NOAA Technical Memorandum NMFS-XXX-##

GAP Production Data Documentation

Bering Sea Survey Team, Gulf of Alaska Survey Team, Aleutian Islands Survey Team

U.S. DEPARTMENT OF COMMERCE

National Oceanic and Atmospheric Administration
National Marine Fisheries Service
Northwest Fisheries Science Center

GAP Production Data Documentation

Bering Sea Survey Team[1,*], Gulf of Alaska Survey Team[1,*] and Aleutian Islands Survey Team[1,*]

1. NOAA Fisheres Alaska Fisheries Science Center, Groundfish Assessment Program

* *Correspondence:* Bering Sea Survey Team nmfs.afsc.gap.metadata@noaa.gov * *Correspondence:* Gulf of Alaska Survey Team nmfs.afsc.gap.metadata@noaa.gov * *Correspondence:* Aleutian Islands Survey Team nmfs.afsc.gap.metadata@noaa.gov

Table of contents

List of Figures

List of Tables

Part I.

Welcome

Report run date: Monday, February 16, 2026

AFSC Bottom Trawl Surveys

AFSC bottom trawl surveys are conducted by the AFSC's Groundfish Assessment Program and Shellfish Assessment Program and are conducted in the Gulf of Alaska, Aleutian Islands, Eastern Bering Sea Slope, Eastern Bering Sea Shelf, and Northern Bering Sea. Each survey is a multispecies survey that collects data on the distribution, abundance, and biological characteristics of fish, crab, and other resources to inform groundfish stock assessment and management. These fishery-independent surveys are conducted in the summer aboard contracted commercial fishing vessels. Specifics regarding each of the surveys can be found below.

Figure 1.: Sorting and weighing fish on deck on the 2022 Bering Sea groundfish survey aboard the F/V Alaska Knight. Credit: Emily Markowitz/NOAA Fisheries.

Documentation Objective

As part of our commitment to open science, reproducibility, and transparency, we provide this metadata guide to compliment our public-domain data.

> Please consider this resource to be a **Living Document**. The code in this repository is regularly being updated and improved. Please refer to releases for finalized products and project milestones.
>
> At this time, these master production and AKFIN tables are **provisional and we are welcoming feedback before the 2024 survey season.** We look forward to hearing from you. Do not hesitate to reach out (to us at either nmfs.afsc.gap.metadata@noaa.gov or GitHub issues, especially if you find discrepancies in the data or want to suggest improvements to infrastructure. Thank you in advance for your collaboration and partnership with us as we develop our future data universe.

User Resources

- Groundfish Assessment Program Bottom Trawl Surveys
- AFSC's Resource Assessment and Conservation Engineering Division
- All AFSC Research Surveys
- Survey code books
- Publications and Data Reports
- Research Surveys conducted at AFSC

Cite this data

Use the below bibtext citations, as cited in our group's citation repository for citing the data created and maintained in this repo. Add "note = {Accessed: mm/dd/yyyy}" to append the day this data was accessed. Included here are AFSC RACE Groundfish and Shellfish Assessment Program's:

- Design-Based Production Data (internal) (NOAA Fisheries Alaska Fisheries Science Center, Goundfish Assessment Program, 2024).

Cite this data

- AFSC RACE Groundfish Data for AKFIN (Alaska Fisheries Information Network (AKFIN), 2024).
- Public Data hosted on the Fisheries One Stop Shop (FOSS) Data Platform (NOAA Fisheries Alaska Fisheries Science Center, 2024).

```
@misc{GAPProducts,
  author = {{NOAA Fisheries Alaska Fisheries Science Center, Goundfish Assessment Program}}
  year = {2023},
  title = {AFSC Goundfish Assessment Program Design-Based Production Data},
  howpublished = {https://www.fisheries.noaa.gov/alaska/science-data/groundfish-
assessment-program-bottom-trawl-surveys},
  publisher = {{U.S. Dep. Commer.}},
  copyright = {Public Domain}
}

@misc{FOSSAFSCData,
  author = {{NOAA Fisheries Alaska Fisheries Science Center}},
  year = {2023},
  title = {Fisheries One Stop Shop Public Data: RACE Division Bottom Trawl Survey Data Quer
  howpublished = {https://www.fisheries.noaa.gov/foss},
  publisher = {{U.S. Dep. Commer.}},
  copyright = {Public Domain}
}

@misc{GAPakfin,
  author = {{Alaska Fisheries Information Network (AKFIN)}},
  institution = {{NOAA Fisheries Alaska Fisheries Science Center, Goundfish Assessment Prog
  year = {2023},
  title = {AFSC Goundfish Assessment Program Design-Based Production Data},
  howpublished = {https://www.psmfc.org/program/alaska-fisheries-information-
network-akfin},
  publisher = {{U.S. Dep. Commer.}},
  copyright = {Public Domain}
}
```

Access Constraints

There are no legal restrictions on access to the data. They reside in public domain and can be freely distributed.

User Constraints: Users must read and fully comprehend the metadata and code of conduct prior to use. Data should not be used beyond the limits of the source scale. Acknowledgement of AFSC Groundfish Assessment Program, as the source from which these data were obtained, in any publications and/or other representations of these data, is suggested.

Suggestions and comments

If the data or metadata can be improved, please create a pull request, submit an issue to the GitHub organization or submit an issue to the code's repository.

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.

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.

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. 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

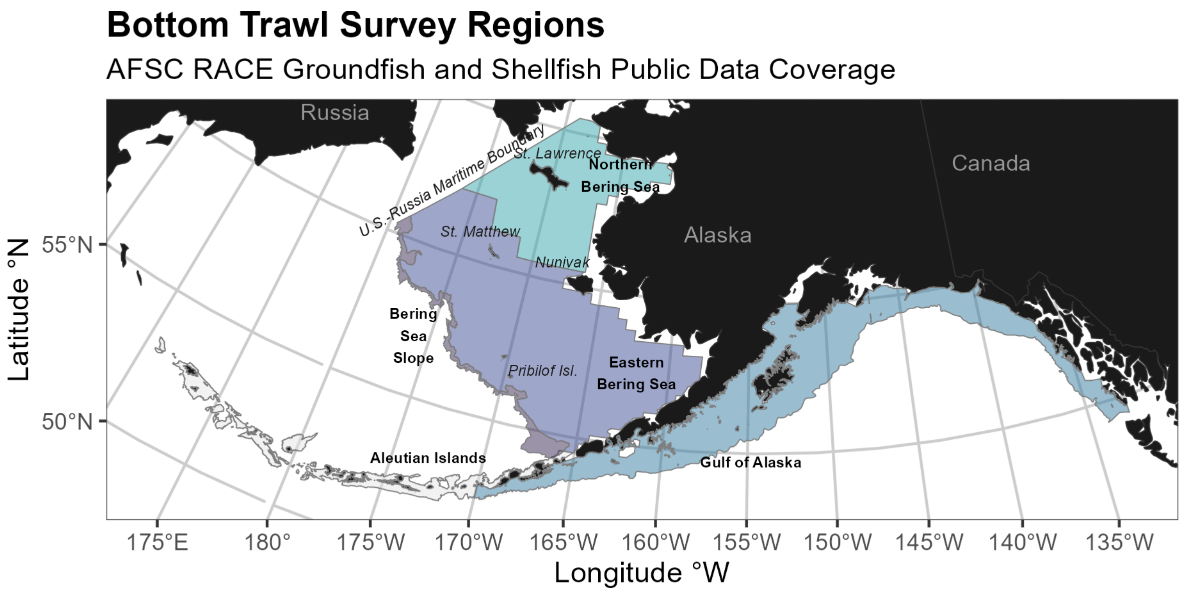


Figure 1.1.: Strata used in the all surveys.

Each survey conducted by the Groundfish Assessment Program are multispecies bottom trawl surveys. We collect environmental and biological data to assess how climate variability and loss of sea 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 to determine fish ages. We use this information in annual stock assessments and to assess the state of the ecosystem. This research is conducted on fishing industry contract vessels.

Table 1.1.: Survey summary stats

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

Survey	Survey Definition ID	Years	Depth (m)	Area (km2)	# Statistical Areas	# Possible Stations
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	2025 - 1982 (43)	1 - 200	492,989.9	28	515
Gulf of Alaska Bottom Trawl Survey	47	2025 - 1990 (17)	1 - 1,000	315,501.9	37	6,939
Northern Bering Sea Crab/Groundfish Survey - Eastern Bering Sea Shelf Survey Extension	143	2025 - 2010 (7)	1 - 100	189,191.4	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, walleye pollock, Pacific cod, sablefish, and other rockfish species.

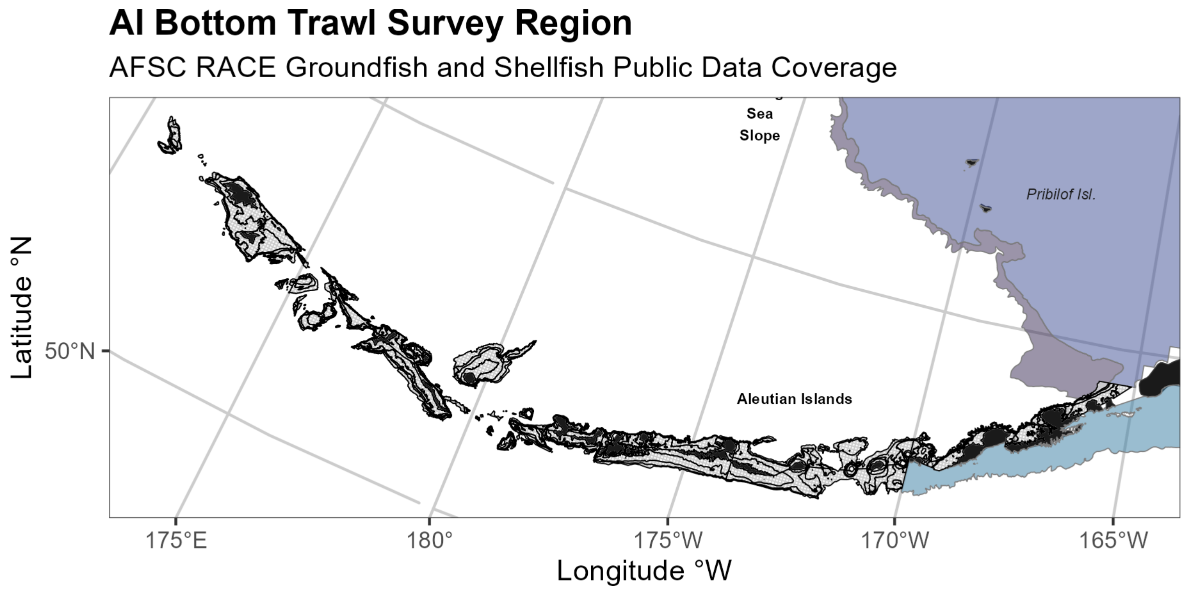


Figure 1.2.: Strata used in the Aleutian Islands bottom trawl survey.

1.5.2. Gulf of Alaska

Most recent data report: (Siple et al., 2024)

- 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, walleye pollock, Pacific cod, flatfish, and other rockfish species.

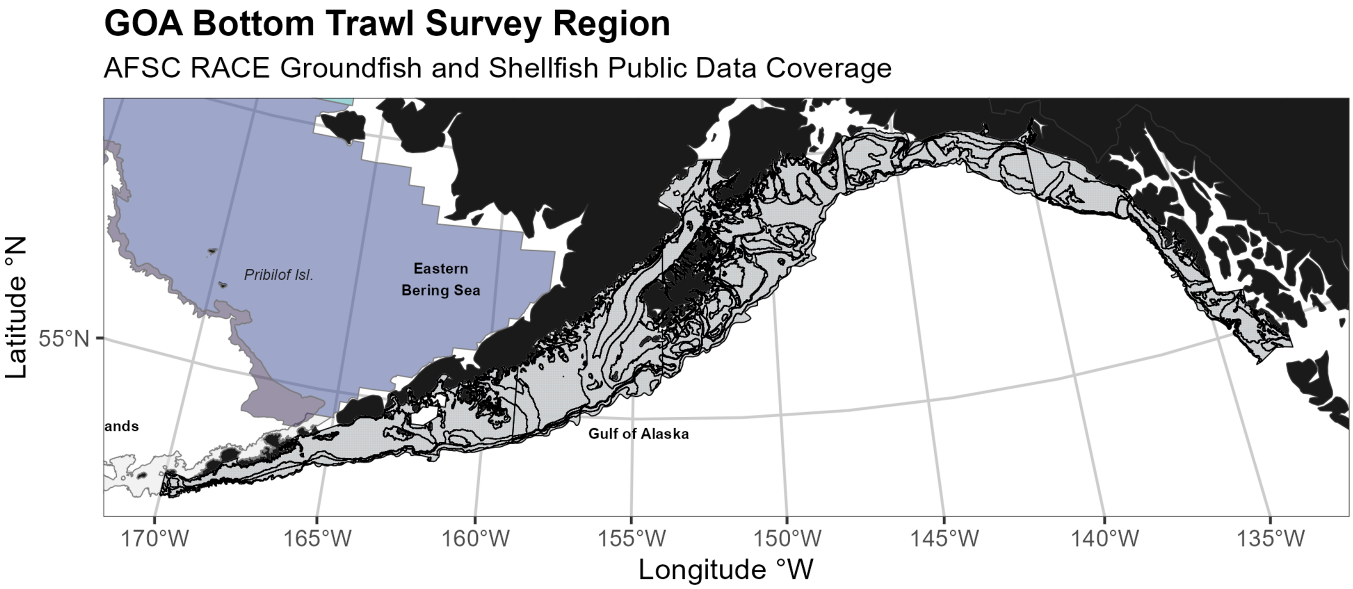


Figure 1.3.: Strata used in the Gulf of Alaska bottom trawl survey.

1.5.3. Eastern Bering Sea Shelf

Most recent data report: (Markowitz et al., 2025)

- 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, Pacific cod, Greenland turbot, yellowfin sole, northern rock sole, red king crab, and snow and Tanner crabs.

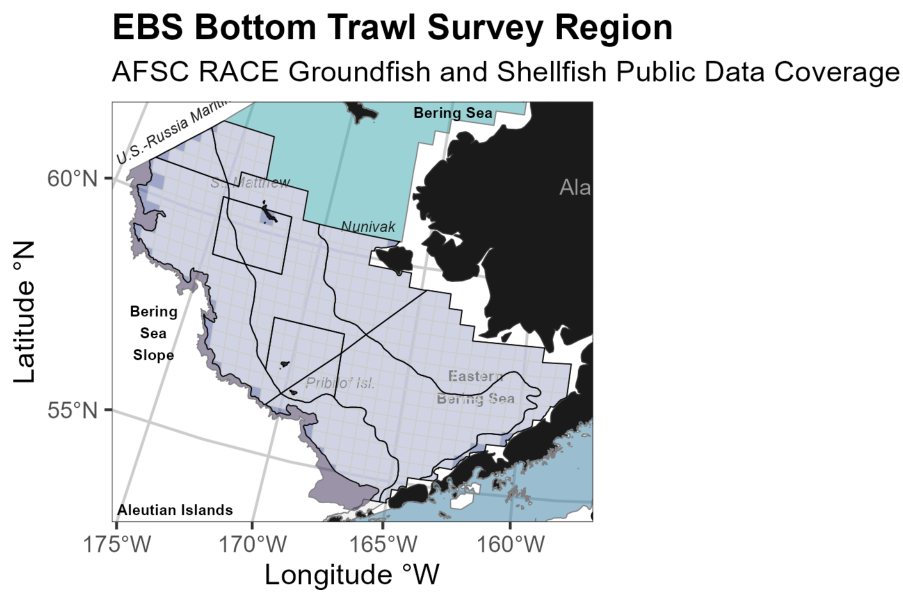


Figure 1.4.: Strata used in the Eastern Bering Sea bottom trawl survey.

1.5.4. Northern Bering Sea

Most recent data report: (Markowitz et al., 2024)

- 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.

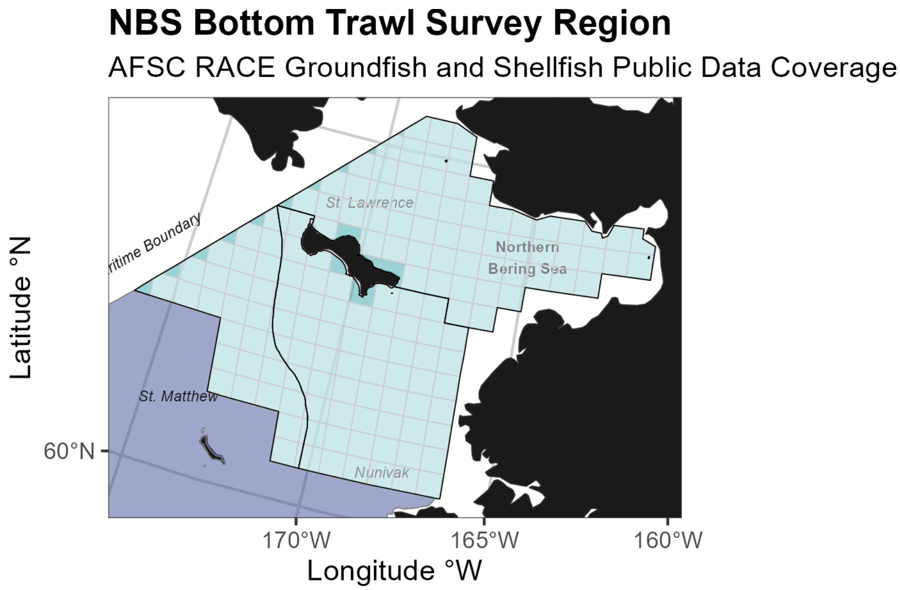


Figure 1.5.: Strata used in the Northern Bering Sea bottom trawl survey.

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, popeye grenadier, walleye pollock, and arrowtooth flounder.

