   
 "  
-- Select columns for output data  
SELECT   
(cp.CPUE\_KGKM2/100) CPUE\_KGHA, -- akgfmaps is expecting hectares  
hh.LATITUDE\_DD\_START LATITUDE,  
hh.LONGITUDE\_DD\_START LONGITUDE  
  
-- Use HAUL data to obtain LATITUDE & LONGITUDE and connect to cruisejoin  
FROM GAP\_PRODUCTS.AKFIN\_CPUE cp  
LEFT JOIN GAP\_PRODUCTS.AKFIN\_HAUL hh  
ON cp.HAULJOIN = hh.HAULJOIN  
  
-- Use CRUISES data to obtain YEAR and SURVEY\_DEFINITION\_ID  
LEFT JOIN GAP\_PRODUCTS.AKFIN\_CRUISE cc  
ON hh.CRUISEJOIN = cc.CRUISEJOIN  
  
-- Filter data  
WHERE cp.SPECIES\_CODE = 30060   
AND cc.SURVEY\_DEFINITION\_ID = 98   
AND cc.YEAR = 2021;")

dat %>%   
 dplyr::arrange(desc(CPUE\_KGHA)) %>%   
 head() %>%   
 flextable::flextable() %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra()

| **CPUE\_KGHA** | **LATITUDE** | **LONGITUDE** |
| --- | --- | --- |
| 10.1768965 | 57.64871 | -173.3735 |
| 6.2734470 | 56.36952 | -169.4604 |
| 3.0252034 | 56.66253 | -171.9549 |
| 1.8214628 | 57.98912 | -173.4816 |
| 0.5535672 | 55.65865 | -168.1804 |
| 0.2813533 | 57.32545 | -173.3217 |

# devtools::install\_github("afsc-gap-products/akgfmaps", build\_vignettes = TRUE)  
library(akgfmaps)  
  
figure <- akgfmaps::make\_idw\_map(  
 x = dat, # Pass data as a data frame  
 region = "bs.south", # Predefined EBS area  
 set.breaks = "jenks", # Gets Jenks breaks from classint::classIntervals()  
 in.crs = "+proj=longlat", # Set input coordinate reference system  
 out.crs = "EPSG:3338", # Set output coordinate reference system  
 grid.cell = c(20000, 20000), # 20x20km grid  
 key.title = "Pacific Ocean perch") # Include in the legend title

[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]

figure$plot +   
 ggplot2::guides(fill=guide\_legend(title = "Pacific Ocean perch\nCPUE (kg/km2)")) |>   
 change\_fill\_color(new.scheme = "grey", show.plot = FALSE)

|  |
| --- |
| EBS Pacific Ocean perch CPUE and [akgfmaps](https://github.com/afsc-gap-products/akgfmaps) map. |

### 8.0.6 Ex. GOA Pacific Ocean perch biomass and abundance

Biomass and abundance for Pacific Ocean perch from 1990 – 2023 for the western/central/eastern GOA management areas as well as for the entire region.

dat <- RODBC::sqlQuery(channel = channel,   
 query =   
 "  
-- Manipulate data to join to  
WITH FILTERED\_STRATA AS (  
SELECT AREA\_ID, DESCRIPTION FROM GAP\_PRODUCTS.AKFIN\_AREA  
WHERE AREA\_TYPE in ('REGULATORY\_AREA', 'REGION')   
AND SURVEY\_DEFINITION\_ID = 47)  
  
-- Select columns for output data  
SELECT   
BIOMASS\_MT,  
POPULATION\_COUNT,   
YEAR,   
DESCRIPTION  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_BIOMASS BIOMASS  
JOIN FILTERED\_STRATA STRATA   
ON STRATA.AREA\_ID = BIOMASS.AREA\_ID  
  
-- Filter data results  
WHERE BIOMASS.SPECIES\_CODE = 30060")

dat0 <- dat %>%   
 janitor::clean\_names() %>%   
 dplyr::select(biomass\_mt, population\_count, year, area = description) %>%  
 pivot\_longer(cols = c("biomass\_mt", "population\_count"),   
 names\_to = "var",   
 values\_to = "val") %>%   
 dplyr::mutate(  
 val = ifelse(var == "biomass\_mt", val/1e6, val/1e9),   
 var = ifelse(var == "biomass\_mt", "Biomass (Mmt)", "Population (B)"),   
 area = gsub(x = area, pattern = " - ", replacement = "\n"),   
 area = gsub(x = area, pattern = ": ", replacement = "\n"),   
 type = sapply(X = strsplit(x = area, split = "\n", fixed = TRUE), `[[`, 2)) %>%   
 dplyr::arrange(type) %>%   
 dplyr::mutate(  
 area = factor(area, levels = unique(area), labels = unique(area), ordered = TRUE))  
  
flextable::flextable(head(dat)) %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra() %>%  
 flextable::colformat\_num(x = ., j = "YEAR", big.mark = "")

| **BIOMASS\_MT** | **POPULATION\_COUNT** | **YEAR** | **DESCRIPTION** |
| --- | --- | --- | --- |
| 157,295.1 | 317,129,408 | 1990 | GOA Region: All Strata |
| 157,295.1 | 317,129,408 | 1990 | GOA Region: All Strata |
| 483,622.6 | 833,902,161 | 1993 | GOA Region: All Strata |
| 483,622.6 | 833,902,161 | 1993 | GOA Region: All Strata |
| 771,412.8 | 1,252,616,603 | 1996 | GOA Region: All Strata |
| 771,412.8 | 1,252,616,603 | 1996 | GOA Region: All Strata |

# install.packages("scales")  
library(scales)  
figure <- ggplot2::ggplot(  
 dat = dat0,   
 mapping = aes(x = year, y = val, color = type)) +  
 ggplot2::geom\_point(size = 3) +   
 ggplot2::facet\_grid(cols = vars(area), rows = vars(var), scales = "free\_y") +   
 ggplot2::scale\_x\_continuous(name = "Year", n.breaks = 3) +  
 ggplot2::scale\_y\_continuous(name = "Estimate", labels = comma) +  
 ggplot2::labs(title = 'GOA Pacific Ocean perch biomass and abundance 1990 – 2023') +   
 ggplot2::guides(color=guide\_legend(title = "Region Type"))+  
 ggplot2::scale\_color\_grey() +  
 ggplot2::theme\_bw() +  
 ggplot2::theme(legend.direction = "horizontal",   
 legend.position = "bottom")  
  
figure

|  |
| --- |
| GOA Pacific Ocean perch biomass and abundance. |

### 8.0.7 Ex. AI rock sole size compositions and ridge plot

Northern and Southern rock sole size composition data from 1991 – 2022 for the Aleutian Islands, with Ridge plot from [ggridges](https://cran.r-project.org/web/packages/ggridges/vignettes/introduction.html).

dat <- RODBC::sqlQuery(channel = channel,   
 query = "  
SELECT   
YEAR,  
LENGTH\_MM / 10 AS LENGTH\_CM,   
SUM(POPULATION\_COUNT) AS POPULATION\_COUNT  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_SIZECOMP   
  
-- 99904 is the AREA\_ID that codes for the whole AI survey region  
WHERE AREA\_ID = 99904  
-- including northern rock sole, southern rock sole, and rock sole unid.  
AND SPECIES\_CODE IN (10260, 10261, 10262)  
-- remove the -9 LENGTH\_MM code  
AND LENGTH\_MM > 0  
-- sum over species\_codes and sexes  
GROUP BY (YEAR, LENGTH\_MM)")

dat0 <- dat %>%   
 janitor::clean\_names() %>%   
 head() %>%   
 flextable::flextable() %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra() %>%  
 flextable::colformat\_num(x = ., j = "year", big.mark = "")  
dat0

| **year** | **length\_cm** | **population\_count** |
| --- | --- | --- |
| 1991 | 23 | 4,625,236 |
| 1991 | 38 | 2,254,964 |
| 1991 | 42 | 820,614 |
| 1991 | 52 | 11,225 |
| 1994 | 16 | 741,246 |
| 1994 | 26 | 9,762,322 |

# install.packages("ggridges")  
library(ggridges)  
figure <- ggplot(dat,   
 mapping = aes(x = LENGTH\_CM,   
 y = YEAR,   
 height = POPULATION\_COUNT,   
 group = YEAR)) +  
 ggridges::geom\_density\_ridges(stat = "identity", scale = 1) +  
 ggplot2::ylab(label = "Year") +  
 ggplot2::scale\_x\_continuous(name = "Length (cm)") +  
 ggplot2::labs(title = paste0('Aleutian Islands Rock sole Size Compositions'),   
 subtitle = paste0(min(dat$YEAR), ' – ', max(dat$YEAR))) +  
 ggplot2::theme\_bw()  
  
figure

|  |
| --- |
| AI Rock sole size compositions and ridge plot. |

### 8.0.8 Ex. 2023 EBS Walleye Pollock Age Compositions and Age Pyramid

Walleye pollock age composition for the EBS standard + NW Area from 2023, with age pyramid plot.

dat <- RODBC::sqlQuery(channel = channel,   
 query = "  
-- Manipulate data to join to  
WITH FILTERED\_STRATA AS (  
SELECT   
AREA\_ID,   
DESCRIPTION   
FROM GAP\_PRODUCTS.AKFIN\_AREA  
WHERE AREA\_TYPE = 'REGION' AND   
SURVEY\_DEFINITION\_ID = 98)  
  
-- Select columns for output data  
SELECT   
AGECOMP.AGE,   
AGECOMP.POPULATION\_COUNT,   
AGECOMP.SEX  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_AGECOMP AGECOMP  
JOIN FILTERED\_STRATA STRATA   
ON STRATA.AREA\_ID = AGECOMP.AREA\_ID  
  
-- Filter data results  
WHERE SPECIES\_CODE = 21740  
AND YEAR = 2023  
AND AGE >= 0")

dat0 <- dat %>%   
 janitor::clean\_names() %>%   
 dplyr::filter(sex %in% c(1,2)) %>%  
 dplyr::mutate(  
 sex = ifelse(sex == 1, "M", "F"),  
 population\_count = # change male population to negative  
 ifelse(sex=="M", population\_count\*(-1), population\_count\*1)/1e9)   
  
flextable::flextable(head(dat)) %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra()

| **AGE** | **POPULATION\_COUNT** | **SEX** |
| --- | --- | --- |
| 1 | 19,777,357 | 1 |
| 2 | 82,284,457 | 1 |
| 3 | 136,609,551 | 1 |
| 4 | 237,196,811 | 1 |
| 5 | 939,749,004 | 1 |
| 6 | 244,651,171 | 1 |

figure <- ggplot2::ggplot(  
 data = dat0,   
 mapping =   
 aes(x = age,  
 y = population\_count,   
 fill = sex)) +  
 ggplot2::scale\_fill\_grey() +  
 ggplot2::geom\_bar(stat = "identity") +  
 ggplot2::coord\_flip() +  
 ggplot2::scale\_x\_continuous(name = "Age") +  
 ggplot2::scale\_y\_continuous(name = "Population (billions)", labels = abs) +  
 ggplot2::ggtitle(label = "2023 EBS Walleye Pollock Age Compositions") +   
 ggplot2::guides(fill = guide\_legend(title = "Sex"))+  
 ggplot2::theme\_bw()  
  
figure

|  |
| --- |
| 2023 EBS Walleye Pollock Age Compositions and Age Pyramid. |

### 8.0.9 Ex. NBS Pacific cod biomass and abundance

Pacific cod biomass and abundance data for the NBS by stratum.

dat <- RODBC::sqlQuery(channel = channel,   
 query =   
 "  
-- Manipulate data to join to  
WITH FILTERED\_STRATA AS (  
SELECT   
AREA\_ID,   
AREA\_NAME,   
DESCRIPTION   
FROM GAP\_PRODUCTS.AKFIN\_AREA  
WHERE AREA\_TYPE in ('STRATUM') AND   
SURVEY\_DEFINITION\_ID = 143)   
  
-- Select columns for output data  
SELECT   
BIOMASS.BIOMASS\_MT,   
BIOMASS.POPULATION\_COUNT,   
BIOMASS.YEAR,   
STRATA.AREA\_NAME  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_BIOMASS BIOMASS   
JOIN FILTERED\_STRATA STRATA   
ON STRATA.AREA\_ID = BIOMASS.AREA\_ID  
  
-- Filter data results  
WHERE BIOMASS.SURVEY\_DEFINITION\_ID IN 143   
AND BIOMASS.SPECIES\_CODE = 21720")

dat0 <- dat %>%   
 janitor::clean\_names() %>%   
 dplyr::select(biomass\_mt, population\_count, year, area = area\_name) %>%  
 pivot\_longer(cols = c("biomass\_mt", "population\_count"),   
 names\_to = "var",   
 values\_to = "val") %>%   
 dplyr::mutate(  
 val = ifelse(var == "biomass\_mt", val/1e6, val/1e9),   
 var = ifelse(var == "biomass\_mt", "Biomass (Mmt)", "Population (B)"),   
 area = factor(area, levels = unique(area), labels = unique(area), ordered = TRUE))  
flextable::flextable(head(dat)) %>%   
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra() %>%  
 flextable::colformat\_num(x = ., j = "YEAR", big.mark = "")

| **BIOMASS\_MT** | **POPULATION\_COUNT** | **YEAR** | **AREA\_NAME** |
| --- | --- | --- | --- |
| 7,462.559 | 4,724,153 | 2010 | Inner Domain |
| 7,462.559 | 4,724,153 | 2010 | Inner Domain |
| 7,462.559 | 4,724,153 | 2010 | Inner Domain |
| 7,462.559 | 4,724,153 | 2010 | Inner Domain |
| 7,462.559 | 4,724,153 | 2010 | Inner Domain |
| 20,983.376 | 3,928,600 | 2010 | Inner Domain |

figure <- ggplot2::ggplot(  
 dat = dat0,   
 mapping = aes(y = val, x = year, fill = area)) +   
 ggplot2::geom\_bar(position="stack", stat="identity") +   
 ggplot2::facet\_grid(rows = vars(var), scales = "free\_y") +  
 ggplot2::scale\_y\_continuous(name = "Estimate", labels = comma) +  
 ggplot2::scale\_x\_continuous(name = "Year", breaks = unique(dat0$year)) +  
 ggplot2::labs(title = 'NBS Pacific cod biomass and abundance by stratum') +   
 ggplot2::guides(fill=guide\_legend(title = "Region Type"))+  
 ggplot2::scale\_fill\_grey() +  
 ggplot2::theme\_bw() +  
 ggplot2::theme(legend.direction = "horizontal",   
 legend.position = "bottom")  
  
figure

|  |
| --- |
| NBS Pacific cod biomass and abundance. |

### 8.0.10 Ex. GOA Pacific Ocean perch biomass and line plot

Pacific Ocean perch biomass totals for GOA between 1984-2021 from GAP\_PRODUCTS.AKFIN\_BIOMASS

dat <- RODBC::sqlQuery(channel = channel,  
 query = "  
-- Select columns for output data  
SELECT  
SURVEY\_DEFINITION\_ID,  
BIOMASS\_MT / 1000 AS BIOMASS\_KMT,  
(BIOMASS\_MT - 2 \* SQRT(BIOMASS\_VAR)) / 1000 AS BIOMASS\_KCI\_DW,  
(BIOMASS\_MT + 2 \* SQRT(BIOMASS\_VAR)) / 1000 AS BIOMASS\_KCI\_UP,  
YEAR  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_BIOMASS  
  
