Marine and Coastal

National Environmental Science Program

FINAL REPORT
Project 5.9

MAKING MARINE ENVIRONMENTAL DATA MORE ASSESSMENT READY

Eduardo Klein-Salas, Benjamin Stepin, Thomas Galindo, Karishma Khana, and Natalia Atkins

© UNIVERSITY OF TASMANIA, INTEGRATED MARINE OBSERVATION SYSTEM

MAY 2026

Preferred citation

Klein-Salas E, Stepin B, Galindo T, Khana K, Atkins N (2026). *Making marine environmental data more assessment-ready: NESP MaC Hub Project 5.9 Final Report*. University of Tasmania / IMOS Integrated Marine Observing System, Hobart.

Copyright

All material in this publication is licensed under a Creative Commons Attribution 4.0 International Licence except content supplied by third parties and logos. Enquiries about the licence and any use of this document should be emailed to Eduardo Klein-Salas at eduardo.kleinsalas@utas.edu.au.

ISBN 978-1-922640-31-4

Acknowledgement

This project is supported with funding from the Australian Government under the National Environmental Science Program. The Marine and Coastal Hub is funded by the Australian Government under the National Environmental Science Program.

Disclaimer

The Hub has exercised due care and skill in preparing and compiling the information and data in this publication. Notwithstanding, the Hub, their employees and advisors disclaim all liability, including liability for negligence and for any loss, damage, injury, expanse or cost incurred by any person as a result of accessing, using or relying on any of the information or data in this publication to the maximum, extent permitted by law.

Cover images

Front: National Reef Monitoring Network. Credit: Antonia Cooper.
Back: Project generated map. Credit: Eduardo Klein-Salas.

This publication is available on the NESP Marine and Coastal Hub website:

www.nespmarinecoastal.edu.au

Contents

Executive summary **12**

1. Introduction **13**

1.1. General justification of the project 13
1.2. Relevant precedents and related initiatives 14
1.3. Project objectives 16
1.4. Expected project outcomes 16

2. Prioritisation of available datasets **18**

2.1. Prioritisation workshop 18
2.2. Workshop results 20
2.3. Key outcomes and cross-cutting themes 23
2.4. Recommended datasets for development 23

3. Selected datasets **25**

3.1. National Seagrass Data Aggregation 25
3.2. Seabird Aggregated Data Product 27
3.3. National Kelp Dataset 28
3.4. NRMN Photo Quadrat Aggregated Data Product 30
3.5. Dugong Aerial Survey Database 32
3.6. Aggregated AMSA Vessel Tracking Data Product 33

4. Extract–Transform–Load process **36**

4.1. Common architectural patterns 36
4.2. Seagrass Aggregated Data Product 37
4.2.1. Extract 37
4.2.2. Transform 37
4.2.3. Load 38
4.3. Seabird Aggregated Data Product 39
4.3.1. Extract 39
4.3.2. Transform 39
4.3.3. Load 40
4.4. National Kelp Dataset 42
4.4.1. Extract 42
4.4.2. Transform 42
4.4.3. Load 43
4.5. NRMN Photo Quadrat Aggregated Data Product 44
4.5.1. Extract 44
4.5.2. Transform 44
4.5.3. Load 45
4.6. Aggregated AMSA Vessel Tracking Data Product 46
4.6.1. Extract 46

4.6.2. Transform 46
4.6.3. Load 47
4.7. Dugong Aerial Survey Database 49
4.7.1. Extract 49
4.7.2. Transform 50
4.7.3. Load 50
4.8. Summary of common benefits 52
5. Training strategy 53
5.1. Purpose and learning objectives 53
5.2. Scope and deliverables 53
5.3. Video series structure 54
5.4. Content design approach 55
5.5. Production timeline and governance 56
5.6. Risks and assumptions 56
6. Conclusions 58
6.1. Summary of achievements 58
6.2. Applications for national assessment and reporting 59
6.3. Data gaps and recommendations for future opportunities 61
6.4. Limitations and future directions 62
6.5. Closing remarks 63
7. References 64
Appendices 67

List of figures

Figure 1: Impact framework diagram for Project 5.9. 17

Figure 2: Spatial distribution of the number of seagrass surveys aggregated into h3 hexagons of ~250Km² (resolution 4). 26

Figure 3: Spatial distribution of seabird occurrences aggregated in the seabird's product. 28

Figure 4: Spatial distribution of the number of Kelp annotations aggregated into h3 hexagons of ~250 Km² (resolution 4). 30

Figure 5: Spatial distribution of the number of benthic annotations aggregated into h3 hexagons of ~250Km² (resolution 4). 31

Figure 6: Spatial aggregation of the number of dugong occurrences in h3 hexagons of ~250Km² (resolution 4). 33

Figure 7: Spatial aggregation of the number of vessel AIS pings in h3 hexagons of ~250Km² (resolution 4). 35

List of tables

Table 1: Workshop participants by organisation 19

Table 2: Datasets recommended for development as assessment-ready data products 24

Table 3: Metadata summary – National Seagrass Data Aggregation 25

Table 4: Metadata summary – Seabird Aggregated Data Product 27

Table 5: Metadata Summary – National Kelp Dataset 28

Table 6: Metadata Summary – NRMN Photo Quadrat Aggregated Data Product 30

Table 7: Metadata Summary – Dugong Aerial Survey Database 32

Table 8: Metadata Summary – Aggregated AMSA Vessel Tracking Data Product 33

Table 9: Output Schema – Seagrass Aggregated Data Product 38

Table 10: Output Schema – Seabird Aggregated Data Product (principal fields shown) 40

Table 11: Output Schema – National Kelp Dataset 43

Table 12: Output Schema – NRMN Photo Quadrat Aggregated Data Product 45

Table 13: Output Schema – Aggregated AMSA Vessel Tracking Data Product 48

Table 14: Additional fields in the rejected records output 48

Table 15: Output Schema – Dugong Optimised Data Product 50

Table 16: Additional fields in the rejected records output 51

Table 17: Planned structure of the training video series 54

Acknowledgement of Country

The Marine and Coastal Hub acknowledges Aboriginal and Torres Strait Islander people as the first peoples and Traditional Owners and custodians of the land and waterways on which we live and work. We honour and pay our respects to Elders past, present and emerging.

Aboriginal and Torres Strait Islander peoples represent the world's oldest living culture.
We celebrate and respect this continuing culture and strive to empower Aboriginal and Torres Strait Islander peoples.

Statement on the use of Artificial Intelligence tools

Generative artificial intelligence tools (specifically Anthropic's Claude) were used as a writing aid during the preparation of this report. These tools assisted with structuring prose, formatting tables and references, and editing for clarity and consistency.

All scientific concepts, technical content, factual claims, analytical interpretations, data descriptions, and project-specific knowledge presented in this report originate entirely from the authors and the project team. The AI tools did not generate, supply or verify any of the underlying ideas, data, methods, or findings. The authors reviewed, verified, and edited all AI-assisted texts to ensure accuracy, completeness, and appropriate representation of the project's work.

The use of AI writing assistance does not diminish the intellectual contribution or responsibility of the authors. Full responsibility for the content of this report, including any errors or omissions, rests with the authors.

Project team

Team composition

Project Lead	Eduardo Klein-Salas
Support Officer	Benjamin Stepin
Data Engineers	Thomas Galindo, Karishma Khanna
Notebook Creators	Denisse Fierro-Arcos, Thomas Galindo, Karishma Khanna
Metadata	Natalia Atkins
Project Support	Nicole Kanisay
Video Production	Reuben Gilles-Clark
Research Assistants	Mathew Youssef, Xinlong Liu, Lizzy Oh

Project Steering Committee

Members

Tim Moltmann (Chair)	UTAS / IMAS
Fabrice Jaine	IMOS
Rowan Trebilco	CSIRO
Anthony Richardson	University of Queensland
Alice McGlashan	DCCEEW, State of the Environment
Penny Johnson	DCCEEW

Invited members

Damien Burrows	JCU, NESP MaC Hub
Alan Jordan	UTAS, NESP MaC Hub
Eleanor Hearder	DCCEEW, Environment Information Australia
Emily Kost	DCCEEW, Environment Information Australia
Steffan Howe	DCCEEW, Parks Australia

Acronyms and abbreviations

Acronym	Full name	Description
ABS	Australian Bureau of Statistics	Commonwealth statistical agency responsible for Environmental Economic Accounts
AIMS	Australian Institute of Marine Science	National research agency for tropical marine science
AIS	Automatic Identification System	International vessel tracking system using radio transponders
AMSA	Australian Maritime Safety Authority	Commonwealth authority responsible for maritime safety and vessel traffic data
AODN	Australian Ocean Data Network	National infrastructure for discovery, access and delivery of marine data
ARDC	Australian Research Data Commons	National programme investing in research data infrastructure
AUV	Autonomous Underwater Vehicle	Robotic underwater platform used for benthic imaging surveys
AWS	Amazon Web Services	Cloud computing platform used for S3 storage and processing
BOO	Biological Ocean Observer	IMOS application aggregating plankton and microbial observations
BOSS	Baited Observation Survey System	Stereo-video system for benthic fish and habitat surveys
BRUVS	Baited Remote Underwater Video System	Video-based method for surveying fish and shark assemblages
CAAB	Codes for Australian Aquatic Biota	Controlled vocabulary for Australian marine species maintained by CSIRO
CATAMI	Collaborative and Annotation Tools for Analysis of Marine Imagery	Standardised classification scheme for benthic image annotation
CBD	Convention on Biological Diversity	International treaty governing the Kunming-Montreal Global Biodiversity Framework
CC-BY	Creative Commons Attribution	Open-access licence requiring attribution (variants include NC, SA)
CSIRO	Commonwealth Scientific and Industrial Research Organisation	Australia's national science agency
CTS	Craft Tracking System	AMSA's operational vessel tracking database combining AIS and other sources

DCCEEW	Department of Climate Change, Energy, the Environment and Water	Australian Government department responsible for environmental policy
DMS	Data Management System	Data system developped for the Great Barrier Reef Authority
DOI	Digital Object Identifier	Persistent identifier for published research outputs
DOV	Diver-Operated Video	Underwater video survey method operated by scientific divers
DwC	Darwin Core	International standard for sharing biodiversity occurrence data
eDNA	Environmental DNA	DNA collected from environmental samples for species detection
EEA	Environmental Economic Accounts	National accounts integrating environmental asset and condition data
EIA	Environment Information Australia	Entity established under the Nature Positive Plan for environmental data supply
EML	Ecological Metadata Language	XML-based metadata standard used in Darwin Core archives
eMoF	ExtendedMeasurementOrFact	Darwin Core extension for contextual measurements linked to occurrences
EPBC	Environment Protection and Biodiversity Conservation	Commonwealth Act governing environmental assessment and species protection
EPSG	European Petroleum Survey Group	Registry of coordinate reference systems (e.g. EPSG:4326 = WGS 84)
ETL	Extract–Transform–Load	Data engineering workflow for ingesting, processing and storing data
FAIR	Findable, Accessible, Interoperable, Reusable	Guiding principles for scientific data management and stewardship
GA	Geoscience Australia	Commonwealth geoscience agency providing national mapping products
GBIF	Global Biodiversity Information Facility	International network and data infrastructure for biodiversity data
GBR	Great Barrier Reef	World Heritage–listed reef system off Queensland, Australia
GBRMPA	Great Barrier Reef Marine Park Authority	Commonwealth agency managing the Great Barrier Reef Marine Park
GIS	Geographic Information System	Software and methods for spatial data analysis and mapping

GMT	Greenwich Mean Time	Time standard equivalent to UTC, used for timestamp normalisation
GPS	Global Positioning System	Satellite-based navigation system used for field survey positioning
H3	Hexagonal Hierarchical Spatial Index	Uber-developed hierarchical hexagonal grid for spatial indexing
IMAS	Institute for Marine and Antarctic Studies	Marine and Antarctic research institute at the University of Tasmania
IMOS	Integrated Marine Observing System	Australia's national collaborative marine observing infrastructure
ISO	International Organization for Standardization	International standards body (e.g. ISO 19115, ISO 8601)
JCU	James Cook University	University in Queensland; custodian of dugong aerial survey data
LSID	Life Science Identifier	Globally unique persistent identifier for biological taxa
MaC	Marine and Coastal	NESP research hub focused on marine and coastal environments
NCRIS	National Collaborative Research Infrastructure Strategy	Australian Government programme funding national research infrastructure
NESP	National Environmental Science Program	Australian Government programme funding applied environmental research
NOPSEMA	National Offshore Petroleum Safety and Environmental Management Authority	Commonwealth regulator for offshore petroleum and greenhouse gas activities
NRMN	National Reef Monitoring Network	IMOS facility for long-term reef monitoring using standardised survey methods
OBIS	Ocean Biodiversity Information System	Global open-access platform for marine species occurrence data (UNESCO-IOC)
OGC	Open Geospatial Consortium	International body developing standards for geospatial web services
RLS	Reef Life Survey	Citizen-science reef monitoring programme contributing to NRMN
S3	Simple Storage Service	AWS cloud object storage used for hosting data products
SAR	Search and Rescue	AMSA-defined maritime region bounding Australian vessel tracking coverage
SOE	State of the Environment	Periodic national assessment of Australia's environmental condition

STAR	Status and Trends of Australia's Oceans Report	IMOS report synthesising ocean observing data for national assessment
TEP	Threatened, Endangered and Protected	Classification for species listed under the EPBC Act
UAV	Unmanned Aerial Vehicle	Drone platform used for aerial mapping of coastal habitats
UNESCO	United Nations Educational, Scientific and Cultural Organization	UN agency; hosts the Intergovernmental Oceanographic Commission
UNSW	University of New South Wales	Australian university represented at the prioritisation workshop
UQ	University of Queensland	Australian university; Steering Committee member affiliation
UTAS	University of Tasmania	Host institution for IMOS and Project 5.9
UWA	University of Western Australia	Australian university contributing to marine data programmes
WAMSI	Western Australian Marine Science Institution	Collaborative marine research institution in Western Australia
WKB	Well-Known Binary	Binary encoding for geometric data (points, polygons)
WKT	Well-Known Text	Text-based encoding for geometric data used in GeoServer exports
WoRMS	World Register of Marine Species	Authoritative global taxonomic database for marine species

Executive summary

Australia possesses world-class marine research data, yet a persistent gap remains between these well-managed research datasets and the assessment-ready information required for State of the Environment (SOE) reportings, Environmental Economic Accounts (EEA), environmental approvals and the national environmental data supply chain envisaged under the Nature Positive Plan. NESP Marine and Coastal Hub Project 5.9 was designed to close this gap by building the data infrastructure needed to routinely transform priority marine datasets into cloud-optimised, analysis-ready products sustained through IMOS/AODN national infrastructure.