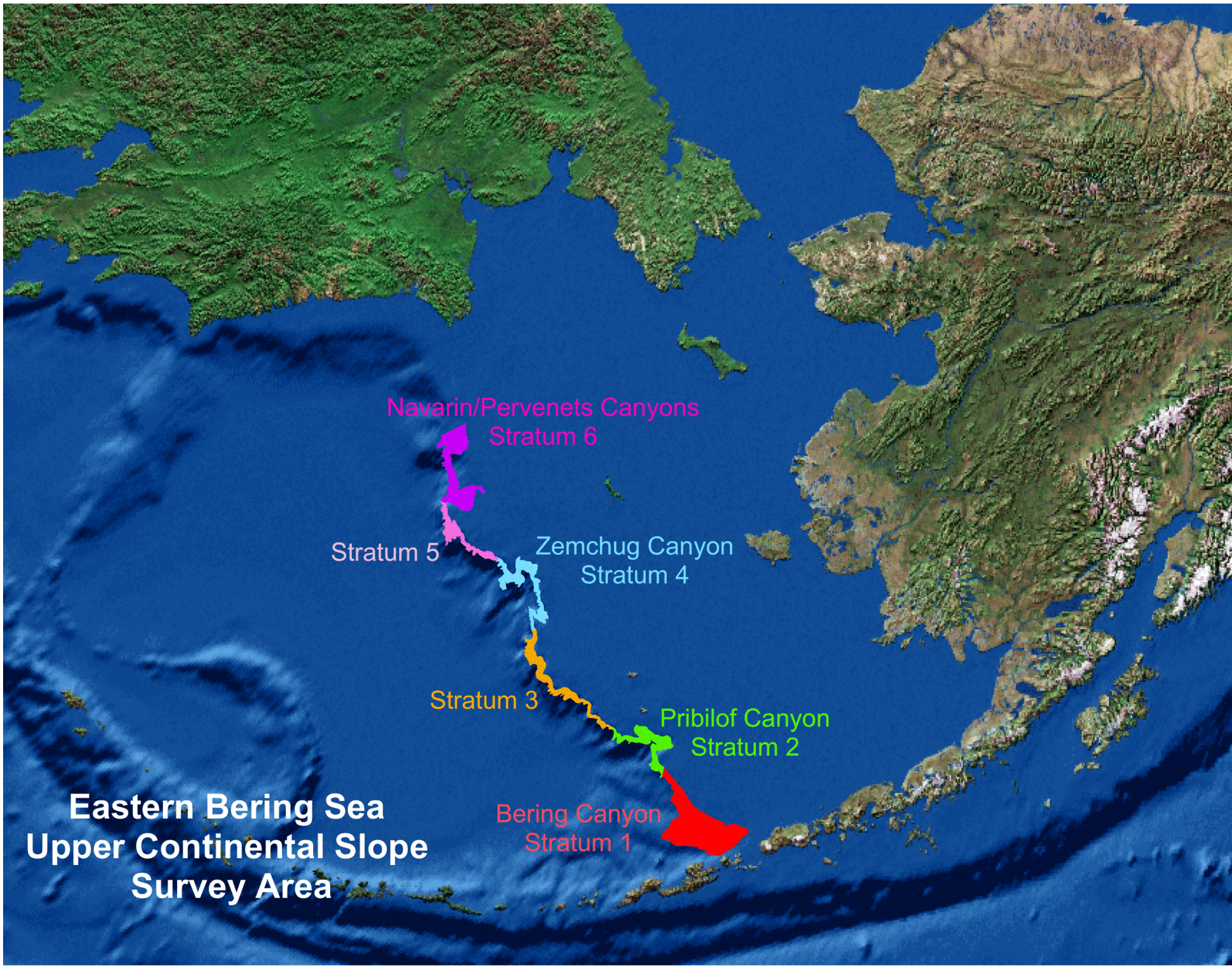


Figure 1.6.: Strata used in the Bering Sea Slope bottom trawl survey.

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.

Table 2.1.: Operational product development timeline.

	January	February	March	April	May	June	July	August
Surveys								
Planning								
Development								
Deployment (survey deliverables)								
Deployment (survey operations)								
Triage (fixing bugs and errors)								

	January	February	March	April	May	June	July	August
User feedback and brain-storming								

2.2. Data workflow from boat to production

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

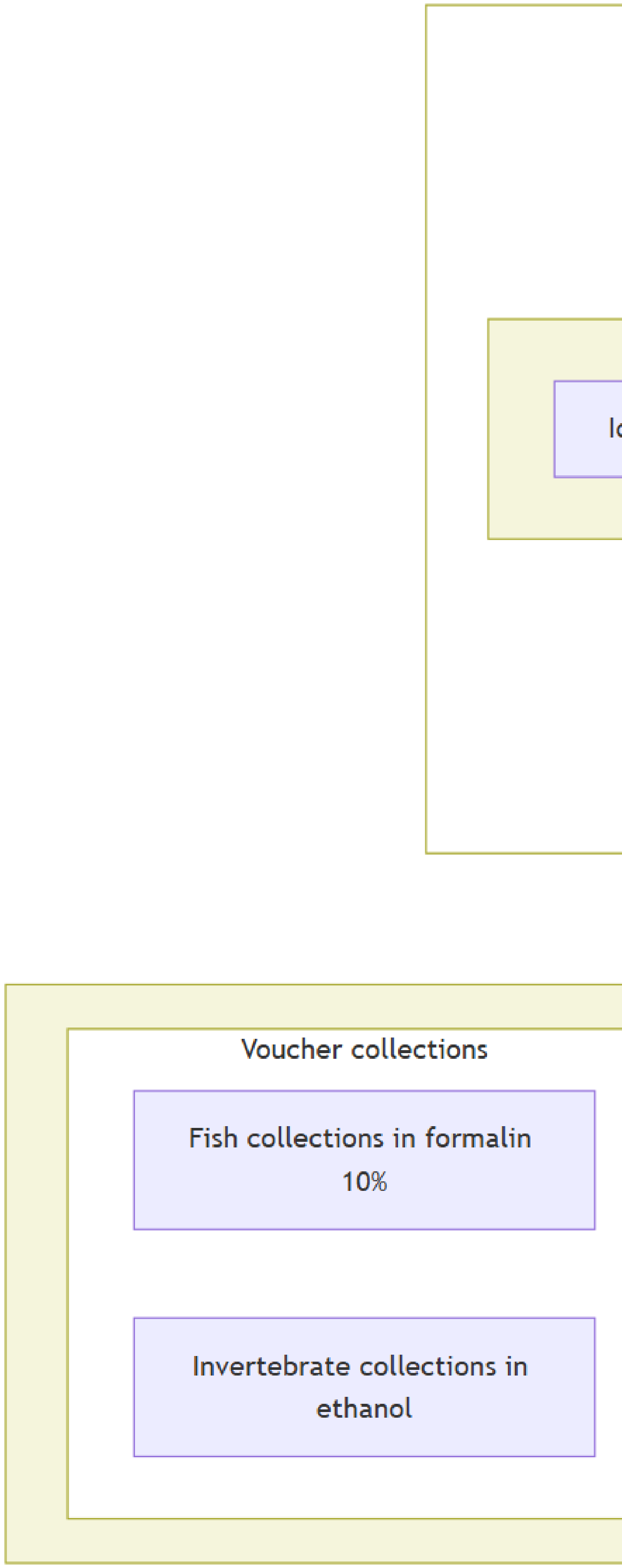
The objective of this process is to take raw data, QA/QC and clean these data, curate standard data products for these survey. Please note, through this process we are not providing "data" (what we consider lower level data material; see the data levels section below) but "data products", which is intended 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` 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**.

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


Voucher collections
Fish collections in formalin 10%
Invertebrate collections in ethanol

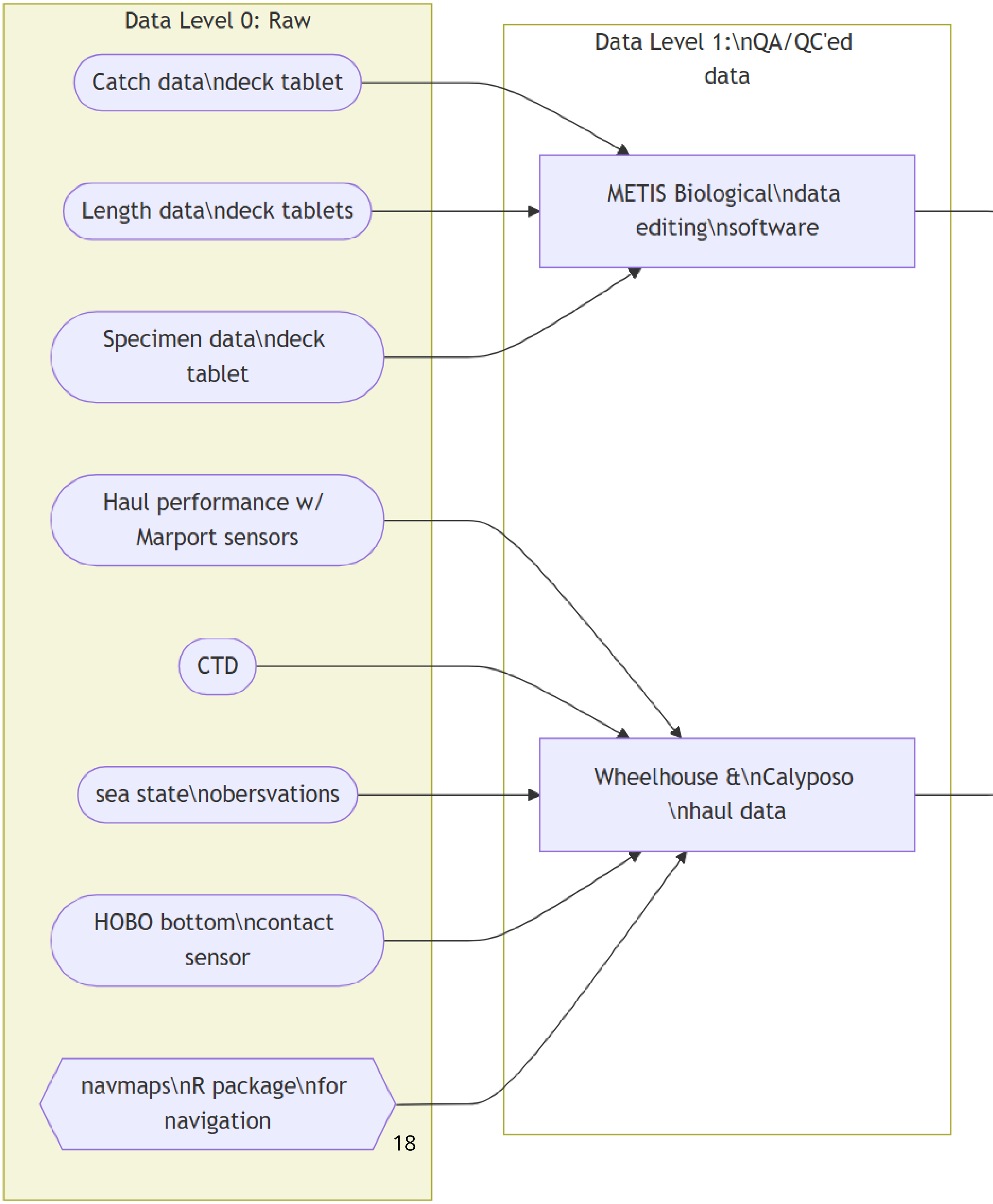


Figure 2.2.: Simplified data workflow from boat to production.

to a production table will be compared and checked to make sure those changes are intentional and documented.

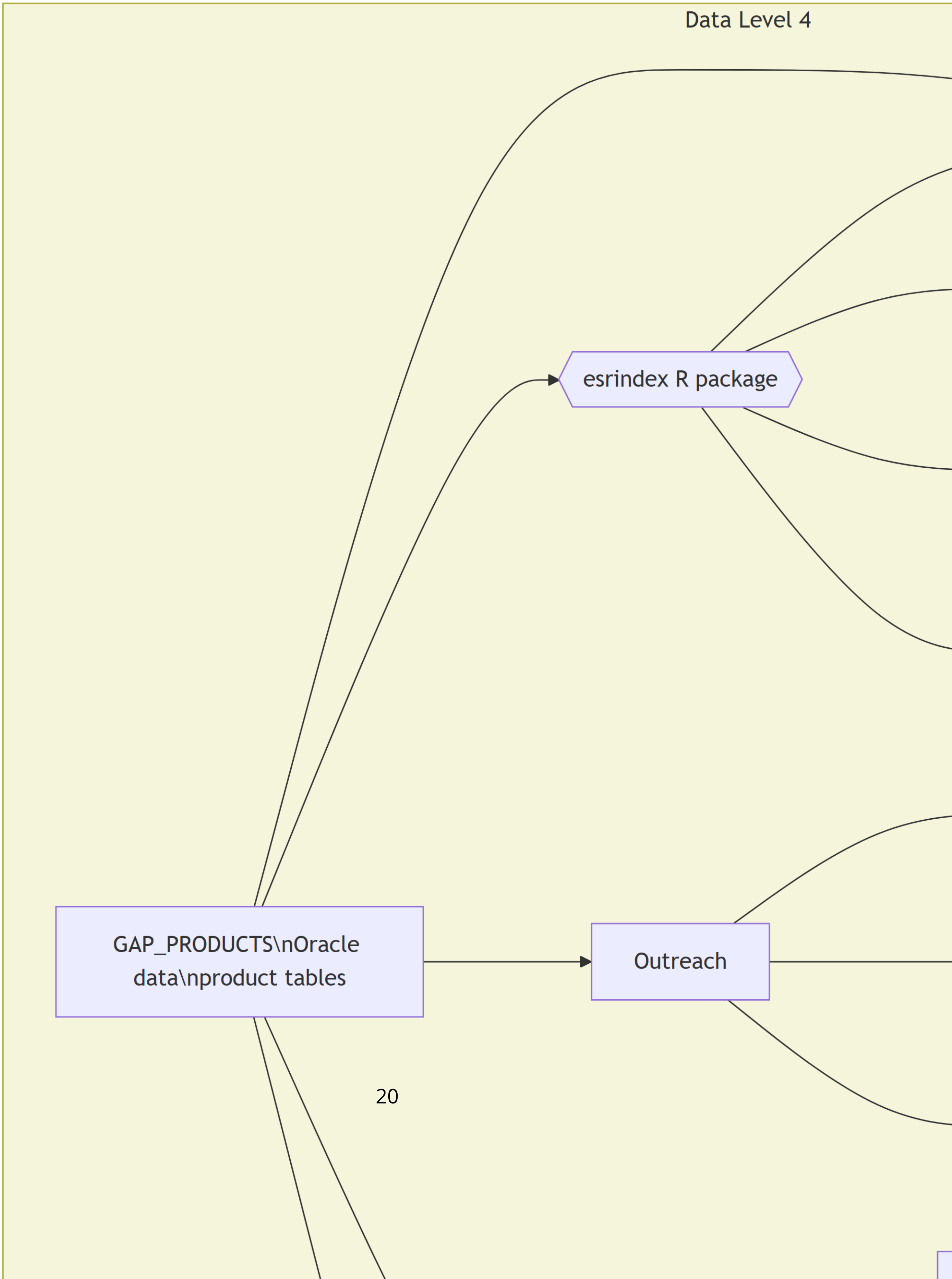
2. Use the `gapindex` R package to calculate the four major standard data products: CPUE, BIOMASS, SIZECOMP, AGECOMP. These tables are compared and checked to their respective locally saved copies and any changes to the tables are vetted and documented. These tables are then uploaded to the GAP_PRODUCTS 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.

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 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 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


Data Level 4
esrindex R package
GAP_PRODUCTS\nOracle data\nproduct tables
Outreach

- **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 2025-09-29) using gapindex v3.0.3: Run completed by: Duane Stevenson

– GAP_PRODUCTS ChangeLog (last produced on 2025-09-22) using gapindex v3.0.3: Run completed by: Duane Stevenson

– GAP_PRODUCTS ChangeLog (last produced on 2025-08-25) using gapindex v3.0.2: Run completed by: Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2025-07-15) using gapindex v2.2.0: Run completed by: Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2025-06-05) using gapindex v3.0.2: Run completed by: Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2025-05-20) using gapindex v3.0.2: Run completed by: Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2025-04-30) using gapindex v3.0.2: Run completed by: Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2025-03-20) using gapindex v3.0.2: Run completed by: Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2024-12-10) using gapindex v3.0.2: Run completed by: Sean Rooney

– GAP_PRODUCTS ChangeLog (last produced on 2024-10-22) using gapindex v2.2.0: Run completed by: Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2024-10-21) using gapindex v2.2.0: Run completed by: Duane Stevenson, Ned Laman, Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2024-09-05) using gapindex v2.2.0: Run completed by: Ned Laman, Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2024-09-03) using gapindex v2.2.0: Run completed by: Ned Laman, Zack Oyafuso

– GAP_PRODUCTS ChangeLog (last produced on 2024-08-29) using gapindex v2.2.0: 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: 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: A development branch version of gapindex called 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: A development branch version of gapindex called 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: A new version of gapindex 2.2.0 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. 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: A new version of gapindex (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: A new version of gapindex (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: 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: 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: 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 from the nmfs-opensci GitHub.

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, version 1.4, available at https://contributor-covenant.org/version/1/4

Part II.

GAP Production Data

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. These data are created using the gapindex R package v2.1.0.

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. 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.

gapindex

Code to generate design-based catch-per-unit-effort (CPUE), indices of abundance, biomass, and size and age compositions from survey data is avaialble from gapindex. See the gapindex documentation for more information. Make sure you have installed R packages devtools, RODBC, and getPass and are connected to the AFSC network or VPN while using this package.

Cite this data

Use the below bibtext citation, as cited in our group's citation repository for citing the data created and maintained in this repository. Add "note = {Accessed: mm/dd/yyyy}" to append the day this data was accessed.

```
[1] "@misc{GAPProducts,"
[2] "  author = {{NOAA Fisheries Alaska Fisheries Science Center, Goundfish Assessment Prog
[3] "  year = {2024}, "
```

```
[4] "  title = {AFSC Goundfish Assessment Program Design-Based Production Data},"
[5] "  howpublished = {https://www.fisheries.noaa.gov/alaska/science-data/groundfish-
assessment-program-bottom-trawl-surveys},"
[6] "  publisher = {{U.S. Dep. Commer.}},"
[7] "  copyright = {Public Domain} "
[8] "}"
```

6. Data description

6.1. Data tables

6.1.1. AGECOMP

Stratum- and region-level age compositions by sex. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Number of rows: 680,450

Number of columns: 10

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

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

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.

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.

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.

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.

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. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Number of rows: 499

Number of columns: 9

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.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

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.

DESCRIPTION

Description

text

VARCHAR2(4000 BYTE)

Description of row observation.

AREA_KM2

Area (km2)

kilometers squared

NUMBER(38,3)

Area in square kilometers.

DEPTH_MIN_M

Area ID minimum depth (m)

meters

NUMBER(38,3)

Minimum depth (meters).

DEPTH_MAX_M

Area ID maximum depth (m)

meters

NUMBER(38,3)

Maximum depth (meters).

6.1.3. BIOMASS

Stratum/subarea/region-level mean CPUE (weight and numbers), total biomass, and total abundance with associated variances. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Number of rows: 2,655,773

Number of columns: 16

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

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.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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_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).

N_HAUL

Valid hauls

count

NUMBER(38,0)

Total number of hauls.

N_WEIGHT

Hauls with catch

count

NUMBER(38,0)

Total number of hauls with positive catch biomass.

N_COUNT

Hauls with taxon counts

numeric

NUMBER(38,0)

Total number of hauls with positive count data.

N_LENGTH

Hauls with taxon lengths

count

NUMBER(38,0)

Total number of hauls with length data.

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.

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.

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_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).

6.1.4. CPUE

Haul-level zero-filled weight and numerical catch-per-unit-effort. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Number of rows: 21,915,584

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

WEIGHT_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

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

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

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

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.

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).

6.1.5. SIZECOMP

Stratum/subarea/region-level size compositions by sex. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Number of rows: 3,234,183

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

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.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

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

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.

6.1.6. SPECIES_YEAR

This is a table

Number of rows: 18

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.

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. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_-products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Number of rows: 1,063

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)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

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.

6.1.8. 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. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_-products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Number of rows: 87

Number of columns: 3

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

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.

6.1.9. TAXON_GROUPS

GAP_PRODUCTS.TAXONOMIC_CLASSIFICATION subsetted for taxonomic classifications accepted by the GAP bottom trawl survey and added GROUP_CODE to denote taxonomic aggregations. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_products). There are no legal restrictions on access to the data. Last updated on 25 October 2024.

Number of rows: 2,777

Number of columns: 22

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.

SPECIES_NAME

Scientific name of species

text

VARCHAR2(255 BYTE)

Scientific name of 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.

ID_RANK

Lowest taxonomic rank

text

VARCHAR2(255 BYTE)

Lowest taxonomic rank of a given species entry.

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.

GENUS_TAXON

Genus phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of genus of a given species.

SUBFAMILY_TAXON

Subfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subfamily of a given species.