-- Filter data results  
WHERE SPECIES\_CODE = 30060  
AND SURVEY\_DEFINITION\_ID = 47  
AND AREA\_ID = 99903  
AND YEAR BETWEEN 1990 AND 2023" ) %>%   
 janitor::clean\_names()

flextable::flextable(head(dat)) %>%  
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra() %>%  
 flextable::colformat\_num(x = ., j = "year", big.mark = "")

| **survey\_definition\_id** | **biomass\_kmt** | **biomass\_kci\_dw** | **biomass\_kci\_up** | **year** |
| --- | --- | --- | --- | --- |
| 47 | 157.2951 | 63.03638 | 251.5538 | 1990 |
| 47 | 483.6226 | 266.33581 | 700.9093 | 1993 |
| 47 | 771.4128 | 364.30515 | 1,178.5204 | 1996 |
| 47 | 727.0635 | -50.06854 | 1,504.1955 | 1999 |
| 47 | 673.1551 | 229.14901 | 1,117.1611 | 2001 |
| 47 | 457.4216 | 313.39204 | 601.4511 | 2003 |

a\_mean <- dat %>%   
 dplyr::group\_by(survey\_definition\_id) %>%   
 dplyr::summarise(biomass\_kmt = mean(biomass\_kmt, na.rm = TRUE),   
 minyr = min(year, na.rm = TRUE),   
 maxyr = max(year, na.rm = TRUE))   
  
figure <-  
 ggplot(data = dat,   
 mapping = aes(x = year,   
 y = biomass\_kmt)) +  
 ggplot2::geom\_point(size = 2.5, color = "grey40") +   
 ggplot2::scale\_x\_continuous(  
 name = "Year",   
 labels = scales::label\_number(  
 accuracy = 1,   
 big.mark = "")) +  
 ggplot2::scale\_y\_continuous(  
 name = "Biomass (Kmt)",   
 labels = comma) +  
 ggplot2::geom\_segment(  
 data = a\_mean,  
 mapping = aes(x = minyr,   
 xend = maxyr,   
 y = biomass\_kmt,   
 yend = biomass\_kmt),  
 linetype = "dashed",   
 linewidth = 2) +  
 ggplot2::geom\_errorbar(  
 mapping = aes(ymin = biomass\_kci\_dw, ymax = biomass\_kci\_up),  
 position = position\_dodge(.9),  
 alpha = 0.5, width=.2) +  
 ggplot2::ggtitle(  
 label = "GOA Pacific Ocean Perch Biomass 1984-2021",   
 subtitle = paste0("Mean = ",   
 formatC(x = a\_mean$biomass\_kmt,   
 digits = 2,   
 big.mark = ",",   
 format = "f"),   
 " Kmt")) +  
 ggplot2::theme\_bw()  
  
figure

|  |
| --- |
| GOA Pacific Ocean perch biomass and line plot. |

### 8.0.11 Ex. 2022 AI Atka mackerel age specimen summary

#### 8.0.11.1 All ages determined:

dat <- RODBC::sqlQuery(channel = channel,  
 query = "  
-- Select columns for output data  
SELECT SURVEY\_DEFINITION\_ID, YEAR, SPECIES\_CODE, AGE  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_SPECIMEN  
JOIN (SELECT HAULJOIN, CRUISEJOIN FROM GAP\_PRODUCTS.AKFIN\_HAUL)  
USING (HAULJOIN)  
JOIN (SELECT CRUISEJOIN, YEAR, SURVEY\_DEFINITION\_ID FROM GAP\_PRODUCTS.AKFIN\_CRUISE)  
USING (CRUISEJOIN)  
  
-- Filter data results  
WHERE GAP\_PRODUCTS.AKFIN\_SPECIMEN.SPECIMEN\_SAMPLE\_TYPE = 1  
AND SPECIES\_CODE = 21921  
AND YEAR = 2022  
AND SURVEY\_DEFINITION\_ID = 52") %>%   
 janitor::clean\_names()

flextable::flextable(head(dat) %>%   
 dplyr::arrange(age)) %>%  
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra() %>%  
 flextable::colformat\_num(x = ., j = c("year", "species\_code"), big.mark = "")

| **survey\_definition\_id** | **year** | **species\_code** | **age** |
| --- | --- | --- | --- |
| 52 | 2022 | 21921 | 3 |
| 52 | 2022 | 21921 | 3 |
| 52 | 2022 | 21921 | 4 |
| 52 | 2022 | 21921 | 4 |
| 52 | 2022 | 21921 | 4 |
| 52 | 2022 | 21921 | 7 |

#### 8.0.11.2 How many of each age was found:

dat <- RODBC::sqlQuery(channel = channel,  
 query = "  
-- Select columns for output data  
SELECT SURVEY\_DEFINITION\_ID, YEAR, SPECIES\_CODE, AGE,   
COUNT(AGE) AS COUNTAGE  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_SPECIMEN  
JOIN (SELECT HAULJOIN, CRUISEJOIN FROM GAP\_PRODUCTS.AKFIN\_HAUL)  
USING (HAULJOIN)  
JOIN (SELECT CRUISEJOIN, YEAR, SURVEY\_DEFINITION\_ID FROM GAP\_PRODUCTS.AKFIN\_CRUISE)  
USING (CRUISEJOIN)  
  
-- Filter data results  
WHERE GAP\_PRODUCTS.AKFIN\_SPECIMEN.SPECIMEN\_SAMPLE\_TYPE = 1  
AND SPECIES\_CODE = 21921  
AND YEAR = 2022  
AND SURVEY\_DEFINITION\_ID = 52  
GROUP BY (YEAR, SURVEY\_DEFINITION\_ID, SPECIES\_CODE, AGE)") %>%   
 janitor::clean\_names()

flextable::flextable(head(dat) %>%   
 dplyr::arrange(age)) %>%  
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra() %>%  
 flextable::colformat\_num(x = ., j = c("year", "species\_code"), big.mark = "")

| **survey\_definition\_id** | **year** | **species\_code** | **age** | **countage** |
| --- | --- | --- | --- | --- |
| 52 | 2022 | 21921 | 1 | 1 |
| 52 | 2022 | 21921 | 2 | 40 |
| 52 | 2022 | 21921 | 6 | 116 |
| 52 | 2022 | 21921 | 7 | 108 |
| 52 | 2022 | 21921 | 8 | 61 |
| 52 | 2022 | 21921 | 11 | 20 |

#### 8.0.11.3 How many otoliths were aged:

Using SQL

dat <- RODBC::sqlQuery(channel = channel,  
 query = "  
-- Select columns for output data  
SELECT SURVEY\_DEFINITION\_ID, YEAR, SPECIES\_CODE,   
COUNT(AGE) AS COUNTAGE  
  
-- Identify what tables to pull data from  
FROM GAP\_PRODUCTS.AKFIN\_SPECIMEN  
JOIN (SELECT HAULJOIN, CRUISEJOIN FROM GAP\_PRODUCTS.AKFIN\_HAUL)  
USING (HAULJOIN)  
JOIN (SELECT CRUISEJOIN, YEAR, SURVEY\_DEFINITION\_ID FROM GAP\_PRODUCTS.AKFIN\_CRUISE)  
USING (CRUISEJOIN)  
  
-- Filter data results  
WHERE GAP\_PRODUCTS.AKFIN\_SPECIMEN.SPECIMEN\_SAMPLE\_TYPE = 1  
AND SPECIES\_CODE = 21921  
AND YEAR = 2022  
AND SURVEY\_DEFINITION\_ID = 52  
GROUP BY (YEAR, SURVEY\_DEFINITION\_ID, SPECIES\_CODE)") %>%   
 janitor::clean\_names()

Using dbplyr:

library(odbc)  
library(keyring)  
library(dplyr)  
library(dbplyr)  
  
channel <- DBI::dbConnect(odbc::odbc(), "akfin", uid = keyring::key\_list("akfin")$username,  
 pwd = keyring::key\_get("akfin", keyring::key\_list("akfin")$username))  
  
dat <- dplyr::tbl(src = channel, dplyr::sql('gap\_products.akfin\_specimen')) %>%   
 dplyr::rename\_all(tolower) %>%   
 dplyr::select(hauljoin, specimen = specimen\_id, species\_code, length = length\_mm,   
 weight = weight\_g, age, sex, age\_method = age\_determination\_method) %>%   
 dplyr::left\_join(dplyr::tbl(akfin, dplyr::sql('gap\_products.akfin\_haul')) %>%  
 dplyr::rename\_all(tolower) %>%   
 dplyr::select(cruisejoin, hauljoin, haul, date\_collected = date\_time\_start,   
 latitude = latitude\_dd\_start, longitude = longitude\_dd\_start),  
 by = join\_by(hauljoin)) %>%   
 dplyr::left\_join(dplyr::tbl(akfin, dplyr::sql('gap\_products.akfin\_cruise')) %>%  
 dplyr::rename\_all(tolower) %>%   
 dplyr::select(cruisejoin, year, vessel = vessel\_id, survey\_definition\_id),  
 by = join\_by(cruisejoin)) %>%   
 dplyr::filter(year == YEAR &  
 survey\_definition\_id == 52 &   
 species\_code %in% spp\_codes &  
 !is.na(age)) %>%   
 dplyr::collect()

Both scripts will produce this table:

flextable::flextable(head(dat)) %>%  
 flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra() %>%  
 flextable::colformat\_num(x = ., j = c("year", "species\_code"), big.mark = "")

| **survey\_definition\_id** | **year** | **species\_code** | **countage** |
| --- | --- | --- | --- |
| 52 | 2022 | 21921 | 1,061 |

# 9. Access API data using R

AKFIN has developed web services (apis) to distribute GAP data. Like the GAP\_PRODUCTS schema, these are under active development. These do not require VPN or an oracle connection but they are protected by Oracle authentication, please contact matt.callahan@noaa.gov for information on how to get an api token to use this option.

The url structure is “https://apex.psmfc.org/akfin/data\_marts/gap\_products/gap\_[base table name]” . For example “https://apex.psmfc.org/akfin/data\_marts/gap\_products/gap\_biomass” is the base url to get data from the akfin\_biomass table. Web services linked to large tables have mandatory parameters to reduce data download size. For example to get agecomp data for Bering Sea pollock in area\_id 10 in 2022 you would use “https://apex.psmfc.org/akfin/data\_marts/gap\_products/gap\_biomass?survey\_definition\_id=98&area\_id=10&species\_code=21740&start\_year=2022&end\_year=2022”.

If you’re using R to pull data through web services you might find the [akfingapdata](https://github.com/MattCallahan-NOAA/akfingapdata/tree/main) (pronounced akfin-gap-data not ak-eff-ing-app-data) R package helpful.

## 9.1 Ex. Direct database query in R using the (akfingapdata readme)[https://github.com/MattCallahan-NOAA/akfingapdata/blob/main/README.Rmd] R package:

## 9.2 Ex. Direct database query in R using the (akfingapdata readme)[https://github.com/MattCallahan-NOAA/akfingapdata/blob/main/README.Rmd] R package:

Sign into akfin with token (need to request token from AKFIN)

akfingapdata::get\_gap\_catch()[,1:6] %>%   
 head() %>%   
 flextable::flextable() %>%  
 flextable::theme\_zebra()

# 10. Data description

The Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC) conducts fisheries-independent bottom trawl surveys to monitor the condition of the demersal fish and crab stocks of Alaska. These data are developed to describe the temporal distribution and abundance of commercially and ecologically important groundfish species, examine the changes in the species composition of the fauna over time and space, and describe the physical environment of the groundfish habitat.

There are no legal restrictions on access to the data. They reside in the public domain and can be freely distributed. Users must read and fully comprehend the metadata prior to use. Data should not be used beyond the limits of the source scale. Acknowledgement of NOAA, as the source from which these data were obtained, in any publications and/or other representations of these data, is suggested. These data are compiled and approved annually after each summer survey season. The data from previous years are unlikely to change substantially once published.

These data are zero-filled (presence and absence) observations from surveys conducted on fishing vessels. These surveys monitor trends in distribution and abundance of groundfish, crab, and bottom-dwelling species in Alaska’s marine ecosystems. These data include estimates of catch-per-unit-effort (CPUE) for all identified species for index stations. Some survey data are excluded, such as non-standard stations, surveys completed in earlier years using different/non-standard gear, and special tows and non-standard data collections.

Though not included in the public data, these surveys also collect oceanographic and environmental data, and biological data such as length, weight, stomach contents (to learn more about diet), otoliths (fish ear bones to learn about age), and tissue samples for genetic analysis, all of which can be shared upon special request. Also not included in the public data are estimated biomass (average total weight of all fish and crabs sampled) of crabs and groundfish that support the creation of annual stock assessments.

## 10.1 Data tables

### 10.1.1 FOSS\_CATCH

Catch-per-unit-effort subsetted for standard GAP bottom trawl stations (i.e., HAULJOIN values associated with ABUNDANCE\_HAUL = Y in RACEBASE.HAUL) with added taxonomic confidence data.

Number of rows: 939,197

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

Total whole number of individuals caught in haul or samples collected.

CPUE\_KGKM2

Weight CPUE (kg/km2)

kilograms per kilometers squared

NUMBER(38,6)

Catch weight (kilograms) per unit effort (area swept by the net, units square kilometers).

CPUE\_NOKM2

Number CPUE (no/km2)

count per kilometers squared

NUMBER(38,6)

Numerical catch per unit effort (area swept by the net, units square kilometers).

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

TAXON\_CONFIDENCE

Taxon confidence rating

category

VARCHAR2(255 BYTE)

Confidence in the ability of the survey team to correctly identify the taxon to the specified level, based solely on identification skill (e.g., not likelihood of a taxon being caught at that station on a location-by-location basis). Quality codes follow: **High**: High confidence and consistency. Taxonomy is stable and reliable at this level, and field identification characteristics are well known and reliable. **Moderate**: Moderate confidence. Taxonomy may be questionable at this level, or field identification characteristics may be variable and difficult to assess consistently. **Low**: Low confidence. Taxonomy is incompletely known, or reliable field identification characteristics are unknown. Documentation: [Species identification confidence in the eastern Bering Sea shelf survey (1982-2008)](http://apps-afsc.fisheries.noaa.gov/Publications/ProcRpt/PR2009-04.pdf), [Species identification confidence in the eastern Bering Sea slope survey (1976-2010)](http://apps-afsc.fisheries.noaa.gov/Publications/ProcRpt/PR2014-05.pdf), and [Species identification confidence in the Gulf of Alaska and Aleutian Islands surveys (1980-2011)](http://apps-afsc.fisheries.noaa.gov/Publications/ProcRpt/PR2014-01.pdf).