A national prioritisation workshop (February 2025; 40+ participants, 16 organisations including DCCEEW, IMOS, AIMS, Parks Australia, GBRMPA, NOPSEMA, CSIRO and ABS) selected six datasets for development: a national seagrass aggregation (83 source datasets, 1967–2024), a seabird product (41 Darwin Core archives, 1939–2025), kelp annotations from Squidle+, NRMN reef benthos photo quadrats, dugong aerial surveys and AMSA vessel tracking (2012–2025). These span the three thematic pillars of marine assessment: habitats, species and pressures.

Automated Extract–Transform–Load pipelines, were orchestrated using Prefect, the AODN standard pipeline tool and validated through defined schemas. Heterogeneous sources were harmonised and converted into standard Parquet files. All biological datasets are aligned to WoRMS and CAAB taxonomies. Products are enriched with H3 hexagonal spatial indices for multi-resolution aggregation and Australian Marine Region tags for alignment with national reporting geographies. Compression ratios exceed 88%, and all products are accessible from Python or R with standard open-source libraries.

All pipelines are integrated into IMOS/AODN operational infrastructure, with products refreshed automatically as new source data arrive – ensuring sustained delivery beyond the project's funding cycle. Products are discoverable through the AODN portal with ISO 19115 metadata and OGC service protocols, interoperable with DCCEEW systems and global platforms including OBIS and GBIF. A companion training programme of screencast videos (9 episodes) with Python and R code repositories builds practitioner capability across government, research and consultancy sectors.

The products have immediate applications for SOE 2026 and the biennial cycle from 2028, EEA preparation, environmental approvals under the EPBC Act (including offshore renewable energy), the Sustainable Ocean Plan, marine park management, and threatened and migratory species assessments. The project demonstrates a replicable model for transforming research data into assessment-ready information within national infrastructure, directly addressing the gap identified by the 2020 Samuel Review.

1. Introduction

1.1. General justification of the project

Australia's marine estate supports globally significant biodiversity and underpins critical ecosystem services that sustain coastal communities, industries and First Nations Sea Country. Effective stewardship of these resources is dependent on timely, authoritative environmental information that can be used in assessment and decision-making processes. Yet, as the 2020 Independent Review of the *Environment Protection and Biodiversity Conservation Act 1999* (the "Samuel Review") observed, "decision-makers, proponents and the community do not have access to the best available data, information and science", resulting in "sub-optimal decision-making, inefficiency and additional cost for business, and poor transparency for the community" (Samuel, 2020).

Australia is fortunate to have world-class marine research facilities and institutions. Through sustained investment in the Integrated Marine Observing System (IMOS), the Australian Ocean Data Network (AODN), and successive rounds of the National Environmental Science Program (NESP), the country has amassed extensive marine research datasets spanning physical oceanography, biological observations, habitat mapping and vessel tracking (IMOS, 2023). The AODN, which manages the IMOS data collection programme, incorporates contributions from universities and government research agencies. The AODN stands at the forefront of marine data management in Australia. Additional mapping tools, principally Seamap Australia (UTAS/IMAS) and eAtlas (AIMS), complement AODN and are supported through NESP. Collectively, these systems manage data that are findable, accessible, interoperable and reusable (FAIR; Wilkinson et al., 2016) and follow international standards for metadata formatting, discovery and sharing.

In practice, however, the best available marine environmental research data are not routinely consumed by the periodic assessment processes that most need them. The State of the Environment (SOE), Environmental Economic Accounts (EEA), environmental approval processes, and systems for tracking environmental trends and outcomes all require data on the status and trend of marine habitats, species and pressures. Making research data "assessment ready" for these processes requires additional analysis, reformatting, aggregation, quality control and infrastructure that fall outside the scope of the original research programmes (Samuel, 2020; Evans, 2024). This gap between well-managed research data and assessment-ready environmental information is the central problem that Project 5.9 was designed to address.

In the context of this project, "assessment readiness" is defined as the state in which data from multiple disparate sources have been harmonised to a minimum set of common variables relevant to the generation of environmental indicators or analysis, quality-controlled against authoritative taxonomic and spatial vocabularies, and made available through a single access point so that users can work with the data efficiently without needing to navigate the complexity of the original source collections. Assessment-ready data do not replace the underlying research datasets; rather, they add a standardised analytical layer above them, bridging the gap between research data management and policy consumption.

Project 5.9 delivers assessment-ready products in Apache Parquet, a cloud-optimised columnar file format. Parquet was selected because it is self-describing: each file embeds the data type, field name and description for every variable, analogous to the global and variable attributes in netCDF files. This

overcomes the poorly described CSV format. Because the data are organised by columns rather than rows, users can read individual variables without downloading the full dataset – a critical advantage when working with products such as the AMSA vessel tracking dataset, which contains more than 200 million records and cannot be practically loaded as a single file on an ordinary computer. Parquet files can also be partitioned across multiple files according to criteria that match anticipated query patterns (for example, by year), so that only the relevant partitions are read during a spatial or temporal filter. Users connect to the data remotely at a documented access point, apply filters for space, time or any other variable, and retrieve only the subset needed for their analysis – without maintaining a local copy. This remote, filter-and-extract access pattern is well established in remote sensing applications and very large Earth observation datasets, and is increasingly adopted for tabular environmental data. It is particularly well suited to modelling workflows and operational information dashboards, where the analytical tool queries the data at source rather than ingesting a static extract.

The Australian Government's response to the Samuel Review, The December 2022 Nature Positive Plan (Australian Government, 2022), established Environment Information Australia (EIA) to deliver "clear, authoritative sources of high-quality environmental information" through a national environmental data supply chain. Marine environmental research data is in scope for this supply chain. With the next SOE due in 2026, and the reporting frequency increasing to biennially from 2028 (previously every five years), the need for routine delivery of assessment-ready marine data products has become increasingly urgent. Additional drivers include the preparation of Environmental Economic Accounts, significant environmental approval processes (including for offshore renewable energy development), the Sustainable Ocean Plan, and Australia's commitments under the Kunming-Montreal Global Biodiversity Framework (CBD, 2022), including the "30 by 30" target.

Project 5.9 responds to this convergence by building the information infrastructure required to routinely deliver marine environmental data and products that are assessment ready in line with the Nature Positive Plan. Unlike short-term, time-bound projects with no plan for sustainability, the data products associated with this project have been built by and integrated into IMOS data infrastructure, ensuring they are sustained over time as part of business-as-usual AODN operations.

1.2. Relevant precedents and related initiatives

Project 5.9 builds on several important precedents that have progressively advanced the availability of assessment-ready marine information in Australia.

IMOS Status and Trends of Australia's Oceans Report (STAR) 2019

The IMOS STAR report (Richardson et al., 2020) provided a proof of concept for transforming research-grade observational data into products suited for environmental assessment. STAR synthesised long-term physical, chemical and biological time series from IMOS observing platforms to characterise the status and trends of Australian ocean conditions. Elements of the STAR report were subsequently taken up in the 2021 State of the Environment report (Trebilco et al., 2022), demonstrating the capability potential for research data to inform national assessments when appropriately packaged. Project 5.9 extends the STAR approach by broadening the range of datasets, delivering them in cloud-optimised formats, and establishing automated data pipelines for sustained, replicable production.

State of the Environment reports (2016 and 2021)

The 2016 and 2021 SOE reports highlighted both the value and the limitations of available marine environmental data. The 2016 Marine chapter (Evans et al., 2017) drew on shipping density products derived from AIS vessel tracking data, several of which were published through AODN. The 2021 report (Trebilco et al., 2022) advanced the use of marine research data and, for the first time, incorporated significant First Nations co-design in its preparation. However, both reports revealed persistent gaps in the availability of standardised, readily accessible data products for key marine habitats, species and pressures. Compiling marine indicators often required bespoke, labour-intensive data extraction and analysis by individual authors, a process that is not conducive to the increased reporting frequency that is now mandated. Project 5.9 directly addresses this limitation by pre-building the aggregated, quality-controlled data products that SOE authors and other assessors require.

AODN Cloud Optimised Initiative and Biological Ocean Observer

IMOS/AODN has been progressively developing cloud-optimised data capabilities, including the conversion of key datasets to Analysis-Ready Cloud-Optimised (ARCO) formats such as cloud-optimised Zarr and Parquet (IMOS, 2023). The Biological Ocean Observer (BOO), developed in collaboration with AODN (Davies et al., 2022) demonstrated the feasibility of aggregating biological observations from disparate sources and serving them through standardised web services. More recently, AODN has received additional funding under the National Collaborative Research Infrastructure (NCRIS) programme for a data uplift initiative, which focuses on making more non-IMOS data available through AODN. Project 5.9 leverages these capabilities and contributes directly to the AODN cloud-optimised ecosystem.

Related NESP MaC Hub and External Projects

Several other NESP MaC Hub projects have contributed data and methodological foundations that Project 5.9 draws upon. Project 3.21 (Evans, 2024) identified priority datasets of relevance to the Gippsland declaration area for offshore renewable energy assessments, highlighting the challenge of data discoverability and the need for systematic aggregation. The Protected Place Management Initiative scoping study (Project 1.1; Gordon and Dunstan, 2022) identified the consistent need across government for improved information to support evidence-based decision-making in protected places. Project 1.17 (Stoeckl et al., 2022) reviewed integrated data requirements for natural resource management, concluding that there is insufficient data available to adequately describe the socio-ecological systems needed for effective policy. These projects collectively reinforced the need for the systematic data infrastructure development undertaken by Project 5.9.

Externally, the Australian Research Data Commons (ARDC) has been investing in a Trusted Environmental Data and Information Supply Chains project focused on offshore renewable energy development in the Gippsland region (ARDC, 2024). While that project focuses on enabling access to potentially sensitive industry data, it is complementary to Project 5.9's focus on making openly accessible research data assessment ready.

1.3. Project objectives

Project 5.9 – Making marine environmental data more assessment ready was designed to address the gap between well-managed marine research data and assessment-ready environmental information products. The project pursued three interlinked objectives:

- **Identify priority datasets** currently managed as research data, through structured consultation with research-users at DCCEEW and other primary stakeholders, documented through a formal prioritisation workshop process.
- **Build the additional information infrastructure** required to transform priority research datasets into analysis-ready, cloud-optimised data products. This encompassed the design and implementation of Extract–Transform–Load (ETL) workflows, schema validation, quality control pipelines, spatial indexing (H3; Brodsky, 2018), and integration with existing AODN infrastructure for sustained delivery.
- **Routinely deliver marine environmental data and products that are assessment ready**, accessible through standard Open Geospatial Consortium (OGC) protocols, stored in cloud-optimised formats (Parquet; Apache Software Foundation, 2024), and discoverable through the AODN portal with ISO 19115–compliant metadata records. The delivery model ensures that products are updated as source data are refreshed, supporting recurring assessment cycles.

These objectives were pursued with explicit attention to sustainability. Unlike many previous efforts that delivered data products as one-off project outputs, the software development and data workflows created by Project 5.9 were integrated into IMOS national research infrastructure from the outset, ensuring that the products will continue to be maintained and updated as part of routine AODN operations beyond the life of the project.

1.4. Expected project outcomes

The project was designed within a research impact framework (Figure 1) that traces the pathway from scientific inputs through to policy and management impacts. The expected outcomes span four interconnected domains:

Improved SOE reporting

By delivering pre-built, quality-controlled, assessment-ready data products for priority marine habitats, species and pressures, the project reduces the analytical burden on Environment Information Australia and SoE assessors, and improves the consistency and timeliness of marine indicators. With SOE reporting frequency increasing to biennial from 2028 (Australian Government, 2022), the availability of routinely updated data products is expected to be a significant enabler of the new reporting cycle.

Enhanced environmental economic accounts

The standardised, spatially indexed datasets produced by the project can feed directly into the preparation of Environmental Economic Accounts, which require consistent spatial and temporal data on environmental assets and their condition (ABS, 2024).

Support for environmental approvals and trend tracking

Assessment-ready marine data products support environmental approval processes, including those for offshore renewable energy development, and ongoing tracking of environmental trends and outcomes. The H3 spatial indexing (Brodsky, 2018) and cloud-optimised formats enable rapid spatial queries and time-series extraction at scales relevant to regulatory decision-making.

Contribution to the national environmental data supply chain

The project's outputs are designed to be consumed by the EIA national environmental data supply chain. By using national and international data standards (OGC protocols, ISO 19115 metadata, Darwin Core (Wieczorek et al., 2012), CAAB and WoRMS (WoRMS Editorial Board, 2024) taxonomies), the products are interoperable with GIS systems and with global biodiversity information systems such as OBIS (OBIS, 2024).

Beyond these direct outcomes, the project is expected to identify gaps where insufficient data currently exists to inform required policy and management responses, thereby guiding future investment in marine observing and data collection. The project also contributes to the NESP MaC Hub's cross-cutting initiatives on Protected Place Management, Threatened and Migratory Species, and Climate Adaptation, by ensuring that the best available marine research data underpins the evidence base for each of these priority areas.

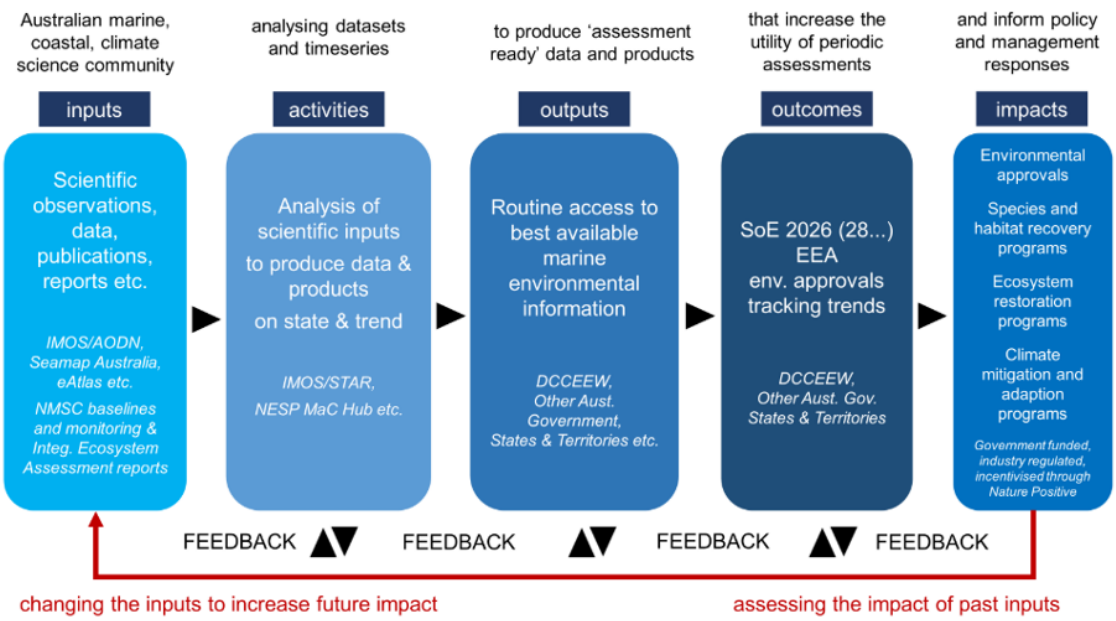


Figure 1: Impact framework diagram for Project 5.9.