FAMILY_TAXON

Family phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of family of a given species.

SUPERFAMILY_TAXON

Superfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superfamily of a given species.

INFRAORDER_TAXON

Infraorder phylogenetic rank

category

VARCHAR2(255 BYTE)

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

SUBORDER_TAXON

Suborder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of suborder of a given species.

ORDER_TAXON

Order phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of order of a given species.

SUPERORDER_TAXON

Superorder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superorder of a given species.

INFRACLASS_TAXON

Infraclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Infraclass phylogenetic rank. Phylogenetic latin rank of infraclass of a given speices.

SUBCLASS_TAXON

Subclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subclass of a given species.

CLASS_TAXON

Class phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of class of a given species.

SUPERCLASS_TAXON

Superclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superclass of a given species.

SUBPHYLUM_TAXON

Subphylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subphylum of a given species.

PHYLUM_TAXON

Phylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of phylum of a given species.

KINGDOM_TAXON

Kingdom phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of kingdom of a given species.

GROUP_CODE

Species or Complex ID

ID key code

NUMBER(38,0)

Equivalent to the SPECIES_CODE if the taxon is reported as a single taxon in GAP_-PRODUCTS, otherwise denotes a SPECIES_CODE of a higher taxonomic group to which the taxon is aggregated in the GAP_PRODUCTS CPUE and BIOMASS tables.

7. Universal Column Metadata

This table is used to string together the various field comments for the tables in GAP_-PRODUCTS. This table was created by the Resource Assessment and Conservation Engineering Division (RACE) Groundfish Assessment Program (GAP) of the Alaska Fisheries Science Center (AFSC). The GitHub repository for the scripts that created this code can be found at (https://github.com/afsc-gap-products/gap_products). There are no legal restrictions on access to the data. Last updated on 22 September 2025.

Table 7.1.: Universal stock metadata that users can use to document their table columns.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
ABUNDANCE_HAUL	Design-based index approved haul	logical	VARCHAR2(255 BYTE)	Logical, describing if this haul was conducted in a standard manner and thus used for design-based index estimates (TRUE) or not (FALSE).

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).
ACTION	Database action	text	VARCHAR2(255 BYTE)	Standard action taken to alter current database record

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
ACTIVE	Vessel active/inactive	logical	VARCHAR2(255 BYTE)	Logical, describing if a vessel is active (TRUE) or not (FALSE).

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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
AGENCY_ACRONYM	Acronym of listed Agency	text abbreviated	VARCHAR2(255 BYTE)	Abbreviated agencies that are affiliated with the Alaska bottom trawl survey. The column agency_acronym is associated with the agency_short and agency_long columns.
AGENCY_JOIN	Agency ID	ID key code	NUMBER(38,0)	Affiliated agency ID key code.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
AGENCY_LONG	Official name of agency	text	VARCHAR2(255 BYTE)	Full official name of affiliated agencies to the Alaska bottom trawl survey. The column agency_long is associated with the agency_acronym and agency_short columns.

Column name from data	Descrip-tive column Name	Units	Oracle data type	Column descrip-tion
AGENCY_-SHORT	Agency shorthand name	text	VAR-CHAR2(255 BYTE)	A sort version of the full official name of affiliated agencies to the Alaska bottom trawl survey. The column agency_-short is associated with the agency_-acronym and agency_-long columns.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).
AGE_DETERMINATION_METHODS	Age determination method	ID key code	NUMBER(38,0)	A unique ID used to identify this age determination method.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
AGE_YEAR	Age bin of taxon	year	NUMBER(38,0)	Age bin of a taxon in years estimated by the age comp estimate.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
AREA-JOIN	Area polygon ID	ID key code	NUMBER(38,0)	A call sign is a designated sequence of letters and numbers that are assigned when a vessel, whether it be a sailing yacht, motor yacht, rib or commercial vessel, receives its Ship Radio Licence. The vessel also receives its MMSI number, so that each vessel is uniquely identified.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
AREA_ID	Area ID	ID key code	NUMBER(38,0)	An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
AREA_ID_FOOTPRINT	Survey Footprint	text	VARCHAR2(40(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.
AREA_KM2	Area (km2)	kilometers squared	NUMBER(38,3)	Area in square kilometers.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
AREA_NAME	Area ID name	text	VARCHAR2(40(BYTE)	Descriptive name of each AREA_ID. These names often identify the region, depth ranges, or other regional information for the area ID.
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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.
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).
CATALOG_NUM	Catalog number	text	VARCHAR2(255 BYTE)	Museum catalog number associated with record

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
CATCHJOII	Catch observation ID	ID key code	NUMBER(38,0)	Unique integer ID assigned to each survey, vessel, year, and catch observation combination.
CLASSIFICATION	Taxonomic classification rank group	category	VARCHAR2(255 BYTE)	Phylogenetic classification group rank for a given species.
CLASS_TAXON	Class phylogenetic rank	category	VARCHAR2(25 BYTE)	Phylogenetic latin rank of class of a given species.
COLLECTED_BY	Person who collected specimen	text	VARCHAR2(255 BYTE)	Initials of person who collected specimen in the field
COMMENTS	Comments	text	VARCHAR2(40 BYTE)	Comments regarding row observation.

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).

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.
COUNTRY_ID	Country code	ID key code	NUMBER(38,0)	Country ID key code of where a vessel, for example, may be from. For a complete list of country ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
CPUE_KGHA	Weight CPUE (kg/ha)	kilograms per hectare	NUMBER(38,6)	Catch weight (kilograms) per unit effort (area swept by the net, units hectares).
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_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).

Column name from data	Descrip-tive column Name	Units	Oracle data type	Column descrip-tion
CPUE_-KGKM2_-VAR	Variance of the mean weight CPUE	kilograms per kilometers squared	NUM-BER(38,6)	The variance of mean catch weight (kilo-grams) per unit effort (area swept by the net, units squared kilome-ters).
CPUE_-NOHA	Number CPUE (no/ha)	count per hectare	NUM-BER(38,6)	Numerical catch per unit effort (area swept by the net, units hectares).
CPUE_-NOKM2	Number CPUE (no/km2)	count per kilometers squared	NUM-BER(38,6)	Numerical catch per unit effort (area swept by the net, units square kilome-ters).

Column name from data	Descrip-tive column Name	Units	Oracle data type	Column descrip-tion
CPUE_-NOKM2_-MEAN	Mean numeric CPUE	count per kilometers squared	NUM-BER(38,6)	The mean of numerical catch per unit effort (area swept by the net, units square kilome-ters).
CPUE_-NOKM2_-VAR	Variance of the mean numeric CPUE	count per kilometers squared	NUM-BER(38,6)	The variance of mean numerical catch per unit effort (area swept by the net, units square kilome-ters).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
CRS	Coordinate reference system	ID key code	VARCHAR2(255 BYTE)	The coordinate reference system (CRS) that shapefiles were created in or areas (like AREA_KM2) are calculated in, as defined by https://spatialreference.org/ (e.g., ”+proj=longlat”, ”EPSG:3338”).

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.

Column name from data	Descrip-tive column Name	Units	Oracle data type	Column descrip-tion
DATABASE	Database source	category	VAR-CHAR2(255 BYTE)	Taxonomic database source, either ITIS or WoRMS.
DATABASE ID	Species ID in database	ID key code	VAR-CHAR2(255 BYTE)	Species ID key code of a species in the taxonomic ”DATABASE” source.
DATE	Date	YYYY-MM-DD	DATE	The date (YYYY-MM-DD) of the event (e.g., cruise).
DATE_-END	End date	YYYY-MM-DD	DATE	The date (YYYY-MM-DD) of the end of the event (e.g., cruise).

Column name from data	Descrip-tive column Name	Units	Oracle data type	Column descrip-tion
DATE_-START	Start date	YYYY-MM-DD	DATE	The date (YYYY-MM-DD) of the beginning of the event (e.g., cruise).
DATE_-TIME	Date and time	MM/DD/YY 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 Anchor-age, AK, USA (UTC/GMT -8 hours).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
DATE_TIME_END	End date and time	MM/DD/YYYY HH::MM	TIMESTAMP	The date (MM/DD/YYYY) and time (HH:MM) of the end of the haul. All dates and times are in Alaska time (AKDT) of Anchorage, AK, USA (UTC/GMT -8 hours).
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
DEPTH_GEAR_M	Depth of gear (m)	meters	NUMBER(38,1)	Depth of gear (meters).
DEPTH_M	Depth (m)	meters	NUMBER(38,1)	Bottom depth (meters).
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
DISTANCE_FISHED_KM	Distance fished (km)	kilometers	NUMBER(38,3)	Distance the net fished (kilometers).
DUMMY	dummy	dummy	VARCHAR2(25 BYTE)	dummy.
DURATION_HR	Tow duration (decimal hr)	hours	NUMBER(38,1)	This is the elapsed time between start and end of a haul (decimal hours).
FAMILY_TAXON	Family phylogenetic rank	category	VARCHAR2(25 BYTE)	Phylogenetic latin rank of family of a given species.
FIELD_ID	Field specimen identification	text	VARCHAR2(255 BYTE)	Field identification for the vouchered specimen
FREQUENCY	Count of observation	count	NUMBER(38,0)	Frequency, or count, of an observation.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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_DEPTH_M	Gear depth	meters	NUMBER(38,1)	Depth gear was deployed at (tenths of a meter). Gear depth plus net height equals bottom depth.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
GEAR_ID	Gear ID	ID key code	NUMBER(38,0)	Type of trawl or gear deployed. For a complete list of vessel gear type ID key codes, review the [code books](https://www.fisheries.noaa.gov/resource/document/groundfish-survey-species-code-manual-and-data-codes-manual).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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.
GENUS_-TAXON	Genus phylogenetic rank	category	VARCHAR2(255 BYTE)	Phylogenetic latin rank of genus of a given species.
GEOMETRY	Spatial geometry	text	VARCHAR2(255 BYTE)	Spatial geometry information (like points, lines, or polygons) a feature.
GONAD_-G	Weight of gonads (g)	grams	NUMBER(38,1)	Weight of specimen gonads (grams).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
GROUP_CODE	Species or Complex ID	ID key code	NUMBER(38,0)	Equivalent to the SPECIES_CODE if the taxon is reported as a single taxon in GAP_PRODUCTS, otherwise denotes a SPECIES_CODE of a higher taxonomic group to which the taxon is aggregated in the GAP_PRODUCTS CPUE and BIOMASS tables.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).
ID_RANK	Lowest taxonomic rank	text	VARCHAR2(255 BYTE)	Lowest taxonomic rank of a given species entry.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
INFRACLASS_-TAXON	Infraclass phylogenetic rank	category	VARCHAR2(255 BYTE)	Infraclass phylogenetic rank. Phylogenetic latin rank of infraclass of a given speices.
INFRAORDER_-TAXON	Infraorder phylogenetic rank	category	VARCHAR2(255 BYTE)	Infraorder phylogenetic rank. Phylogenetic latin rank of infraorder of a given speices.
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/).
KINGDOM_-TAXON	Kingdom phylogenetic rank	category	VARCHAR2(255 BYTE)	Phylogenetic latin rank of kingdom of a given species.

Column name from data	Descrip-tive column Name	Units	Oracle data type	Column descrip-tion
LATI-TUDE_DD	Latitude (decimal degrees)	decimal degrees	NUM-BER(38,6)	Latitude (one hundred thou-sandth of a decimal degree).
LATI-TUDE_-DD_END	End latitude (decimal degrees)	decimal degrees	NUM-BER(38,6)	Latitude (one hundred thou-sandth of a decimal degree) of the end of the haul.
LATI-TUDE_-DD_-START	Start latitude (decimal degrees)	decimal degrees	NUM-BER(38,6)	Latitude (one hundred thou-sandth of a decimal degree) of the start of the haul.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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_MM_MEAN	Mean length at age weighted by numbers at length	numeric	NUMBER(38,3)	Mean length (millimeters).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
LENGTH_MM_SD	Standard deviation of length at age weighted by numbers at length	numeric	NUMBER(38,3)	Variance of mean length.
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
LONGITUDE_DD	Longitude (decimal degrees)	decimal degrees	NUMBER(38,6)	Longitude (one hundred thousandth of a decimal degree).
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.
MATURITY	Specimen maturity code	ID key code	NUMBER(38,0)	The maturity code or the condition identified by the maturity code.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
METADATA_COLNAME	Column name	text	VARCHAR2(40(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(40(BYTE)	Long name for the column.
METADATA_DATATYPE	Oracle datatype code	text	VARCHAR2(4000 BYTE)	Oracle data type of data column.
METADATA_SENTENCE	Sentence	text	VARCHAR2(40(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(40(BYTE)	Type of sentence to have in table metadata.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
METADATA_UNITS	Units	category	VARCHAR2(4000 BYTE)	Units of the column.
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.

Column name from data	Descrip-tive column Name	Units	Oracle data type	Column descrip-tion
NET_-WIDTH_M	Net width (m)	meters	NUM-BER(38,1)	Measured or estimated distance (meters) between wingtips of the trawl.
NEW_ID	New specimen identifica-tion	text	VAR-CHAR2(255 BYTE)	Confirmed taxonomist identifica-tion of the vouchered specimen
NEW_-SPECIES_-CODE	New species code	ID key code	NUM-BER(10,0)	Species code associated with new species name
NEW_-SPECIES_-NAME	New species name	text	VAR-CHAR2(255 BYTE)	Updated taxonomic name
N_COUNT	Hauls with taxon counts	numeric	NUM-BER(38,0)	Total number of hauls with positive count data.
N_HAUL	Valid hauls	count	NUM-BER(38,0)	Total number of hauls.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
N_-LENGTH	Hauls with taxon lengths	count	NUMBER(38,0)	Total number of hauls with length data.
N_-SAMPLE	Hauls with sample	count	NUMBER(38,0)	Total number of hauls with positive sample collection.
N_SPECIMENS	Number of specimens in the lot	count	NUMBER(38,0)	Number of specimens in the voucher lot
N_-WEIGHT	Hauls with catch	count	NUMBER(38,0)	Total number of hauls with positive catch biomass.
OLD_-SPECIES_-CODE	Old species code	ID key code	NUMBER(10,0)	Species code associated with old species name
OLD_-SPECIES_-NAME	Old species name	text	VARCHAR2(255 BYTE)	Taxonomic name previously used in the database

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
ORDER_TAXON	Order phylogenetic rank	category	VARCHAR2(255 BYTE)	Phylogenetic latin rank of order of a given species.
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).
PHYLUM_TAXON	Phylum phylogenetic rank	category	VARCHAR2(255 BYTE)	Phylogenetic latin rank of phylum of a given species.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
POLYGON_WKB	Polygon binary string	code string	VARCHAR2(255 BYTE)	Well-known binary (WKB) representation of geometry for a AREA_JOIN polygon. WKB is used to transfer and store the same information in a more compact form convenient for computer processing but that is not human-readable.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
POLYGON_WKT	Polygon well known text	code string	VARCHAR2(255 BYTE)	Well-known text (WKT) representation of geometry for a AREA_JOIN polygon. WKT is a text markup language for representing vector geometry objects.
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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.
PRESERVATIVE	Chemical specimen stored in	text	VARCHAR2(255 BYTE)	Chemical specimen currently stored in
PRINCIPAL_INVESTIGATOR	Principle investigator	text	VARCHAR2(25 BYTE)	First and last name of principal investigator for a project.
PROJECT_TITLE	Title of special project	text	VARCHAR2(255 BYTE)	Special project title.
PROJECT_TITLE_SHORT	Short title of special project	text	VARCHAR2(25 BYTE)	Special project short title (short version of PROJECT_TITLE).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
RANK_ID	Taxonomic rank	category	VARCHAR2(255 BYTE)	The taxonomic rank of a taxon identification.
REASON	Reason for taxonomic change	text	VARCHAR2(255 BYTE)	Reason for taxonomic change; pulled directly from online database (i.e. WoRMS or ITIS)

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
SEX	Sex of a specimen	ID key code	NUMBER(38,0)	Sex of a specimen where ”1” = ”Male”, ”2” = ”Female”, ”3” = Unsexed.

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).
SPECIES_NAME	Scientific name of species	text	VARCHAR2(255 BYTE)	Scientific name of species.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
SPECIES_-NAME_-ACCEPTED	Scientific name used in taxonomic database	text	VARCHAR2(255 BYTE)	Scientific name of species used in taxonomic ”DATABASE” column.
SPECIES_-NAME_-SURVEY	Scientific name used in survey data	text	VARCHAR2(255 BYTE)	Scientific name of species historically or currently used in the survey.
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

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
STANDARD_LENGTH_MM	Standard length of specimens (mm)	numeric	VARCHAR2(255 BYTE)	Standard length of specimen or range of lengths if multiple specimens in lot; measured by taxonomists in lab
STATION	Station ID	ID key code	VARCHAR2(255 BYTE)	Alphanumeric designation for the station established in the design of a survey.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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.
SUBCLASS_TAXON	Subclass phylogenetic rank	category	VARCHAR2(255 BYTE)	Phylogenetic latin rank of subclass of a given species.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
SUBFAMILY_TAXON	Subfamily phylogenetic rank	category	VARCHAR2(255 BYTE)	Phylogenetic latin rank of subfamily of a given species.
SUBMISSION_DATE	Date	YYYY-MM-DD	DATE	Date special projects were due to be submitted for the upcoming survey season.
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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.
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.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
SURVEY	Survey name	text	VARCHAR2(25 BYTE)	Name and description of survey. The column survey is associated with the srvy and survey_definition_id columns.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
SURVEY_ID	Survey ID raw	ID key code	NUMBER(38,0)	The survey ID uniquely identifies a survey instance.
SURVEY_NAME	Survey name official	text	VARCHAR2(255 BYTE)	Long name of the survey conducted
SURVEY_SPECIES	Species used in survey	logical	BINARY_DOUBLE	Designates whether or not species name is accepted/actively used in the RACE surveys
TAXONOMIST	Taxonomist	text	VARCHAR2(255 BYTE)	Taxonomist(s) who re-identified specimen(s)

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
				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

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
				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

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
TRAWLABL	Trawlable stations	logical	BINARY_-DOUBLE	Logical, describing if stations are trawlable (TRUE) or not (FALSE).

Column name from data	Descrip- tive column Name	Units	Oracle data type	Column descrip- tion
VESSEL_-CALL-SIGN	Vessel call sign	ID key code	NUM-BER(38,0)	A call sign is a designated sequence of letters and numbers that are assigned when a vessel, whether it be a sailing yacht, motor yacht, rib or com- mercial vessel, receives its Ship Radio Licence. The vessel also receives its MMSI number, so that each vessel is uniquely identified.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
VESSEL_COAST_GUARD_NUMBER	Vessel coast guard number	ID key code	NUMBER(38,0)	Official Identification number as defined by www.dco.uscg.mil. The Official Number (O/N) is the 6 or 7 digit number awarded to the vessel at the time it is first documented with the US Coast Guard. This number remains with the vessel indefinitely and should be marked in accordance with 46 CFR 67.121.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
VESSEL_IMO	Vessel international maritime organization number	ID key code	NUMBER(38,0)	The International Maritime Organization (IMO) number consists of the letters "IMO" followed by a unique, seven-digit number: the pattern is "NNNNNNN", where N is a single-digit number, e.g., "1234567"
VESSEL_LENGTH_M	Vessel length (m)	meters	NUMBER(38,0)	The length of vessel in meters.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
VESSEL_MMSI	Vessel maritime mobile service identities	ID key code	NUMBER(38,0)	Maritime Mobile Service Identities (MMSIs) are nine-digit numbers used by maritime digital selective calling (DSC), automatic identification systems (AIS) and certain other equipment to uniquely identify a ship or a coast radio station.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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).