WEIGHT\_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

Weight (thousandths of a kilogram) of individuals in a haul by taxon.

### 10.1.2 FOSS\_HAUL

Standard GAP bottom trawl haul (station) data.

Number of rows: 33,334

Number of columns: 27

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

AREA\_SWEPT\_KM2

Area swept (km)

kilometers

NUMBER(38,6)

The area the net covered while the net was fishing (kilometers squared), defined as the distance fished times the net width.

BOTTOM\_TEMPERATURE\_C

Bottom temperature (degrees Celsius)

degrees Celsius

NUMBER(38,1)

Bottom temperature (tenths of a degree Celsius); NA indicates removed or missing values.

CRUISE

Cruise Name

ID key code

NUMBER(38,0)

This is a six-digit integer identifying the cruise number of the form: YYYY99 (where YYYY = year of the cruise; 99 = 2-digit number and is sequential; 01 denotes the first cruise that vessel made in this year, 02 is the second, etc.).

CRUISEJOIN

Cruise ID

ID key code

NUMBER(38,0)

Unique integer ID assigned to each survey, vessel, and year combination.

DATE\_TIME

Date and time

MM/DD/YYYY HH::MM

DATE

The date (MM/DD/YYYY) and time (HH:MM) of the haul. All dates and times are in Alaska time (AKDT) of Anchorage, AK, USA (UTC/GMT -8 hours).

DEPTH\_M

Depth (m)

degrees Celsius

NUMBER(38,1)

Bottom depth (meters).

DISTANCE\_FISHED\_KM

Distance fished (km)

degrees Celsius

NUMBER(38,3)

Distance the net fished (thousands of kilometers).

DURATION\_HR

Tow duration (decimal hr)

hours

NUMBER(38,1)

This is the elapsed time between start and end of a haul (decimal hours).

HAUL

Haul number

ID key code

NUMBER(38,0)

This number uniquely identifies a sampling event (haul) within a cruise. It is a sequential number, in chronological order of occurrence.

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

This is a unique numeric identifier assigned to each (vessel, cruise, and haul) combination.

LATITUDE\_DD\_END

End latitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Latitude (one hundred thousandth of a decimal degree) of the end of the haul.

LATITUDE\_DD\_START

Start latitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Latitude (one hundred thousandth of a decimal degree) of the start of the haul.

LONGITUDE\_DD\_END

End longitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Longitude (one hundred thousandth of a decimal degree) of the end of the haul.

LONGITUDE\_DD\_START

Start longitude (decimal degrees)

decimal degrees

NUMBER(38,6)

Longitude (one hundred thousandth of a decimal degree) of the start of the haul.

NET\_HEIGHT\_M

Net height (m)

meters

NUMBER(38,1)

Measured or estimated distance (meters) between footrope and headrope of the trawl.

NET\_WIDTH\_M

Net width (m)

meters

NUMBER(38,1)

Measured or estimated distance (meters) between wingtips of the trawl.

PERFORMANCE

Haul performance code

category

NUMBER(38,0)

This denotes what, if any, issues arose during the haul. For more information, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SRVY

Survey abbreviation

text abbreviated

VARCHAR2(255 BYTE)

Abbreviated survey names. The column srvy is associated with the survey and survey\_definition\_id columns. Northern Bering Sea (NBS), Southeastern Bering Sea (EBS), Bering Sea Slope (BSS), Gulf of Alaska (GOA), Aleutian Islands (AI).

STATION

Station ID

ID key code

VARCHAR2(255 BYTE)

Alpha-numeric designation for the station established in the design of a survey.

STRATUM

Stratum ID

ID key code

NUMBER(10,0)

RACE database statistical area for analyzing data. Strata were designed using bathymetry and other geographic and habitat-related elements. The strata are unique to each survey region. Stratum of value 0 indicates experimental tows.

SURFACE\_TEMPERATURE\_C

Surface temperature (degrees Celsius)

degrees Celsius

NUMBER(38,1)

Surface temperature (tenths of a degree Celsius); NA indicates removed or missing values.

SURVEY

Survey name

text

VARCHAR2(255 BYTE)

Name and description of survey. The column survey is associated with the srvy and survey\_definition\_id columns.

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_NAME

Survey name official

text

VARCHAR2(255 BYTE)

Long name of the survey conducted

VESSEL\_ID

Vessel ID

ID key code

NUMBER(38,0)

ID number of the vessel used to collect data for that haul. The column vessel\_id is associated with the vessel\_name column. Note that it is possible for a vessel to have a new name but the same vessel id number. For a complete list of vessel ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

VESSEL\_NAME

Vessel name

text

VARCHAR2(255 BYTE)

Name of the vessel used to collect data for that haul. The column vessel\_name is associated with the vessel\_id column. Note that it is possible for a vessel to have a new name but the same vessel id number. For a complete list of vessel ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

### 10.1.3 FOSS\_SPECIES

These datasets, FOSS\_CATCH, FOSS\_CPUE\_PRESONLY, FOSS\_HAUL, and FOSS\_SPECIES, when full joined by the HAULJOIN variable, includes zero-filled (presence and absence) observations and catch-per-unit-effort (CPUE) estimates for all identified species at for index stations. These tables were created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). There are legal restrictions on access to the data. These data are not intended for public dissemination and should not be shared without the explicit written consent of the data managers and owners (NOAA Fisheries). The GitHub repository for the scripts that created this code can be found at https://github.com/afsc-gap-products/gap\_products. For more information about codes used in the tables, please refer to the survey code books (https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual). These data were last updated March 04, 2024.

Number of rows: 1,894

Number of columns: 6

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

COMMON\_NAME

Taxon common name

text

VARCHAR2(255 BYTE)

The common name of the marine organism associated with the scientific\_name and species\_code columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

ID\_RANK

Lowest taxonomic rank

text

VARCHAR2(255 BYTE)

Lowest taxonomic rank of a given species entry.

ITIS

Integrated taxonomic information system (ITIS) serial number

ID key code

NUMBER(38,0)

Species code as identified in the Integrated Taxonomic Information System (https://itis.gov/).

SCIENTIFIC\_NAME

Taxon scientific name

text

VARCHAR2(255 BYTE)

The scientific name of the organism associated with the common\_name and species\_code columns. For a complete taxon list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

WORMS

World register of marine species (WoRMS) taxonomic serial number

ID key code

NUMBER(38,0)

Species code as identified in the World Register of Marine Species (WoRMS) (https://www.marinespecies.org/).

### 10.1.4 FOSS\_SURVEY\_SPECIES

This reference dataset contains the full list of species by survey to be used to zero-fill FOSS\_CATCH and FOSS\_HAUL for each survey. These tables were created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). There are legal restrictions on access to the data. These data are not intended for public dissemination and should not be shared without the explicit written consent of the data managers and owners (NOAA Fisheries). The GitHub repository for the scripts that created this code can be found at https://github.com/afsc-gap-products/gap\_products. For more information about codes used in the tables, please refer to the survey code books (https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual). These data were last updated March 04, 2024.

Number of rows: 5,030

Number of columns: 2

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

SURVEY\_DEFINITION\_ID

Survey ID

ID key code

NUMBER(38,0)

The survey definition ID key code is an integer that uniquely identifies a survey region/survey design. The column survey\_definition\_id is associated with the srvy and survey columns. Full list of survey definition IDs are in RACE\_DATA.SURVEY\_DEFINITIONS and in the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

### 10.1.5 FOSS\_TAXON\_GROUP

This reference dataset contains suggested search groups for simplifying species selection in the FOSS data platform so users can better search through FOSS\_CATCH. These tables were created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). There are legal restrictions on access to the data. These data are not intended for public dissemination and should not be shared without the explicit written consent of the data managers and owners (NOAA Fisheries). The GitHub repository for the scripts that created this code can be found at https://github.com/afsc-gap-products/gap\_products. For more information about codes used in the tables, please refer to the survey code books (https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual). These data were last updated March 04, 2024.

Number of rows: 33,721

Number of columns: 3

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

CLASSIFICATION

Taxonomic classification rank group

category

VARCHAR2(255 BYTE)

Phylogenetic classification group rank for a given species.

RANK\_ID

Taxonomic rank

category

VARCHAR2(255 BYTE)

The taxonomic rank of a taxon identification.

SPECIES\_CODE

Taxon code

ID key code

NUMBER(38,0)

The species code of the organism associated with the common\_name and scientific\_name columns. For a complete species list, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

# 11. Using the FOSS platform

## 11.1 Select and filter

|  |
| --- |
| AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform. |

Select, filter, and download this and other NOAA Fisheries data from the [Fisheries One Stop Shop (FOSS)](https://www.fisheries.noaa.gov/foss) platform. A user guide for the FOSS platform can be found [here](https://www.fisheries.noaa.gov/foss/f?p=215:7:7542600605674:::::). To begin a report, select the kind of data you need: Haul and catch data, Haul data only, All observed species.

In this example, we’ll select for 2023 eastern Bering Sea Arctic cod data. Here, we used the Search Species box to search for species with the term “cod” in their common names and selected “Pacific cod” from that shortened list.

### 11.1.1 Catch and haul

|  |
| --- |
| Catch data on the AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform. |

### 11.1.2 Haul

|  |
| --- |
| Haul data on the AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform. |

### 11.1.3 Species

|  |
| --- |
| All species observed by survey on the AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform. |

## 11.2 Search options

The user must select a option in each of the three option boxes as they appear for catch, haul, and species:

* Survey: Each survey has different in design, time series, and history. More information on each survey and their designs can be found in our [annual data reports](https://www.fisheries.noaa.gov/alaska/science-data/groundfish-assessment-program-bottom-trawl-surveys#data-products).
* Year: Surveys are not conducted in all years, so only data from the years for which the survey was conducted will be returned.
* Species: Common name of all species ever encountered in the survey. Find more information about these species in our [survey code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

For a given box, select one or a few options from the options box (list on the left) to query. To select multiple options, hold down the CTRL key while clicking on the options of interest, or click and drag down the list. Once the options you wish to be included in your query are highlighted, click the right-pointing arrow (>) to move them into the “selection box” (list on the right). This can also be achieved by double clicking the option item of interest. If you accidentally select an option that you do not want to query, simply select the unwanted option from the selection box and click the left-pointing arrow (<).

If you wish to select all options from the options box and send them to the selection box, simply click the double right-pointing arrow (>>). If you want to unselect all options from the selection box, use the double left-pointing arrow (<<) or the reset icon.

To find a specific species or group more quickly you can use the Search Species option to quickly narrow the options. Search for parts of species common names in the Search Species box by entering a term and clicking the search button. The platform will return a shorter list in the Speices options box of only species that contain a match to that search term.

Use the Reset All Parameters button to reset all parameters for entire form.

|  |
| --- |
| Diagram of selection and search tools available on the FOSS platfrom. |

## 11.3 Run report

Click the RUN REPORT button. Below the select and filter area, the results of your query will appear below the page in the format you selected. To change the format, make a different selection and run the report again. Further modifications to your results can be made by clicking on the Actions button above your data. Here you can download your data, select columns included in your results, and apply a variety of filters and mathematical tools.

|  |
| --- |
| Example data returned from running the report. |

## 11.4 API

APIs, or Application Programming Interfaces, allows users to pull data through a IDE, or integrated development environment, like RStudio or VS Code. Explore the API pages for each of the data pages (Haul and catch data, Haul data only, All observed species).

# 12. Use data

Learn how to pull and use this data through the

* [API and R programming language](https://afsc-gap-products.github.io/gap_products/content/foss-api-r.html)
* [API and python programming language using the afscgap python package](https://afsc-gap-products.github.io/gap_products/content/foss-api-py.html)
* [Oracle and R programming language (AFSC scientists only)](https://afsc-gap-products.github.io/gap_products/content/foss-oracle-r.html)

# 13. Access via API and R

**An application programming interface (API) is a way for two or more computer programs to communicate with each other.** More information about how to amend API links can be found [here](https://docs.oracle.com/en/database/oracle/oracle-rest-data-services/22.3/books.html#AELIG90103/). Useful introductions to using APIs in R can be found [here](https://www.dataquest.io/blog/r-api-tutorial/).

There are three tables the user can pull from the API. Learn more about them on the [FOSS data description page](https://afsc-gap-products.github.io/gap_products/content/foss-metadata.html). Here, you can see them in their raw JSON format: - haul: (https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_haul/)[https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_haul/] - catch: (https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_catch/)[https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_catch/] - species: (https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_species/)[https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_species/]

Here are some examples of how to use the data with R:

# 14. Examples of all species in all survey regions in all years

## 14.1 Ex. Load all rows of the catch, haul, and species data tables

Note that without specifying, a basic query to the API will only return 25 entries.

### 14.1.1 Haul data:

# link to the API  
api\_link\_haul <- 'https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_haul/'

**Load first 25 rows of data:**

res <- httr::GET(url = api\_link\_haul)  
# res ## Test connection  
  
## convert from JSON format  
data <- jsonlite::fromJSON(base::rawToChar(res$content))  
  
# Find how many rows and columns are in the data pull  
print(paste0("rows: ", dim(data$items)[1], "; cols: ", dim(data$items)[2]))

[1] "rows: 25; cols: 28"

**Load all data:**

Since the maxim number of rows a user can pull is 10,000 rows in a query, the user needs to cycle through by offsetting to the next 10,000 rows (as is shown here).

dat <- data.frame()  
for (i in seq(0, 500000, 10000)){  
 ## find how many iterations it takes to cycle through the data  
 print(i)  
 ## query the API link  
 res <- httr::GET(url = paste0(api\_link\_haul, "?offset=",i,"&limit=10000"))  
 ## convert from JSON format  
 data <- jsonlite::fromJSON(base::rawToChar(res$content))   
   
 ## if there are no data, stop the loop  
 if (is.null(nrow(data$items))) {  
 break  
 }  
   
 ## bind sub-pull to dat data.frame  
 dat <- dplyr::bind\_rows(dat,   
 data$items %>%  
 dplyr::select(-links)) # necessary for API accounting, but not part of the dataset)  
}