2. Prioritisation of available datasets

A structured prioritisation process was conducted at the beginning of Project 5.9 to ensure that the datasets selected for transformation into assessment-ready products were those most likely to meet the needs of key research and reporting users. This chapter describes the prioritisation workshop, summarises the key discussions and their outcomes, and presents the datasets recommended for development within the project's scope and timeline.

2.1. Prioritisation workshop

Workshop design and objectives

The NESP 5.9 kick-off workshop was held on 20 February 2025 in Hobart, with remote participation from Canberra-based government agencies and other partners. The workshop was the first milestone of the project and was designed to bring together data custodians, research users and infrastructure providers to jointly identify priority datasets and agree upon the scope of work for the project's twelve-month development period.

The workshop pursued three interlinked objectives. First, to identify priority user needs across the principal assessment and policy processes that consume marine environmental data. These included the State of the Environment, Parks Australia management effectiveness evaluation, GBRMPA Reef health assessments, Environmental Economic Accounts (ABS), Blue Carbon accounting, threatened, endangered and protected (TEP) and migratory species assessments, environmental approvals for offshore renewable energy, the Sustainable Ocean Plan, and Environment Information Australia. Second, to map available data sources across NESP projects, AODN, IMOS facilities (including the National Reef Monitoring Network, Reef Life Survey, Underwater Marine Imagery via Squidle+, the Animal Tracking Facility, the Biological Ocean Observer and OceanCurrent), AusSeabed (Geoscience Australia), Seamap Australia (UTAS), and Global Archive (UWA). Third, to reach agreement on potential priorities using a simple scoring framework that assessed each candidate dataset against three criteria:

a. **Scope** – whether the dataset fell within the project's remit (Yes / No / Maybe);

b. **Readiness** – the current level of data maturity and accessibility (High / Medium / Low); and

c. **Potential priority** – a combined judgement reflecting both scope and readiness, used to guide final selection.

This framework allowed workshop participants to distinguish between datasets that were both desirable and feasible within the project timeline, and those that, while important, would require more extensive development than the current project could support.

Participants

The workshop convened over 40 participants from 16 organisations spanning Commonwealth government agencies, research institutions, data infrastructure providers and industry partners (Table 1). Representation from DCCEEW was particularly strong, with 11 attendees drawn from sections responsible for SoE reporting, marine species conservation, migratory species, blue carbon, sustainable ocean planning, enterprise science partnerships and environmental approvals. This breadth of government representation ensured that the prioritisation reflected the full range of current and emerging assessment requirements.

Table 1: Workshop participants by organisation

Organisation	Representatives
NESP MaC Hub	Damien Burrows, Alan Jordan, Paul Hedge, Tim Moltmann
IMOS	Benjamin Stepin, Eduardo Klein Salas, Fabrice Jaine
AIMS	Eric Lawrey
DCCEEW	Alice McGlashan, Amanda Richley, Anna Copley, Chris Okey, David Logan, Karita Negandhi, Milena Kiatkoski Kim, Peter Yates, Rebecca Haughey, Scott Francis, Shannon Scullion
Geoscience Australia	Andrew Carroll, Donna-Marie Audas
UWA	Tim Langlois
WAMSI / SEAF	Brendan Busch, Chris Gentle
NOPSEMA / OIR	Cam Sim
CSIRO	Clair Davies, Gabriela S. Pilo, Jason Everett, Jodie van de Kamp, Katherin Tattersall, Piers Dunstan, Rowan Trebilco
ARDC	Julia Martin, Kerry Levett, Paula Andrea Martinez
GBRMPA	Karen Chong Seng, Neil Collier
UNSW	Mitch Lyons
ABS	Penny Johnson
IMAS / UTAS	Emma Flukes, Jacquomo Monk, Peter Walsh, Vanessa Lucieer
Parks Australia	Steffan Howe
Greybits	Ariell Friedman

Workshop structure

The workshop was organised into three thematic sessions, each structured around the major categories of marine environmental information identified in the SOE framework and aligned with the Nature Positive Plan's data requirements. Session 1 (11:15–12:15 AEST) addressed the state and trend of marine habitats and communities, covering seabed habitats (seagrass, mangroves, salt marsh, kelp, rocky reefs, corals, sponges, bryozoans), water column habitats (canyons, seamounts, oceanic reefs) and associated communities. Session 2 (12:15–13:15 AEST) focused on the status of marine species, including sharks and rays, marine fish, marine invertebrates, seabirds, marine turtles, sea snakes and marine mammals. Session 3 (13:45–14:45 AEST) considered marine ecosystem processes and pressures, encompassing water clarity, connectivity, food webs, microbial communities, primary and secondary productivity, species redistribution, climate and climate change pressures, and population pressures.

Prior to the thematic sessions, the project team presented an overview of the current data landscape, noting that the AODN catalogue held over 14,000 metadata records, the IMOS catalogue contained 1,712 records, the AODN portal served 318 datasets, and the NESP MaC Hub held 174 project records. Of the 120 datasets proposed under the Hub, 41 were available at the time of the workshop, with average accessibility and maturity ratings that were very good for completed datasets. The presentation also highlighted the existing cloud-optimised capabilities within AODN and the range of complementary data systems (Seamap Australia, eAtlas, AusSeabed, Global Archive, CSIRO Marine National Facility) available for integration.

2.2. Workshop results

The workshop discussions yielded detailed assessments of data availability, quality and readiness across the three thematic areas. The following sections summarise the key findings from each session.

Habitats and communities

Seagrass

The seagrass discussion, led by Vanessa Lucieer (IMAS), Emma Flukes (IMAS) and Eric Lawrey (AIMS), identified a strong foundation of existing datasets that could be integrated into a national seagrass layer. Key sources included Seamap Australia, which continuously assimilates seagrass mapping data from projects across Australia; remote sensing datasets for temperate seagrass mapping produced under NESP project 3.6 with accompanying validation surveys; and multiple national seagrass mapping projects from JCU's Centre for Tropical Water and Aquatic Research (TropWATER). Participants highlighted data gaps in areas such as Bass Strait and emphasised the importance of validation surveys and ground-truthing remote sensing products. A consensus was reached that the volume and quality of available seagrass data were sufficient to warrant development of a national aggregated product, provided that careful attention was paid to harmonising the disparate source schemas, date conventions and species encodings.

Mangroves and salt marshes

Updates were presented on the Geoscience Australia 10-metre resolution annual product mapping intertidal seagrass, mangroves and salt marsh, along with ongoing validation efforts and the challenge of differentiating salt marsh from salt flats. National mangrove extent mapping, particularly in northern Australia, was noted as progressing well. While these datasets were considered important, they were assessed as lower priority for Project 5.9 due to the Geoscience Australia product already providing a well-maintained national baseline with regular updates, that are predominantly raster-based products that differ from the tabular, point-observation data that form the core of the project's ETL infrastructure.

Kelp

The kelp discussion, led by Chris Okey (DCCEEW), Alan Jordan (IMAS) and Jacquomo Monk (IMAS), identified challenges in mapping kelp due to depth variations and complex habitat structures. Reef Life Survey, the National Reef Monitoring Network, AUV data and machine learning tools were highlighted as sources for improved kelp detection. The need for a national kelp dataset integrating existing resources was strongly endorsed. The Squidle+ marine imagery platform was identified as a practical source of kelp annotations that could be transformed into a standardised product using a Darwin Core–compliant approach.

Water column habitats and communities

Discussions addressed the use of the IMOS Biological Ocean Observer for analysing plankton and microbial community data, and the expansion of acoustic monitoring through National Reference Stations and Ships of Opportunity. These datasets were noted as important for long-term ecosystem process monitoring but were assessed as requiring further development before they could be delivered as assessment-ready products within the project timeframe.

Species

Seabirds and shorebirds

The need for coordinated national data collection for improved seabird conservation strategies was a strong theme. Multiple Darwin Core archives held by the OBIS Australia Node at CSIRO were identified as a high-readiness source that could be aggregated into a single harmonised dataset. The archives span at-sea observations, vessel-based wildlife surveys and satellite tracking data, with temporal coverage extending from the late 1930s to present. Participants agreed that an aggregated seabird product would directly support SOE reporting, Parks Australia management effectiveness evaluation and migratory species assessments.

Sharks and rays

Tracking data and population assessments for key species were discussed, with Global Archive, Reef Life Survey and BRUVS data identified as relevant monitoring sources. While important, the complexity of integrating these diverse data streams was noted, and participants assessed this area as requiring further scoping before a cloud-optimised product could be delivered.

Marine fish

Assessment of fish populations through long-term underwater video data, including Reef Life Survey and BRUVS, was discussed. These data streams were considered complementary to the reef benthos products and could potentially be developed in future iterations of the project, noting that some of these datasets are already delivered by AODN.

Marine mammals and turtles

Challenges in data sharing and integration for monitoring were highlighted. Population assessments using tracking networks and acoustic monitoring data were discussed. Dugong aerial survey data from JCU was identified as a specific candidate for development, given the long-term, systematic nature of the survey programme and its direct relevance to threatened species assessments in northern Australian waters.

Ecosystem processes and pressures

Vessel traffic

The AMSA vessel tracking dataset was identified as a high-priority pressure layer with strong policy demand from SOE reporting, the Sustainable Ocean Plan and environmental approvals processes. Monthly Shapefile deliveries from AMSA, covering AIS and Craft Tracking System data from 2012 to present, were assessed as having high readiness for transformation into a cloud-optimised product. The workshop noted that previous SOE reports (2016 and 2021) had already drawn on shipping density products, but these had been produced as one-off analytical exercises rather than as routinely updated data products. Participants from DCCEEW and NOPSEMA confirmed strong demand for a sustained, standardised vessel traffic product.

Other pressures

Discussions also covered national microplastics monitoring, updates on fishing pressure layers, development of national underwater noise guidelines and acoustic monitoring data integration. These were recognised as important emerging datasets but were assessed as requiring further standardisation and development before they could be delivered as assessment-ready products within the project scope.

Oceanographic data and ecosystem modelling

The use of satellite data for monitoring sea surface temperature and anomalies, connectivity models for assessing marine ecosystem trends, and infrastructure needs for long-term reproducible monitoring were discussed. These capabilities were noted as complementary to the assessment-ready products being developed and were identified as potential areas for future extension of the project’s approach, noting that many of these products are already available at AODN.

2.3. Key outcomes and cross-cutting themes

Several cross-cutting themes emerged from the workshop discussions that informed the project's approach to data product development.

Standardisation and interoperability

Participants consistently emphasised the need for standardised data collection methods and the adoption of international standards, particularly Darwin Core for biodiversity observations, to enable interoperability with national (AODN, OBIS Australia) and global (OBIS, GBIF) data systems. This requirement directly shaped the project's decision to adopt Darwin Core, WoRMS and CAAB controlled vocabularies in its ETL pipelines.

Integration into National Frameworks

A recurring theme was the importance of integrating datasets into existing national infrastructure rather than creating stand-alone products. This reinforced the project's design principle of building data pipelines within the AODN/IMOS infrastructure for long-term maintenance as business-as-usual operations.

Collaboration and partnerships

The workshop highlighted the need to strengthen partnerships between Government agencies, research institutions and industry to improve data sharing agreements and enhance accessibility. The breadth of organisations represented at the workshop itself demonstrated the collaborative foundation on which the project would build.

2.4. Recommended datasets for development

Based on the workshop deliberations and subsequent consultation with the project steering group, six datasets were recommended for development as cloud-optimised, assessment-ready data products within the Project 5.9 scope (Table 2). Selection was guided by the prioritisation framework applied during the workshop, balancing policy demand, data readiness and technical feasibility within the project's twelve-month timeframe and available resources.

The recommended datasets span three of the workshop's thematic areas: habitats and communities (seagrass, kelp, reef benthos), species (seabirds, dugongs) and pressures (vessel traffic). Together, they provide assessment-ready information products that directly serve the SOE 2026 reporting requirements, Environmental Economic Accounts, environmental approvals for offshore renewable energy, Parks Australia management effectiveness evaluation and threatened and migratory species assessments.

Datasets not selected for immediate development – including mangroves and salt marshes, water column communities, sharks and rays, marine fish, and pollution pressure layers - were documented as candidates for future iterations of the assessment-ready data pipeline, should additional resources become available or should the datasets reach a higher level of readiness.

The workshop recommended that annual follow-up workshops be facilitated to reassess progress on identified priorities and discuss technology advancements, including the potential role of hyperspectral satellite data and expanded eDNA sampling.

The following chapters describe each of the recommended datasets in detail (Chapter 4), present the ETL processes used to transform them into cloud-optimised products (Chapter 5), and demonstrate examples of the aggregated products and their analytical applications (Chapter 6).

Table 2: Datasets recommended for development as assessment-ready data products

Theme	Dataset	Scope	Readiness	Priority	Rationale
Habitats and communities	Seagrass aggregated product	Y	H	High	National integration of 50+ datasets from Seamap Australia, JCU, state agencies; harmonised to CAAB/WoRMS taxonomy
Habitats and communities	Kelp annotations (Squidle+)	Y	H	High	Public kelp image annotations from Squidle+ platform; Darwin Core translation with WoRMS alignment
Habitats and communities	NRMN photo quadrat (reef benthos)	Y	H	High	Reef Life Survey Method annotations via Squidle+ and legacy database; coral and benthic cover
Species	Seabird aggregated product	Y	H	High	40+ Darwin Core resources from OBIS Australia Node; tracking and at-sea observations 1939–present
Species	Dugong aerial surveys	Y	M	Medium–High	JCU aerial survey database covering northern Australian waters; requires coordination with data custodians
Pressures	AMSA vessel tracking	Y	H	High	Monthly AIS/CTS Shapefile deliveries 2012–present; harmonised to a Parquet dataset with H3 indexing

3. Selected datasets

This chapter describes each of the six datasets selected for development as cloud-optimised, assessment-ready data products under Project 5.9. For each dataset, a summary of the metadata record is presented, followed by a description of the data content, spatial and temporal coverage, contributing sources, observational methods, taxonomic framework and licensing. The technical details of how each dataset was transformed through the ETL process are presented separately in section 4.