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
VESSEL_OWNER	Vessel owner	text	VARCHAR2(255 BYTE)	Name of vessel owner or company.
VESSEL_TONNAGE	Vessel tonnage	metric tons	NUMBER(38,0)	The tonnage of vessel in metric tons.
VOUCHER	Voucher number	numeric	NUMBER(38,0)	The voucher number of the specimen within a single haul
WEIGHT_G	Specimen weight (g)	grams	NUMBER(38,1)	Weight of specimen (grams).
WEIGHT_KG	Sample or taxon weight (kg)	kilograms	NUMBER(38,3)	Weight (thousandths of a kilogram) of individuals in a haul by taxon.
WIRE_LENGTH_M	Trawl wire length	meters	NUMBER(38,0)	Length of wire deployed during a given haul in meters.

Column name from data	Descriptive column Name	Units	Oracle data type	Column description
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/).
YEAR	Survey year	year	NUMBER(10,0)	Year the observation (survey) was collected.
YEAR_CHANGED	Year changed	numeric	DATE	Year change implemented in database

Part III.

AKFIN

These data are used directly by stock assessors and are provided to The Alaska Fisheries Information Network (AKFIN).

The Alaska Fisheries Information Network

The Alaska Fisheries Information Network (AKFIN) is a regional program that consolidates and supports the processing, analysis, and reporting of fisheries data for Alaskan fisheries. AKFIN integrates this information into a single data management system using consistent methods and standardized formats. The resulting data enables fishery managers, scientists, and associated agencies to supervise fisheries resources more effectively and efficiently. The AKFIN database contains much of the data needed to complete stock assessments, including GAP trawl survey data.

Data Access Options

Direct database connection If you are an AFSC employee you may access the AKFIN oracle database directly while on the NOAA network or VPN. Note that this is a separate database from the AFSC oracle database referenced above, and requires separate credentials. If you do not already have an AKFIN account you can request one here. NOAA IT will need to add AKFIN access to your tnsnames.ora file (they do this frequently). Once your connection is established data may be accessed through SQL queries using SQL developer, R, or python.

AKFIN Answers

(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 AKFIN direct database credentials. The distribution of GAP_PRODUCTS on AKFIN Answers is planned but not yet implemented. The RACE Survey tab on the stock assessment dashboard contains reports generated from now depreciated tables that predated the GAP_PRODUCTS tables. AKFIN will keep these reports for reference but they will not be updated 2024 onward.

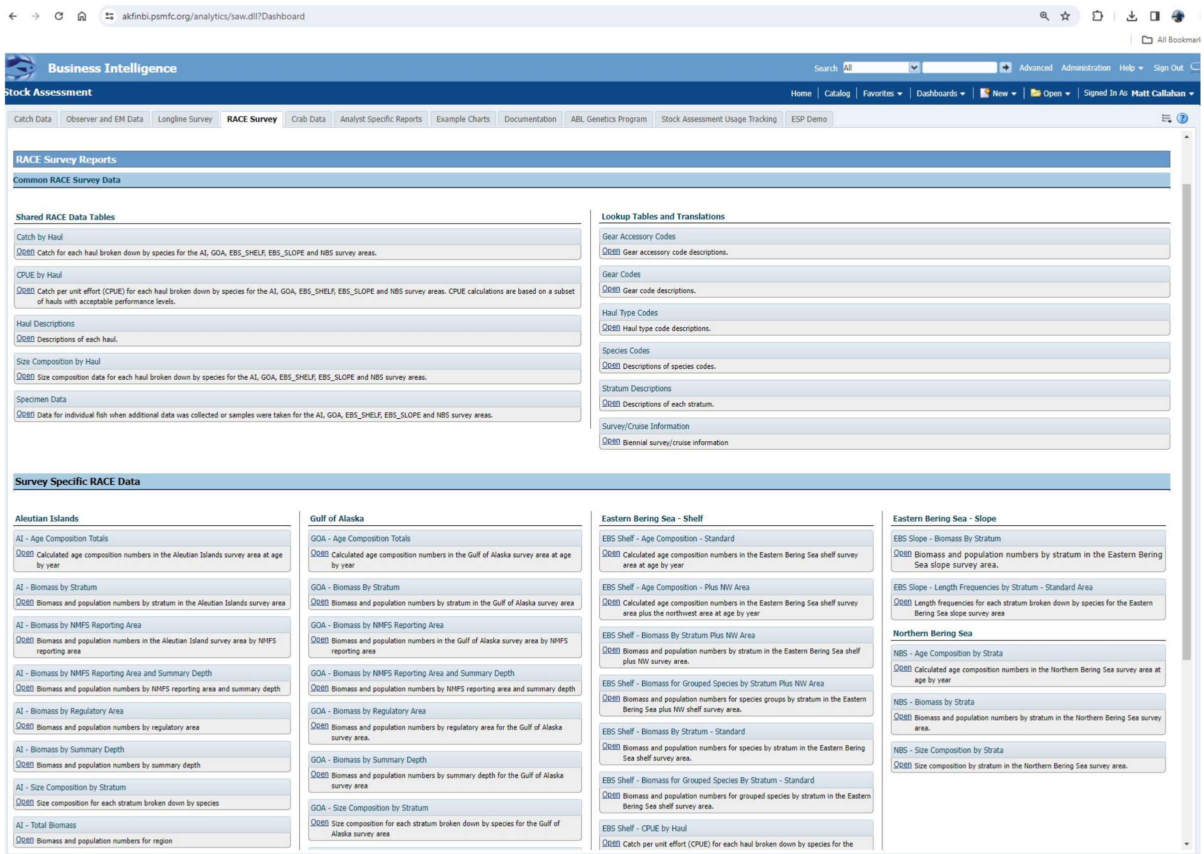


Figure 7.1.: AKFIN platfrom.

Web Service

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 (pronounced akfin-gap-data not ak-eff-ing-app-data) R package helpful.

Cite this data

Use the below bibtext citation, as cited in our group’s citation repository for citing the data created and maintained in this repo (Alaska Fisheries Information Network (AKFIN), 2024). Add “note = {Accessed: mm/dd/yyyy}” to append the day this data was accessed.

```
 [1] "@misc{GAPakfin,"
 [2] "  author = {{Alaska Fisheries Information Network (AKFIN)}}, "
 [3] "  institution = {{NOAA Fisheries Alaska Fisheries Science Center, Groundfish Assessme
 [4] "  year = {2024}, "
 [5] "  title = {AFSC Groundfish Assessment Program Design-Based Production Data},"
 [6] "  howpublished = {https://akfinbi.psmfc.org/analytics/},"
 [7] "  url = {https://www.psmfc.org/program/alaska-fisheries-information-network-
akfin},"
 [8] "  publisher = {{U.S. Dep. Commer.}},"
 [9] "  copyright = {Public Domain} "
[10] "}"
```

8. Data description

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.

8.1. Data tables

8.1.1. AKFIN_AGECOMP

snapshot table for snapshot GAP_PRODUCTS.AKFIN_AGECOMP

Number of rows: 687,698

Number of columns: 10

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

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

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.

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.

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.

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.

8.1.2. AKFIN_AREA

snapshot table for snapshot GAP_PRODUCTS.AKFIN_AREA

Number of rows: 503

Number of columns: 9

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.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

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.

DESCRIPTION

Description

text

VARCHAR2(4000 BYTE)

Description of row observation.

AREA_KM2

Area (km2)

kilometers squared

NUMBER(38,3)

Area in square kilometers.

DEPTH_MIN_M

Area ID minimum depth (m)

meters

NUMBER(38,3)

Minimum depth (meters).

DEPTH_MAX_M

Area ID maximum depth (m)

meters

NUMBER(38,3)

Maximum depth (meters).

8.1.3. AKFIN_BIOMASS

snapshot table for snapshot GAP_PRODUCTS.AKFIN_BIOMASS

Number of rows: 2,655,773

Number of columns: 16

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

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.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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_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).

N_HAUL

Valid hauls

count

NUMBER(38,0)

Total number of hauls.

N_WEIGHT

Hauls with catch

count

NUMBER(38,0)

Total number of hauls with positive catch biomass.

N_COUNT

Hauls with taxon counts

numeric

NUMBER(38,0)

Total number of hauls with positive count data.

N_LENGTH

Hauls with taxon lengths

count

NUMBER(38,0)

Total number of hauls with length data.

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.

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.

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_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).

8.1.4. AKFIN_CATCH

snapshot table for snapshot GAP_PRODUCTS.AKFIN_CATCH

Number of rows: 998,924

Number of columns: 6

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

CATCHJOIN

Catch observation ID

ID key code

NUMBER(38,0)

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

WEIGHT_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

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

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

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

8.1.5. AKFIN_CPUE

snapshot table for snapshot GAP_PRODUCTS.AKFIN_CPUE

Number of rows: 21,915,584

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

WEIGHT_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

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

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

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

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.

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).

8.1.6. AKFIN_CRUISE

snapshot table for snapshot GAP_PRODUCTS.AKFIN_CRUISE

Number of rows: 180

Number of columns: 10

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

CRUISEJOIN

Cruise ID

ID key code

NUMBER(38,0)

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

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.).

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

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.

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.

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.

DATE_START

Start date

YYYY-MM-DD

DATE

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

DATE_END

End date

YYYY-MM-DD

DATE

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

8.1.7. AKFIN_HAUL

snapshot table for snapshot GAP_PRODUCTS.AKFIN_HAUL

Number of rows: 35,111

Number of columns: 25

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

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.

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.

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.

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).

DURATION_HR

Tow duration (decimal hr)

hours

NUMBER(38,1)

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

DISTANCE_FISHED_KM

Distance fished (km)

kilometers

NUMBER(38,3)

Distance the net fished (kilometers).

NET_WIDTH_M

Net width (m)

meters

NUMBER(38,1)

Measured or estimated distance (meters) between wingtips 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_HEIGHT_M

Net height (m)

meters

NUMBER(38,1)

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

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.

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.

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.

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.

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.

STATION

Station ID

ID key code

VARCHAR2(255 BYTE)

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

DEPTH_GEAR_M

Depth of gear (m)

meters

NUMBER(38,1)

Depth of gear (meters).

DEPTH_M

Depth (m)

meters

NUMBER(38,1)

Bottom depth (meters).

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.

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.

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.

WIRE_LENGTH_M

Trawl wire length

meters

NUMBER(38,0)

Length of wire deployed during a given haul in meters.

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.

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.

8.1.8. AKFIN_LENGTH

snapshot table for snapshot GAP_PRODUCTS.AKFIN_LENGTH

Number of rows: 4,573,329

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

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

FREQUENCY

Count of observation

count

NUMBER(38,0)

Frequency, or count, of an observation.

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.

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.

8.1.9. AKFIN_METADATA_COLUMN

snapshot table for snapshot GAP_PRODUCTS.AKFIN_METADATA_COLUMN

Number of rows: 173

Number of columns: 5

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

METADATA_COLNAME_DESC

Column description

text

VARCHAR2(4000 BYTE)

Description of the column.

METADATA_COLNAME

Column name

text

VARCHAR2(4000 BYTE)

Name of the column in a table.

METADATA_COLNAME_LONG

Column name spelled out

text

VARCHAR2(4000 BYTE)

Long name for the column.

METADATA_UNITS

Units

category

VARCHAR2(4000 BYTE)

Units of the column.

METADATA_DATATYPE

Oracle datatype code

text

VARCHAR2(4000 BYTE)

Oracle data type of data column.

8.1.10. AKFIN_SIZECOMP

snapshot table for snapshot GAP_PRODUCTS.AKFIN_SIZECOMP

Number of rows: 3,305,896

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

AREA_ID

Area ID

ID key code

NUMBER(38,0)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

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.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

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

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.

8.1.11. AKFIN_SPECIMEN

snapshot table for snapshot GAP_PRODUCTS.AKFIN_SPECIMEN

Number of rows: 604,594

Number of columns: 12

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

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

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

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.

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.

SEX

Sex of a specimen

ID key code

NUMBER(38,0)

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

WEIGHT_G

Specimen weight (g)

grams

NUMBER(38,1)

Weight of specimen (grams).

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.

MATURITY

Specimen maturity code

ID key code

NUMBER(38,0)

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

GONAD_G

Weight of gonads (g)

grams

NUMBER(38,1)

Weight of specimen gonads (grams).

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.

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.

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.

8.1.12. AKFIN_STRATUM_GROUPS

snapshot table for snapshot GAP_PRODUCTS.AKFIN_STRATUM_GROUPS

Number of rows: 1,066

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)

An ID code representing a broad range of spatial areas from statistical sampling strata to management and regional areas composed of multiple strata.

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.

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.

8.1.13. AKFIN_SURVEY_DESIGN

snapshot table for snapshot GAP_PRODUCTS.AKFIN_SURVEY_DESIGN

Number of rows: 88

Number of columns: 3

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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.

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

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.

8.1.14. AKFIN_TAXONOMIC_CLASSIFICATION

snapshot table for snapshot GAP_PRODUCTS.AKFIN_TAXONOMIC_CLASSIFICATION

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.

SUPERCLASS_TAXON

Superclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superclass of a given species.

SUBPHYLUM_TAXON

Subphylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subphylum of a given species.

PHYLUM_TAXON

Phylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of phylum of a given species.

KINGDOM_TAXON

Kingdom phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of kingdom of a given species.

SPECIES_NAME

Scientific name of species

text

VARCHAR2(255 BYTE)

Scientific name of 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.

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.

ID_RANK

Lowest taxonomic rank

text

VARCHAR2(255 BYTE)

Lowest taxonomic rank of a given species entry.

DATABASE_ID

Species ID in database

ID key code

VARCHAR2(255 BYTE)

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

DATABASE

Database source

category

VARCHAR2(255 BYTE)

Taxonomic database source, either ITIS or WoRMS.

GENUS_TAXON

Genus phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of genus of a given species.

SUBFAMILY_TAXON

Subfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subfamily of a given species.

FAMILY_TAXON

Family phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of family of a given species.

SUPERFAMILY_TAXON

Superfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superfamily of a given species.

SUBORDER_TAXON

Suborder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of suborder of a given species.

ORDER_TAXON

Order phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of order of a given species.

SUPERORDER_TAXON

Superorder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superorder of a given species.

SUBCLASS_TAXON

Subclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subclass of a given species.

8.1.15. AKFIN_TAXONOMIC_GROUPS

snapshot table for snapshot GAP_PRODUCTS.AKFIN_TAXONOMIC_GROUPS

Number of rows: 2,777

Number of columns: 22

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.

SPECIES_NAME

Scientific name of species

text

VARCHAR2(255 BYTE)

Scientific name of 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.

ID_RANK

Lowest taxonomic rank

text

VARCHAR2(255 BYTE)

Lowest taxonomic rank of a given species entry.

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.

GENUS_TAXON

Genus phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of genus of a given species.

SUBFAMILY_TAXON

Subfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subfamily of a given species.

FAMILY_TAXON

Family phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of family of a given species.

SUPERFAMILY_TAXON

Superfamily phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superfamily of a given species.

INFRAORDER_TAXON

Infraorder phylogenetic rank

category

VARCHAR2(255 BYTE)

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

SUBORDER_TAXON

Suborder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of suborder of a given species.

ORDER_TAXON

Order phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of order of a given species.

SUPERORDER_TAXON

Superorder phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superorder of a given species.

INFRACLASS_TAXON

Infraclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Infraclass phylogenetic rank. Phylogenetic latin rank of infraclass of a given speices.

SUBCLASS_TAXON

Subclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subclass of a given species.

CLASS_TAXON

Class phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of class of a given species.

SUPERCLASS_TAXON

Superclass phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of superclass of a given species.

SUBPHYLUM_TAXON

Subphylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of subphylum of a given species.

PHYLUM_TAXON

Phylum phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of phylum of a given species.

KINGDOM_TAXON

Kingdom phylogenetic rank

category

VARCHAR2(255 BYTE)

Phylogenetic latin rank of kingdom of a given species.

GROUP_CODE

Species or Complex ID

ID key code

NUMBER(38,0)

Equivalent to the SPECIES_CODE if the taxon is reported as a single taxon in GAP_-PRODUCTS, otherwise denotes a SPECIES_CODE of a higher taxonomic group to which the taxon is aggregated in the GAP_PRODUCTS CPUE and BIOMASS tables.

9. 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.

9.0.1. 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.

```r
library(RODBC)
channel <- gapindex::get_connected()
```

Data SQL Query Examples:

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

9.0.1. 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()
```

9.0.2. 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()
```

Table 9.1.: CPUE for all EBS and NBS stations with associated haul, cruise, and species information.

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/Ground-fish Bottom Trawl Survey	1	CHAPMAN	877
198,203	1,982	98	Eastern Bering Sea Crab/Groundfish Bottom Trawl Survey	1	CHAPMAN	877

9.0.3. 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, CPU
  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()
```

Table 9.2.: CPUE for all stations contained in the Shumagin region (AREA_ID = 919).

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	10,044.840[illegible]	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

9.0.4. Ex. EBS Pacific Ocean perch CPUE and `akgfmaps` map

Pacific Ocean perch catch-per-unit-effort estimates for EBS in 2021 from `GAP_PRODUCTS.AKFIN_CPUE` and map constructed using `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, but can take any units
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()
```

Table 9.3.: EBS Pacific Ocean perch CPUE and `akgfmaps` map.

CPUE_-KGHA	LATITUDE	LONGI-TUDE
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
```

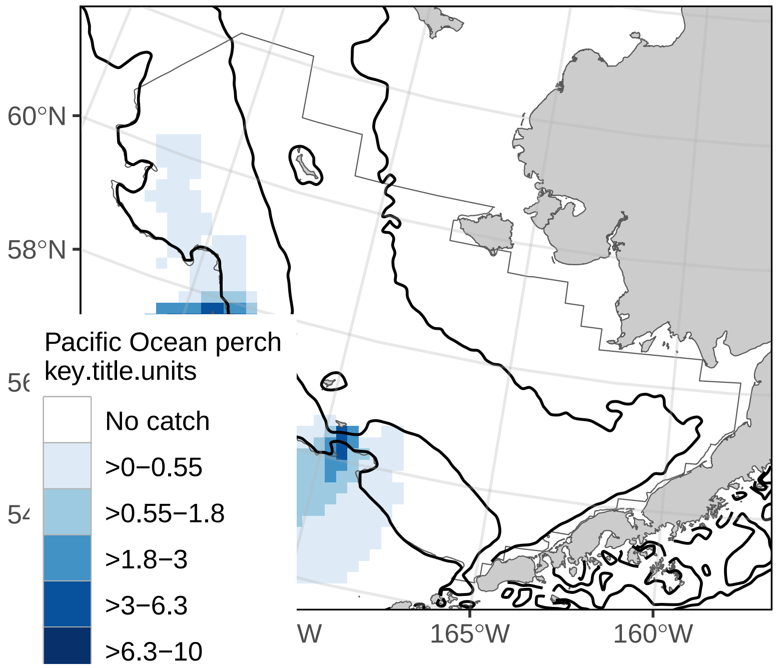


Figure 9.1.: EBS Pacific Ocean perch CPUE and `akgfmaps` map.

9.0.5. 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
-- Use the AREA records associated with the GOA stratification prior to 2025
AND DESIGN_YEAR = 1984)

-- 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
AND BIOMASS.YEAR BETWEEN 1990 AND 2023")
```

```
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(j = "YEAR", big.mark = "")
```

Table 9.4.: GOA Pacific Ocean perch biomass and abundance.