[1] 0  
[1] 10000  
[1] 20000  
[1] 30000  
[1] 40000

summary(dat)

year srvy survey survey\_name   
 Min. :1982 Length:33334 Length:33334 Length:33334   
 1st Qu.:1996 Class :character Class :character Class :character   
 Median :2005 Mode :character Mode :character Mode :character   
 Mean :2005   
 3rd Qu.:2014   
 Max. :2023   
   
 survey\_definition\_id cruise cruisejoin hauljoin   
 Min. : 47.00 Min. :198201 Min. : -766 Min. : -23126   
 1st Qu.: 47.00 1st Qu.:199601 1st Qu.: -691 1st Qu.: -13535   
 Median : 78.00 Median :200501 Median : -612 Median : -3958   
 Mean : 74.51 Mean :200507 Mean : 300438 Mean : 295896   
 3rd Qu.: 98.00 3rd Qu.:201401 3rd Qu.: 837800 3rd Qu.: 821743   
 Max. :143.00 Max. :202302 Max. :1225395 Max. :1225635   
   
 haul stratum station vessel\_id   
 Min. : 1.0 Min. : 10.0 Length:33334 Min. : 1.0   
 1st Qu.: 56.0 1st Qu.: 31.0 Class :character 1st Qu.: 88.0   
 Median :112.0 Median : 50.0 Mode :character Median : 94.0   
 Mean :117.6 Mean :129.1 Mean :106.9   
 3rd Qu.:170.0 3rd Qu.:141.0 3rd Qu.:147.0   
 Max. :355.0 Max. :794.0 Max. :178.0   
   
 vessel\_name date\_time latitude\_dd\_start longitude\_dd\_start  
 Length:33334 Length:33334 Min. :51.19 Min. :-180.0   
 Class :character Class :character 1st Qu.:55.02 1st Qu.:-170.7   
 Mode :character Mode :character Median :57.18 Median :-165.2   
 Mean :56.89 Mean :-140.4   
 3rd Qu.:58.98 3rd Qu.:-154.4   
 Max. :65.34 Max. : 180.0   
   
 latitude\_dd\_end longitude\_dd\_end bottom\_temperature\_c surface\_temperature\_c  
 Min. :51.19 Min. :-180.0 Min. :-2.10 Min. :-1.100   
 1st Qu.:55.02 1st Qu.:-170.7 1st Qu.: 2.70 1st Qu.: 5.800   
 Median :57.18 Median :-165.2 Median : 4.10 Median : 7.500   
 Mean :56.89 Mean :-140.4 Mean : 3.84 Mean : 7.832   
 3rd Qu.:58.97 3rd Qu.:-154.4 3rd Qu.: 5.20 3rd Qu.: 9.400   
 Max. :65.35 Max. : 180.0 Max. :15.30 Max. :18.100   
 NA's :4 NA's :4 NA's :1601 NA's :849   
 depth\_m distance\_fished\_km duration\_hr net\_width\_m   
 Min. : 9 Min. :0.135 Min. :0.0250 Min. : 7.51   
 1st Qu.: 68 1st Qu.:1.498 1st Qu.:0.2710 1st Qu.:15.59   
 Median : 102 Median :2.527 Median :0.4900 Median :16.40   
 Mean : 138 Mean :2.207 Mean :0.4007 Mean :16.42   
 3rd Qu.: 156 3rd Qu.:2.831 3rd Qu.:0.5090 3rd Qu.:17.21   
 Max. :1200 Max. :4.334 Max. :0.9800 Max. :23.82   
   
 net\_height\_m area\_swept\_km2 performance   
 Min. : 0.000 Min. :0.002314 Min. :0.0000   
 1st Qu.: 2.399 1st Qu.:0.024251 1st Qu.:0.0000   
 Median : 5.886 Median :0.039562 Median :0.0000   
 Mean : 4.841 Mean :0.036404 Mean :0.2777   
 3rd Qu.: 6.799 3rd Qu.:0.047326 3rd Qu.:0.0000   
 Max. :11.038 Max. :0.077795 Max. :7.0000   
 NA's :3269

# Find how many rows and columns are in the data pull  
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

[1] "rows: 33334; cols: 27"

# save outputs for later comparison  
dat\_haul\_api <- dat

# Print the first few lines of the data   
head(dat, 3)

| **year** | **srvy** | **survey** | **survey\_name** | **survey\_definition\_id** | **cruise** | **cruisejoin** | **hauljoin** | **haul** | **stratum** | **station** | **vessel\_id** | **vessel\_name** | **date\_time** | **latitude\_dd\_start** | **longitude\_dd\_start** | **latitude\_dd\_end** | **longitude\_dd\_end** | **bottom\_temperature\_c** | **surface\_temperature\_c** | **depth\_m** | **distance\_fished\_km** | **duration\_hr** | **net\_width\_m** | **net\_height\_m** | **area\_swept\_km2** | **performance** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2015 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 201501 | -698 | -14,399 | 133 | 41 | M-20 | 162 | ALASKA KNIGHT | 2015-07-04T11:01:28Z | 58.97713 | -169.8382 | 59.00117 | -169.8264 | 0.0 | 5.2 | 64 | 2.758 | 0.518 | 16.880 | 2.085 | 0.046555 | 0 |
| 2005 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 200501 | -612 | -13,759 | 71 | 31 | F-04 | 88 | ARCTURUS | 2005-06-19T17:23:13Z | 56.65513 | -165.8663 | 56.68046 | -165.8501 | 3.7 | 7.6 | 79 | 2.989 | 0.521 | 17.185 | 2.223 | 0.051366 | 0 |
| 2014 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 201401 | -689 | -11,934 | 89 | 41 | M-19 | 162 | ALASKA KNIGHT | 2014-07-01T15:08:44Z | 58.98618 | -169.1757 | 59.01007 | -169.1831 | 1.0 | 6.4 | 54 | 2.688 | 0.500 | 15.756 | 2.130 | 0.042352 | 0 |

### 14.1.2 Catch data:

# link to the API  
api\_link\_catch <- 'https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_catch/'

# api\_link\_catch <- 'https://dev-apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_catch/' # test

**Load first 25 rows of data:**

res <- httr::GET(url = api\_link\_catch)  
# res ## Test connection  
  
## convert from JSON format  
data <- jsonlite::fromJSON(base::rawToChar(res$content))  
  
# Find how many rows and columns are in the data pull  
print(paste0("rows: ", dim(data$items)[1], "; cols: ", dim(data$items)[2]))

[1] "rows: 25; cols: 8"

**Load all data:**

Since the maxim number of rows a user can pull is 10,000 rows in a query, the user needs to cycle through by offsetting to the next 10,000 rows (as is shown here).

dat <- data.frame()  
for (i in seq(0, 1000000, 10000)){  
 ## find how many iterations it takes to cycle through the data  
 print(i)  
 ## query the API link  
 res <- httr::GET(url = paste0(api\_link\_catch, "?offset=",i,"&limit=10000"))  
 ## convert from JSON format  
 data <- jsonlite::fromJSON(base::rawToChar(res$content))   
   
 ## if there are no data, stop the loop  
 if (is.null(nrow(data$items))) {  
 break  
 }  
   
 ## bind sub-pull to dat data.frame  
 dat <- dplyr::bind\_rows(dat,   
 data$items %>%  
 dplyr::select(-links)) # necessary for API accounting, but not part of the dataset)  
}

[1] 0  
[1] 10000  
[1] 20000  
[1] 30000  
[1] 40000  
[1] 50000  
[1] 60000  
[1] 70000  
[1] 80000  
[1] 90000  
[1] 100000  
[1] 110000  
[1] 120000  
[1] 130000  
[1] 140000  
[1] 150000  
[1] 160000  
[1] 170000  
[1] 180000  
[1] 190000  
[1] 200000  
[1] 210000  
[1] 220000  
[1] 230000  
[1] 240000  
[1] 250000  
[1] 260000  
[1] 270000  
[1] 280000  
[1] 290000  
[1] 300000  
[1] 310000  
[1] 320000  
[1] 330000  
[1] 340000  
[1] 350000  
[1] 360000  
[1] 370000  
[1] 380000  
[1] 390000  
[1] 400000  
[1] 410000  
[1] 420000  
[1] 430000  
[1] 440000  
[1] 450000  
[1] 460000  
[1] 470000  
[1] 480000  
[1] 490000  
[1] 500000  
[1] 510000  
[1] 520000  
[1] 530000  
[1] 540000  
[1] 550000  
[1] 560000  
[1] 570000  
[1] 580000  
[1] 590000  
[1] 600000  
[1] 610000  
[1] 620000  
[1] 630000  
[1] 640000  
[1] 650000  
[1] 660000  
[1] 670000  
[1] 680000  
[1] 690000  
[1] 700000  
[1] 710000  
[1] 720000  
[1] 730000  
[1] 740000  
[1] 750000  
[1] 760000  
[1] 770000  
[1] 780000  
[1] 790000  
[1] 800000  
[1] 810000  
[1] 820000  
[1] 830000  
[1] 840000  
[1] 850000  
[1] 860000  
[1] 870000  
[1] 880000  
[1] 890000  
[1] 900000  
[1] 910000  
[1] 920000  
[1] 930000  
[1] 940000

summary(dat)

hauljoin species\_code cpue\_kgkm2 cpue\_nokm2   
 Min. : -23126 Min. : 1 Min. : 0 Min. : 13   
 1st Qu.: -13968 1st Qu.:21333 1st Qu.: 5 1st Qu.: 56   
 Median : -5200 Median :43010 Median : 40 Median : 207   
 Mean : 286755 Mean :47620 Mean : 1146 Mean : 4379   
 3rd Qu.: 816166 3rd Qu.:72751 3rd Qu.: 316 3rd Qu.: 1079   
 Max. :1225635 Max. :99999 Max. :3226235 Max. :21780780   
 NA's :118525   
 count weight\_kg taxon\_confidence   
 Min. : 1 Min. : 0.001 Length:939197   
 1st Qu.: 2 1st Qu.: 0.168 Class :character   
 Median : 7 Median : 1.480 Mode :character   
 Mean : 172 Mean : 38.354   
 3rd Qu.: 41 3rd Qu.: 11.700   
 Max. :867119 Max. :18187.700   
 NA's :118525

# Find how many rows and columns are in the data pull  
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

[1] "rows: 939197; cols: 7"

# save outputs for later comparison  
dat\_catch\_api <- dat

# Print the first few lines of the data   
head(dat\_catch\_api, 3)

| **hauljoin** | **species\_code** | **cpue\_kgkm2** | **cpue\_nokm2** | **count** | **weight\_kg** | **taxon\_confidence** |
| --- | --- | --- | --- | --- | --- | --- |
| 1,180,846 | 10200 | 413.670836 | 1,455.98431 | 34 | 9.660 | 1 |
| 1,180,867 | 66120 | 25.864144 | 1,363.44531 | 33 | 0.626 | 1 |
| 1,180,871 | 24200 | 0.794012 | 39.70062 | 1 | 0.020 | 1 |

### 14.1.3 Species data:

Since there are less than 10,000 rows of species data (and the maxim number of rows a user can pull from this API is 10,000 rows in a query), we can simply call ?offset=0&limit=10000 in our query call.

# link to the API  
api\_link\_species <- 'https://apps-st.fisheries.noaa.gov/ods/foss/afsc\_groundfish\_survey\_species/'

res <- httr::GET(url = paste0(api\_link\_species, "?offset=0&limit=10000"))  
  
# res ## Test connection  
  
## convert from JSON format  
data <- jsonlite::fromJSON(base::rawToChar(res$content))  
dat <- data$items %>%  
 dplyr::select(-links) # necessary for API accounting, but not part of the dataset  
  
summary(dat)

species\_code scientific\_name common\_name id\_rank   
 Min. : 1 Length:1894 Length:1894 Length:1894   
 1st Qu.:30105 Class :character Class :character Class :character   
 Median :70061 Mode :character Mode :character Mode :character   
 Mean :58788   
 3rd Qu.:80543   
 Max. :99999   
   
 worms itis   
 Min. : 51 Min. : 46861   
 1st Qu.: 134194 1st Qu.: 79433   
 Median : 254533 Median : 157165   
 Mean : 299858 Mean : 209114   
 3rd Qu.: 368813 3rd Qu.: 167413   
 Max. :1699283 Max. :1206057   
 NA's :233 NA's :388

# Find how many rows and columns are in the data pull  
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

[1] "rows: 1894; cols: 6"

# save outputs for later comparison  
dat\_species\_api <- dat

# Print the first few lines of the data   
head(data$items, 3)

| **species\_code** | **scientific\_name** | **common\_name** | **id\_rank** | **worms** | **itis** | **links** |
| --- | --- | --- | --- | --- | --- | --- |
| 460 | Bathyraja trachura | roughtail skate | species | 271538 | 160942 | [[data.frame]] |
| 495 | Bathyraja violacea | Okhotsk skate | species | 271540 | 564249 | [[data.frame]] |
| 710 | Hydrolagus colliei | spotted ratfish | species | 271406 | 161015 | [[data.frame]] |

### 14.1.4 Create zero-filled data for 2023 eastern Bering Sea walleye pollock and plot

It is important to create and have access to zero-fill (presence and absence) so you can do simple analyses and plot data.

# come up with full combination of what species should be listed for what hauls/surveys  
# for zero-filled data, all species caught in a survey need to have zero or non-zero row entries for a haul  
comb <- dplyr::full\_join(  
 # find all species that have been caught, by survey  
 x = dplyr::left\_join(dat\_catch\_api, dat\_haul\_api, by = "hauljoin") %>%  
 dplyr::select(survey\_definition\_id, species\_code) %>%  
 dplyr::distinct(),  
 # find all haul events (hauljoins), by survey  
 y = dat\_haul\_api %>%  
 dplyr::select(survey\_definition\_id, hauljoin) %>%  
 dplyr::distinct(),  
 relationship = "many-to-many",  
 by = "survey\_definition\_id"  
) %>%   
 dplyr::select(-survey\_definition\_id) # redundant  
  
head(comb, 3)

species\_code hauljoin  
1 10200 -22236  
2 10200 -22190  
3 10200 -20871

# Join data to make a full zero-filled CPUE dataset  
  
dat\_zerofill\_api <- comb %>%  
 # add species data to unique species by survey table  
 dplyr::left\_join(dat\_species\_api) %>% # , "species\_code"  
 # add haul data  
 dplyr::left\_join(dat\_haul\_api) %>% # , c("hauljoin")  
 # add catch data  
 dplyr::left\_join(dat\_catch\_api) %>% # , c("species\_code", "hauljoin")  
 # modify zero-filled rows  
 dplyr::mutate(  
 cpue\_kgkm2 = ifelse(is.na(cpue\_kgkm2), 0, cpue\_kgkm2),  
 cpue\_nokm2 = ifelse(is.na(cpue\_nokm2), 0, cpue\_nokm2),  
 count = ifelse(is.na(count), 0, count),  
 weight\_kg = ifelse(is.na(weight\_kg), 0, weight\_kg))  
  
## Find how many rows and columns are in the data pull.   
## Because all of the data have been full joined together,   
## there should be the maximum number of rows.   
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

[1] "rows: 1894; cols: 6"

head(dat, 3) %>%   
 flextable::flextable() %>% flextable::theme\_zebra()

| **species\_code** | **scientific\_name** | **common\_name** | **id\_rank** | **worms** | **itis** |
| --- | --- | --- | --- | --- | --- |
| 460 | Bathyraja trachura | roughtail skate | species | 271,538 | 160,942 |
| 495 | Bathyraja violacea | Okhotsk skate | species | 271,540 | 564,249 |
| 710 | Hydrolagus colliei | spotted ratfish | species | 271,406 | 161,015 |

### 14.1.5 Visualize CPUE data in distribution map

Using the zero-filled data from the previous example, we can make a few plots!