3.1. National Seagrass Data Aggregation

Table 3: Metadata summary – National Seagrass Data Aggregation

Title	National Seagrass Data Aggregation
Temporal coverage	1967 – 2024
Geographic coverage	Australian coastal waters (national)
Contributing datasets	83 source datasets
Output format	Parquet (long format)
Licence	CC-BY 4.0 (some contributing datasets carry NC or SA conditions)
Update frequency	As needed
Status	Complete
Access point	s3://data-uplift-public/stored/datauplift/seagrass/seagrass.parquet

The National Seagrass Data Aggregation is the most complex product developed under Project 5.9. It harmonises 83 individual datasets from across Australia into a single, standardised resource suitable for ecological assessment, long-term monitoring and habitat modelling. Source data were contributed by institutions including the Institute for Marine and Antarctic Studies (IMAS), James Cook University's Centre for Tropical Water and Aquatic Research (TropWATER), state Government agencies in New South Wales, Queensland, Western Australia, South Australia, Victoria, Northern Territory and Tasmania, and research organisations including CSIRO, AIMS, UWA and UQ. Data was accessed primarily through Seamap Australia's GeoServer infrastructure and institutional repositories.

The aggregated product encompasses observations from a wide range of survey methodologies, reflecting the diversity of approaches used across contributing programmes. These include direct field surveys using on-foot GPS delineation of intertidal meadows at low tide; helicopter overflights

for rapid visual assessment of meadow extent and condition; boat-based transects with snorkel and photo-quadrat surveys for subtidal habitats; underwater video, bathyscope and side-scan sonar deployments used in state monitoring programmes; UAV (drone) orthomosaics providing centimetre-scale mapping of shallow reef-flat meadows; and high-resolution multispectral satellite imagery (WorldView-2, IKONOS, QuickBird-2) and medium-resolution PlanetScope imagery for landscape-scale mapping of seagrass cover, species composition and biomass.

The geographic coverage spans the full extent of Australian coastal waters, with particularly dense data coverage in the Great Barrier Reef, Torres Strait, Gulf of Carpentaria, Hervey Bay, Moreton Bay, Cockburn Sound, and temperate southern coastlines including Tasmania and Victoria. The temporal record extends from 1967 (the earliest Cockburn Sound historical mapping) through to 2024.

The product records presence and absence of 13 seagrass species, including *Cymodocea rotundata*, *C. serrulata*, *Enhalus acoroides*, *Halophila capricornia*, *H. decipiens*, *H. ovalis*, *H. spinulosa*, *H. tricostata*, *Halodule uninervis*, *Syringodium isoetifolium*, *Thalassodendron ciliatum*, *Thalassia hemprichii* and *Zostera muelleri* subsp. *capricorni*. All taxa have been mapped to the Codes for Australian Aquatic Biota (CAAB, 2024) controlled vocabulary and aligned with the World Register of Marine Species (WoRMS) taxonomy, including AphiaID, rank and taxonomic status. Where source datasets recorded species as “sp.” or “spp.”, the ETL pipeline rolls these up to the nearest resolved parent taxon. Optional biomass observations are retained where available from the source data.

The final product is delivered as a long-format Parquet file (one record per species observation per site visit), with H3 spatial indices at resolution 15 (~1 m²) and Australian Marine Region tags assigned through spatial overlay. The Parquet format reduces the combined source data volume from over 2 GB of GeoServer exports to approximately 230 MB, a compression ratio exceeding 88%.

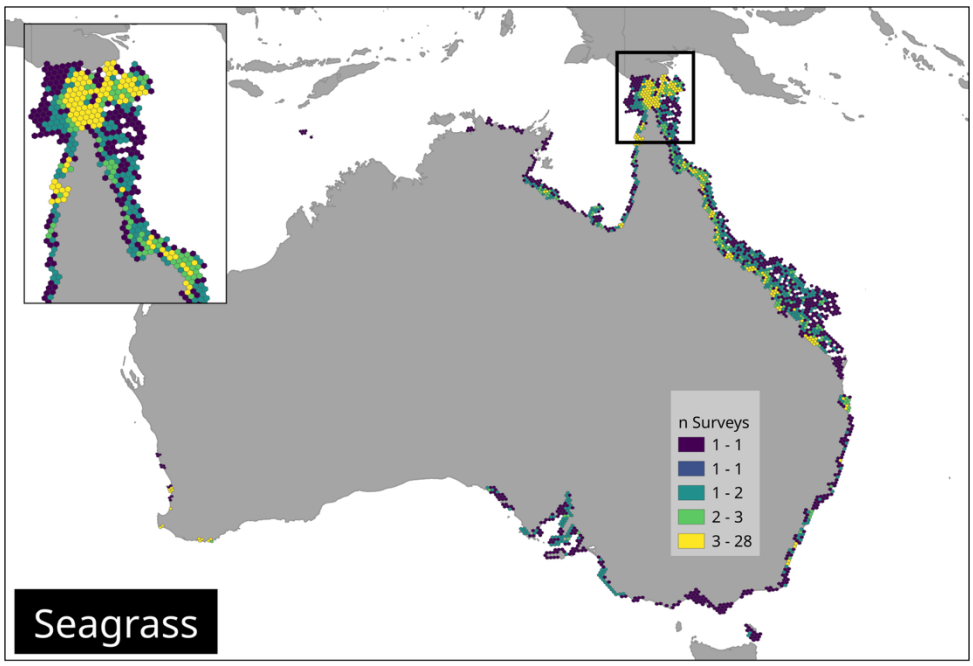


Figure 2: Spatial distribution of the number of seagrass surveys aggregated into h3 hexagons of ~250Km² (resolution 4).

3.2. Seabird Aggregated Data Product

Table 4: Metadata summary – Seabird Aggregated Data Product

Title	Seabird Aggregated Data Product
Temporal coverage	1939 – 2025
Geographic coverage	Global (bounding box: 180°W–80°E, 71°S–63°N); primarily Australian and Southern Ocean waters
Contributing datasets	41 Darwin Core archives from OBIS Australia Node
Output format	Parquet (~5 MB)
Licence	CC-BY 4.0 (some contributing datasets carry NC conditions)
Update frequency	Daily (automated refresh)
Status	Complete
Access point	s3://data-uplift-public/stored/datauplift/seabird/seabird.parquet

The Seabird Aggregated Data Product unifies 41 heterogeneous Darwin Core archives published through the OBIS Australia Node at CSIRO into a single harmonised dataset. The contributing archives span a long temporal range, from historic South Pacific sea-bird logs dating from1939 through to RV Investigator wildlife observations from 2025. The datasets encompass at-sea vessel-based observation programmes, satellite and geolocation tracking studies, colony monitoring surveys and citizen-science sighting records.

Contributing organisations include the Australasian Seabird Group, CSIRO Marine National Facility, Institute for Marine and Antarctic Studies (IMAS/UTAS), Deakin University, Macquarie University, Monash University, Lund University, the New Zealand Department of Conservation, Birds Australia, Taronga Zoo, BMT and the Centre for Whale Research. Principal investigators and observers span multiple generations of seabird research, from historic voyage observers (C. Fleming, J.D. Gibson, Neil Cheshire) to contemporary tracking researchers (Aymeric Fromant, Luke Halpin, Graeme Taylor).

The product targets the class Aves with a focus on orders Procellariiformes (albatrosses, petrels, shearwaters) and Charadriiformes (shorebirds, gulls, terns), though records span all marine-associated avian taxa observed during at-sea surveys. Taxonomic validation is performed against the World Register of Marine Species (WoRMS) to ensure that each observation is linked to an accepted taxon with correct hierarchy and status. Life-stage terms are validated against the GBIF Life Stage Vocabulary.

The geographic coverage is primarily concentrated in Australian marine waters and the Southern Ocean, with additional records from the Australasian, Indo-Pacific and sub-Antarctic regions, reflecting the wide-ranging movements of the tracked and observed species. Environmental and

contextual measurements from the Darwin Core Extended Measurement Or Fact extension – including sea state, sea-surface temperature, wind speed, wind direction and depth – are integrated where available, providing valuable co-located environmental context for each observation.

The final product is a compact Parquet file of approximately 5 MB, representing a substantial compression from the multi-file Darwin Core archive sources. Each record is enriched with WoRMS taxonomic hierarchy, H3 index at resolution 4 (~1,700 km², as it has a global geographical bounding box), survey type, dataset-level metadata and an ETL processing timestamp for provenance tracking.

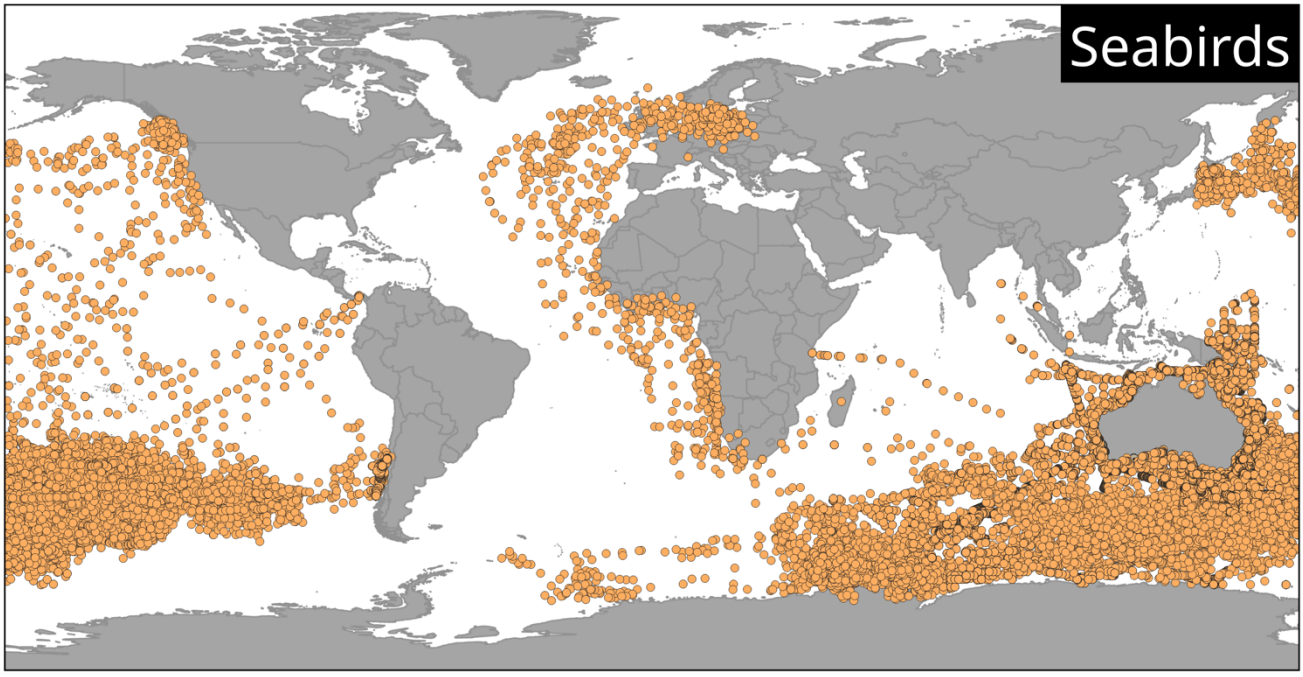


Figure 3: Spatial distribution of seabird occurrences aggregated in the seabird's product.

3.3. National Kelp Dataset

Table 5: Metadata Summary – National Kelp Dataset

Title	National Kelp Dataset
Temporal coverage	2008 – 2025
Geographic coverage	Southern and eastern Australian coastal waters (113.4°E–153.5°E, 43.7°S–27.1°S)
Contributing datasets	Squidle+ public annotation sets
Output format	Parquet (~2 MB)

Licence	CC-BY 4.0
Update frequency	Ongoing (scheduled refresh)
Status	Complete
Access point	s3://data-uplift-public/stored/datauplift/kelp/kelp.parquet

The National Kelp Dataset aggregates kelp-related annotations from public marine image surveys hosted on the Squidle+ platform and translates them into the Darwin Core standard for interoperability with national and global biodiversity information systems. Squidle+ is a web-based platform for the annotation and analysis of georeferenced marine imagery, widely used across Australian marine research programmes including the IMOS Autonomous Underwater Vehicle (AUV) facility, the National Reef Monitoring Network and numerous university and Government survey campaigns (Friedman et al., 2026).

Because Squidle+ supports user-defined controlled vocabularies, kelp labels are not always consistent across annotation sets and campaigns. To ensure specificity, the ETL pipeline applies a curated Kelp Label Allow List comprising over 50 label identifiers drawn from multiple catalogues. This Allow List targets canopy-forming and habitat-forming macroalgae from the phylum Ochrophyta, class Phaeophyceae, with particular emphasis on the orders Laminariales and Fucales. Key taxa represented include *Ecklonia radiata, Phyllospora comosa, Macrocystis pyrifera, Durvillaea* spp., *Cystophora* spp., *Scytothalia dorycarpa, Lessonia corrugata* and *Perithalia caudata*, among others. All labels are taxonomically aligned with WoRMS, providing standardised scientific names, Life Science Identifiers (LSIDs) and full taxonomic hierarchy from Kingdom to Genus.

Each observation is classified with a basis of record of Machine Observation (photograph-based detection) and an occurrence status of “present”, reflecting the annotation-based detection methodology. Media linkage is maintained for each record through resolvable Squidle+ API URLs, enabling visual verification of individual annotations. Records prior to 28 September 2007 are excluded as a quality threshold, and observations are constrained to the Australian marine region.

The geographic coverage spans the temperate and subtropical coastal waters of southern and eastern Australia, from approximately 113°E to 153°E and 44°S to 27°S, encompassing the major kelp forest ecosystems of the Great Southern Reef, the south-east Australian coast and the eastern seaboard. The final product is delivered as a Parquet file of approximately 2 MB, supporting automated scheduled updates as new public annotations are added to the Squidle+ platform.