BIOMASS_MT	POPULATION_COUNT	YEAR	DESCRIPTION
31,073.15	60,087,711	1990	CENTRAL GOA - INPFC
100,321.48	174,708,361	1990	EASTERN GOA - INPFC
24,435.56	79,343,919	1990	WESTERN GOA - INPFC
155,830.19	314,139,991	1990	GOA Region: All Strata
256,345.03	454,133,028	1993	CENTRAL GOA - INPFC
147,912.16	230,314,654	1993	EASTERN GOA - INPFC

```r
# 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
```

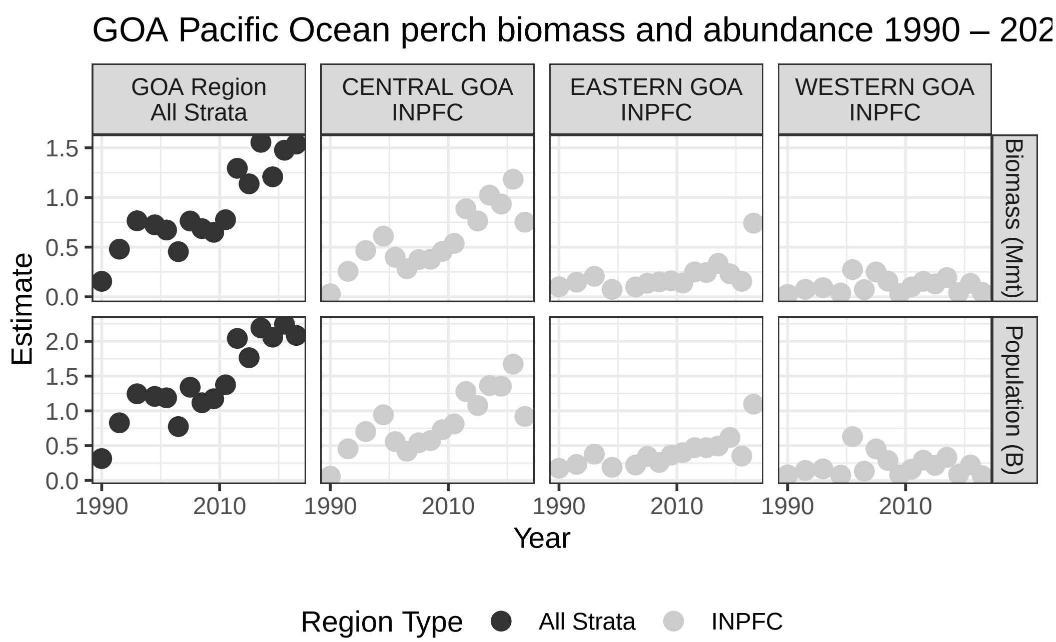


Figure 9.2.: GOA Pacific Ocean perch biomass and abundance.

9.0.6. 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`.

```
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(j = "year", big.mark = "")
dat0
```

Table 9.5.: AI Rock sole size compositions and ridge plot.

year	length_-cm	popula-tion_-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
```

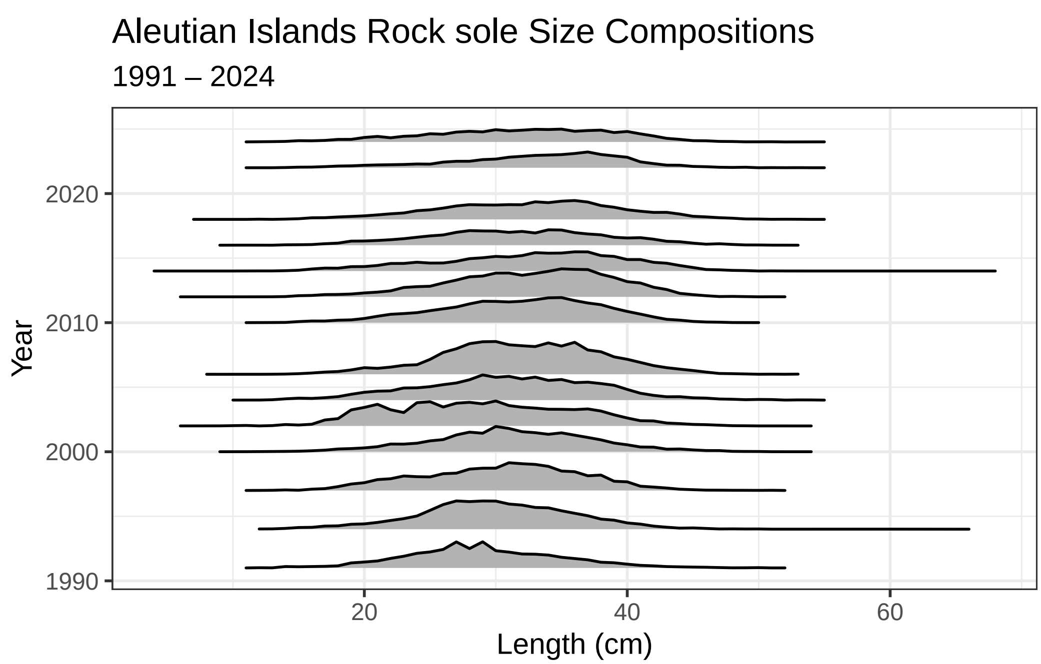


Figure 9.3.: AI Rock sole size compositions and ridge plot.

9.0.7. 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
-- Filter for EBS Standard + NW Area
WHERE AREA_ID = 99900)

-- 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()
```

Table 9.6.: EBS Walleye Pollock Age Compositions and Age Pyramid.

AGE	POPULATION_COUNT	SEX
1	22,060,172	1
2	123,165,369	1
3	136,542,62	1
4	252,538,747	1
5	964,790,93	1
6	242,135,720	1

```r
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 (Standard Area + NW) walleye pollock Age Composition")
  ggplot2::guides(fill = guide_legend(title = "Sex"))+
  ggplot2::theme_bw()

figure
```

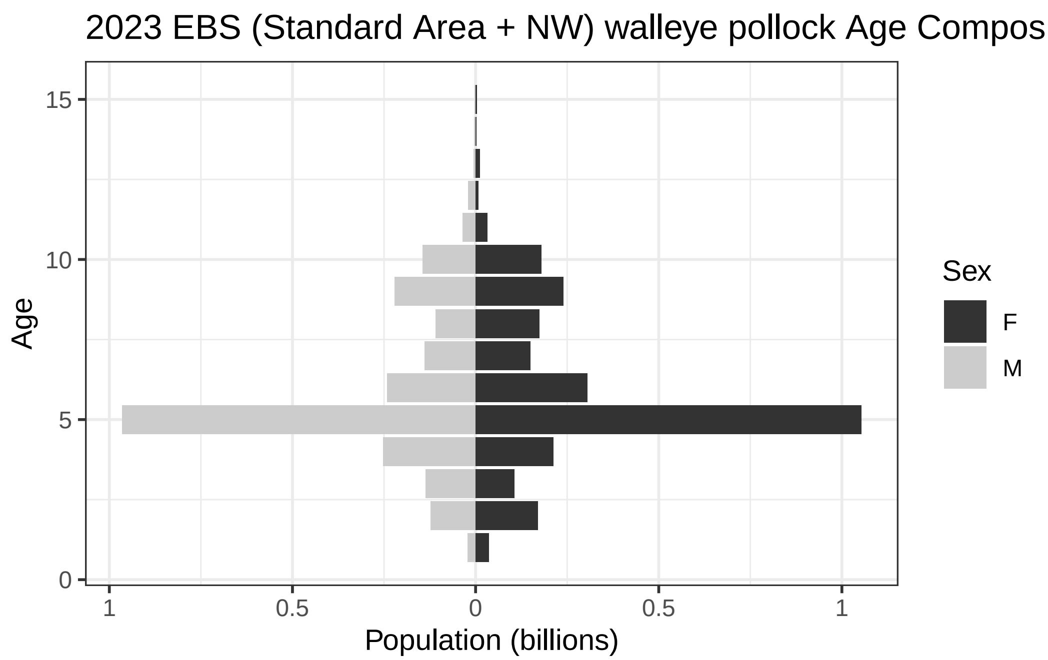


Figure 9.4.: 2023 EBS Walleye Pollock Age Compositions and Age Pyramid.

9.0.8. Ex. NBS Pacific cod biomass and abundance

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

```
dat <- RODBC::sqlQuery(channel = channel,
                       query =
                         "
SELECT YEAR, AREA_ID AS STRATUM, AREA_NAME, BIOMASS_MT, POPULATION_COUNT
FROM GAP_PRODUCTS.AKFIN_BIOMASS

JOIN ( -- join with area table
SELECT AREA_ID, AREA_NAME
FROM GAP_PRODUCTS.AKFIN_AREA
WHERE AREA_TYPE = 'STRATUM'
AND SURVEY_DEFINITION_ID = 143
AND DESIGN_YEAR = 2022)

USING (AREA_ID)
```

```
-- Filter data results to NBS Pacific cod
WHERE SURVEY_DEFINITION_ID IN 143
AND SPECIES_CODE = 21720
ORDER BY YEAR, STRATUM")
```

```
dat0 <- dat |>
  janitor::clean_names() |>
  dplyr::select(year, area_name, biomass_mt, population_count) |>
  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_name, levels = unique(area_name), labels = unique(area_name), ordere
flextable::flextable(dat) |>
  flextable::fit_to_width(max_width = 6) |>
  flextable::theme_zebra() |>
  flextable::colformat_num(j = "YEAR", big.mark = "")
```

Table 9.7.: NBS Pacific cod biomass and abundance.

YEAR	STRATUM	AREA_-NAME	BIOMASS_MT	POPULA-TION_-COUNT
2010	70	Inner Domain	7,462.5586	4,724,153
2010	71	Inner Domain	20,983.3757	3,928,600
2010	81	Middle Domain	680.4357	250,837
2017	70	Inner Domain	132,490.151[illegible]	66,187,245
2017	71	Inner Domain	147,971.45[illegible]	65,078,489

Data SQL Query Examples:

YEAR	STRATUM	AREA_-NAME	BIOMASS_MT	POPULA-TION_-COUNT
2017	81	Middle Domain	7,089.8740	4,191,118
2019	70	Inner Domain	107,096.729	102,734,142
2019	71	Inner Domain	194,846.7230	73,495,085
2019	81	Middle Domain	63,061.2780	25,926,805
2021	70	Inner Domain	95,849.9833	68,767,498
2021	71	Inner Domain	53,814.6332	17,941,471
2021	81	Middle Domain	77,917.1083	42,991,939
2022	70	Inner Domain	96,500.6975	60,433,135
2022	71	Inner Domain	26,747.0747	10,447,602
2022	81	Middle Domain	30,487.2782	15,157,597
2023	70	Inner Domain	76,708.4327	39,605,860
2023	71	Inner Domain	19,130.0040	8,459,469
2023	81	Middle Domain	12,507.8566	4,128,368
2025	70	Inner Domain	48,587.6745	31,324,540
2025	71	Inner Domain	11,129.2287	4,456,827

Data SQL Query Examples:

YEAR	STRATUM	AREA_-NAME	BIOMASS_MT	POPULA-TION_-COUNT
2025	81	Middle Domain	9,665.6621	7,262,957

```
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 = "Domain Type "))+
  ggplot2::scale_fill_grey() +
  ggplot2::theme_bw() +
  ggplot2::theme(legend.direction = "horizontal",
                 legend.position = "bottom")

figure
```

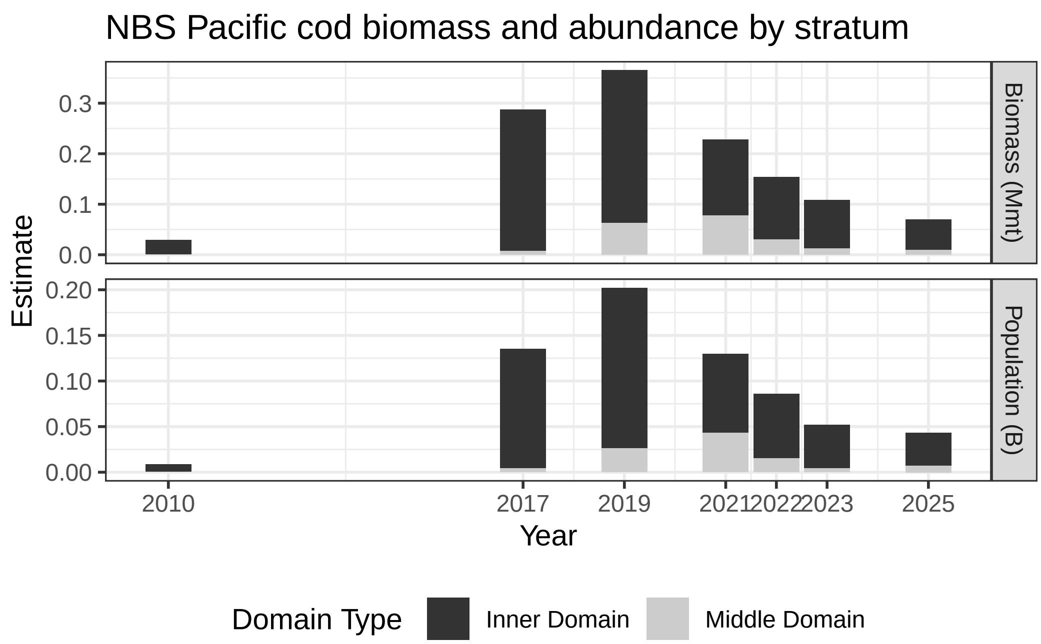


Figure 9.5.: NBS Pacific cod biomass and abundance.

9.0.9. 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 / 1000000 AS BIOMASS_MMT,
(BIOMASS_MT - 2 * SQRT(BIOMASS_VAR)) / 1000000 AS BIOMASS_CI_DW,
(BIOMASS_MT + 2 * SQRT(BIOMASS_VAR)) / 1000000 AS BIOMASS_CI_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(j = "year", big.mark = "")
```

Table 9.8.: GOA Pacific Ocean perch biomass and line plot.

survey_-definition_id	biomass_-mmt	biomass_-ci_dw	biomass_-ci_up	year
47	0.1558302	0.0618137(	0.2498467	1990
47	0.4796687	0.26596329	0.6933741	1993
47	0.7651705	0.36044598	1.1698950	1996
47	0.7243655	-0.05238029	1.5011113	1999
47	0.6723673	0.2290337ƽ	1.1157008	2001
47	0.4543899	0.31077353	0.5980063	2003

```
a_mean <- dat |>
  dplyr::group_by(survey_definition_id) |>
  dplyr::summarise(biomass_mmt = mean(biomass_mmt, 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_mmt)) +
  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 (Mmt)",
    labels = comma) +
  ggplot2::geom_segment(
    data = a_mean,
    mapping = aes(x = minyr,
                  xend = maxyr,
                  y = biomass_mmt,
                  yend = biomass_mmt),
    linetype = "dashed",
    linewidth = 2) +
  ggplot2::geom_errorbar(
    mapping = aes(ymin = biomass_ci_dw, ymax = biomass_ci_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_mmt,
                              digits = 2,
                              big.mark = ",",
                              format = "f"),
                      " Mmt")) +
  ggplot2::theme_bw()

figure
```

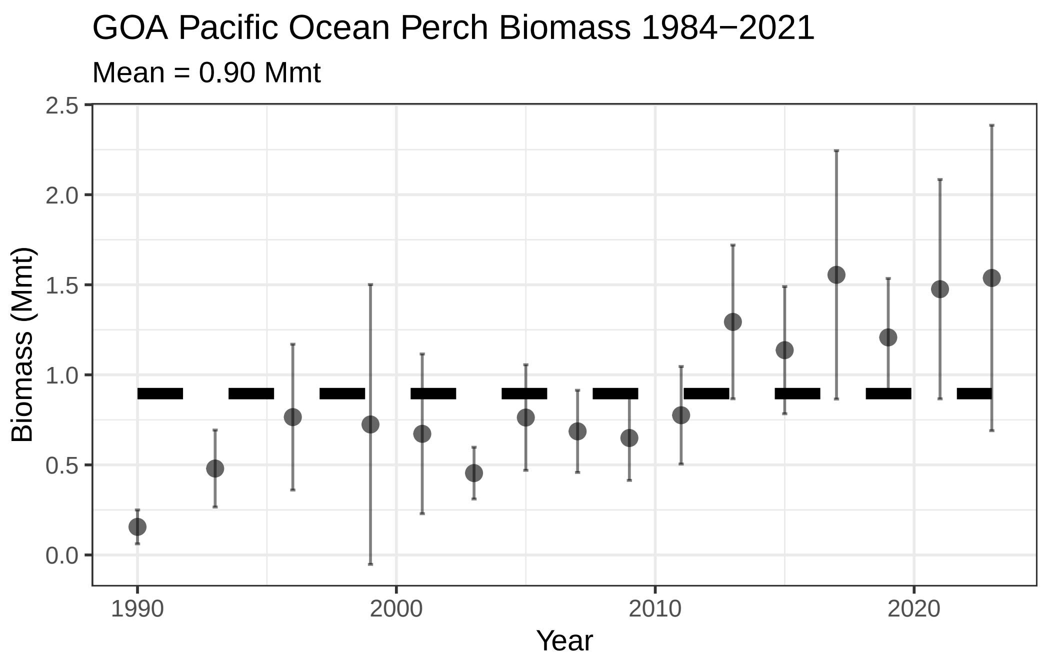


Figure 9.6.: GOA Pacific Ocean perch biomass and line plot.

9.0.10. Ex. 2022 AI Atka mackerel age specimen summary

9.0.10.1. 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(j = c("year", "species_code"), big.mark = "")
```

Table 9.9.: 2022 AI Atka mackerel age specimen summary: all ages determined.

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

9.0.10.2. 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 AGE >= 0
AND SPECIES_CODE = 21921
AND YEAR = 2022
AND SURVEY_DEFINITION_ID = 52
GROUP BY (YEAR, SURVEY_DEFINITION_ID, SPECIES_CODE, AGE)

ORDER BY AGE") |>
  janitor::clean_names()
```

```
flextable::flextable(dat) |>
  flextable::fit_to_width(max_width = 6) |>
  flextable::theme_zebra() |>
  flextable::colformat_num(j = c("year", "species_code"), big.mark = "")
```

Table 9.10.: Ex.: 2022 AI Atka mackerel age specimen summary: how many of each age were determined.

survey_definition_id	year	species_code	age	countage
52	2022	21921	1	1
52	2022	21921	2	40
52	2022	21921	3	295
52	2022	21921	4	119
52	2022	21921	5	130
52	2022	21921	6	116
52	2022	21921	7	108
52	2022	21921	8	61

survey_definition_id	year	species_code	age	countage
52	2022	21921	9	88
52	2022	21921	10	73
52	2022	21921	11	20
52	2022	21921	12	9
52	2022	21921	13	1

9.0.10.3. 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
                                     latitude = latitude_dd_start, longitude = longitude_dd
                     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_definitio
                     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(j = c("year", "species_code"), big.mark = "")
```

Data SQL Query Examples:

Table 9.11.: 2022 AI Atka mackerel age specimen summary: how many otoliths were aged. This quiery was created using SQL.

survey_-definition_id	year	species_-code	countage
52	2022	21921	1,061

10. Access API data via 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 (pronounced **akfin-gap-data** not **ak-eff-ing-app-data**) R package helpful.