Here is some example data of 2023 through 2019 (year %in% 2019:2023) eastern and northern Bering Sea (srvy == c("EBS", "NBS)) walleye pollock (species\_code == 21740).

dat0 <- dat\_zerofill\_api %>%   
 dplyr::filter(year %in% c(2019:2023) &   
 srvy == c("EBS", "NBS") &   
 species\_code == 21740)  
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

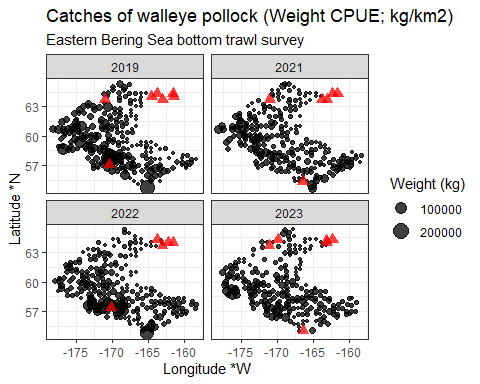
[1] "rows: 1894; cols: 6"

head(dat0, 3) %>% flextable::flextable() %>% flextable::theme\_zebra()

| **species\_code** | **hauljoin** | **scientific\_name** | **common\_name** | **id\_rank** | **worms** | **itis** | **year** | **srvy** | **survey** | **survey\_name** | **survey\_definition\_id** | **cruise** | **cruisejoin** | **haul** | **stratum** | **station** | **vessel\_id** | **vessel\_name** | **date\_time** | **latitude\_dd\_start** | **longitude\_dd\_start** | **latitude\_dd\_end** | **longitude\_dd\_end** | **bottom\_temperature\_c** | **surface\_temperature\_c** | **depth\_m** | **distance\_fished\_km** | **duration\_hr** | **net\_width\_m** | **net\_height\_m** | **area\_swept\_km2** | **performance** | **cpue\_kgkm2** | **cpue\_nokm2** | **count** | **weight\_kg** | **taxon\_confidence** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 21,740 | -21,778 | Gadus chalcogrammus | walleye pollock | species | 300,735 | 934,083 | 2,022 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202,201 | -755 | 196 | 41 | S-26 | 94 | VESTERAALEN | 2022-07-28T16:27:01Z | 61.00249 | -174.1504 | 60.99662 | -174.2016 | -0.8 | 8.5 | 81 | 2.851 | 0.511 | 17.783 | 2.357 | 0.050699 | 0 | 5,103.065 | 11,439.99 | 580 | 258.722 | 1 |
| 21,740 | -21,769 | Gadus chalcogrammus | walleye pollock | species | 300,735 | 934,083 | 2,022 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202,201 | -756 | 193 | 90 | U-29 | 162 | ALASKA KNIGHT | 2022-07-27T15:43:43Z | 61.65277 | -176.4648 | 61.67708 | -176.4805 | 0.0 | 8.2 | 105 | 2.828 | 0.508 | 19.732 | 1.752 | 0.055802 | 0 | 6,025.222 | 13,619.56 | 760 | 336.220 | 1 |
| 21,740 | -21,528 | Gadus chalcogrammus | walleye pollock | species | 300,735 | 934,083 | 2,022 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202,201 | -755 | 135 | 41 | M-23 | 94 | VESTERAALEN | 2022-07-13T07:24:02Z | 58.97864 | -171.7825 | 59.00292 | -171.7754 | 0.1 | 7.9 | 87 | 2.729 | 0.516 | 17.598 | 2.264 | 0.048025 | 0 | 4,088.542 | 10,390.43 | 499 | 196.352 | 1 |

### 14.1.6 Plot locations

library(ggplot2)  
  
ggplot2::ggplot(data = dat0 %>% dplyr::filter(cpue\_kgkm2 != 0),   
 mapping = aes(x = longitude\_dd\_start,   
 y = latitude\_dd\_start,   
 size = cpue\_kgkm2)) +   
 ggplot2::geom\_point(alpha = .75) +  
 ggplot2::geom\_point(data = dat0 %>% dplyr::filter(cpue\_kgkm2 == 0),   
 color = "red",   
 shape = 17,  
 alpha = .75,  
 size = 3) +  
 ggplot2::xlab("Longitude \*W") +  
 ggplot2::ylab("Latitude \*N") +  
 ggplot2::ggtitle(label = "Catches of walleye pollock (Weight CPUE; kg/km2)",   
 subtitle = "Eastern Bering Sea bottom trawl survey") +  
 ggplot2::scale\_size\_continuous(name = "Weight (kg)") +   
 ggplot2::facet\_wrap(facets = vars(year)) +   
 ggplot2::theme\_bw()



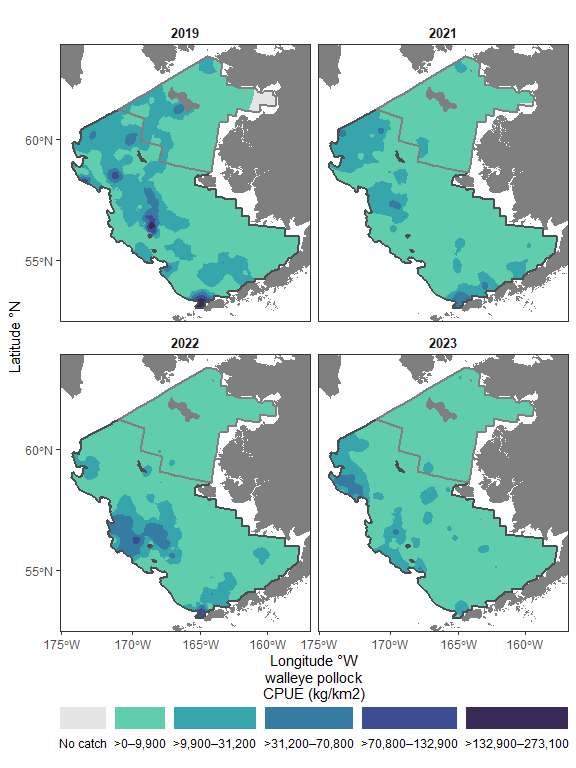
### 14.1.7 Plot inverse-distance weighted modeled product of locations

This map is constructed using [akgfmaps](https://github.com/afsc-gap-products/akgfmaps). To make IDW plots, you must have data from all stations surveyed, even if no fish of interest were found there.

idw <- akgfmaps::make\_idw\_stack(  
 x = dat0 %>%   
 dplyr::select(COMMON\_NAME = common\_name,   
 CPUE\_KGHA = cpue\_kgkm2,   
 LATITUDE = latitude\_dd\_start,   
 LONGITUDE = longitude\_dd\_start,   
 year),   
 grouping.vars = "year",   
 region = "bs.all", # Predefined EBS area  
 set.breaks = "jenks", # Gets Jenks breaks from classint::classIntervals()  
 in.crs = "+proj=longlat", # Set input coordinate reference system  
 out.crs = "EPSG:3338", # Set output coordinate reference system  
 extrapolation.grid.type = "sf")

[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]

shps <- akgfmaps::get\_base\_layers(  
 select.region = "bs.all",   
 include.corners = TRUE,   
 set.crs = "EPSG:3338")  
  
# set.breaks <- akgfmaps::eval\_plot\_breaks(CPUE = dat0$cpue\_kgkm2, n.breaks = 5)  
# set.breaks <- as.vector(unlist(set.breaks[set.breaks$style == "pretty", -1]))  
set.breaks <- c(0, 50000, 100000, 150000, 200000, 250000)  
  
figure\_print <- ggplot() +  
 # add map of alaska  
 ggplot2::geom\_sf(data = shps$akland,  
 color = NA,  
 fill = "grey50") +  
 # add IDW plots  
 geom\_sf(data = idw$extrapolation.stack,  
 mapping = aes(fill = var1.pred),  
 na.rm = FALSE,  
 show.legend = TRUE,   
 color = NA) +  
 ggplot2::scale\_fill\_manual(  
 name = "walleye pollock\nCPUE (kg/km2)",  
 values = c("gray90",  
 viridis::viridis(  
 option = "mako",  
 direction = -1,  
 n = length(set.breaks)-1,  
 begin = 0.20,  
 end = 0.80)),  
 na.translate = FALSE, # Don't use NA  
 drop = FALSE) +   
 # seperate plots by year  
 ggplot2::facet\_wrap(facets = vars(year), nrow = 2) +   
 # add survey area  
 ggplot2::geom\_sf(  
 data = shps$survey.area,   
 mapping = aes(color = SURVEY,   
 geometry = geometry),   
 fill = "transparent",   
 linewidth = 1,   
 show.legend = FALSE) +  
 ggplot2::scale\_color\_manual(  
 name = " ",   
 values = c("grey30", "grey50"),  
 breaks = shps$survey.area$SURVEY,  
 labels = shps$survey.area$SRVY) +   
 # lat/lon axis and map bounds  
 ggplot2::scale\_x\_continuous(name = "Longitude °W",  
 breaks = seq(-180, -150, 5)) +  
 ggplot2::scale\_y\_continuous(name = "Latitude °N",  
 breaks = seq(50, 65, 5)) + # seq(52, 62, 2)  
 ggplot2::coord\_sf(xlim = sf::st\_bbox(shps$survey.area)[c(1,3)],  
 ylim = sf::st\_bbox(shps$survey.area)[c(2,4)]) +  
 # add theme aesthetics  
 ggplot2::guides(  
 fill = guide\_legend(  
 order = 1,  
 title.position = "top",  
 label.position = "bottom",  
 title.hjust = 0.5,  
 override.aes = list(color = NA),  
 nrow = 1),  
 color = "none") +  
 ggplot2::theme(   
 panel.background = element\_rect(fill = "white", colour = NA),   
 panel.border = element\_rect(fill = NA, colour = "grey20"),   
 strip.background = element\_blank(),   
 strip.text = element\_text(size = 10, face = "bold"),   
 legend.text = element\_text(size = 9),  
 legend.background = element\_rect(colour = "transparent",   
 fill = "transparent"),  
 legend.key = element\_rect(colour = "transparent",   
 fill = "transparent"),  
 legend.position = "bottom",   
 legend.box = "horizontal",  
 legend.box.spacing = unit(0, "pt"), # reduce space between legend & plot  
 legend.margin=margin(0, 0, 0, 0) )  
  
figure\_print



# 15. Examples or one species in one survey region in one year

## 15.1 Ex. Show catch data for 2023 eastern Bering Sea Walleye Pollock

Data downloads and joins for just one species, survey, and year are much faster and easier to do.

First, because year is identified in the haul table, we need to identify all of the hauls (or more specifically, hauljoin codes) that were completed in the eastern Bering Sea ("srvy":"EBS") in 2023 ("year":2023).

## query the API link  
res <- httr::GET(url = paste0(api\_link\_haul, '?limit=10000&q={"year":2023,"srvy":"EBS"}'))  
  
## convert from JSON format  
data <- jsonlite::fromJSON(base::rawToChar(res$content))   
dat <- data$items %>%  
 dplyr::select(-links) # necessary for API accounting, but not part of the dataset  
  
## show summary of data to make sure it is subset correctly  
summary(dat %>% dplyr::mutate(srvy = as.factor(srvy)))

year srvy survey survey\_name   
 Min. :2023 EBS:376 Length:376 Length:376   
 1st Qu.:2023 Class :character Class :character   
 Median :2023 Mode :character Mode :character   
 Mean :2023   
 3rd Qu.:2023   
 Max. :2023   
 survey\_definition\_id cruise cruisejoin hauljoin   
 Min. :98 Min. :202301 Min. :-760.0 Min. :-23019   
 1st Qu.:98 1st Qu.:202301 1st Qu.:-760.0 1st Qu.:-22776   
 Median :98 Median :202301 Median :-759.0 Median :-22539   
 Mean :98 Mean :202301 Mean :-759.5 Mean :-22552   
 3rd Qu.:98 3rd Qu.:202301 3rd Qu.:-759.0 3rd Qu.:-22333   
 Max. :98 Max. :202301 Max. :-759.0 Max. :-22110   
 haul stratum station vessel\_id   
 Min. : 7.00 Min. :10.00 Length:376 Min. :134.0   
 1st Qu.: 65.75 1st Qu.:31.00 Class :character 1st Qu.:134.0   
 Median :114.00 Median :41.00 Mode :character Median :162.0   
 Mean :114.16 Mean :39.22 Mean :148.3   
 3rd Qu.:161.25 3rd Qu.:50.00 3rd Qu.:162.0   
 Max. :224.00 Max. :90.00 Max. :162.0   
 vessel\_name date\_time latitude\_dd\_start longitude\_dd\_start  
 Length:376 Length:376 Min. :54.66 Min. :-178.2   
 Class :character Class :character 1st Qu.:57.00 1st Qu.:-172.7   
 Mode :character Mode :character Median :58.02 Median :-168.9   
 Mean :58.26 Mean :-168.8   
 3rd Qu.:59.50 3rd Qu.:-165.2   
 Max. :62.01 Max. :-158.3   
 latitude\_dd\_end longitude\_dd\_end bottom\_temperature\_c surface\_temperature\_c  
 Min. :54.68 Min. :-178.2 Min. :-1.600 Min. : 1.700   
 1st Qu.:57.01 1st Qu.:-172.7 1st Qu.: 1.200 1st Qu.: 4.200   
 Median :58.02 Median :-168.9 Median : 2.700 Median : 6.550   
 Mean :58.26 Mean :-168.8 Mean : 2.249 Mean : 6.386   
 3rd Qu.:59.50 3rd Qu.:-165.2 3rd Qu.: 3.500 3rd Qu.: 8.525   
 Max. :62.01 Max. :-158.3 Max. : 5.400 Max. :11.000   
 depth\_m distance\_fished\_km duration\_hr net\_width\_m   
 Min. : 20.00 Min. :1.065 Min. :0.1890 Min. :12.90   
 1st Qu.: 54.75 1st Qu.:2.805 1st Qu.:0.5100 1st Qu.:16.66   
 Median : 74.00 Median :2.889 Median :0.5180 Median :17.27   
 Mean : 80.75 Mean :2.854 Mean :0.5129 Mean :17.15   
 3rd Qu.:105.00 3rd Qu.:2.945 3rd Qu.:0.5260 3rd Qu.:17.83   
 Max. :171.00 Max. :3.849 Max. :0.6560 Max. :20.29   
 net\_height\_m area\_swept\_km2 performance   
 Min. :1.300 Min. :0.02017 Min. :0.0000   
 1st Qu.:1.875 1st Qu.:0.04725 1st Qu.:0.0000   
 Median :2.064 Median :0.04944 Median :0.0000   
 Mean :2.107 Mean :0.04892 Mean :0.1075   
 3rd Qu.:2.343 3rd Qu.:0.05134 3rd Qu.:0.0000   
 Max. :3.196 Max. :0.06369 Max. :6.2200

## Find how many rows and columns are in the data pull.   
## The eastern Bering Sea survey has 376 stations in it,   
## so this should have a similar number of rows.   
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

[1] "rows: 376; cols: 27"

# save outputs for later comparison  
dat\_haul\_ex <- dat

# Print the first few lines of the data   
head(dat\_haul\_ex, 3)

| **year** | **srvy** | **survey** | **survey\_name** | **survey\_definition\_id** | **cruise** | **cruisejoin** | **hauljoin** | **haul** | **stratum** | **station** | **vessel\_id** | **vessel\_name** | **date\_time** | **latitude\_dd\_start** | **longitude\_dd\_start** | **latitude\_dd\_end** | **longitude\_dd\_end** | **bottom\_temperature\_c** | **surface\_temperature\_c** | **depth\_m** | **distance\_fished\_km** | **duration\_hr** | **net\_width\_m** | **net\_height\_m** | **area\_swept\_km2** | **performance** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2023 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202301 | -760 | -22,950 | 194 | 41 | S-26 | 134 | NORTHWEST EXPLORER | 2023-07-25T10:20:59Z | 60.99106 | -174.1873 | 61.01197 | -174.1552 | -0.8 | 9.1 | 83 | 2.902 | 0.507 | 18.427 | 1.890 | 0.053475 | 0 |
| 2023 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202301 | -760 | -22,621 | 124 | 42 | J-22 | 134 | NORTHWEST EXPLORER | 2023-06-30T08:19:37Z | 57.98877 | -170.9448 | 57.98921 | -170.9935 | 2.3 | 7.1 | 86 | 2.887 | 0.508 | 17.772 | 1.945 | 0.051308 | 0 |
| 2023 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202301 | -759 | -22,710 | 155 | 61 | G-25 | 162 | ALASKA KNIGHT | 2023-07-08T15:09:48Z | 56.98981 | -172.6523 | 57.01371 | -172.6623 | 3.6 | 8.4 | 122 | 2.726 | 0.498 | 17.692 | 1.653 | 0.048228 | 0 |

### 15.1.1 Identify species\_code for walleye pollock

In the catch data, we itemize species catches by species\_code. To find out which species\_code to use, you can check variations on the following code. Note that here the word pollock is case sensitive. All species common\_name entries are lower case except for proper nouns (e.g., “Pacific”). The notation for finding a string is to use % around the phrase. Since % is a reserved character in a URL, you have to replace % with %25. Similarly, %20 needs to be used in place of a space (e.g., between “walleye” and “pollock”: "walleye%20pollock"}').