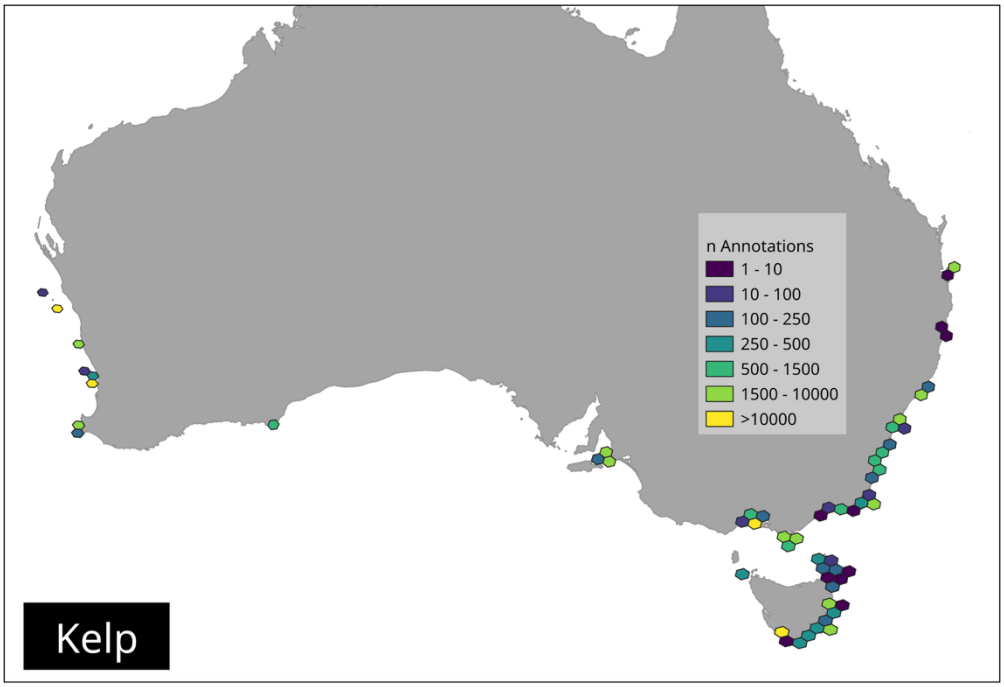


Figure 4: Spatial distribution of the number of Kelp annotations aggregated into h3 hexagons of ~250 Km² (resolution 4).

3.4. NRMN Photo Quadrat Aggregated Data Product

Table 6: Metadata Summary – NRMN Photo Quadrat Aggregated Data Product

Title	NRMN Photo Quadrat Aggregated Data Product
Temporal coverage	Aligned with IMOS NRMN monitoring programme
Geographic coverage	Australian temperate and tropical reef sites
Contributing datasets	IMOS National Reef Monitoring Network legacy database; Squidle+ private NRMN annotations
Output format	Parquet (~1 MB, public and private variants)
Licence	Public component: open access; Private component: restricted
Update frequency	Ongoing (scheduled refresh)
Status	Complete
Access point	s3://data-uplift-public/stored/datauplift/nrnm/nrmn.parquet

The NRMN Photo Quadrat Aggregated Data Product provides standardised benthic community composition data derived from the IMOS National Reef Monitoring Network's Reef Life Survey Method 13 (photo quadrat surveys). This product is distinctive in drawing from two complementary sources: a static export from the legacy NRMN database and a dynamic extraction from the Squidle+ annotation platform via its REST API. This dual-source approach ensures that both historical survey data and contemporary annotations are captured in the harmonised product.

The Squidle+ component accesses private NRMN data shared through a dedicated Squidle+ account and involves navigating the platform's complex entity-relationship model. This model encompasses annotations, annotation sets, label schemes, media objects, deployments and campaigns. Annotations are produced using two methods: full biodiversity surveys, which classify every point on the photo quadrat; and targeted coral surveys, which focus specifically on coral and bleaching indicators. Each method requires a distinct aggregation process to calculate survey-level benthic cover estimates.

Label scheme translation is a critical step in this product. The NRMN programme has used multiple classification schemes over time, principally the CATAMI 1.4 scheme and the RLS Catalogue. The ETL pipeline normalises annotations across these different labelling schemes to produce consistent benthic cover categories. For the legacy static export, additional transformations include the construction of dataset identifiers, updating of RLS lineage records and the translation of bleached and dead coral columns into standardised tag fields.

The final product is delivered in both public and private variants. The public version (Parquet and CSV formats) contains survey-aggregated benthic cover data suitable for open distribution. The private version (Parquet only) retains finer-grained annotation data subject to data sharing agreements. Both versions benefit from modern compression, reducing the output to approximately 1 MB.

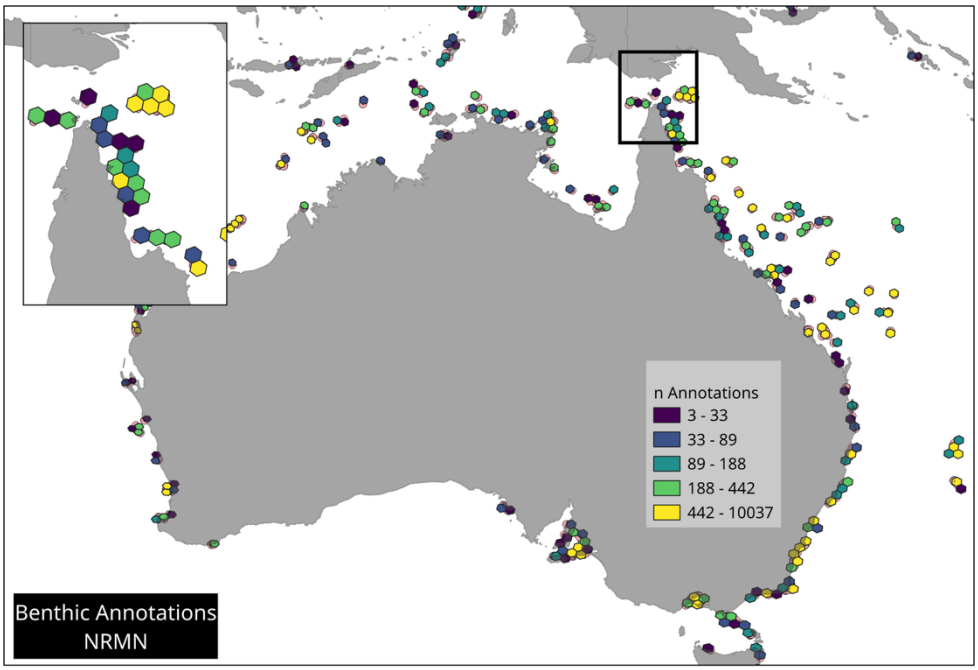


Figure 5: Spatial distribution of the number of benthic annotations aggregated into h3 hexagons of ~250Km² (resolution 4).

3.5. Dugong Aerial Survey Database

Table 7: Metadata Summary – Dugong Aerial Survey Database

Title	Dugong Aerial Survey Database
Temporal coverage	1984 – 2023
Geographic coverage	Northern Australian coastal waters (nine broad survey regions)
Contributing datasets	James Cook University (JCU)
Output format	Parquet
Licence	CC-BY 4.0
Update frequency	As surveys are conducted
Status	Completed
Access point	s3://data-uplift-public/stored/datauplift/seabird/dugongs.parquet

The Dugong Aerial Survey Database contains data from systematic aerial surveys for dugongs (*Dugong dugon*) undertaken by James Cook University across nine broad regions of the Australian coast since 1984 (Cleguer and Marsh, 2023). The survey regions are: Shark Bay (WA), Exmouth/Ningaloo (WA), Pilbara (WA), Gulf of Carpentaria (NT and QLD), Torres Strait (QLD), northern Great Barrier Reef (QLD), southern Great Barrier Reef (QLD), Hervey Bay (QLD) and Moreton Bay (QLD). Together these regions encompass the core habitat range of the Australian dugong population, from the tropical waters of the Torres Strait south to the subtropical waters of south-east Queensland and the warm temperate waters of Western Australia's Shark Bay.

The database represents one of the most comprehensive long-term aerial survey programmes for a marine mammal species globally, with surveys spanning four decades. Data from these surveys have been fundamental to population assessments and conservation management of dugongs in Australian waters and have contributed directly to national threatened species recovery plans and international conservation reporting.

At the time of writing, the development of this dataset into a cloud-optimised assessment-ready product is ongoing. Coordination with the JCU data custodians is required to finalise data sharing arrangements, validate the survey metadata and complete the ETL transformation into the standardised Parquet format used by the other Project 5.9 products. The dataset was assessed as medium–high priority at the prioritisation workshop due to the long-term, systematic nature of the survey programme and its direct relevance to threatened species assessments under the EPBC Act.

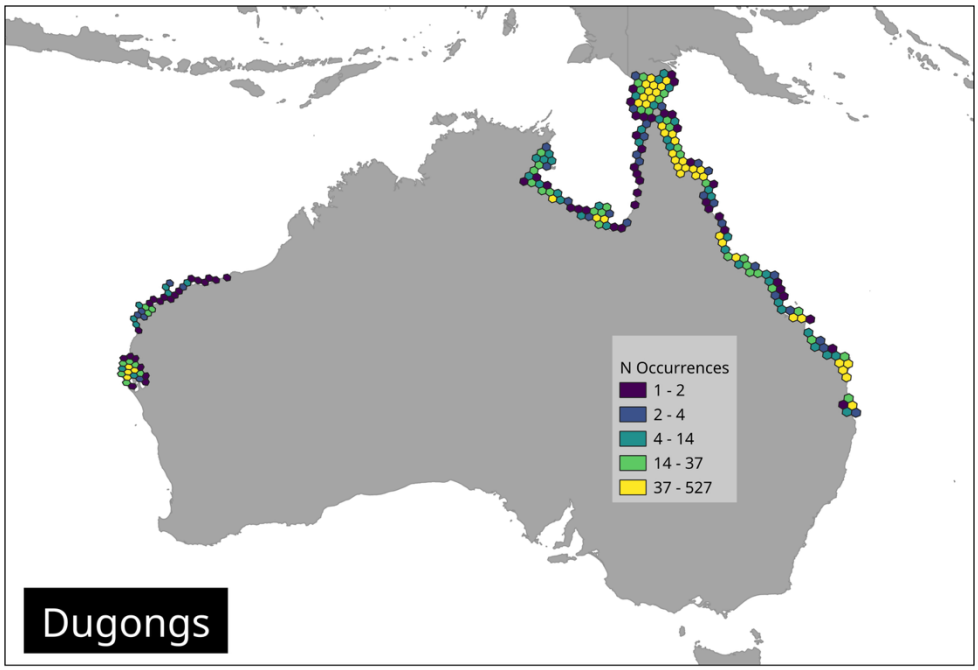


Figure 6: Spatial aggregation of the number of dugong occurrences in h3 hexagons of ~250Km² (resolution 4).

3.6. Aggregated AMSA Vessel Tracking Data Product

Table 8: Metadata Summary – Aggregated AMSA Vessel Tracking Data Product

Title	Aggregated AMSA Vessel Tracking Data 2012–2025
Temporal coverage	September 2012 – 2025
Geographic coverage	Australian Search and Rescue Region (75°E–163°E, 86.6°S–2°S)
Contributing datasets	Australian Maritime Safety Authority (AMSA)
Output format	Parquet (yearly master files)
Licence	CC-BY-NC 4.0
Update frequency	Ongoing (monthly ingestion)
Status	Complete
Access point	s3://data-uplift-public/stored/datauplift/amsa/

The Aggregated AMSA Vessel Tracking Data Product is the only pressure-layer dataset developed under Project 5.9, and provides a harmonised, analysis-ready record of vessel movements throughout the Australian Search and Rescue Region over more than a decade. The source data originates from AMSA's Craft Tracking System (CTS) and Mariweb databases, which collect Automatic Identification System (AIS) signals from ships using both terrestrial and satellite receivers (AMSA, 2024). AIS is an internationally mandated tracking system for commercial vessels, and the combined CTS and AIS data provide comprehensive coverage of vessel traffic in Australian waters.

AMSA publishes vessel tracking data as monthly spatial datasets in nested ZIP archives containing Shapefiles, distributed through its operational data service. Each record represents a single vessel position report defined by geographic coordinates and a UTC transmission timestamp, along with vessel attributes including vessel type, name and identification codes. The monthly deliveries of vessel traffic data have been available since September 2012, providing over 13 years of continuous recording at the time of product completion.

The geographic coverage encompasses the full Australian Search and Rescue Region, extending from 75°E to 163°E and from 86.6°S to 2°S, covering the Indian Ocean, Pacific Ocean and Southern Ocean near to Australia. This extensive spatial footprint captures vessel traffic across major international shipping lanes, coastal navigation corridors, fishing grounds and offshore resource extraction areas.

The product has direct relevance to multiple assessment processes. Previous SOE reports (2016, 2021) drew on shipping density products to characterise marine pressures, but these were produced as bespoke, one-off analyses. The Project 5.9 product provides a routinely updated, standardised alternative that can support recurring SOE reporting cycles, Environmental Economic Accounts, environmental approval processes for offshore renewable energy, the Sustainable Ocean Plan and vessel strike risk assessments for marine megafauna. The metadata record documents several related historical products, including the Australian Ship Reporting System summaries (1999–2015), Queensland AIS data (2013), vessel traffic density and vessel strike risk maps (2013–2015) and the CSIRO Futures of Seafood commercial vessel transiting dataset; establishing the AMSA product within a broader lineage of vessel traffic analysis in Australian waters.

The processed product is delivered as yearly master Parquet files with H3 spatial indexing at resolution 8 (~0,7 Km²), enabling efficient spatial aggregation and temporal analysis. Each record is enriched with provenance fields (source file name and processing date) and validated through a quality control process that rejects duplicate, spatially invalid or out-of-bounds records. The licence is CC-BY-NC 4.0, reflecting the source data copyright held by the Commonwealth of Australia through AMSA.

The six datasets described in this chapter represent a cross-section of the marine environmental data required for national assessments. Together, they demonstrate the breadth of data types, source formats, institutional origins and observational methodologies that the Project 5.9 ETL infrastructure was designed to accommodate. The technical processes used to transform each of these datasets into cloud-optimised, assessment-ready products are described in the following chapter.

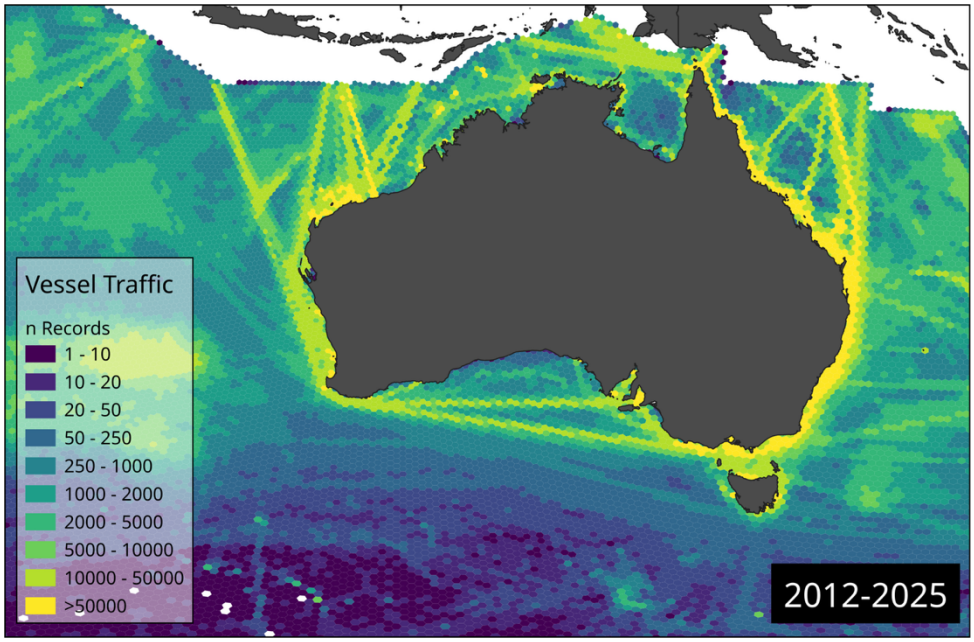


Figure 7: Spatial aggregation of the number of vessel AIS pings in h3 hexagons of ~250Km² (resolution 4).