10.0.1. Load packages and helpful functions

10.1. Ex. Direct database query in R using the akfingapdata R package README:

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

```
akfingapdata::get_gap_catch()[,1:6] |>
  head() |>
  flextable::flextable() |>
  flextable::theme_zebra()
```

Part IV.

Public Data (FOSS)

The final, validated survey data are publicly accessible soon after surveys are completed on the Fisheries One Stop Shop (FOSS) platform. This data includes catch, haul, and environmental data collected at each station. On the FOSS data platform, users can interactively select, view, and download data. Descriptive documentation and user-examples are available on the metadata page.

This data contains all of the catch, environmental, and haul data from the fisheries-independent Groundfish and Shellfish Assessment Program surveys in the Bering Sea, Aleutian Islands, and Gulf of Alaska. This data is sought after by the general public, private entities, and NOAA partners alike, including tribal organizations, K-12 classrooms, academic institutions, for-profit groups, and non-profit groups. This data is compiled and approved once a year after each summer survey season and is available for open access.

Part V.

Collaborators and data users

Below are a few packages and products currently using this data. If you have developed a product, performed an analysis, or exhibited this data in any way, reach out so we can showcase your hard work.

- **NOAA Fisheries Distribution Mapping and Analysis Portal**; *NOAA Fisheries Office of Science and Technology*
- **Pull data with python and explore the in-browser visualization tool. Reference their example Python notebook**; *The Eric and Wendy Schmidt Center for Data Science and the Environment at UC Berkeley, including sam.pottinger@berkeley.edu, ccmartinez@berkeley.edu, gzarpellon@berkeley.edu, and kkoy@berkeley.edu.*

Access Constraints

User Constraints: Users must read and fully comprehend the metadata prior to use. Data should not be used beyond the limits of the source scale. Acknowledgment of AFSC Groundfish Assessment Program, as the source from which these data were obtained, in any publications and/or other representations of these data, is suggested.

General questions and more specific data requests can be sent to nmfs.afsc.gap.metadata@noaa.gov or submitted as an issue on our GitHub Organization. The version of this data used for stock assessments can be found through the Alaska Fisheries Information Network (AKFIN). For questions about the eastern Bering Sea surveys, contact Duane Stevenson (Duane.Stevenson@noaa.gov). For questions about the Gulf of Alaska or Aleutian Islands surveys, contact Ned Laman (Ned.Laman@noaa.gov). For questions specifically about crab data in any region, contact Mike Litzow (Mike.Litzow@noaa.gov), the Shellfish Assessment Program lead.

For questions, comments, and concerns specifically about the Fisheries One Stop Shop (FOSS) platform, please contact us using the Comments page on the FOSS webpage.

Cite this data

Use the below bibtext citation, as cited in our group's citation repository for citing the data created and maintained in this repository. Add "note = {Accessed: mm/dd/yyyy}" to append the day this data was accessed.

Cite this data

```
[1] "@misc{FOSSAFSCData,"
[2] "  author = {{NOAA Fisheries Alaska Fisheries Science Center}},"
[3] "  year = {2024}, "
[4] "  title = {Fisheries One Stop Shop Public Data: RACE Division Bottom Trawl Survey Data
[5] "  howpublished = {https://www.fisheries.noaa.gov/foss},"
[6] "  publisher = {{U.S. Dep. Commer.}},"
[7] "  copyright = {Public Domain} "
[8] "}"
```

11. 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.

11.1. Data tables

11.1.1. FOSS_CATCH

snapshot table for snapshot GAP_PRODUCTS.FOSS_CATCH

Number of rows: 917,401

Number of columns: 7

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

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).

COUNT

Taxon count

count, whole number resolution

NUMBER(38,0)

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

WEIGHT_KG

Sample or taxon weight (kg)

kilograms

NUMBER(38,3)

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

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), Species identification confidence in the eastern Bering Sea slope survey (1976-2010), and Species identification confidence in the Gulf of Alaska and Aleutian Islands surveys (1980-2011).

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.

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).

11.1.2. FOSS_HAUL

snapshot table for snapshot GAP_PRODUCTS.FOSS_HAUL

Number of rows: 34,839

Number of columns: 27

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

YEAR

Survey year

year

NUMBER(10,0)

Year the observation (survey) was collected.

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).

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.

SURVEY_NAME

Survey name official

text

VARCHAR2(255 BYTE)

Long name of the survey conducted

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.

HAULJOIN

Haul ID

ID key code

NUMBER(38,0)

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

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.

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.

STATION

Station ID

ID key code

VARCHAR2(255 BYTE)

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

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.

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.

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).

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_START

Start longitude (decimal degrees)

decimal degrees

NUMBER(38,6)

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

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.

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.

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.

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.

DEPTH_M

Depth (m)

meters

NUMBER(38,1)

Bottom depth (meters).

DISTANCE_FISHED_KM

Distance fished (km)

kilometers

NUMBER(38,3)

Distance the net fished (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).

NET_WIDTH_M

Net width (m)

meters

NUMBER(38,1)

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

NET_HEIGHT_M

Net height (m)

meters

NUMBER(38,1)

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

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.

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.

11.1.3. FOSS_SPECIES

snapshot table for snapshot GAP_PRODUCTS.FOSS_SPECIES

Number of rows: 1,000

Number of columns: 7

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.

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.

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.

ID_RANK

Lowest taxonomic rank

text

VARCHAR2(255 BYTE)

Lowest taxonomic rank of a given species entry.

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/).

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/).

11.1.4. FOSS_SURVEY_SPECIES

snapshot table for snapshot GAP_PRODUCTS.FOSS_SURVEY_SPECIES

Number of rows: 2,746

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.

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.

11.1.5. FOSS_TAXON_GROUP

snapshot table for snapshot GAP_PRODUCTS.FOSS_TAXON_GROUP

Number of rows: 10,309

Number of columns: 3

Column name from data

Descriptive column Name

Units

Oracle data type

Column description

RANK_ID

Taxonomic rank

category

VARCHAR2(255 BYTE)

The taxonomic rank of a taxon identification.

CLASSIFICATION

Taxonomic classification rank group

category

VARCHAR2(255 BYTE)

Phylogenetic classification group rank for 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.

12. Using the FOSS platform

12.1. Select and filter

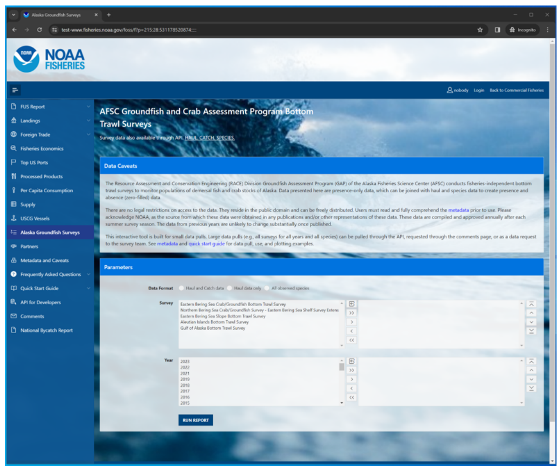


Figure 12.1.: AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform.

Select, filter, and download this and other NOAA Fisheries data from the Fisheries One Stop Shop (FOSS) platform. A user guide for the FOSS platform can be found here. 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.

12.1.1. Catch and haul

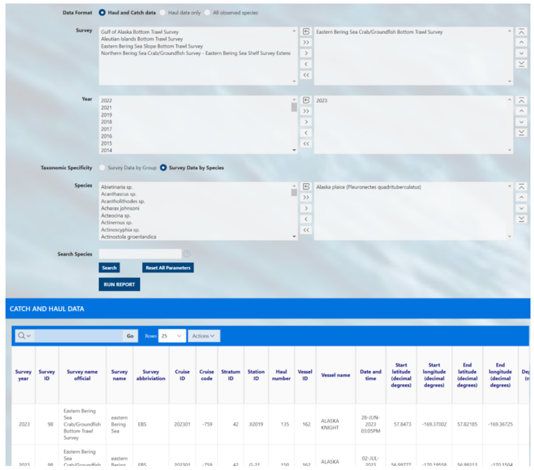


Figure 12.2.: Catch data on the AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform.

12.1.2. Haul

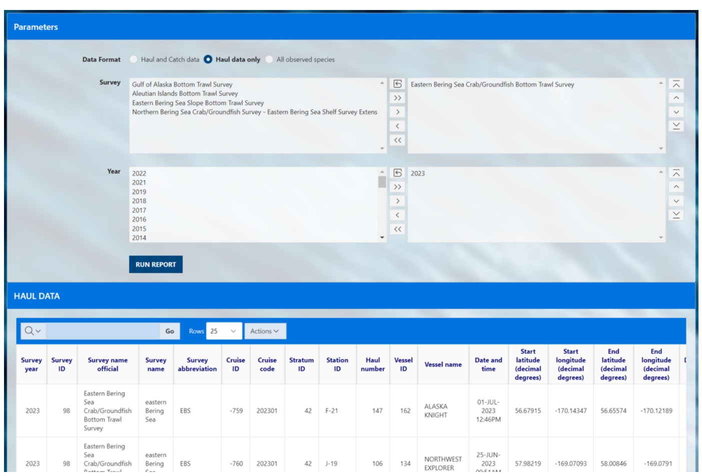


Figure 12.3.: Haul data on the AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform.

12.1.3. Species

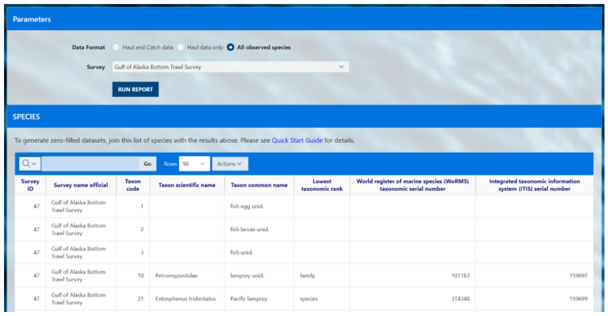


Figure 12.4.: All species observed by survey on the AFSC Groundfish and Crab Assessment Program Bottom Trawl Survey data interface on the Fisheries One Stop Shop platform.

12.2. 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.
- `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.

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.

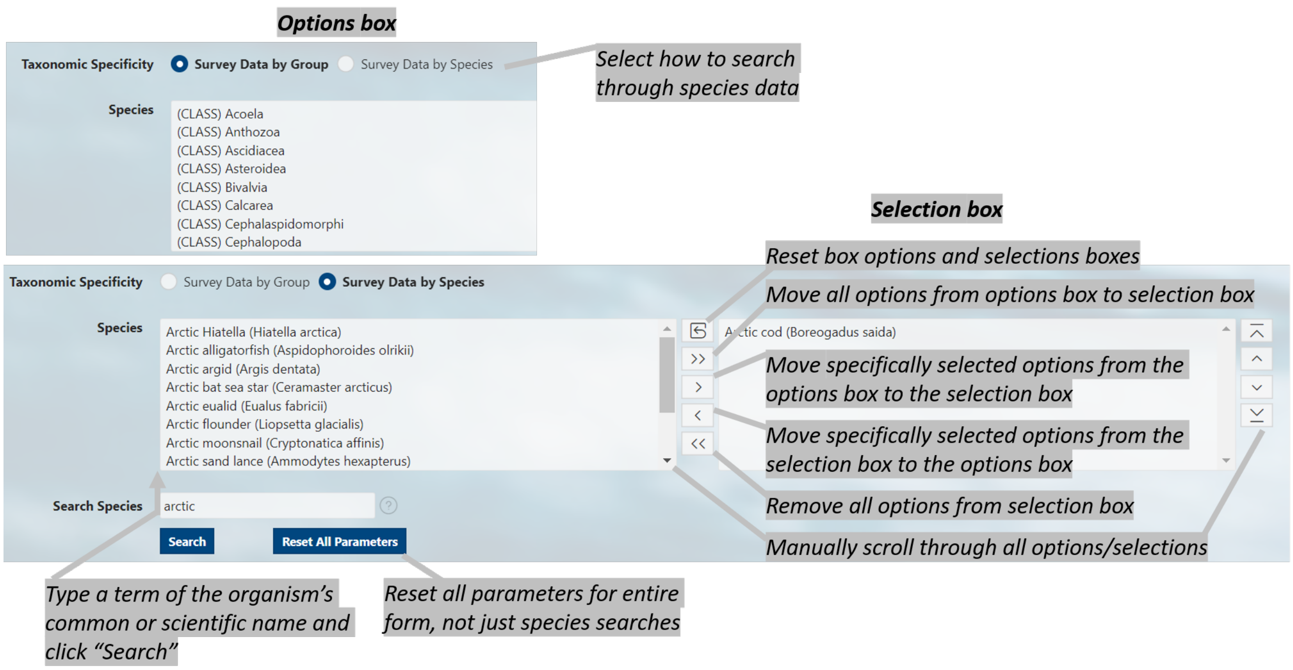


Figure 12.5.: Diagram of selection and search tools available on the FOSS platfrom.

12.3. 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.

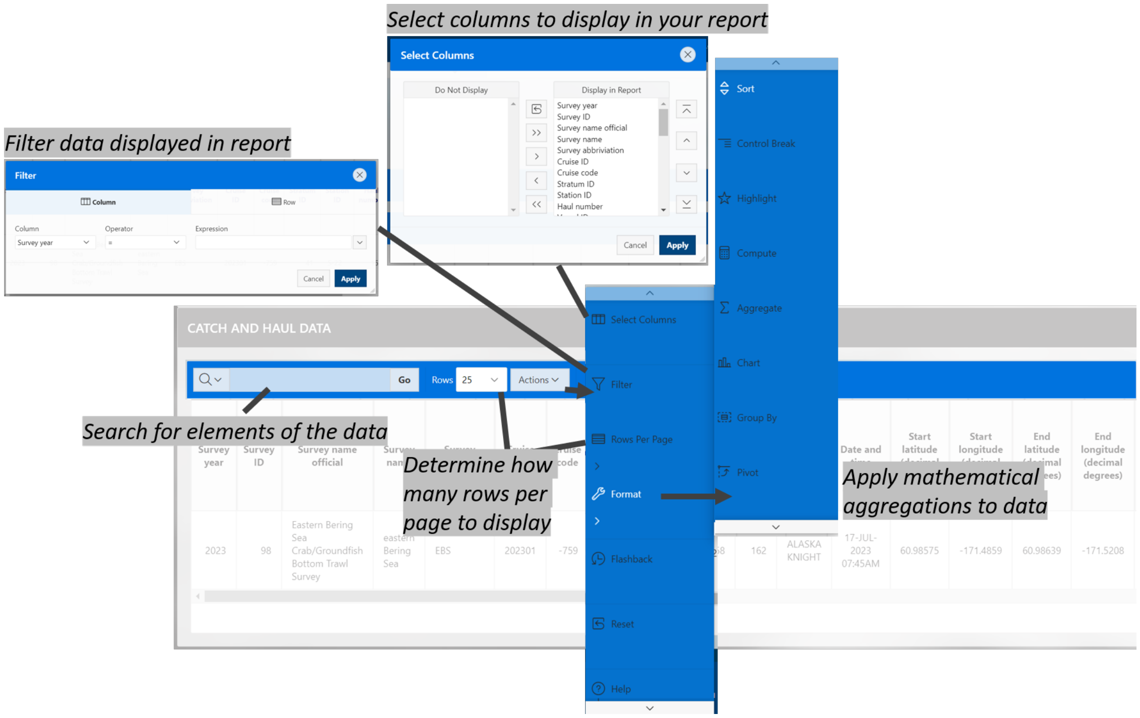


Figure 12.6.: Example data returned from running the report.

12.4. 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`).

13. Use data

Learn how to pull and use this data through the

- API and R programming language
- API and python programming language using the `afscgap` python package
- Oracle and R programming language (AFSC scientists only)

14. 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. Useful introductions to using APIs in `R` can be found here.

There are three tables the user can pull from the API. Learn more about them on the FOSS data description page. Here, you can see them in their raw JSON format:

- 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/
- 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:

14.1. 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. Load haul data

```
# link to the API
api_link_haul <- 'https://apps-st.fisheries.noaa.gov/ods/foss/afsc_groundfish_survey_haul/'
```

14.1.1.1. Load first 25 rows of data

```
res <- httr::GET(url = api_link_haul)
# res ## Test connection

## convert from JSON format
dat <- jsonlite::fromJSON(base::rawToChar(res$content))$items

# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

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

14.1.1.2. 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
}
```

```
[1] 0
[1] 10000
```

```
[1] 20000
[1] 30000
[1] 40000
```

Explore the data contents:

```
# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

```
[1] "rows: 34839; cols: 27"
```

```
# learn about the structure of the data
summary(dat)
```

```
      year          srvy               survey          survey_name       
 Min.   :1982   Length:34839       Length:34839       Length:34839      
 1st Qu.:1997   Class :character   Class :character   Class :character  
 Median :2006   Mode  :character   Mode  :character   Mode  :character  
 Mean   :2006                                                           
 3rd Qu.:2015                                                           
 Max.   :2025                                                           
                                                                        
 survey_definition_id     cruise         cruisejoin         hauljoin      
 Min.   : 47.00       Min.   :198201   Min.   :   -777   Min.   : -24955  
 1st Qu.: 47.00       1st Qu.:199701   1st Qu.:   -700   1st Qu.: -14898  
 Median : 78.00       Median :200601   Median :   -618   Median :  -4846  
 Mean   : 74.77       Mean   :200595   Mean   : 286597   Mean   : 281205  
 3rd Qu.: 98.00       3rd Qu.:201501   3rd Qu.: 827462   3rd Qu.: 802366  
 Max.   :143.00       Max.   :202502   Max.   :1225395   Max.   :1225635  
                                                                          
      haul          stratum         station            vessel_id  
 Min.   :  1.0   Min.   : 10.0   Length:34839       Min.   :  1  
 1st Qu.: 56.0   1st Qu.: 31.0   Class :character   1st Qu.: 88  
 Median :111.0   Median : 50.0   Mode  :character   Median : 94  
 Mean   :116.6   Mean   :129.4                      Mean   :109  
 3rd Qu.:169.0   3rd Qu.:141.0                      3rd Qu.:147  
 Max.   :355.0   Max.   :794.0                      Max.   :178  
                                                                 
 vessel_name         date_time         latitude_dd_start longitude_dd_start
```

```
 Length:34839       Length:34839       Min.   :51.19     Min.   :-180.0
 Class :character   Class :character   1st Qu.:55.02     1st Qu.:-170.7
 Mode  :character   Mode  :character   Median :57.24     Median :-165.3
                                       Mean   :56.90     Mean   :-140.1
                                       3rd Qu.:58.98     3rd Qu.:-154.5
                                       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.   :-2.100       Min.   :-1.100
 1st Qu.:55.02   1st Qu.:-170.7   1st Qu.: 2.700       1st Qu.: 5.800
 Median :57.25   Median :-165.3   Median : 4.100       Median : 7.500
 Mean   :56.90   Mean   :-140.1   Mean   : 3.843       Mean   : 7.809
 3rd Qu.:58.99   3rd Qu.:-154.5   3rd Qu.: 5.200       3rd Qu.: 9.300
 Max.   :65.35   Max.   : 180.0   Max.   :15.300       Max.   :18.100
 NA's   :4       NA's   :4        NA's   :1599         NA's   :853
    depth_m        distance_fished_km  duration_hr      net_width_m
 Min.   :   9.0   Min.   :0.135      Min.   :0.0250   Min.   : 7.51
 1st Qu.:  68.0   1st Qu.:1.498      1st Qu.:0.2710   1st Qu.:15.58
 Median : 101.0   Median :2.537      Median :0.4900   Median :16.40
 Mean   : 136.9   Mean   :2.208      Mean   :0.4009   Mean   :16.43
 3rd Qu.: 155.0   3rd Qu.:2.834      3rd Qu.:0.5100   3rd Qu.:17.22
 Max.   :1200.0   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.: 2.378   1st Qu.:0.024317   1st Qu.:0.0000
 Median : 5.816   Median :0.039604   Median :0.0000
 Mean   : 4.794   Mean   :0.036397   Mean   :0.2782
 3rd Qu.: 6.768   3rd Qu.:0.047225   3rd Qu.:0.0000
 Max.   :11.038   Max.   :0.077795   Max.   :7.0000
 NA's   :3270
```