## query the API link. Use:   
res <- httr::GET(url = paste0(api\_link\_species, '?q={%22common\_name%22:%22walleye%20pollock%22}'))  
# OR  
res <- httr::GET(url = paste0(api\_link\_species, '?q={"common\_name":{"$like":"%25pollock%25"}}'))  
# OR  
res <- httr::GET(url = paste0(api\_link\_species, '?q={"common\_name":"walleye%20pollock"}'))  
  
## convert from JSON format  
data <- jsonlite::fromJSON(base::rawToChar(res$content))   
  
# save outputs for later comparison  
dat\_species\_ex <- data$items %>% dplyr::select(-links) # necessary for API accounting, but not part of the dataset

dat\_species\_ex

| **species\_code** | **scientific\_name** | **common\_name** | **id\_rank** | **worms** | **itis** |
| --- | --- | --- | --- | --- | --- |
| 21740 | Gadus chalcogrammus | walleye pollock | species | 300735 | 934083 |

### 15.1.2 Then, apply the hauljoins and species\_code to catch query

We’ll use the data from the haul and species table we collected before to select 2023 eastern Bering Sea walleye pollock catch data.

## query the API link  
# data for all walleye pollock caught in all 2023 eastern Bering Sea survey hauls  
dat <- data.frame()  
# there must be a better way to select multiple values for one parameter,   
# but saving that, we will loop through each hauljoin and collect the data of interest  
for (i in 1:nrow(dat\_haul\_ex)) {  
 res <- httr::GET(url = paste0(  
 api\_link\_catch,   
 '?q={"species\_code":21740,"hauljoin":', dat\_haul\_ex$hauljoin[i],'}'))  
 ## convert from JSON format  
 data <- jsonlite::fromJSON(base::rawToChar(res$content))   
 if (length(data$items) != 0) {  
 dat <- dplyr::bind\_rows(  
 dat,  
 data$items %>%   
 dplyr::select(-links)) # necessary for API accounting, but not part of the dataset  
 }  
}  
  
## show summary of data to make sure it is subset correctly  
summary(dat)

hauljoin species\_code cpue\_kgkm2 cpue\_nokm2   
 Min. :-23019 Min. :21740 Min. : 10.34 Min. : 18.26   
 1st Qu.:-22777 1st Qu.:21740 1st Qu.: 1454.44 1st Qu.: 2278.72   
 Median :-22540 Median :21740 Median : 3286.76 Median : 5875.33   
 Mean :-22553 Mean :21740 Mean : 6364.85 Mean : 11565.45   
 3rd Qu.:-22324 3rd Qu.:21740 3rd Qu.: 6956.25 3rd Qu.: 12512.98   
 Max. :-22110 Max. :21740 Max. :148679.68 Max. :202321.08   
 NA's :1   
 count weight\_kg taxon\_confidence  
 Min. : 1 Min. : 0.492 Mode:logical   
 1st Qu.: 114 1st Qu.: 71.560 NA's:374   
 Median : 286 Median : 162.310   
 Mean : 574 Mean : 315.419   
 3rd Qu.: 619 3rd Qu.: 350.399   
 Max. :9997 Max. :7346.495   
 NA's :1

## Find how many rows and columns are in the data pull.   
## The eastern Bering Sea survey has 376 stations in it,   
## and pollock are often found in throughout the region  
## so this should have a similar number of rows.   
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

[1] "rows: 374; cols: 7"

# save outputs for later comparison  
dat\_catch\_ex <- dat

# Print the first few lines of the data   
head(dat, 3)

| **hauljoin** | **species\_code** | **cpue\_kgkm2** | **cpue\_nokm2** | **count** | **weight\_kg** | **taxon\_confidence** |
| --- | --- | --- | --- | --- | --- | --- |
| -22,950 | 21740 | 2,101.761 | 5,123.875 | 274 | 112.392 |  |
| -22,621 | 21740 | 5,768.932 | 10,563.703 | 542 | 295.991 |  |
| -22,710 | 21740 | 1,840.596 | 2,363.753 | 114 | 88.769 |  |

For reference and to help break down the above query, see these other query examples:

# data for haul -22775 (i.e., one specific haul)?  
res <- httr::GET(url = paste0(api\_link\_catch,   
 '?offset=',i,'&limit=10000&q={"hauljoin":-22775}'))  
  
# data for all walleye pollock (i.e., one species) caught in all years and surveys  
res <- httr::GET(url = paste0(api\_link\_catch,   
 '?offset=',i,'&limit=10000&q={"species\_code":21740}'))

### 15.1.3 Create zero-filled data for 2023 eastern Bering Sea walleye pollock and plot

It is important to create and have access to zero-fill (presence and absence) so you can do simple analyses and plot data.

dat <- dplyr::full\_join(  
 dat\_haul\_ex,  
 dat\_catch\_ex) %>%   
 dplyr::full\_join(  
 dat\_species\_ex) %>%   
 # modify zero-filled rows  
 dplyr::mutate(  
 cpue\_kgkm2 = ifelse(is.na(cpue\_kgkm2), 0, cpue\_kgkm2),  
 cpue\_nokm2 = ifelse(is.na(cpue\_nokm2), 0, cpue\_nokm2),  
 count = ifelse(is.na(count), 0, count),  
 weight\_kg = ifelse(is.na(weight\_kg), 0, weight\_kg))  
  
## show summary of data to make sure it is subset correctly  
summary(dat)

year srvy survey survey\_name   
 Min. :2023 Length:376 Length:376 Length:376   
 1st Qu.:2023 Class :character Class :character Class :character   
 Median :2023 Mode :character Mode :character Mode :character   
 Mean :2023   
 3rd Qu.:2023   
 Max. :2023   
   
 survey\_definition\_id cruise cruisejoin hauljoin   
 Min. :98 Min. :202301 Min. :-760.0 Min. :-23019   
 1st Qu.:98 1st Qu.:202301 1st Qu.:-760.0 1st Qu.:-22776   
 Median :98 Median :202301 Median :-759.0 Median :-22539   
 Mean :98 Mean :202301 Mean :-759.5 Mean :-22552   
 3rd Qu.:98 3rd Qu.:202301 3rd Qu.:-759.0 3rd Qu.:-22333   
 Max. :98 Max. :202301 Max. :-759.0 Max. :-22110   
   
 haul stratum station vessel\_id   
 Min. : 7.00 Min. :10.00 Length:376 Min. :134.0   
 1st Qu.: 65.75 1st Qu.:31.00 Class :character 1st Qu.:134.0   
 Median :114.00 Median :41.00 Mode :character Median :162.0   
 Mean :114.16 Mean :39.22 Mean :148.3   
 3rd Qu.:161.25 3rd Qu.:50.00 3rd Qu.:162.0   
 Max. :224.00 Max. :90.00 Max. :162.0   
   
 vessel\_name date\_time latitude\_dd\_start longitude\_dd\_start  
 Length:376 Length:376 Min. :54.66 Min. :-178.2   
 Class :character Class :character 1st Qu.:57.00 1st Qu.:-172.7   
 Mode :character Mode :character Median :58.02 Median :-168.9   
 Mean :58.26 Mean :-168.8   
 3rd Qu.:59.50 3rd Qu.:-165.2   
 Max. :62.01 Max. :-158.3   
   
 latitude\_dd\_end longitude\_dd\_end bottom\_temperature\_c surface\_temperature\_c  
 Min. :54.68 Min. :-178.2 Min. :-1.600 Min. : 1.700   
 1st Qu.:57.01 1st Qu.:-172.7 1st Qu.: 1.200 1st Qu.: 4.200   
 Median :58.02 Median :-168.9 Median : 2.700 Median : 6.550   
 Mean :58.26 Mean :-168.8 Mean : 2.249 Mean : 6.386   
 3rd Qu.:59.50 3rd Qu.:-165.2 3rd Qu.: 3.500 3rd Qu.: 8.525   
 Max. :62.01 Max. :-158.3 Max. : 5.400 Max. :11.000   
   
 depth\_m distance\_fished\_km duration\_hr net\_width\_m   
 Min. : 20.00 Min. :1.065 Min. :0.1890 Min. :12.90   
 1st Qu.: 54.75 1st Qu.:2.805 1st Qu.:0.5100 1st Qu.:16.66   
 Median : 74.00 Median :2.889 Median :0.5180 Median :17.27   
 Mean : 80.75 Mean :2.854 Mean :0.5129 Mean :17.15   
 3rd Qu.:105.00 3rd Qu.:2.945 3rd Qu.:0.5260 3rd Qu.:17.83   
 Max. :171.00 Max. :3.849 Max. :0.6560 Max. :20.29   
   
 net\_height\_m area\_swept\_km2 performance species\_code   
 Min. :1.300 Min. :0.02017 Min. :0.0000 Min. :21740   
 1st Qu.:1.875 1st Qu.:0.04725 1st Qu.:0.0000 1st Qu.:21740   
 Median :2.064 Median :0.04944 Median :0.0000 Median :21740   
 Mean :2.107 Mean :0.04892 Mean :0.1075 Mean :21740   
 3rd Qu.:2.343 3rd Qu.:0.05134 3rd Qu.:0.0000 3rd Qu.:21740   
 Max. :3.196 Max. :0.06369 Max. :6.2200 Max. :21740   
 NA's :2   
 cpue\_kgkm2 cpue\_nokm2 count weight\_kg   
 Min. : 0 Min. : 0 Min. : 0.0 Min. : 0.00   
 1st Qu.: 1431 1st Qu.: 2238 1st Qu.: 111.5 1st Qu.: 70.64   
 Median : 3273 Median : 5842 Median : 280.0 Median : 161.44   
 Mean : 6331 Mean : 11473 Mean : 569.4 Mean : 313.74   
 3rd Qu.: 6946 3rd Qu.: 12345 3rd Qu.: 611.5 3rd Qu.: 349.81   
 Max. :148680 Max. :202321 Max. :9997.0 Max. :7346.49   
   
 taxon\_confidence scientific\_name common\_name id\_rank   
 Mode:logical Length:376 Length:376 Length:376   
 NA's:376 Class :character Class :character Class :character   
 Mode :character Mode :character Mode :character   
   
   
   
   
 worms itis   
 Min. :300735 Min. :934083   
 1st Qu.:300735 1st Qu.:934083   
 Median :300735 Median :934083   
 Mean :300735 Mean :934083   
 3rd Qu.:300735 3rd Qu.:934083   
 Max. :300735 Max. :934083   
 NA's :2 NA's :2

## Find how many rows and columns are in the data pull.   
## Because all of the data have been full joined together,   
## there should be the maximum number of rows.   
print(paste0("rows: ", dim(dat)[1], "; cols: ", dim(dat)[2]))

[1] "rows: 376; cols: 38"

head(dat, 3) %>%   
 flextable::flextable()

| year | srvy | survey | survey\_name | survey\_definition\_id | cruise | cruisejoin | hauljoin | haul | stratum | station | vessel\_id | vessel\_name | date\_time | latitude\_dd\_start | longitude\_dd\_start | latitude\_dd\_end | longitude\_dd\_end | bottom\_temperature\_c | surface\_temperature\_c | depth\_m | distance\_fished\_km | duration\_hr | net\_width\_m | net\_height\_m | area\_swept\_km2 | performance | species\_code | cpue\_kgkm2 | cpue\_nokm2 | count | weight\_kg | taxon\_confidence | scientific\_name | common\_name | id\_rank | worms | itis |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2,023 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202,301 | -760 | -22,950 | 194 | 41 | S-26 | 134 | NORTHWEST EXPLORER | 2023-07-25T10:20:59Z | 60.99106 | -174.1873 | 61.01197 | -174.1552 | -0.8 | 9.1 | 83 | 2.902 | 0.507 | 18.427 | 1.890 | 0.053475 | 0 | 21,740 | 2,101.761 | 5,123.875 | 274 | 112.392 |  | Gadus chalcogrammus | walleye pollock | species | 300,735 | 934,083 |
| 2,023 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202,301 | -760 | -22,621 | 124 | 42 | J-22 | 134 | NORTHWEST EXPLORER | 2023-06-30T08:19:37Z | 57.98877 | -170.9448 | 57.98921 | -170.9935 | 2.3 | 7.1 | 86 | 2.887 | 0.508 | 17.772 | 1.945 | 0.051308 | 0 | 21,740 | 5,768.932 | 10,563.703 | 542 | 295.991 |  | Gadus chalcogrammus | walleye pollock | species | 300,735 | 934,083 |
| 2,023 | EBS | eastern Bering Sea | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 98 | 202,301 | -759 | -22,710 | 155 | 61 | G-25 | 162 | ALASKA KNIGHT | 2023-07-08T15:09:48Z | 56.98981 | -172.6523 | 57.01371 | -172.6623 | 3.6 | 8.4 | 122 | 2.726 | 0.498 | 17.692 | 1.653 | 0.048228 | 0 | 21,740 | 1,840.596 | 2,363.753 | 114 | 88.769 |  | Gadus chalcogrammus | walleye pollock | species | 300,735 | 934,083 |

### 15.1.4 Visualize CPUE data in distribution map

Using the zero-filled data from the previous example, we can make a few plots!