4. Extract–Transform–Load process

This chapter describes the Extract–Transform–Load (ETL) processes used to convert the six selected datasets from their heterogeneous source formats into cloud-optimised, assessment-ready data products. For each dataset the narrative covers three phases: extraction of source data; transformation and validation; and loading of the final product to cloud storage. Each section concludes with a table describing the output file structure, including field names, data types and descriptions.

4.1. Common architectural patterns

Although the source data for each product differ substantially in format, schema and complexity, all five ETL pipelines share a common technical architecture that ensures consistency, reproducibility and long-term maintainability across the Project 5.9 portfolio.

All pipelines are orchestrated using Prefect (Prefect Technologies, Inc.), a modern workflow management framework that provides scheduling, retry logic, observability and dependency management. Each pipeline is structured as a Prefect flow comprising discrete extract, transform and load tasks, enabling independent monitoring and selective re-execution of individual stages without the need to reprocess the entire pipeline.

Schema validation is enforced at multiple stages using Pydantic models (Colvin et al., 2026) that wrap PyArrow schemas (Apache Arrow Developers, 2026). Source data are validated against per-dataset input schemas at the extraction stage, and the final output is validated against a strict output schema before loading. This approach catches type mismatches, missing fields and constraint violations early in the pipeline, ensuring that only conforming data reaches the output products. Where columns are missing from source data, the schema enforcement layer adds them as null-valued fields and reorders all columns to a consistent sequence.

Data processing is performed using a combination of PyArrow for type-safe columnar operations, Polars for efficient in-memory transformations and, in the case of the AMSA product, pyogrio for geospatial file reading. All pipelines produce output in Apache Parquet format with Snappy compression, providing substantial reductions in file size compared with the original formats, while supporting efficient columnar access, predicate pushdown and compatibility with cloud-native analytical tools. Across the five completed products, compression ratios range from approximately 88% (seagrass) to over 95% (seabirds), reducing the combined source data volume from several gigabytes to tens of megabytes.

Spatial enrichment is applied where appropriate. The seagrass product assigns H3 hexagonal spatial indices at resolution 15 and overlays Australian Marine Region tags. The AMSA vessel tracking product uses H3 indexing at resolution 8, reflecting the coarser spatial precision of vessel position reports. Both resolutions are configurable through the pipeline's JSON configuration files. Where taxonomic data are present, all products align species identifiers to the World Register of Marine Species (WoRMS) controlled vocabulary, providing standardised scientific names, AphiaID identifiers, taxonomic rank and acceptance status.

All final products are loaded to Amazon Web Services S3 cloud storage, whereby they are served through the AODN data delivery infrastructure. Each pipeline supports idempotent re-execution: reprocessing a source file produces the same output without creating duplicate records, enabling safe recovery from failures and routine scheduled updates as new source data become available.

4.2. Seagrass Aggregated Data Product

4.2.1. Extract

The seagrass product aggregates 83 source datasets from institutions including IMAS, JCU TropWATER, Seamap, Australia GeoServer exports and various state Government agencies. Source data arrives in CSV format, having been previously exported from institutional GeoServer instances and research repositories. Because the contributing datasets differ widely in column naming, geometry encoding, temporal representation and species nomenclature, each dataset is validated at extraction time against its own custom Pydantic-wrapped PyArrow schema. The minimum required fields across all datasets are a geometry column (typically in Well-Known Text format), a date field and one or more species indicators, though the specific column names and encodings vary by dataset.

Extraction is managed through JSON configuration files (one per source dataset) that specify the S3 storage location, file format and expected schema. These configuration files serve as a declarative record of each dataset's structure, ensuring that the extraction process is transparent, version-controlled and reproducible. There are plans to replace the current static exports with dynamic GeoServer extractions in future iterations of the pipeline.

4.2.2. Transform

Transformation of the seagrass data proceeds through a two-stage process. In the first stage, each source dataset undergoes a bespoke transformation to convert it into a harmonised intermediary format. In the second stage, the intermediary datasets are combined and post-processed into the final product.

Geometry extraction

Most source datasets encode spatial information as WKT geometry strings from GeoServer export queries. The pipeline extracts latitude and longitude from the centroid of each geometry. Where geometry columns are unavailable, raw latitude and longitude fields are used directly.

Date normalisation

Date handling is the most labour-intensive component of the per-dataset transformations, owing to the extreme inconsistency of temporal encodings across contributing programmes. Source dates include ISO 8601 strings, abbreviated month-year formats ("Dec-19"), date ranges ("2015–2016"), multi-date strings ("10,17,19,27/5/22") and compound expressions ("December-2010, November-2011"). Each non-standard encoding is manually audited and mapped to separate start_year, start_month and start_day integer fields. Where only partial temporal information is available (for example, month and year without day), the unresolved components are recorded as zero.

Species harmonisation

Species presence and absence are encoded in source datasets using two principal methods: delimited strings (for example, “HO, HA, AO” representing abbreviated species codes) and one-hot encoded Boolean columns (one column per species). In both cases, species identifiers are mapped to the Codes for Australian Aquatic Biota (CAAB) controlled vocabulary through a curated lookup table maintained in the pipeline’s enumeration module. Ambiguous records using “sp.” or “spp.” designations are rolled up to the nearest resolved parent taxon. All CAAB codes are further aligned with WoRMS to provide internationally recognised AphiaID identifiers, taxonomic rank and acceptance status.

Post-processing

Once all datasets have been transformed to the intermediary wide format, they are vertically stacked and pivoted from wide to long format, producing one record per species observation per site visit. H3 spatial indices at resolution 15 and Australian Marine Region tags are assigned through spatial overlay. The final long-format table is validated against the output schema before loading.

4.2.3. Load

The validated output is written as a single Parquet file with Snappy compression to AWS S3, reducing the combined source volume from over 2 GB of GeoServer exports to approximately 230 MB.

Table 9: Output Schema – Seagrass Aggregated Data Product

Field	Type	Description
source	string	Name of the contributing source dataset
source_id	string	Row identifier within the source dataset
metadata_link	string	URL to the source dataset metadata record
start_year	int16	Start year of the observation
start_month	int8	Start month of the observation (0 if unknown)
start_day	int8	Start day of the observation (0 if unknown)
end_year	int16	End year of the observation, if applicable
end_month	int8	End month of the observation, if applicable
end_day	int8	End day of the observation, if applicable
geometry	string	WKT-encoded geometry of the observation (EPSG:4326)
lat	float32	Latitude of the geometry centroid (decimal degrees)
lon	float32	Longitude of the geometry centroid (decimal degrees)

h3_cell_15	string	H3 hexagonal spatial index at resolution 15
australian_marine_regions_tags	string	Australian Marine Region identifiers applicable to the observation
present	boolean	Whether the species was recorded as present
biomass	float32	Biomass of the species, where available (g dry weight m^{-2})
scientific_name	string	Scientific name per WoRMS catalogue
aphia_id	int64	WoRMS AphiaID identifier
rank	string	Taxonomic rank per WoRMS (e.g. Species, Genus)
status	string	Taxonomic acceptance status per WoRMS

4.3. Seabird Aggregated Data Product

4.3.1. Extract

The seabird product draws on 41 Darwin Core archives published through the OBIS Australia Node at CSIRO. Each archive is a structured ZIP file containing an EML metadata document (eml.xml), a schema description file (meta.xml), one or more core tables (in this case, Occurrence tables) and optional extension tables, most importantly the Extended Measurement Or Fact (eMoF) extension. The extraction process reads each archive, parses the EML metadata to obtain dataset-level information (name, temporal extent, geographic bounds, contacts, abstract and publication version), and translates the core Occurrence and eMoF tables into intermediary Parquet files validated against strict Darwin Core schemas.

Being sourced from an OBIS Node, the data are generally of high quality and conforms to the Darwin Core standard. Nevertheless, two validation checks are applied during extraction. First, a timestamp check confirms that eventDate values conform to ISO 8601 patterns. Second, a taxonomic consistency check ensures that the scientificName and scientificNameID fields match the current accepted taxonomy in WoRMS, flagging any deprecated or ambiguous identifiers for correction.

4.3.2. Transform

The individual Darwin Core Occurrence Parquet files are aggregated vertically into a single table. Pre-filtering steps remove rows with missing latitude or longitude, rows with missing organism quantity, rows in the eMoF extension that lack both eventID and occurrenceID references, and columns that are entirely null across all contributing datasets.

ExtendedMeasurementOrFact pivot

The eMoF extension presents contextual measurements (sea state, sea-surface temperature, wind speed, depth, salinity and others) in a long-format table with a many-to-one relationship to the

Occurrence table. To produce a flat analytical product, the eMoF table is pivoted to wide format, with each retained measurementType becoming a separate column. The retention threshold is set at 10,000 non-null values, corresponding to approximately 10% presence across the combined dataset. This threshold retains eleven measurement types: sea state, sea-surface temperature, depth, salinity, wind direction, air pressure, wind speed, wind force (Beaufort scale), cloud cover, atmospheric temperature and cloud cover in oktas.

Measurement merging

Several measurement types are recorded under variant names across different contributing datasets. Where careful analysis of the underlying values confirms compatibility, semantically equivalent measurement types are merged: wind speed and wind force (Beaufort scale) are combined into a single wind speed field; variant wind direction labels are unified; air temperature and atmospheric temperature are merged; and duplicate depth columns are consolidated. Any mutations to the underlying measurement data are reported to AODN for review.

Taxonomic enrichment

The WoRMS taxonomic hierarchy is mapped to each record via the scientificNameID field, providing Kingdom, Phylum, Class, Family and Genus assignments. Life-stage values are validated against the GBIF Life Stage Vocabulary. The output table is enriched with dataset-level metadata (dataset name, source URL, publication version) and an ETL processing timestamp for provenance tracking.

4.3.3. Load

The final product is written as a single Parquet file of approximately 5 MB. The compact size reflects both the efficiency of columnar compression and the removal of the highly redundant long-format eMoF structure.

Table 10: Output Schema – Seabird Aggregated Data Product (principal fields shown)

Field	Type	Description
occurrenceID	string	Unique identifier for the observation record
eventDate	date	Date of the observation (ISO 8601)
decimalLatitude	float64	Latitude of the observation (decimal degrees)
decimalLongitude	float64	Longitude of the observation (decimal degrees)
scientificName	string	Scientific name of the observed taxon, validated against WoRMS
scientificNameID	string	WoRMS Life Science Identifier (LSID) for the taxon
kingdom	string	Taxonomic kingdom (WoRMS hierarchy)
phylum	string	Taxonomic phylum (WoRMS hierarchy)

class	string	Taxonomic class (WoRMS hierarchy)
family	string	Taxonomic family (WoRMS hierarchy)
genus	string	Taxonomic genus (WoRMS hierarchy)
vernacularName	string	Common name of the observed taxon
basisOfRecord	string	Nature of the record (e.g. HumanObservation)
occurrenceStatus	string	Presence or absence of the organism
individualCount	int64	Number of individuals observed
organismQuantity	string	Quantity of organism recorded
organismQuantityType	string	Unit or method of quantity measurement
lifeStage	string	Life stage, validated against GBIF vocabulary
sex	string	Sex of the observed individual(s)
recordedBy	string	Name(s) of the observer(s)
samplingProtocol	string	Survey method used
minimumDepthInMeters	int64	Minimum depth of the observation (m)
maximumDepthInMeters	int64	Maximum depth of the observation (m)
sea_state	string	Sea state at time of observation (from eMoF pivot)
sea_surface_temperature	string	Sea-surface temperature at time of observation (°C, from eMoF pivot)
wind_speed	string	Wind speed at time of observation (kn or Beaufort, from eMoF pivot)
wind_direction	string	Wind direction at time of observation (degrees, from eMoF pivot)
air_pressure	string	Atmospheric pressure at time of observation (hPa, from eMoF pivot)
atmospheric_temperature	string	Air temperature at time of observation (°C, from eMoF pivot)
depth	string	Water depth at observation location (m, from eMoF pivot)
salinity	string	Sea-surface salinity (psu, from eMoF pivot)
cloud_cover	string	Cloud cover at time of observation (oktas, from eMoF pivot)
dataset_name	string	Name of the contributing Darwin Core archive
source_url	string	Download URL for the source dataset and version
etl_timestamp	timestamp	Date and time of ETL processing (UTC)

4.4. National Kelp Dataset

4.4.1. Extract

The kelp product draws its source data from public annotations hosted on the Squidle+ marine imagery platform via its REST API. Because Squidle+ supports user-defined controlled vocabularies, kelp annotations are not uniformly labelled across annotation sets and campaigns. To address this, the pipeline applies a curated Kelp Label Allow List comprising over 50 label identifiers drawn from six catalogues: Australian Morphospecies Catalogue; Benthic Habitat and Relief Schema for BRUV/BOSS/DOV Imagery; CATAMI 1.4, Hunter Extended RLS Scheme; RLS Australian Coral Species List; RLS Catalogue and SQUIDLE 1.0.

Extraction is a three-step API process. First, the pipeline queries Squidle+ for all public annotation sets containing at least one label matching the Kelp Label Allow List. Second, the full annotations for those sets are exported. Third, all vocabulary elements associated with the kelp labels are exported to support downstream taxonomic translation. The raw annotations, WoRMS vocabulary mappings and the label-to-WoRMS translation table are passed to the transform stage.

4.4.2. Transform

Cleaning

The dataset is first pruned to retain only records classified as kelp, removing all unrelated biological observations. A media conflict resolution step then identifies images with multiple overlapping annotation layers and rejects any media where kelp labels are contradictory across annotation sets, ensuring high data integrity.