```r
# Print the first few lines of the data
dat |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("year", "cruise", "cruisejoin"),
      big.mark = "") |>
  flextable::theme_zebra()
```

year	srvy	survey	survey_-name	survey_-defini-tion_id	cruise	cruisejoin	hauljoin	haul
1989	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Groun fish Bottom Trawl Survey	98	198901	159	11,795	153
1989	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Ground-fish Bottom Trawl Survey	98	198901	159	11,796	154
1989	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Groun fish Bottom Trawl Survey	98	198901	159	11,797	155

```
# save outputs for later comparison
dat_haul_api <- dat
```

14.1.2. Load catch data

```
# link to the API
api_link_catch <- 'https://apps-st.fisheries.noaa.gov/ods/foss/afsc_groundfish_survey_catch
```

14.1.2.1. Load first 25 rows of data

```
res <- httr::GET(url = api_link_catch)
# res ## Test connection

## convert from JSON format
dat <- jsonlite::fromJSON(base::rawToChar(res$content))$items

# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

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

14.1.2.2. 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, 100000, 10000)){
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
}
```

Explore the data contents:

```r
# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

```
[1] "rows: 917401; cols: 7"
```

```r
# learn about the structure of the data
summary(dat)
```

```
    hauljoin         species_code     cpue_kgkm2           cpue_nokm2
 Min.   : -24955   Min.   :    1   Min.   :      0.0   Min.   :       12.9
 1st Qu.: -15284   1st Qu.:20510   1st Qu.:      5.6   1st Qu.:       58.5
 Median :  -5742   Median :40500   Median :     48.9   Median :      214.8
 Mean   : 271952   Mean   :45282   Mean   :   1236.9   Mean   :     4613.4
 3rd Qu.: 802106   3rd Qu.:71890   3rd Qu.:    371.8   3rd Qu.:     1146.5
 Max.   :1225635   Max.   :99999   Max.   :3226234.7   Max.   :21780780.3
                                                       NA's   :91873
     count             weight_kg         taxon_confidence
 Min.   :      1.0   Min.   :    0.001   Length:917401
 1st Qu.:      2.0   1st Qu.:    0.200   Class :character
 Median :      8.0   Median :    1.814   Mode  :character
 Mean   :    181.2   Mean   :   41.359
 3rd Qu.:     43.0   3rd Qu.:   13.800
 Max.   :867119.0   Max.   :18187.700
 NA's   :91873
```

```r
# Print the first few lines of the data
dat |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("species_code"),
      big.mark = "") |>
  flextable::theme_zebra()
```

hauljoin	species_-code	cpue_-kgkm2	cpue_-nokm2	count	weight_kg	taxon_-confi-dence
-24,470	68578	84.5611	1,297.341	61	3.976	
-24,470	68580	2,703.0414	24,798.350	1,166	127.095	
-24,470	68590	348.6019	3,062.575	144	16.391	

```
# save outputs for later comparison
dat_catch_api <- dat
```

14.1.3. Load 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_spe
```

```
res <- httr::GET(url = paste0(api_link_species, "?offset=0&limit=10000"))

## 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
```

Explore the data contents:

```
# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

```
[1] "rows: 1014; cols: 6"
```

```
# learn about the structure of the data
summary(dat)
```

```
  species_code   scientific_name    common_name         id_rank         
 Min.   :    1   Length:1014        Length:1014        Length:1014       
 1st Qu.:22177   Class :character   Class :character   Class :character  
 Median :66868   Mode  :character   Mode  :character   Mode  :character  
 Mean   :50653                                                           
 3rd Qu.:75077                                                           
 Max.   :99999                                                           
                                                                         
     worms              itis        
 Min.   :     51   Min.   :  46861  
 1st Qu.: 127206   1st Qu.:  97781  
 Median : 254573   Median : 162045  
 Mean   : 293224   Mean   : 217907  
 3rd Qu.: 342060   3rd Qu.: 167487  
 Max.   :1699296   Max.   :1206057  
 NA's   :82        NA's   :132      
```

```r
# Print the first few lines of the data
dat |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("species_code", "worms", "itis"), #
      big.mark = "") |>
  flextable::theme_zebra()
```

species_code	scientific_name	common_name	id_rank	worms	itis
1		fish egg unid.			
2		fish larvae unid.			
3		fish unid.			

```r
# save outputs for later comparison
dat_species_api <- dat
```

14.2. Ex. Create zero-filled data using data loaded in last example

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

First prepare a table with all combinations 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) # now, redundant
```

Explore the data contents:

```
print(paste0("rows: ", nrow(comb), "; cols: ", ncol(comb)))
```

```
[1] "rows: 22070179; cols: 2"
```

```
comb |> head(3) |>
  flextable::flextable()  |>
  flextable::colformat_num(
      j = c("species_code", "hauljoin"),
      big.mark = "") |>
  flextable::theme_zebra()
```

species_-code	hauljoin
68578	11795
68578	11796
68578	11797

Now, using that table of combinations (here, called `comb`), join data to make a full zero-filled CPUE dataset. When all of the data have been full joined together, there should be the maximum number of rows in `comb`.

```
dat <- comb |>
  # add species data
  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/clean up 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))
```

```
TRUE Joining with `by = join_by(species_code)`
TRUE Joining with `by = join_by(hauljoin)`
TRUE Joining with `by = join_by(species_code, hauljoin)`
```

Explore the data contents:

```
# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

```
[1] "rows: 22070179; cols: 38"
```

```r
# learn about the structure of the data
summary(dat)
```

```
  species_code      hauljoin       scientific_name    common_name
 Min.   :    1   Min.   : -24955   Length:22070179    Length:22070179
 1st Qu.:21800   1st Qu.: -14713   Class :character   Class :character
 Median :66770   Median :  -5025   Mode  :character   Mode  :character
 Mean   :50356   Mean   : 289418
 3rd Qu.:74983   3rd Qu.: 816069
 Max.   :99999   Max.   :1225635

   id_rank              worms              itis              year
 Length:22070179    Min.   :     51   Min.   :  46861   Min.   :1982
 Class :character   1st Qu.: 126737   1st Qu.:  97160   1st Qu.:1997
 Mode  :character   Median : 254508   Median : 160846   Median :2007
                    Mean   : 266901   Mean   : 202211   Mean   :2006
                    3rd Qu.: 291581   3rd Qu.: 167452   3rd Qu.:2015
                    Max.   :1699296   Max.   :1206057   Max.   :2025
                    NA's   :1614127   NA's   :2538236
     srvy              survey          survey_name        survey_definition_id
 Length:22070179    Length:22070179    Length:22070179    Min.   : 47.0
 Class :character   Class :character   Class :character   1st Qu.: 47.0
 Mode  :character   Mode  :character   Mode  :character   Median : 52.0
                                                          Mean   : 69.1
                                                          3rd Qu.: 98.0
                                                          Max.   :143.0

     cruise          cruisejoin           haul         stratum
 Min.   :198201   Min.   :    -777   Min.   :  1   Min.   : 10.0
 1st Qu.:199701   1st Qu.:    -700   1st Qu.: 59   1st Qu.: 31.0
 Median :200701   Median :    -623   Median :116   Median : 61.0
 Mean   :200607   Mean   : 294711    Mean   :122   Mean   :140.5
 3rd Qu.:201501   3rd Qu.: 837799    3rd Qu.:176   3rd Qu.:211.0
 Max.   :202502   Max.   :1225395    Max.   :355   Max.   :794.0

   station            vessel_id      vessel_name         date_time
 Length:22070179    Min.   :  1.0   Length:22070179    Length:22070179
 Class :character   1st Qu.: 88.0   Class :character   Class :character
 Mode  :character   Median : 94.0   Mode  :character   Mode  :character
                    Mean   :111.2
```

```
                   3rd Qu.:148.0
                   Max.   :178.0

latitude_dd_start longitude_dd_start latitude_dd_end longitude_dd_end
Min.   :51.19     Min.   :-180.0     Min.   :51.19   Min.   :-180.0
1st Qu.:54.73     1st Qu.:-169.9     1st Qu.:54.73   1st Qu.:-169.9
Median :56.99     Median :-163.4     Median :56.99   Median :-163.4
Mean   :56.65     Mean   :-137.1     Mean   :56.65   Mean   :-137.1
3rd Qu.:58.68     3rd Qu.:-152.1     3rd Qu.:58.68   3rd Qu.:-152.1
Max.   :65.34     Max.   : 180.0     Max.   :65.35   Max.   : 180.0
                                     NA's   :2251    NA's   :2251
bottom_temperature_c surface_temperature_c    depth_m       distance_fished_km
Min.   :-2.10        Min.   :-1.10        Min.   :   9.0   Min.   :0.135
1st Qu.: 3.10        1st Qu.: 5.90        1st Qu.:  71.0   1st Qu.:1.481
Median : 4.30        Median : 7.60        Median : 109.0   Median :1.675
Mean   : 4.12        Mean   : 8.06        Mean   : 141.3   Mean   :2.096
3rd Qu.: 5.40        3rd Qu.: 9.70        3rd Qu.: 166.0   3rd Qu.:2.804
Max.   :15.30        Max.   :18.10        Max.   :1200.0   Max.   :4.334
NA's   :1094226      NA's   :593432
 duration_hr     net_width_m     net_height_m     area_swept_km2
Min.   :0.025   Min.   : 7.51   Min.   : 0.00    Min.   :0.002314
1st Qu.:0.269   1st Qu.:15.54   1st Qu.: 2.58    1st Qu.:0.023848
Median :0.305   Median :16.34   Median : 6.17    Median :0.028048
Mean   :0.380   Mean   :16.38   Mean   : 5.18    Mean   :0.034451
3rd Qu.:0.500   3rd Qu.:17.16   3rd Qu.: 6.88    3rd Qu.:0.046246
Max.   :0.980   Max.   :23.82   Max.   :11.04    Max.   :0.077795
                                NA's   :1723925
 performance       cpue_kgkm2          cpue_nokm2              count
Min.   :0.0000   Min.   :      0.0   Min.   :        0.0   Min.   :     0.00
1st Qu.:0.0000   1st Qu.:      0.0   1st Qu.:        0.0   1st Qu.:     0.00
Median :0.0000   Median :      0.0   Median :        0.0   Median :     0.00
Mean   :0.2917   Mean   :     51.4   Mean   :      172.6   Mean   :     6.78
3rd Qu.:0.0000   3rd Qu.:      0.0   3rd Qu.:        0.0   3rd Qu.:     0.00
Max.   :7.0000   Max.   :3226234.7   Max.   :21780780.3   Max.   :867119.00

  weight_kg         taxon_confidence
Min.   :    0.000   Length:22070179
1st Qu.:    0.000   Class :character
Median :    0.000   Mode  :character
Mean   :    1.719
3rd Qu.:    0.000
```

```
 Max.   :18187.700
```

```
# Print the first few lines of the data
dat |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("species_code", "hauljoin", "year", "cruise", "cruisejoin", "worms", "itis"), #
      big.mark = "") |>
  flextable::theme_zebra()
```

species_-code	hauljoin	scien-tific_-name	com-mon_-name	id_rank	worms	itis	year	srvy
68578	11795	Hyas lyratus	Pacific lyre crab	species	442167	98422	1989	EBS
68578	11796	Hyas lyratus	Pacific lyre crab	species	442167	98422	1989	EBS
68578	11797	Hyas lyratus	Pacific lyre crab	species	442167	98422	1989	EBS

```
# save outputs for later comparison
dat_zerofill_api <- dat
```

14.3. Ex. Visualize zero-filled data for 2023 eastern Bering Sea walleye pollock in 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 %in% c("EBS", "NBS)`) walleye pollock (`species_code == 21740`).

```
dat <- dat_zerofill_api |>
  dplyr::filter(year %in% 2019:2023 &
                srvy %in% c("EBS", "NBS") &
                species_code == 21740) |>
  dplyr::select(year, common_name, longitude_dd_start, latitude_dd_start, cpue_kgkm2)

# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

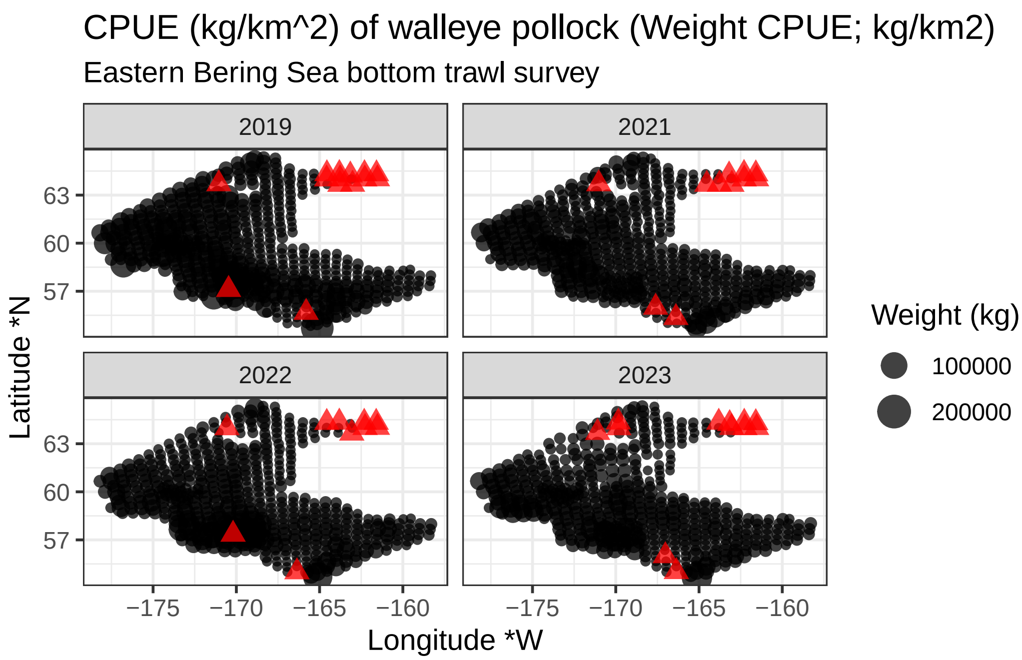
```
[1] "rows: 2052; cols: 5"
```

```
# # learn about the structure of the data
# summary(dat)

# Print the first few lines of the data
dat |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("year"),
      big.mark = "") |>
  flextable::theme_zebra()
```

year	common_name	longitude_dd_start	latitude_dd_start	cpue_kgkm2
2021	walleye pollock	-172.1196	57.35342	4,836.466
2021	walleye pollock	-171.4952	57.32779	12,709.182
2021	walleye pollock	-171.3904	57.00871	18,944.559

14.3.1. Plot locations on map

```
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 = "CPUE (kg/km^2) 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()
```



14.3.2. Plot inverse-distance weighted plot of CPUE

This map is constructed using `akgfmaps`. To make IDW plots, you must have data from all stations surveyed, even if no fish of interest were found there.

These plots are similar to those published in the annual Bering Sea data reports.

```
# devtools::install_github("afsc-gap-products/akgfmaps", build_vignettes = TRUE)
library(akgfmaps)
idw <- akgfmaps::make_idw_stack(
  x = dat |>
    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
```

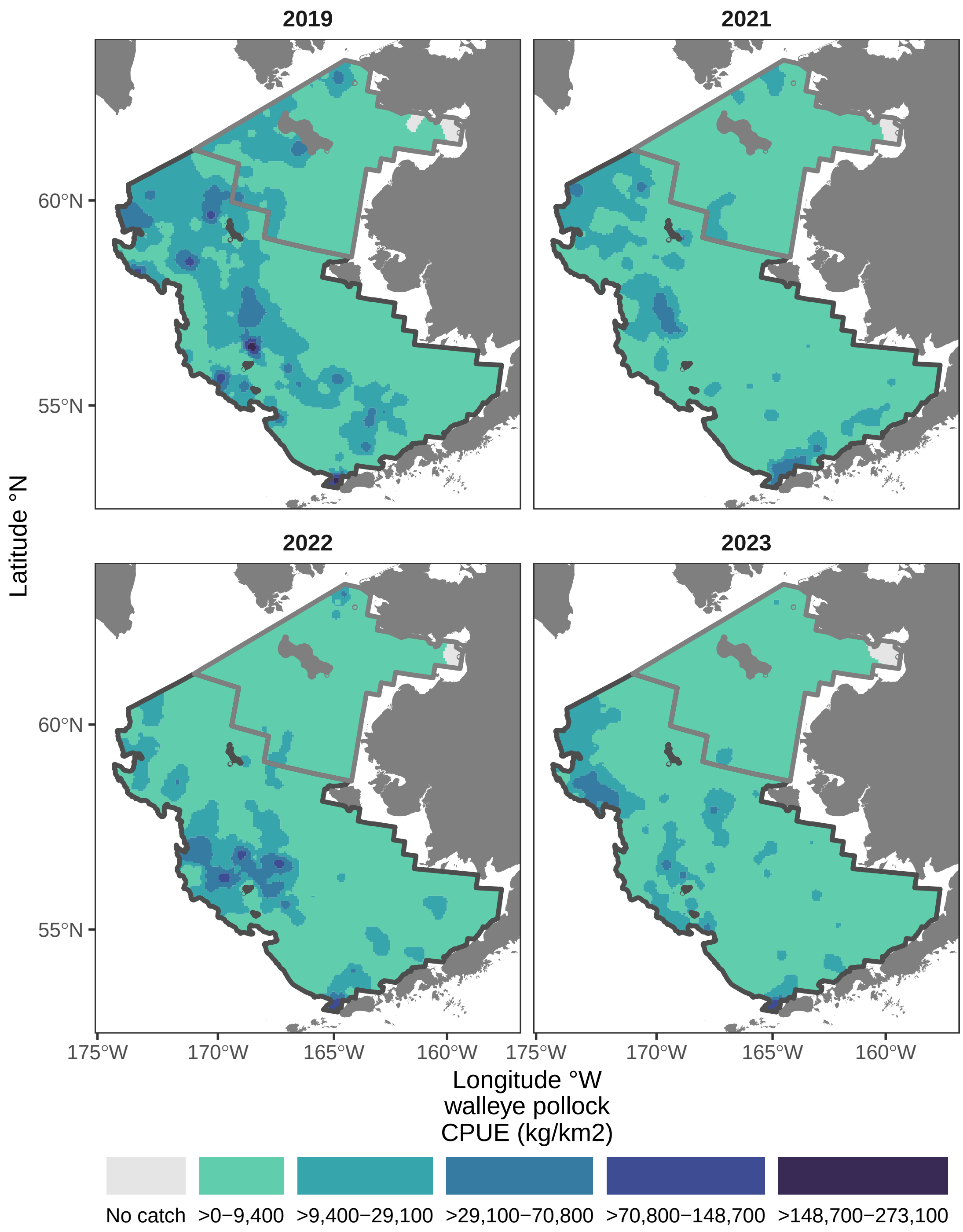
```
  out.crs = "EPSG:3338", # Set output coordinate reference system
  extrapolation.grid.type = "sf")
```

```
[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")

shps$survey.area$SRVY <- c("EBS", "NBS")
shps$survey.area$SURVEY <- c("EBS", "NBS")

# set.breaks <- akgfmaps::eval_plot_breaks(CPUE = dat$cpue_kgkm2, n.breaks = 5)
# set.breaks <- as.vector(unlist(set.breaks[set.breaks$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
```


2019
2021
2022
2023
60°N
55°N
60°N
55°N
Latitude °N
175°W
170°W
165°W
160°W
175°W
170°W
165°W
160°W
Longitude °W
walleye pollock
CPUE (kg/km2)
No catch
>0−9,400
>9,400−29,100
>29,100−70,800
>70,800−148,700
>148,700−273,100

14.4. Ex. Show catch data for 2023 eastern Bering Sea Walleye Pollock (one species in one survey region in one year)

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`).

Note: Check how many rows and columns are in the data pull. The eastern Bering Sea survey (before 2024) has 376 stations in it, and pollock are often found in throughout the region so this should have a similar number of rows.