### 15.1.5 Plot locations

library(ggplot2)  
  
ggplot2::ggplot(data = dat %>% dplyr::filter(cpue\_kgkm2 != 0),   
 mapping = aes(x = longitude\_dd\_start,   
 y = latitude\_dd\_start,   
 size = cpue\_kgkm2)) +   
 ggplot2::geom\_point(alpha = .75) +  
 ggplot2::geom\_point(data = dat %>% dplyr::filter(cpue\_kgkm2 == 0),   
 color = "red",   
 shape = 17,  
 alpha = .75,  
 size = 3) +  
 ggplot2::xlab("Longitude \*W") +  
 ggplot2::ylab("Latitude \*N") +  
 ggplot2::ggtitle(label = "Catches of walleye pollock (Weight CPUE; kg/km2)",   
 subtitle = "2023 eastern Bering Sea bottom trawl survey") +  
 ggplot2::scale\_size\_continuous(name = "Weight (kg)") +   
 ggplot2::theme\_bw()



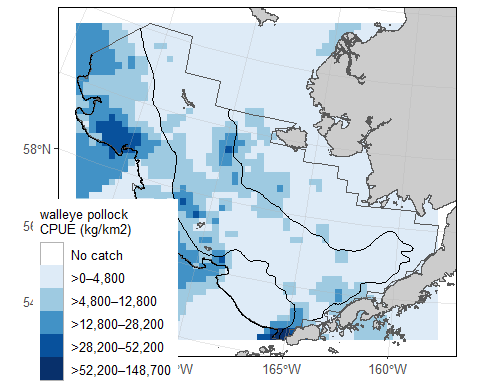
### 15.1.6 Plot inverse-distance weighted modeled product of locations

This map is constructed using [akgfmaps](https://github.com/afsc-gap-products/akgfmaps)

# devtools::install\_github("afsc-gap-products/akgfmaps", build\_vignettes = TRUE)  
library(akgfmaps)  
  
figure0 <- akgfmaps::make\_idw\_map(  
 CPUE\_KGHA = dat$cpue\_kgkm2, # calculates the same, regardless of units.   
 LATITUDE = dat$latitude\_dd\_start,   
 LONGITUDE = dat$longitude\_dd\_start,   
 region = "bs.south", # Predefined EBS area  
 set.breaks = "jenks", # Gets Jenks breaks from classint::classIntervals()  
 in.crs = "+proj=longlat", # Set input coordinate reference system  
 out.crs = "EPSG:3338", # Set output coordinate reference system  
 grid.cell = c(20000, 20000))

[inverse distance weighted interpolation]  
[inverse distance weighted interpolation]

figure0$plot + # 20x20km grid  
 ggplot2::guides(fill=guide\_legend(title = "walleye pollock\nCPUE (kg/km2)"))



# 16. Other query examples

## 16.1 Ex. Combination of year, srvy, stratum

Show haul data where year is less than 1989, srvy = “EBS”, and stratum is not equal to 81.

res <- httr::GET(  
 url = paste0(api\_link\_haul,   
 '?&limit=10000&q={"year":{"$lt":1989},"stratum":{"$ne":"81"},"srvy":"EBS"}}'))  
data <- jsonlite::fromJSON(base::rawToChar(res$content))  
dat <- data$items %>%   
 dplyr::select(-links) # necessary for API accounting, but not part of the dataset)

# 17. Access via API and Python

### 17.0.1 {afscgap} Library Installation

author: Sam Pottinger (sam.pottinger@berkeley.edu; GitHub::sampottinger) date: May 13, 2023

The third-party afscgap Python package interfaces with FOSS to access AFSC GAP data. It can be installed via pip:

#The reticulate package provides a comprehensive set of tools for interoperability between Python and R.   
library(reticulate)

pip install afscgap  
pip install git+https://github.com/SchmidtDSE/afscgap.git@main

For more information on installation and deployment, see the [library documentation](https://pyafscgap.org).

### 17.0.2 Basic query

This first example queries for Pacific glass shrimp (*Pasiphaea pacifica*) in the Gulf of Alaska in 2021. The library will automatically generate HTTP queries, converting from Python types to [ORDS](https://www.oracle.com/database/technologies/appdev/rest.html) query syntax.

import afscgap  
  
query = afscgap.Query()  
query.filter\_year(eq=2021)  
query.filter\_srvy(eq='GOA')  
query.filter\_scientific\_name(eq='Pasiphaea pacifica')  
  
results = query.execute()

The results variable in this example is an iterator that will automatically perform pagination behind the scenes.

### 17.0.3 Iterating with a for loop

The easiest way to interact with results is a simple for loop. This next example determines the frequency of different catch per unit effort where Pacific glass shrimp were reported:

import afscgap  
  
# Mapping from CPUE to count  
count\_by\_cpue = {}  
  
# Build query  
query = afscgap.Query()  
query.filter\_year(eq=2021)  
query.filter\_srvy(eq='GOA')  
query.filter\_scientific\_name(eq='Pasiphaea pacifica')  
results = query.execute()  
  
# Iterate through results and count  
for record in results:  
 cpue = record.get\_cpue\_weight(units='kg/ha')  
 cpue\_rounded = round(cpue)  
 count = count\_by\_cpue.get(cpue\_rounded, 0) + 1  
 count\_by\_cpue[cpue\_rounded] = count  
  
# Print the result  
print(count\_by\_cpue)

Note that, in this example, only records with Pacific glass shrimp are included (“presence-only” data). See zero catch inference below. In other words, it reports on CPUE only for hauls in which Pacific glass shrimp were recorded, excluding some hauls like those in which Pacific glass shrimp were not found at all.

### 17.0.4 Iterating with functional programming

A for loop is not the only option for iterating through results. List comprehensions and other functional programming methods can be used as well.

import statistics  
  
import afscgap  
  
# Build query  
query = afscgap.Query()  
query.filter\_year(eq=2021)  
query.filter\_srvy(eq='GOA')  
query.filter\_scientific\_name(eq='Pasiphaea pacifica')  
results = query.execute()  
  
# Get temperatures in Celsius  
temperatures = [record.get\_bottom\_temperature(units='c') for record in results]  
  
# Take the median  
print(statistics.median(temperatures))

This example reports the median temperature in Celcius for when Pacific glass shrimp was reported.

### 17.0.5 Load into Pandas

The results from the afscgap package are serializable and can be loaded into other tools like [Pandas](https://pandas.pydata.org/). This example loads Pacific glass shrimp from 2021 Gulf of Alaska into a data frame.

import pandas  
  
import afscgap  
  
query = afscgap.Query()  
query.filter\_year(eq=2021)  
query.filter\_srvy(eq='GOA')  
query.filter\_scientific\_name(eq='Pasiphaea pacifica')  
results = query.execute()  
  
pandas.DataFrame(results.to\_dicts())

Specifically, to\_dicts provides an iterator over a dictionary form of the data that can be read into tools like Pandas.

### 17.0.6 Advanced filtering

Queries so far have focused on filters requiring equality but range queries can be built as well.

import afscgap  
  
# Build query  
query = afscgap.Query()  
query.filter\_year(min\_val=2015, max\_val=2019) # Note min/max\_val  
query.filter\_srvy(eq='GOA')  
query.filter\_scientific\_name(eq='Pasiphaea pacifica')  
results = query.execute()  
  
# Sum weight  
weights = map(lambda x: x.get\_weight(units='kg'), results)  
total\_weight = sum(weights)  
print(total\_weight)

This example queries for Pacific glass shrimp data between 2015 and 2019, summing the total weight caught. Note that most users will likely take advantage of built-in Python to [ORDS](https://www.oracle.com/database/technologies/appdev/rest.html) query generation which dictates how the library communicates with the API service. However, users can provide raw ORDS queries as well using [manual filtering](https://pyafscgap.org/devdocs/afscgap.html#manual-filtering).

### 17.0.7 Zero-catch inference

Until this point, these examples use presence-only data. However, the afscgap package can infer negative or “zero catch” records as well.

import afscgap  
  
# Mapping from CPUE to count  
count\_by\_cpue = {}  
  
# Build query  
query = afscgap.Query()  
query.filter\_year(eq=2021)  
query.filter\_srvy(eq='GOA')  
query.filter\_scientific\_name(eq='Pasiphaea pacifica')  
query.set\_presence\_only(False) # Added to earlier example  
results = query.execute()  
  
# Iterate through results and count  
for record in results:  
 cpue = record.get\_cpue\_weight(units='kg/ha')  
 cpue\_rounded = round(cpue)  
 count = count\_by\_cpue.get(cpue\_rounded, 0) + 1  
 count\_by\_cpue[cpue\_rounded] = count  
  
# Print the result  
print(count\_by\_cpue)

This example revisits the earlier snippet for CPUE counts but set\_presence\_only(False) directs the library to look at additional data on hauls, determining which hauls did not have Pacific glass shrimp. This lets the library return records for hauls in which Pacific glass shrimp were not found. This can be seen in differences in counts reported:

| Rounded CPUE | Count with set\_presence\_only(True) | Count with set\_presence\_only(False) |
| --- | --- | --- |
| 0 kg/ha | 44 | 521 |
| 1 kg/ha | 7 | 7 |
| 2 kg/ha | 1 | 1 |

Put simply, while the earlier example showed CPUE counts for hauls in which Pacific glass shrimp were seen, this revised example reports for all hauls in the Gulf of Alaska in 2021.

### 17.0.8 More information

Please see the [API documentation](https://pyafscgap.org/devdocs/afscgap.html) for the Python library for additional details.

# 18. Access via Oracle and R (AFSC only)

If the user has access to the AFSC Oracle database, the user can use SQL developer to view and pull the FOSS public data directly from the GAP\_PRODUCTS Oracle schema.

### 18.0.1 Connect to Oracle from R

Many users will want to access the data from Oracle using R. The user will need to install the RODBC R package and ask OFIS (IT) connect R to Oracle. Then, use the following code in R to establish a connection from R to Oracle:

Here, the user can write in their username and password directly into the RODBC connect function. Never save usernames or passwords in scripts that may be intentionally or unintentionally shared with others. If no username and password is entered in the function, pop-ups will appear on the screen asking for the username and password.

library(gapindex)  
channel <- gapindex::get\_connected()

### 18.0.2 Ex. Wholesale download data and join data in R

locations <- c(  
 "GAP\_PRODUCTS.FOSS\_CATCH",  
 "GAP\_PRODUCTS.FOSS\_HAUL",  
 "GAP\_PRODUCTS.FOSS\_SPECIES"  
)  
  
print(Sys.Date())  
  
error\_loading <- c() # log if any tables are unable to download   
for (i in 1:length(locations)){  
 print(locations[i])  
 a <- RODBC::sqlQuery(channel, paste0("SELECT \* FROM ", locations[i], "; "))  
 if (is.null(nrow(a))) { # if an error in downloading has occurred  
 error\_loading <- c(error\_loading, locations[i])  
 } else { # if no error in downloading has occurred  
 write.csv(x = a,   
 # change file name to be more computer file storage friendly  
 here::here(paste0(tolower(gsub(  
 pattern = '.',   
 replacement = "\_",   
 x = locations[i],   
 fixed = TRUE)),  
 ".csv")))  
 }  
}  
error\_loading

Join downloaded files into presence-only table

# Load data  
library(dplyr)  
library(here)  
library(readr)  
catch <- readr::read\_csv(file = here::here("data/gap\_products\_foss\_catch.csv"))[,-1] # remove "row number" column  
haul <- readr::read\_csv(file = here::here("data/gap\_products\_foss\_haul.csv"))[,-1] # remove "row number" column  
species <- readr::read\_csv(file = here::here("data/gap\_products\_foss\_species.csv"))[,-1] # remove "row number" column  
  
dat <-   
 # join haul and catch data to unique species by survey table  
 dplyr::left\_join(haul, catch) %>%   
 # join species data to unique species by survey table  
 dplyr::left\_join(species) %>%   
 # modify zero-filled rows  
 dplyr::mutate(  
 CPUE\_KGKM2 = ifelse(is.null(CPUE\_KGKM2), 0, CPUE\_KGKM2), # just in case  
 CPUE\_KGHA = CPUE\_KGKM2/100, # Hectares  
 CPUE\_NOKM2 = ifelse(is.null(CPUE\_NOKM2), 0, CPUE\_NOKM2), # just in case  
 CPUE\_NOHA = CPUE\_NOKM2/100, # Hectares  
 COUNT = ifelse(is.null(COUNT), 0, COUNT),  
 WEIGHT\_KG = ifelse(is.null(WEIGHT\_KG), 0, WEIGHT\_KG) )

Join downloaded files into zero-filled table

# Load data  
library(dplyr)  
library(here)  
library(readr)  
catch <- readr::read\_csv(file = here::here("data/gap\_products\_foss\_catch.csv"))[,-1] # remove "row number" column  
haul <- readr::read\_csv(file = here::here("data/gap\_products\_foss\_haul.csv"))[,-1] # remove "row number" column  
species <- readr::read\_csv(file = here::here("data/gap\_products\_foss\_species.csv"))[,-1] # remove "row number" column  
  
# come up with full combination of what species should be listed for what hauls/surveys  
# for zero-filled data, all species caught in a survey need to have zero or non-zero row entries for a haul  
comb <- dplyr::full\_join(  
 x = dplyr::left\_join(catch, haul, by = "HAULJOIN") %>%  
 dplyr::select(SURVEY\_DEFINITION\_ID, SPECIES\_CODE) %>%  
 dplyr::distinct(),  
 y = haul %>%  
 dplyr::select(SURVEY\_DEFINITION\_ID, HAULJOIN) %>%  
 dplyr::distinct(),   
 by = "SURVEY\_DEFINITION\_ID",   
 relationship = "many-to-many"  
)  
  
# Join data to make a full zero-filled CPUE dataset  
dat <- comb %>%   
 # add species data to unique species by survey table  
 dplyr::left\_join(species, "SPECIES\_CODE") %>%   
 # add catch data  
 dplyr::full\_join(catch, c("SPECIES\_CODE", "HAULJOIN")) %>%   
 # add haul data  
 dplyr::full\_join(haul) %>% # , c("SURVEY\_DEFINITION\_ID", "HAULJOIN")  
 # modify zero-filled rows  
 dplyr::mutate(  
 CPUE\_KGKM2 = ifelse(is.null(CPUE\_KGKM2), 0, CPUE\_KGKM2),  
 CPUE\_KGHA = CPUE\_KGKM2/100, # Hectares  
 CPUE\_NOKM2 = ifelse(is.null(CPUE\_NOKM2), 0, CPUE\_NOKM2),  
 CPUE\_NOHA = CPUE\_NOKM2/100, # Hectares  
 COUNT = ifelse(is.null(COUNT), 0, COUNT),  
 WEIGHT\_KG = ifelse(is.null(WEIGHT\_KG), 0, WEIGHT\_KG) )

### 18.0.3 Ex. Join data using Oracle

To join these tables in Oracle, you may use a variant of the following code:

SELECT   
hh.YEAR,  
hh.SRVY,   
hh.SURVEY,  
hh.SURVEY\_DEFINITION\_ID,  
hh.SURVEY\_NAME,  
hh.CRUISE,  
hh.CRUISEJOIN,   
hh.HAUL,  
hh.HAULJOIN,  
hh.STRATUM,  
hh.STATION,  
hh.VESSEL\_ID,  
hh.VESSEL\_NAME,   
hh.DATE\_TIME,  
hh.LATITUDE\_DD\_START,   
hh.LONGITUDE\_DD\_START,   
hh.LATITUDE\_DD\_END,  
hh.LONGITUDE\_DD\_END,   
hh.BOTTOM\_TEMPERATURE\_C,  
hh.SURFACE\_TEMPERATURE\_C,  
hh.DEPTH\_M,  
cc.SPECIES\_CODE,  
ss.ITIS,  
ss.WORMS,  
ss.COMMON\_NAME,   
ss.SCIENTIFIC\_NAME,  
ss.ID\_RANK,  
CASE WHEN cc.CPUE\_KGKM2 IS NULL THEN 0 ELSE cc.CPUE\_KGKM2 END AS CPUE\_KGKM2,  
CASE WHEN cc.CPUE\_NOKM2 IS NULL THEN 0 ELSE cc.CPUE\_NOKM2 END AS CPUE\_NOKM2,  
CASE WHEN cc.COUNT IS NULL THEN 0 ELSE cc.COUNT END AS COUNT,  
CASE WHEN cc.WEIGHT\_KG IS NULL THEN 0 ELSE cc.WEIGHT\_KG END AS WEIGHT\_KG,  
CASE WHEN cc.TAXON\_CONFIDENCE IS NULL THEN NULL ELSE cc.TAXON\_CONFIDENCE END AS TAXON\_CONFIDENCE,  
hh.AREA\_SWEPT\_KM2,   
hh.DISTANCE\_FISHED\_KM,  
hh.DURATION\_HR,   
hh.NET\_WIDTH\_M,  
hh.NET\_HEIGHT\_M,  
hh.PERFORMANCE   
FROM GAP\_PRODUCTS.FOSS\_SURVEY\_SPECIES sv  
FULL OUTER JOIN GAP\_PRODUCTS.FOSS\_SPECIES ss  
ON sv.SPECIES\_CODE = ss.SPECIES\_CODE  
FULL OUTER JOIN GAP\_PRODUCTS.FOSS\_HAUL hh  
ON sv.SURVEY\_DEFINITION\_ID = hh.SURVEY\_DEFINITION\_ID  
FULL OUTER JOIN GAP\_PRODUCTS.FOSS\_CATCH cc  
ON sv.SPECIES\_CODE = cc.SPECIES\_CODE  
AND hh.HAULJOIN = cc.HAULJOIN

### 18.0.4 Ex. Subset data

Here, we are pulling EBS Pacific cod from 2010 - 2021:

# Pull data  
data <- RODBC::sqlQuery(  
channel = channel,   
query =   
"SELECT \* FROM GAP\_PRODUCTS.FOSS\_CATCH cc  
JOIN GAP\_PRODUCTS.FOSS\_HAUL hh  
ON cc.HAULJOIN = hh.HAULJOIN  
WHERE SRVY = 'EBS'   
AND SPECIES\_CODE = 21720 -- 'Pacific cod'   
AND YEAR >= 2010   
AND YEAR < 2021")  
  
flextable::flextable(data[1:3,]) %>%   
 flextable::theme\_zebra()

| **HAULJOIN** | **SPECIES\_CODE** | **CPUE\_KGKM2** | **CPUE\_NOKM2** | **COUNT** | **WEIGHT\_KG** | **TAXON\_CONFIDENCE** | **YEAR** | **SRVY** | **SURVEY** | **SURVEY\_DEFINITION\_ID** | **SURVEY\_NAME** | **CRUISE** | **CRUISEJOIN** | **HAULJOIN.1** | **HAUL** | **STRATUM** | **STATION** | **VESSEL\_ID** | **VESSEL\_NAME** | **DATE\_TIME** | **LATITUDE\_DD\_START** | **LONGITUDE\_DD\_START** | **LATITUDE\_DD\_END** | **LONGITUDE\_DD\_END** | **BOTTOM\_TEMPERATURE\_C** | **SURFACE\_TEMPERATURE\_C** | **DEPTH\_M** | **DISTANCE\_FISHED\_KM** | **DURATION\_HR** | **NET\_WIDTH\_M** | **NET\_HEIGHT\_M** | **AREA\_SWEPT\_KM2** | **PERFORMANCE** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -19,288 | 21,720 | 449.8301 | 1,876.1759 | 83 | 19.90 | 1 | 2,019 | EBS | eastern Bering Sea | 98 | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 201,901 | -727 | -19,288 | 96 | 20 | O-18 | 162 | ALASKA KNIGHT | 2019-06-29 06:54:00 | 59.68079 | -168.6144 | 59.65546 | -168.6178 | 5.1 | 7.8 | 39 | 2.821 | 0.505 | 15.682 | 2.227 | 0.044239 | 0 |
| -19,252 | 21,720 | 413.4828 | 248.0897 | 12 | 20.00 | 1 | 2,019 | EBS | eastern Bering Sea | 98 | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 201,901 | -727 | -19,252 | 76 | 31 | G-03 | 162 | ALASKA KNIGHT | 2019-06-24 15:52:02 | 57.01591 | -166.4752 | 56.99137 | -166.4601 | 4.1 | 9.7 | 74 | 2.880 | 0.528 | 16.795 | 2.126 | 0.048370 | 0 |
| -18,731 | 21,720 | 946.3481 | 2,592.1327 | 118 | 43.08 | 1 | 2,019 | EBS | eastern Bering Sea | 98 | Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey | 201,901 | -726 | -18,731 | 11 | 31 | I-13 | 94 | VESTERAALEN | 2019-06-04 13:15:57 | 57.69052 | -160.2580 | 57.66518 | -160.2640 | 5.5 | 7.5 | 54 | 2.840 | 0.520 | 16.029 | 2.200 | 0.045522 | 0 |

### 18.0.5 Ex. Find all species found in the eastern Bering Sea (EBS) survey in 2023

# Pull data  
data <- RODBC::sqlQuery(  
channel = channel,   
query =   
"SELECT DISTINCT   
ss.COMMON\_NAME,  
ss.SCIENTIFIC\_NAME,   
ss.ID\_RANK,   
ss.WORMS  
FROM GAP\_PRODUCTS.FOSS\_CATCH cc -- get species codes  
LEFT JOIN GAP\_PRODUCTS.FOSS\_SPECIES ss -- get species info  
ON cc.SPECIES\_CODE = ss.SPECIES\_CODE  
LEFT JOIN GAP\_PRODUCTS.FOSS\_HAUL hh -- filter by year and survey  
ON cc.HAULJOIN = hh.HAULJOIN  
WHERE hh.YEAR = 2023  
AND hh.SURVEY\_DEFINITION\_ID = 98 -- EBS survey  
ORDER BY COMMON\_NAME")  
  
flextable::flextable(data[1:3,]) %>%   
 # flextable::fit\_to\_width(max\_width = 6) %>%   
 flextable::theme\_zebra()

| **COMMON\_NAME** | **SCIENTIFIC\_NAME** | **ID\_RANK** | **WORMS** |
| --- | --- | --- | --- |
| Alaska great-tellin | Megangulus luteus | species | 423,511 |
| Alaska plaice | Pleuronectes quadrituberculatus | species | 254,564 |
| Alaska razor | Siliqua alta | species | 413,689 |

# 19. Open source code

## 19.1 R Packages

### 19.1.1 [akgfmaps R package](https://github.com/afsc-gap-products/akgfmaps)

Bttom trawl survey maps layers and plotting examples. **POC:** Sean Rohan

### 19.1.2 [coldpool R package](https://github.com/afsc-gap-products/coldpool)

Cold pool area and temperature data products for the Bering Sea. **POC:** Sean Rohan

### 19.1.3 [akfishcondition R package](https://github.com/afsc-gap-products/akfishcondition)

Groundfish morphometric condition indicators for fish in the Bering Sea, Aleutian Islands, and Gulf of Alaska. **POC:** Sean Rohan

### 19.1.4 [gapindex R package](https://github.com/afsc-gap-products/gapindex)

Calculation of Design-Based Indices of Abundance and Composition for AFSC GAP Bottom Trawl Surveys. **POC:** Zack Oyafuso and Margaret Siple

# 20. Production run notes

# 21. R Version Metadata

R version 4.4.0 (2024-04-24 ucrt)  
Platform: x86\_64-w64-mingw32/x64  
Running under: Windows 10 x64 (build 19045)  
  
Matrix products: default  
  
  
locale:  
[1] LC\_COLLATE=English\_United States.utf8   
[2] LC\_CTYPE=English\_United States.utf8   
[3] LC\_MONETARY=English\_United States.utf8  
[4] LC\_NUMERIC=C   
[5] LC\_TIME=English\_United States.utf8   
  
time zone: America/Los\_Angeles  
tzcode source: internal  
  
attached base packages:  
[1] stats graphics grDevices utils datasets methods base   
  
loaded via a namespace (and not attached):  
 [1] compiler\_4.4.0 fastmap\_1.2.0 cli\_3.6.2 tools\_4.4.0   
 [5] htmltools\_0.5.8.1 rstudioapi\_0.16.0 yaml\_2.3.8 rmarkdown\_2.27   
 [9] knitr\_1.47 jsonlite\_1.8.8 xfun\_0.44 digest\_0.6.35   
[13] rlang\_1.1.4 evaluate\_0.24.0

### 21.0.1 NOAA README

This repository is a scientific product and is not official communication of the National Oceanic and Atmospheric Administration, or the United States Department of Commerce. All NOAA GitHub project code is provided on an ‘as is’ basis and the user assumes responsibility for its use. Any claims against the Department of Commerce or Department of Commerce bureaus stemming from the use of this GitHub project will be governed by all applicable Federal law. Any reference to specific commercial products, processes, or services by service mark, trademark, manufacturer, or otherwise, does not constitute or imply their endorsement, recommendation or favoring by the Department of Commerce. The Department of Commerce seal and logo, or the seal and logo of a DOC bureau, shall not be used in any manner to imply endorsement of any commercial product or activity by DOC or the United States Government.

### 21.0.2 NOAA License

Software code created by U.S. Government employees is not subject to copyright in the United States (17 U.S.C. §105). The United States/Department of Commerce reserve all rights to seek and obtain copyright protection in countries other than the United States for Software authored in its entirety by the Department of Commerce. To this end, the Department of Commerce hereby grants to Recipient a royalty-free, nonexclusive license to use, copy, and create derivative works of the Software outside of the United States.

# 22. Acknowledgments

# 23. Community Acknowledgments

We would like to thank the many communities of Alaska and their members who have helped contribute to this body of work. The knowledge, experiences, and insights have been instrumental in expanding the scope of our science and knowledge to encompass the many issues that face this important ecosystem. We appreciate feedback from those residing in the region that are willing to share their insights and participation in an open dialog about how we can improve our collective knowledge of the ecosystem and the region.

# 24. Land Acknowledgements

We would like to thank the many communities of the Bering Strait region and their members who have helped contribute to this document. The knowledge, experiences, and insights of the people of the Bering Strait region have been instrumental in expanding the scope of our science and knowledge to encompass the many issues that face this important ecosystem. We appreciate feedback from those residing in the region that are willing to share their insights, including the local names used for the species covered by this document, identifying species of interest or concern that should be included in this document, and participation in an open dialog about how we can improve our collective knowledge of the ecosystem and the region.

NOAA Fisheries Alaska Fisheries Science Center’s work is conducted in the waters and along the coastlines of Alaska, which include the traditional home lands and waters of the Inupiat, Yupiit, Siberian Yupiit, Unangax, Alutiiq/Sugpiaq, Eyak, Dena’ina Athabascan, Tlingit, Haida, and Tsimshian who have stewarded their lands and waters since time immemorial. We are indebted to these peoples for their wisdom and knowledge of their lands and waters.

This document was prepared in the greater Seattle area, which are the traditional lands of the Coast Salish people, including the Duwamish people, past and present. We are grateful for their continued sharing of vision, wisdom, values, and leadership.

# 25. Technical Acknowledgments

This quarto book is based off the [NOAA-quarto-book](https://github.com/nmfs-opensci/NOAA-quarto-book) GitHub repo designed by Eli Holmes.

This repo and GitHub Action was based on the tutorial by Openscapes [quarto-website-tutorial](https://github.com/Openscapes/quarto-website-tutorial) by Julia Lowndes and Stefanie Butland.

## 25.1 Partners

Scientists from the Alaska Fisheries Science Center conduct these bottom trawl surveys with participation from the Alaska Department of Fish & Game (ADF&G), the International Pacific Halibut Commission (IPHC), and universities. This research is conducted on chartered fishing vessels.

## 25.2 Collaborators

Our data are used in many annual publications, including but not limited to the list below:

* [Alaska Stock Assessments](https://www.fisheries.noaa.gov/alaska/population-assessments/alaska-stock-assessments)
* [North Pacific Groundfish Stock Assessment and Fishery Evaluation Reports](https://www.fisheries.noaa.gov/alaska/population-assessments/north-pacific-groundfish-stock-assessment-and-fishery-evaluation)
* [Groundfish Economic Status Reports for the Gulf of Alaska and Bering Sea and Aleutian Islands](https://www.fisheries.noaa.gov/alaska/commercial-fishing/groundfish-economic-status-reports-gulf-alaska-and-bering-sea-and-aleutian-islands)
* [Alaska Marine Ecosystem Status Report Database](https://www.fisheries.noaa.gov/resource/data/alaska-marine-ecosystem-status-report-archive)
* [Southeast Alaska Coastal Monitoring Survey Reports](https://www.fisheries.noaa.gov/alaska/commercial-fishing/southeast-alaska-coastal-monitoring-survey-reports)
* [Alaska Fisheries Life History Database](https://www.fisheries.noaa.gov/resource/data/alaska-fisheries-life-history-database)
* [Essential Fish Habitat Research Plan in Alaska](https://www.fisheries.noaa.gov/alaska/habitat-conservation/essential-fish-habitat-research-plan-alaska)

# 26. References

Hoff, G. R. (2016). *Results of the 2016 eastern Bering Sea upper continental slope survey of groundfishes and invertebrate resources* (NOAA Tech. Memo. NOAA-AFSC-339). U.S. Dep. Commer. <https://doi.org/10.7289/V5/TM-AFSC-339>

Markowitz, E. H., Dawson, E. J., Wassermann, S., Anderson, A. B., Rohan, S. K., Charriere, B. K., and Stevenson, D. E. (In prep). *Results of the 2023 eastern and northern Bering Sea continental shelf bottom trawl survey of groundfish and invertebrate fauna* [NOAA Tech. Memo.]. U.S. Dep. Commer.

Von Szalay, P. G., Raring, N. W., Siple, M. C., Dowlin, A. N., Riggle, B. C., and Laman, E. A. and. (2023). *Data report: 2022 Aleutian Islands bottom trawl survey* (AFSC Processed Rep. No. 2023-07; p. 230). U.S. Dep. Commer. <https://doi.org/10.25923/85cy-g225>