Darwin Core translation

Raw annotations are mapped to the Darwin Core standard through several operations. Taxonomic alignment joins the local Squidle+ label identifiers with the WoRMS vocabulary, enriching each record with the full taxonomic hierarchy from Kingdom to Genus and a globally unique scientificNameID (Life Science Identifier). A unique occurrenceID is constructed by concatenating the internal annotation identifier with the deployment identifier, ensuring traceability to the original survey. Timestamps are parsed from diverse source formats into a unified eventDate in GMT. Raw pose coordinates (latitude and longitude from the survey platform) are mapped to Darwin Core decimalLatitude and decimalLongitude fields. The basisOfRecord is set to MachineObservation (photograph-based detection) and occurrenceStatus is set to “present”. Media linkage is maintained by constructing resolvable Squidle+ API URLs for each record, enabling visual verification of individual annotations.

Temporal and spatial filtering

A quality threshold excludes all records prior to 28 September 2007, and a geofence constrains observations to the Australian marine region, removing any erroneous or out-of-scope records.

Validation

The final Polars DataFrame is validated against the expected output schema, confirming correct data types and null constraints before conversion to a PyArrow Table for serialisation.

4.4.3. Load

The validated product is written as a Parquet file of approximately 2 MB to AWS S3. The pipeline supports automated scheduled updates, allowing new public annotations added to the Squidle+ platform to be incorporated without manual intervention.

Table 11: Output Schema – National Kelp Dataset

Field	Type	Description
occurrenceID	string	Unique identifier (annotation ID + deployment ID)
eventDate	timestamp	Date and time of the observation (GMT)
decimalLatitude	float64	Latitude of the observation (decimal degrees)
decimalLongitude	float64	Longitude of the observation (decimal degrees)
scientificName	string	Scientific name per WoRMS catalogue
scientificNameID	string	WoRMS Life Science Identifier (LSID)
kingdom	string	Taxonomic kingdom (WoRMS hierarchy)
phylum	string	Taxonomic phylum (WoRMS hierarchy)
class	string	Taxonomic class (WoRMS hierarchy)
family	string	Taxonomic family (WoRMS hierarchy)
genus	string	Taxonomic genus (WoRMS hierarchy)
basisOfRecord	string	Nature of the record (MachineObservation)
occurrenceStatus	string	Detection status (present)
associatedMedia	string	Resolvable Squidle+ API URL for the source image

4.5. NRMN Photo Quadrat Aggregated Data Product

4.5.1. Extract

The NRMN product is distinctive in drawing from two complementary data sources. The first is a static export of survey-aggregated benthic cover data from the legacy NRMN database, securely cached in S3 storage. The second is a dynamic extraction of raw point annotations from the Squidle+ platform via its REST API, accessing private NRMN data shared through a dedicated Squidle+ account (aodn@nrmn.org.au).

The Squidle+ extraction requires three separate API calls: export of all annotation sets on the RLS platform; export of all annotations within those sets; and export of all labels with their translations to the desired label schemes. The resulting JSON responses are validated using Pydantic models to ensure structural integrity before proceeding to transformation. The annotation set exports contain deeply nested JSON structures encompassing campaign and deployment hierarchies, media objects with geolocation poses, point coordinates within images and user attribution metadata, all of which must be flattened into a tabular structure.

4.5.2. Transform

Static export transformation

The legacy database export is already aggregated to the survey level and requires relatively light transformation. Scalar fields are added to identify the source database (NRMN), programme (RLS), and annotation method (full biodiversity). A unique dataset identifier is constructed from the survey identifier and label scheme resolution. The RLS taxonomic lineage is enriched using Squidle+ label scheme translations, for example translating “Crustose coralline algae” into its full hierarchical path “Biota > Macroalgae > Crustose coralline algae”. Bleached and dead coral indicator columns are consolidated into a single tags field aligned with the Squidle+ tagging convention. Column names are standardised to match the unified output schema.

Squidle+ export transformation

The dynamic Squidle+ data arrives as raw point annotations rather than survey-aggregated cover estimates, requiring substantially more processing. Two distinct annotation methods are present in the NRMN programme: full biodiversity surveys, which classify every point on the photo quadrat; and targeted coral surveys, which focus specifically on coral and bleaching indicators. Each method follows a distinct aggregation pathway to correctly calculate survey-level benthic cover percentages from the raw point counts.

Label scheme translation

A critical step in the NRMN pipeline is the normalisation of annotations across different labelling schemes. The NRMN programme has used multiple classification schemes over time, principally CATAMI 1.4 and the RLS Catalogue. Squidle+ facilitates translation between label schemes, and the pipeline exploits this capability to produce consistent benthic cover categories with both the original label, its RLS equivalent and full RLS and CATAMI lineage paths. This translation enables temporal comparison of surveys annotated under different schemes.

Validation

Both transformed streams are validated against a common output schema specifying 27 fields covering site identification, spatial and temporal coordinates, label scheme information, cover percentages and provenance metadata.

4.5.3. Load

The final product is delivered in two variants. The private survey table is stored as a Parquet file for internal use by the NRMN programme, retaining finer-grained annotation data subject to data sharing agreements. The public survey table is stored as both a Parquet file and a CSV file for open distribution. Both variants benefit from Parquet compression, reducing the output to approximately 1 MB from the combined static and Squidle+ source volumes.

Table 12: Output Schema – NRMN Photo Quadrat Aggregated Data Product

Field	Type	Description
dataset_id	string	Unique identifier for the survey (survey ID + resolution)
database	string	Source database identifier (NRMN)
program	string	Programme name (RLS)
country	string	Country of the survey site
area	string	Broad geographic area of the survey
location	string	Location name within the area
site_code	string	Standardised code for the monitoring site
site_name	string	Human-readable site name
latitude	float64	Latitude of the survey site (decimal degrees)
longitude	float64	Longitude of the survey site (decimal degrees)
date	date32	Date of the survey
hour	string	Hour of the survey, where available
depth	float64	Depth at the survey site (m)
visibility	float64	Underwater visibility at the survey site (m)
label_scheme	string	Label classification scheme used for the annotation
original_label	string	Original benthic category label as annotated

rls_label	string	Label translated to RLS Catalogue scheme
rls_lineage	string	Full hierarchical path in the RLS Catalogue
catami_label	string	Label translated to CATAMI 1.4 scheme
catami_lineage	string	Full hierarchical path in CATAMI 1.4
tags	string	Condition tags (e.g. bleached, dead)
label_count	float64	Number of annotation points assigned to this label
dataset_total_label_count	float64	Total annotation points in the survey
percent_cover	float64	Percentage cover (label_count/dataset_total_label_count × 100)
sq_annotation_set_id	float64	Squidle+ annotation set identifier (null for static export)
sq_annotation_set_owner	string	Squidle+ annotation set owner (null for static export)
method	string	Annotation method (full biodiversity or targeted coral)

4.6. Aggregated AMSA Vessel Tracking Data Product

4.6.1. Extract

AMSA publishes vessel tracking data as monthly ZIP archives containing Shapefiles, distributed through its operational data service. The archive structure is nested: an outer ZIP contains an inner ZIP, which in turn contains the Shapefile component files (.shp, .dbf, .shx, .prj). The extraction process downloads each monthly archive from S3 into memory, parses the year and month from the outer filename using resilient regular expression patterns that accommodate variant naming conventions (month abbreviation versus full month name, varying month/year ordering), and extracts the inner contents to a temporary directory. Shapefiles are read using pyogrio with Arrow integration, and geometry columns are converted to Well-Known Binary (WKB) for safe transport through the pipeline. A fallback stream-based unzip method handles archives that fail standard extraction.

4.6.2. Transform

Column standardisation

All column names are uppercased and known misspellings are mapped to standard names. The pipeline handles variants including TMESTAMP, TIMETAMP and TIMESAMP (all mapped to TIMESTAMP), and LATITUDE/LONGITUDE (mapped to LAT/LON).

Coordinate recovery

Where LAT and LON columns exist, their values are used directly. Where these columns are missing or contain null values, coordinates are derived from the Shapefile geometry by decoding WKB point objects to extract X (longitude) and Y (latitude) values. All coordinates are rounded to five decimal places.

Timestamp normalisation

The TIMESTAMP field is parsed using a coalesced multi-format strategy that handles date-only values, 24-hour datetime strings and 12-hour datetime strings with AM/PM markers. All timestamps are normalised to millisecond-precision UTC.

Enrichment

Each record is enriched with three provenance fields: SOURCE_FILE_NAME (identifying the monthly Shapefile from which the record originated), PROCESSED_DATE (the UTC timestamp of ETL processing) and H3_INDEX (computed at a configurable resolution, currently set to 8).

Waterfall validation

Validation is applied as a sequential waterfall, with each stage preserving and labelling its rejected records. Stage 1 removes exact duplicate rows. Stage 2 rejects records with missing (null) LAT or LON values. Stage 3 rejects records with coordinates outside global bounds (−90 ≤ LAT ≤ 90, −180 ≤ LON ≤ 180). Stage 4 rejects records with coordinates outside the configured Australian Search and Rescue Region bounding box (75°E–163°E, 86.6°S–2°S). Rejected rows from all stages are concatenated into a single rejected DataFrame, each tagged with a REJECTION_REASON, and written to a separate output location for traceability and diagnostics.

Schema enforcement

A strict output schema is applied: missing expected columns are added as nulls, columns are reordered consistently and the dataset is cast to the target Arrow schema. The typed table is then converted back to Polars for writing.

4.6.3. Load

Validated records are loaded into a yearly master Parquet file using an idempotent upsert strategy. The pipeline loads the existing yearly master (if present), removes any rows matching the current SOURCE_FILE_NAME, appends the newly validated rows and overwrites the master file. This approach allows safe reprocessing of individual monthly deliveries without creating duplicate records. Rejected records are written as separate Parquet files per source, stored alongside the master in a dedicated rejected-records directory.

Table 13: Output Schema – Aggregated AMSA Vessel Tracking Data Product

Field	Type	Description
CRAFT_ID	string	Unique vessel identifier from AMSA source data
LAT	float64	Latitude of the vessel position report (decimal degrees, 5 d.p.)
LON	float64	Longitude of the vessel position report (decimal degrees, 5 d.p.)
TIMESTAMP	Timestamp [ms, UTC]	Date and time of the position report (UTC)
COURSE	float64	Vessel course over ground (degrees)
SPEED	float64	Vessel speed over ground (knots)
TYPE	string	Primary vessel type classification
SUBTYPE	string	Secondary vessel type classification
LENGTH	float64	Vessel length (m)
BEAM	float64	Vessel beam / width (m)
DRAUGHT	float64	Vessel draught (m)
H3_INDEX	string	H3 hexagonal spatial index at resolution 8
SOURCE_FILE_NAME	string	Name of the monthly Shapefile from which the record was extracted
PROCESSED_DATE	timestamp	Date and time of ETL processing (UTC)

Table 14: Additional fields in the rejected records output

Field	Type	Description
REJECTION_REASON	string	Reason for rejection (Duplicate, Missing Coordinates, Invalid Global Coords, or Outside SAR Region)

4.7. Dugong Aerial Survey Database

4.7.1. Extract

The dugong pipeline consolidates historical and ongoing aerial survey data from two distinct source formats into a single unified product aligned with the Darwin Core standard. The first source is a flat Parquet file containing historically aggregated sighting records from the Data Management System (DMS). The second comprises JCU geospatial Shapefile archives, each containing Shapefiles that split survey data into two primary tables: Sightings and Herds.

DMS extraction

The DMS Parquet file is read directly from S3 into memory. Because this source is already in a structured tabular format, extraction is straightforward, though column naming and typing require standardisation to match the target Darwin Core schema.

JCU shapefile extraction

The JCU archives are extracted to a temporary directory and their component Shapefiles are read using pyogrio with Arrow-native integration, separating the Sightings and Herds tables. Region abbreviations are derived by lowercasing the raw region string and mapping it to a standard abbreviation through a strict dictionary (for example, “northern great barrier reef” maps to “nGBR”). An eventID is constructed using the pattern {region_abbreviation}-{YYYY}-{MM}, providing a unique survey-level identifier.

Calf and adult unrolling

A distinctive feature of the dugong pipeline is the “explode” logic that converts aggregated group counts into discrete occurrence records by life stage. For each sighting record, adult counts are calculated by subtracting the number of calves from the total group count. Individual row structures are then generated for adult and calf occurrence types, and zero-count rows are filtered out before the structures are exploded into discrete records. This unrolling ensures that the output product records individual life-stage occurrences rather than aggregated group totals, conforming to Darwin Core occurrence conventions.

Defensive handling

The extraction logic includes robust handling for inconsistencies across historical files and survey teams, including variant column naming conventions (for example, number_in_group versus n_in_gp, calves versus n_calves), inconsistent region naming casing, dates formatted as raw strings versus native datetimes, and conflicting schema types across different Shapefiles where numeric fields in one file may be presented as strings in another.

Waterfall validation

Validation is applied at the point of extraction using a sequential waterfall. Records with missing latitude or longitude values are rejected with the reason “Missing Coordinates”. Records with missing date or time fields are rejected with the reason “Missing Date”. Records with zero group counts are filtered out during the explode phase rather than rejected explicitly. All rejected records are tagged with a RejectionReason field and routed to a separate error output, preserving native

types for stable fields while stringifying highly variable source fields to avoid schema merge conflicts between the heterogeneous Shapefile structures.

4.7.2. Transform

The validated and concatenated master DataFrame from both DMS and JCU sources undergoes two spatial enrichment operations.

H3 spatial indexing

An h3Index is computed at resolution 4 using the polars_h3 library, based on the validated decimalLatitude and decimalLongitude fields. Resolution 4 was selected to reflect the broad spatial scale of aerial surveys, where individual sighting positions represent approximate locations within a survey transect rather than precise point observations.

Australian Marine Region tagging

Point-in-polygon spatial joins are executed to append australianMarineRegionsTags to each record, enabling direct filtering by national marine management regions without additional GIS processing.

All columns are mapped to Darwin Core field names during extraction (for example, lat to decimalLatitude, turbidity to turbidityScale). The scientificName field is hardcoded to Dugong dugon and the scientificNameID to the corresponding WoRMS LSID (urn:lsid:marinespecies.org:taxname: 220227), as this is a single-species product.

4.7.3. Load

The final enriched dataset is written as a single Parquet file with Snappy compression (dugong.parquet) to the optimised S3 bucket. Rejected records are written as a separate Parquet file (master_dugong_dataset_rejected.parquet) to the error bucket, ensuring complete traceability and zero data loss during the extraction process.