```
## 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
```

```r
## Find how many rows and columns are in the data pull.
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

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

```
# save outputs for later comparison
dat_haul_ex <- dat
```

```
# Print the first few lines of the data
dat_haul_ex |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("year", "hauljoin", "cruise"),
      big.mark = "") |>
  flextable::theme_zebra()
```

year	srvy	survey	survey_-name	survey_-defini-tion_id	cruise	cruisejoin	hauljoin	haul
2023	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Groun fish Bottom Trawl Survey	98	202301	-759	-22283	64
2023	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Ground-fish Bottom Trawl Survey	98	202301	-759	-22284	63

year	srvy	survey	survey_-name	survey_-defini-tion_id	cruise	cruisejoin	hauljoin	haul
2023	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Groun fish Bottom Trawl Survey	98	202301	-759	-22285	66

14.4.1. 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
# 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 n
```

```
# Print the first few lines of the data
dat_species_ex |>
  head(3) |>
```

```
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("species_code"),
      big.mark = "") |>
  flextable::theme_zebra()
```

Table 14.8.: Walleye pollock species information.

species_code	scientific_name	common_name	id_rank	worms	itis
21740	Gadus chalcogrammus	walleye pollock	species	300,735	934,083

14.4.2. 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
  }
}
```

Explore data:

```r
# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

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

```r
# learn about the structure of the data
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.:  2281.20
 Median :-22540   Median :21740   Median :  3286.76   Median :  5863.07
 Mean   :-22553   Mean   :21740   Mean   :  6364.85   Mean   : 11540.65
 3rd Qu.:-22324   3rd Qu.:21740   3rd Qu.:  6956.25   3rd Qu.: 12456.99
 Max.   :-22110   Max.   :21740   Max.   :148679.68   Max.   :202321.08
     count          weight_kg         taxon_confidence
 Min.   :   1.0   Min.   :   0.492   Length:374
 1st Qu.: 113.2   1st Qu.:  71.560   Class :character
 Median : 284.0   Median : 162.310   Mode  :character
 Mean   : 572.8   Mean   : 315.419
 3rd Qu.: 616.5   3rd Qu.: 350.399
 Max.   :9997.0   Max.   :7346.495
```

```r
# Print the first few lines of the data
dat |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("hauljoin", "species_code"),
      big.mark = "") |>
  flextable::theme_zebra()
```

hauljoin	species_-code	cpue_-kgkm2	cpue_-nokm2	count	weight_kg	taxon_-confi-dence
-22283	21740	644.3718	963.6891	43	28.752	High
-22284	21740	3,554.5913	6,998.5742	317	161.005	High
-22285	21740	6,940.8696	9,808.6744	382	270.313	High

```
# save outputs for later comparison
dat_catch_ex <- dat
```

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}'))
```

14.4.3. 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))
```

Explore data

```r
# Find how many rows and columns are in the data pull
print(paste0("rows: ", nrow(dat), "; cols: ", ncol(dat)))
```

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

```r
# learn about the structure of the data
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.:  2268   1st Qu.: 112.0   1st Qu.:  70.64  
 Median :  3273   Median :  5842   Median : 280.0   Median : 161.44  
 Mean   :  6331   Mean   : 11479   Mean   : 569.8   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         
 Length:376         Length:376         Length:376         Length:376        
 Class :character   Class :character   Class :character   Class :character  
 Mode  :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
```

```r
# Print the first few lines of the data
dat |>
  head(3) |>
  flextable::flextable() |>
  flextable::colformat_num(
      j = c("year", "cruise", "cruisejoin", "species_code"),
      big.mark = "") |>
  flextable::theme_zebra()
```

year	srvy	survey	survey_-name	survey_-defini-tion_id	cruise	cruisejoin	hauljoin	haul
2023	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Groun fish Bottom Trawl Survey	98	202301	-759	-22,283	64
2023	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Ground-fish Bottom Trawl Survey	98	202301	-759	-22,284	63

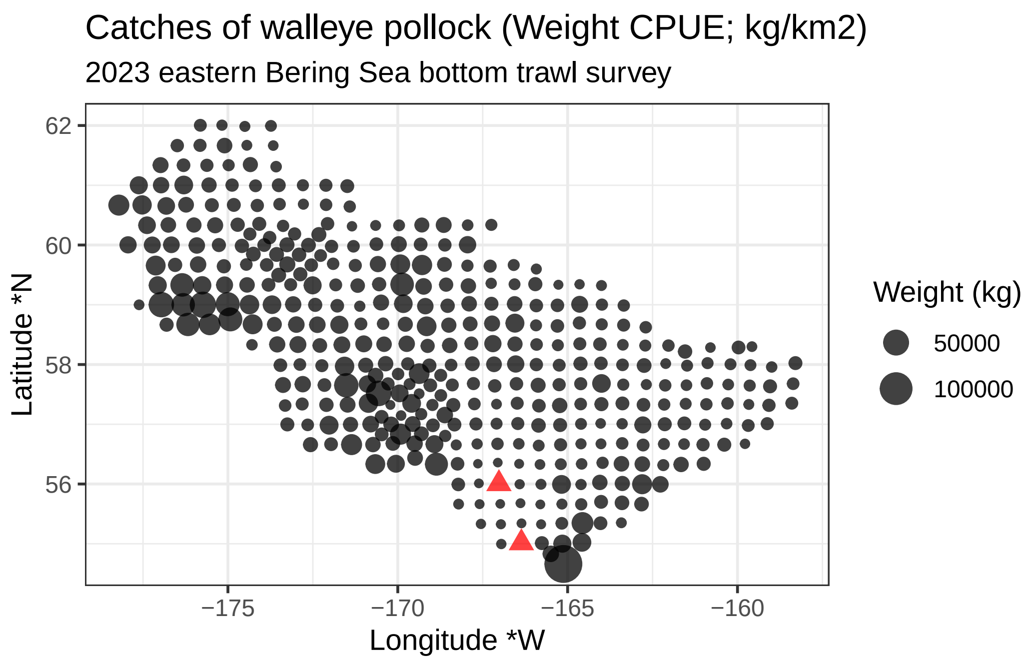
year	srvy	survey	survey_-name	survey_-defini-tion_id	cruise	cruisejoin	hauljoin	haul
2023	EBS	eastern Bering Sea	Eastern Bering Sea Crab/Groun fish Bottom Trawl Survey	98	202301	-759	-22,285	66

14.4.4. Visualize CPUE data in distribution map

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

14.5. 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()
```



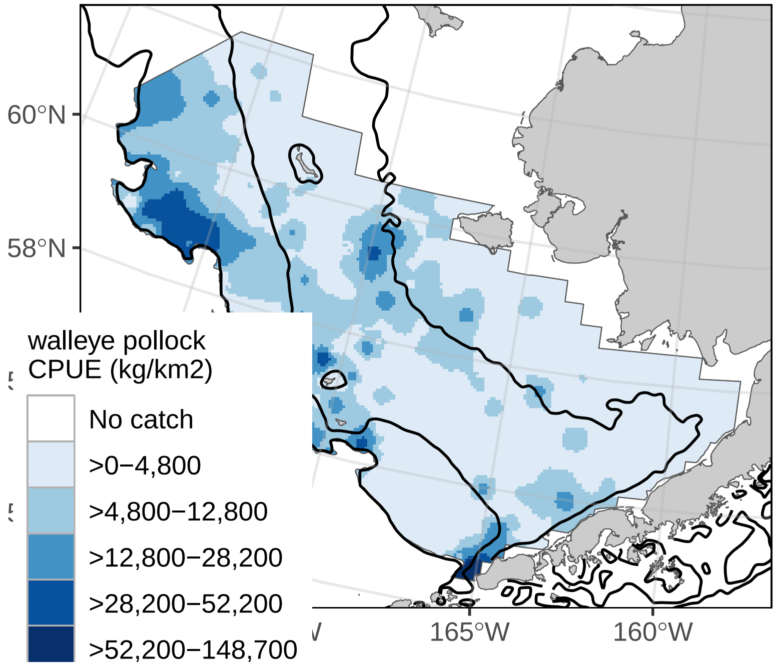
14.5.1. Plot inverse-distance weighted modeled product of locations

This map is constructed using `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
  extrapolation.grid.type = "sf")
```

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

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



15. Access via API and Python

15.0.1. {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
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.

15.0.2. 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 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.

15.0.3. 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.

15.0.4. 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.

15.0.5. Load into Pandas

The results from the `afscgap` package are serializable and can be loaded into other tools like Pandas. 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.

15.0.6. 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 query generation which dictates how the library communicates with the API service. However, users can provide raw ORDS queries as well using manual filtering.

15.0.7. 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.

15.0.8. More information

Please see the API documentation for the Python library for additional details.

16. Access via Oracle and R (AFSC Staff 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.

16.0.1. 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()
```

16.0.2. 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] # remo
haul <- readr::read_csv(file = here::here("data/gap_products_foss_haul.csv"))[,-1] # remove
species <- readr::read_csv(file = here::here("data/gap_products_foss_species.csv"))[,-1] #

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] # remo
haul <- readr::read_csv(file = here::here("data/gap_products_foss_haul.csv"))[,-1] # remove
species <- readr::read_csv(file = here::here("data/gap_products_foss_species.csv"))[,-1] #

# 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 en
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) )
```

16.0.3. 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_CONFI
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
```

16.0.4. Ex. Subset data

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

```r
# 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_-CONFI-DENCE	YEAR	SRVY
-19,461	21,720	646.8800	200.3345	10	32.29	High	2,019	EBS
-19,446	21,720	1,212.7733	164.7230	8	58.90	High	2,019	EBS
-19,422	21,720	313.8482	108.5980	6	17.34	High	2,019	EBS

16.0.5. 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 skate	Arctoraja parmifera	species	1,577,324

Part VI.

Data Products & Tools

To accompany these data, we also produce data products to make using our data more accessible and straightforward.

Table 16.3.: Survey of products developed by GAP

Product	Point of Contact AI	Point of Contact GOA	Point of Contact BS	Descri
Data				
Finalized bottom trawl data	Susanne McDermott	Ned Laman	Duane Stevenson	NOAA- trawl da post-su
Data requests	Alexandra Dowlin		Chris Anderson	To requ NOAA- trawl ra
Species codebook	Chris Anderson			List of c identifie NOAA- surveys
Survey protocols	Em?			Docum NOAA- bottom
Analysis				
Design-based indices for target species	Susanne McDermott	Ned Laman	Duane Stevenson	Standa and ab NOAA- trawl su
Design-based age or length composition	Susanne McDermott	Ned Laman	Duane Stevenson	Standa age co NOAA- trawl su
Model-based indices, age comps (stock assessment), area occupied, and COG (ESP)	Lewis Barnett			Spatiot indices compos NOAA- trawl su

Product	Point of Contact AI	Point of Contact GOA	Point of Contact BS	Descri
Annual bottom and surface temperature summary (ESR, stock assessment)	Rebecca Howard		Sean Rohan & Lewis Barnett	Summa and sur historic
Bering Sea cold pool index and temperature data products (ESR, ESP, stock assessment)	-		Sean Rohan & Lewis Barnett	Create EBS, ca temper visualiz
Annual fish condition (ESR)	Rebecca Howard, Sean Rohan, & Bianca Prohaska	Rebecca Howard & Bianca Prohaska	Bianca Prohaska & Sean Rohan	Ground in the E of Alasl
Rockfish indices vs environmental gradients (ESR)	Alexandra Dowlin & Christina Conrath		-	GOA/A abunda environ
Structure-Forming Invertebrates-Habitat Areas of Particular Concern (SFI-HAPC) (ESR)	Christina Conrath		Thaddeus Buser	Relative hydroco anemo and AI
Forage fishes (ESR)	-	Megsie Siple	-	Relative sandfis GOA ar
Miscellaneous species (ESR)	Sarah Friedman		Thaddeus Buser	Relative poache AI surv
Jellies (ESR)	Alexandra Dowlin		Thaddeus Buser	Relative and AI
Essential fish habitat	Megsie Siple		Sean Rohan	Habitat on spec every fi
Visualization Tools				
Alaska groundfish maps (CPUE, etc.)	Megsie Siple		Sean Rohan	
Communication				

Product	Point of Contact AI	Point of Contact GOA	Point of Contact BS	Descri
Annual survey data report	Megsie Siple, Bethany	Riggle, Alex Dowlin	Emily Markowitz, Sophia Wassermann, Nicole Charriere, Chris Anderson	Alaska Technic survey availab for each (https:/
ADF&G report of research activities	Alexandra Dowlin		Nicole Charriere & Rebecca Haehn	Report activity waters.
IPHC report of research activities	Ned Laman		Rebecca Haehn	
Plan team survey results presentation	Megsie Siple, Susanne McDermott	Megsie Siple, Ned Laman	Duane Stevenson	NOAA- their fin Plan Te attachn https:// council ground
Community highlights report	Susanne McDermott		Emily Markowitz	Compil NOAA- findings
Bottom Trawl Survey Temperature and Progress Maps	Ned Laman		Emily Markowitz	Near re temper Islands Bottom

17. Open source code

17.1. R Packages

17.1.1. akgfmaps R package

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

17.1.2. coldpool R package

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

17.1.3. akfishcondition R package

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

17.1.4. gapindex R package

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

Part VII.

Contact us

Thank you for using our data guide!

> This code is always in development. Find code used for various reports in the code releases.

This code is primarally maintained by:

Emily Markowitz (Emily.Markowitz AT noaa.gov; @EmilyMarkowitz-NOAA)

Zack Oyafuso (Zack.Oyafuso AT noaa.gov; @zoyafuso-NOAA)

Sarah Friedman (Sarah.Friedman AT noaa.gov; @SarahFriedman-NOAA)

Alaska Fisheries Science Center,

National Marine Fisheries Service,

National Oceanic and Atmospheric Administration,

Seattle, WA 98195

General questions and more specific data requests can be sent to nmfs.afsc.gap.metadata@noaa.gov or submitted as an issue on our GitHub Organization. The version of this data used for stock assessments can be found through the Alaska Fisheries Information Network (AKFIN). For questions about the eastern Bering Sea surveys, contact Duane Stevenson (Duane.Stevenson@noaa.gov). For questions about the Gulf of Alaska or Aleutian Islands surveys, contact Ned Laman (Ned.Laman@noaa.gov). For questions specifically about crab data in any region, contact Mike Litzow (Mike.Litzow@noaa.gov), the Shellfish Assessment Program lead.

For questions, comments, and concerns specifically about the Fisheries One Stop Shop (FOSS) platform, please contact us using the Comments page on the FOSS webpage.

18. Production run notes

Report run date: Monday, February 16, 2026

19. R Version Metadata

```
R version 4.5.2 (2025-10-31 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 22631)

Matrix products: default
  LAPACK version 3.12.1

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_4.5.2    fastmap_1.2.0     cli_3.6.5         tools_4.5.2
 [5] htmltools_0.5.9   otel_0.2.0        rstudioapi_0.18.0 yaml_2.3.12
 [9] tinytex_0.58      rmarkdown_2.30    knitr_1.51        jsonlite_2.0.0
[13] xfun_0.56         digest_0.6.39     rlang_1.1.7       evaluate_1.0.5
```

19.0.1. 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.

19.0.2. 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.

20. Acknowledgments

21. 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.

22. 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.

23. Technical Acknowledgments

This quarto book is based off the NOAA-quarto-book GitHub repo designed by Eli Holmes.

This repo and GitHub Action was based on the tutorial by Openscapes quarto-website-tutorial by Julia Lowndes and Stefanie Butland.

23.1. 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.

23.2. Collaborators

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

- Alaska Stock Assessments
- North Pacific Groundfish Stock Assessment and Fishery Evaluation Reports
- Groundfish Economic Status Reports for the Gulf of Alaska and Bering Sea and Aleutian Islands
- Alaska Marine Ecosystem Status Report Database
- Southeast Alaska Coastal Monitoring Survey Reports
- Alaska Fisheries Life History Database
- Essential Fish Habitat Research Plan in Alaska

24. Citations and References

25. Access Constraints

There are no legal restrictions on access to the data. They reside in public domain and can be freely distributed.

User Constraints: 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 AFSC Groundfish Assessment Program, as the source from which these data were obtained, in any publications and/or other representations of these data, is suggested.

26. References

Alaska Fisheries Information Network (AKFIN). (2024). *AFSC groundfish assessment program design-based production data*. NOAA Fisheries Alaska Fisheries Science Center, Groundfish Assessment Program; https://akfinbi.psmfc.org/analytics/; U.S. Dep. Commer. https://www.psmfc.org/program/alaska-fisheries-information-network-akfin

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, C. B., Rohan, S. K., Charriere, B. K., and Stevenson, D. E. (2024). *Results of the 2023 eastern and northern Bering Sea continental shelf bottom trawl survey of groundfish and invertebrate fauna* (NOAA Tech. Memo. NMFS-AFSC-487; p. 242). U.S. Dep. Commer. https://doi.org/10.25923/2mry-yx09

Markowitz, E. H., Wassermann, S., Rohan, S. K., Charriere, B. K., Anderson, C. B., and Stevenson, D. E. (2025). *Results of the 2024 eastern and northern Bering Sea continental shelf bottom trawl survey of groundfish and invertebrate fauna* (NOAA Tech. Memo. NMFS-AFSC-499; p. 203). U.S. Dep. Commer. https://doi.org/10.25923/8qa3-x785

NOAA Fisheries Alaska Fisheries Science Center. (2024). *Fisheries one stop shop public data: RACE division bottom trawl survey data query*. https://www.fisheries.noaa.gov/foss; U.S. Dep. Commer.

NOAA Fisheries Alaska Fisheries Science Center, Goundfish Assessment Program. (2024). *AFSC goundfish assessment program design-based production data*. https://www.fisheries.noaa.gov/alaska/science-data/groundfish-assessment-program-bottom-trawl-surveys; U.S. Dep. Commer.

Siple, M. C., Szalay, P. G. von, Raring, N. W., Dowlin, A. N., and Riggle, B. C. (2024). *Data report: 2023 gulf of alaska bottom trawl survey* (NOAA Tech. Memo. AFSC processed report; 2024-09). U.S. Dep. Commer. https://doi.org/10.25923/gbb1-x748

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. 2023-07; p. 230). U.S. Dep. Commer. https://doi.org/10.25923/85cy-g225