Table 15: Output Schema – Dugong Optimised Data Product

Field	Type	Description
eventID	string	Unique survey-level identifier ({region}-{YYYY}-{MM})
occurrenceID	string	Unique occurrence-level identifier
occurrenceType	string	Life stage of the occurrence record (adult or calf)
scientificName	string	Scientific name (hardcoded: Dugong dugon)
scientificNameID	string	WoRMS LSID (urn:lsid:marinespecies.org:taxname:220227)
eventDate	date	Date of the aerial survey
decimalLatitude	float64	Latitude of the sighting (decimal degrees)

decimalLongitude	float64	Longitude of the sighting (decimal degrees)
locationID	string	Identifier for the survey region
locationName	string	Name of the survey region
geometry	binary (WKB)	Point geometry in Well-Known Binary format
organismQuantity	int64	Number of individuals in this occurrence record
organismQuantityType	string	Unit of quantity (individuals)
recordedBy	string	Name(s) of the survey observer(s)
beaufortScale	string	Sea state at time of observation (Beaufort scale)
turbidityScale	string	Water turbidity at time of observation
h3Index	string	H3 hexagonal spatial index at resolution 4
australianMarineRegionsTags	string	Australian Marine Region identifiers applicable to the sighting

Table 16: Additional fields in the rejected records output

Field	Type	Description
rejection_reason	string	Reason for rejection (Missing Coordinates or Missing Date)

4.8. Summary of common benefits

The ETL processes described in this chapter share several design properties that collectively advance the project's objective of making marine environmental data routinely consumable by assessment processes.

Reproducibility

All pipelines are version-controlled, configuration-driven and orchestrated through Prefect, ensuring that any product can be regenerated from source data with identical results. The original source formats – relational databases, nested archives, multi-file Darwin Core structures and bespoke GeoServer exports – each require specialised domain knowledge to access and interpret. The ETL pipelines encapsulate this knowledge in code, eliminating the need for downstream users to navigate complex source structures.

Space efficiency

Modern columnar compression reduces the combined source data volume from several gigabytes to tens of megabytes, with compression ratios ranging from approximately 88% (seagrass) to over 99% (seabirds). This reduction benefits both storage costs and network transfer times for end users.

Time efficiency

Scheduled updates allow users to access current aggregated data without re-implementing the aggregation, validation and harmonisation steps. For products with dynamic sources (seabirds, kelp, NRMN, AMSA), new data are incorporated through automated pipeline runs without manual intervention.

Cloud-native delivery

All products are stored in Parquet format on S3, supporting efficient columnar access, predicate pushdown and integration with cloud-native analytical tools in both Python and R. This delivery model aligns with the AODN's broader cloud-optimised data strategy and the national Trusted Environmental Data Supply Chain initiative.

5. Training strategy

A key deliverable of Project 5.9 is building the capacity of practitioners, researchers and government assessors to use the assessment-ready data products independently and effectively. The project's original workplan proposed a face-to-face training workshop to fulfil this objective. Following consultation with the project Steering Group, the training component was redesigned as a series of short screencast videos accompanied by companion code repositories. This format offers several advantages over a single workshop event: the videos can be accessed asynchronously by a geographically distributed audience; viewed repeatedly at the learner's own pace; and updated incrementally as datasets or tools evolve. The shift in format extends the potential reach of the training from the small number of participants who could attend a single event in one location to the broader community of marine data users across government, research and consultancy sectors. All videos are publicly available in the IMOS YouTube channel https://www.youtube.com/@IMOS5395.

5.1. Purpose and learning objectives

The purpose of the training series is to equip users with the practical skills needed to access, explore, map and analyse Project 5.9 data products for marine assessment and reporting workflows. The target audience includes Government assessors within DCCEEW, marine park managers at Parks Australia and the Great Barrier Reef Marine Park Authority, environmental consultants, university researchers and postgraduate students. Participants are assumed to be familiar with marine spatial concepts and environmental data types, and to have a working knowledge of either Python or R, however there is no assumption of prior experience with cloud-optimised data formats or the specific technical infrastructure underpinning the Project 5.9 products.

The learning objectives of the series are to enable users to: understand what cloud-optimised data formats are and when they are advantageous over traditional download-and-analyse workflows; connect programmatically to each of the featured Project 5.9 datasets through AODN and cloud endpoints; create maps of observations at multiple spatial resolutions; aggregate point and polygon data into H3 hexagons at different scales and visualise summary statistics; identify locations suitable for trend analysis based on criteria such as temporal coverage, sampling consistency and data completeness; and extract and visualise time series for selected hexagons or regions as a basis for preliminary trend assessment.

5.2. Scope and deliverables

The training package comprises three components. The primary deliverable is a series of six to ten screencast videos, each under ten minutes in length, progressing from introductory concepts through to applied analytical workflows. The videos use four featured datasets - seabirds, benthic annotations (kelp and NRMN), marine vessel traffic and seagrass -as worked examples throughout the series. Each video follows a consistent pedagogical pattern: introduce the concept or dataset; demonstrate the workflow in a live coding environment; and conclude with a summary and link to companion materials.

The second deliverable is a set of companion code repositories in both Python and R, mirroring all workflows demonstrated in the screencasts. The Python repository uses standard geospatial and dataframe libraries including Polars, pyarrow and h3-py. The R repository uses the arrow, sf, and H3-compatible packages to implement analogous workflows. Library versions are pinned in example environments to ensure long-term reproducibility. The parallel structure of the Python and R repositories allows users to work in either language to translate steps directly between the two implementations.

The third deliverable is a set of companion Jupyter notebooks documenting data endpoints, key library requirements and example queries for each dataset, providing a quick-reference resource for users who prefer to work interactively rather than follow the video sequence linearly.

Custom analyses for individual stakeholders and development of new datasets or substantial re-engineering of existing Project 5.9 products are outside the scope of the training deliverables.

5.3. Video series structure

The video series is structured as a progressive sequence, with each instalment building on concepts introduced in earlier videos. Table 17 summarises the planned content of each video.

The first three videos establish conceptual foundations - what the data products are, how cloud-optimised formats work and why H3 spatial indexing is used - before any coding is required. Videos 4 and 5 provide the practical entry point, walking users through environment setup, library configuration and their first programmatic connection to a cloud-hosted dataset. Videos 6 through 9 build analytical capability progressively, moving from spatial visualisation through multi-resolution aggregation to temporal trend extraction. The parallel Python and R structure throughout videos 4 to 9 ensures that users working in either language can follow the complete analytical sequence without needing to translate between languages on their own.

Table 17: Planned structure of the training video series

Video	Title	Content summary
1	NESP 5.9 overview and use cases	Title card and introduction to the project and the four featured datasets. High-level explanation of “assessment-ready” data and typical user scenarios (mapping, trends, reporting). Brief tour of the AODN portal pages and metadata records. Preview montage of the analytical outputs the series will produce.
2	Cloud-optimised formats: pros and cons	Definition of cloud-optimised formats with a visual comparison of traditional download versus partial cloud reads. Demonstration using a Project 5.9 dataset to show tiling, chunking and partial reads. Balanced discussion of benefits (performance, scalability, reproducibility) and limitations (network dependency, endpoint complexity, library requirements). Summary of when to choose cloud-optimised access versus local files.

3	H3 spatial indexing concepts	Definition of spatial indices with visual comparison of triangular, rectangular and hexagonal grids. Introduction to Uber's H3 hierarchical hexagonal index with interactive zoom demonstration. Discussion of advantages (uniqueness, scalability, efficiency, multi-language library support) and limitations (fixed resolutions, loss of original coordinate precision). Explanation of why H3 was adopted for Project 5.9 products.
4–5	Connecting to data in Python and R	Project structure, environment setup and key libraries. Walk-through of connecting to a vector dataset (e.g. seabirds or vessel traffic), filtering by time and region, and previewing attributes in a dataframe. Connection to a second dataset (e.g. seagrass or benthic data), demonstrating lazy cloud access and metadata inspection. Quick map preview to visualise the data spatially. Parallel structure across Python and R videos.
6–7	Mapping and H3 aggregation in Python and R	Mapping observations at native resolution. Re-mapping at coarser resolutions using grid cells or resampled rasters. Aggregation of point data into H3 hexagons at two resolutions (fine and coarse) with density or summary statistic visualisation. Demonstration of how multi-resolution views support selection of appropriate analysis scales.
8–9	Identifying trend sites and extracting time series in Python and R	Transition from spatial ("where is it?") to temporal ("how is it changing?") questions. Application of selection rules to identify candidate hexagons for trend analysis (e.g. minimum years of data, minimum observations per year). Extraction of time series for one or two selected locations, plotting with basic trend lines. Discussion of caveats including sampling design, autocorrelation and the need for advanced statistical methods. Series wrap-up, recap of key skills and invitation for feedback.

5.4. Content design approach

Each screencast follows a consistent four-part pattern designed to maximise both immediate comprehension and long-term reusability of the material.

Introduce the dataset and access pathway

Each video opens with a brief description of the relevant dataset's purpose, key attributes and provenance (for example, seabird occurrences from OBIS Australia Node Darwin Core archives; benthic habitat classifications from Squidle+ annotations; vessel tracking positions from AMSA AIS data; or seagrass presence and species records from Seamap Australia and state agency programs). Access pathways are shown, including the AODN portal pages, institutional metadata records and cloud storage endpoints.

Demonstrate the workflow in a live coding environment

The core of each video is a narrated live coding demonstration, showing the complete workflow from data connection through to analytical output. Code is written and executed in real time (or with minimal editing for clarity), allowing viewers to see both the expected outputs and the intermediate steps. The Python demonstrations use Jupyter notebooks; the R demonstrations use RStudio workflows.

Explain cloud-optimised concepts in context

Rather than treating cloud infrastructure as a separate topic after the introductory videos, each analytical video reinforces cloud-optimised concepts as they arise naturally in the workflow. For example, the Vessel Traffic video demonstrates lazy loading and partial reads from Parquet on S3, while the H3 aggregation video shows how the pre-computed spatial index enables efficient group-by operations without full dataset materialisation.

5.5. Production timeline and governance

Video production follows a six-week cycle. In Week 1, the datasets, access mechanisms and technical stack are confirmed, and a detailed content outline is drafted for each video. Weeks 2 and 3 are devoted to developing and testing the Python and R code examples, with internal feedback sought from data custodians at IMOS, AODN and contributing institutions to ensure technical accuracy and reproducibility. In Weeks 4 and 5, the screencasts are recorded, edited and reviewed, with IMOS and NESP branding applied and accessibility standards met (including closed captions for all audio content). Week 6 covers stakeholder review and sign-off by the project Steering Group, followed by upload of the videos and publication of the companion code repositories.

Quality assurance is maintained through three review layers. A technical review confirms the accuracy, reproducibility and performance of all code examples and analytical workflows. A plain-language review ensures that content is able to be understood by the target audience of technically literate but non-specialist users. Regular check-ins with NESP 5.9 leadership and the project Steering Group provided strategic oversight and alignment with the broader project objectives.

5.6. Risks and assumptions

The training strategy rests on two principal assumptions. First, that the Project 5.9 datasets are accessible via stable cloud endpoints in their final formats by the time video production begins. Second, that the core audience has working skills with either Python or R, and marine spatial concepts; and therefore does not require instruction in general programming or geospatial fundamentals.

Three risks have been identified with corresponding mitigations. The first is the instability of data access endpoints during or after production. This is mitigated by maintaining close coordination with internal AODN Data Pipeline teams and using documented, versioned endpoints that can be updated in the companion code repositories without re-recording video content. The second risk is that the analytical complexity may be too high for short-format videos. This is addressed through tight scripting of each screencast, offloading detailed code and explanatory notes to the companion

repositories and notebooks so that the videos focus on workflow logic rather than code syntax. The third risk is that future changes to Python or R libraries could render demonstrated workflows non-functional. This is mitigated by relying on well-maintained, mainstream libraries, pinning specific library versions in the example environments, and documenting version requirements clearly in the repository metadata.

6. Conclusions

Project 5.9 set out to address a specific and well-documented problem: Australia possesses world-class marine research data, but these data are not routinely available in forms that can be consumed by the national assessment processes that most need them. Through a structured programme of stakeholder consultation, technical infrastructure development and institutional integration, the project has delivered a suite of cloud-optimised, assessment-ready data products that directly respond to this gap. This concluding chapter summarises the project's principal achievements and considers the potential applications of its outputs across the range of policy and reporting processes they were designed to serve.

6.1. Summary of achievements

The project delivered against all three of its interlinked objectives. The project's prioritisation workshop, convened in February 2025 with over 40 participants from 16 organisations, provided a transparent and collaborative mechanism to identify the datasets of highest value for national assessment. Six priority datasets were selected, spanning three thematic areas aligned with the State of the Environment framework: habitats and communities (national seagrass aggregation, kelp annotations, NRMN reef benthos), species (seabirds, dugong aerial surveys) and pressures (AMSA vessel tracking).

For each dataset, the project designed and implemented automated Extract–Transform–Load pipelines that convert heterogeneous source data – ranging from 83 disparate seagrass datasets with inconsistent schemas to 41 Darwin Core seabird archives with complex relational structures – into standardised Parquet products. These products are enriched with H3 spatial indices, Australian Marine Region tags and internationally recognised taxonomic identifiers from WoRMS and CAAB, and are validated against strict schemas at every stage of the pipeline. Compression ratios exceeding 88% reduce multi-gigabyte source collections to compact, analytically efficient files accessible from Python, R or any Parquet-compatible tool.

Crucially, the ETL pipelines and data products are not stand-alone project outputs. They have been integrated into IMOS national research infrastructure and are maintained as part of routine AODN operations, ensuring that products are refreshed automatically as new source data arrive and that they will persist beyond the life of the NESP funding cycle. A companion training programme of screencast videos and open-source code repositories in Python and R was developed to build practitioner capability across the target user community.

6.2. Applications for national assessment and reporting

The data products developed by Project 5.9 were designed to serve multiple, recurring assessment processes. Their potential applications span the following domains.

State of the Environment reporting

The SOE 2026 reporting represents the most immediate application for the Project 5.9 products. The seagrass, seabird, kelp, reef benthos and vessel tracking datasets directly address indicators required for the marine chapter, providing pre-built, quality-controlled data layers on the status and distribution of key habitats, species and pressures. The standardised temporal fields and H3 spatial indexing enable SOE authors to extract national and regional summaries, map observation density and assess temporal coverage without the bespoke, labour-intensive data wrangling that characterised previous reporting cycles. With the SOE reporting frequency increasing from every five years to every two years from 2028, the availability of routinely updated, analysis-ready products is expected to be a significant enabler of the accelerated cycle. The automated refresh built into the AODN delivery model means that SOE 2028 and subsequent editions can draw on the same trusted products with updated data, establishing a consistent baseline across reporting periods.

National environmental data supply chain

The Nature Positive Plan established Environment Information Australia to deliver a national environmental data supply chain providing clear, authoritative sources of high-quality environmental information. The Project 5.9 products are designed to feed directly into this supply chain. By adhering to national and international data standards – OGC protocols for web service delivery, ISO 19115 metadata for discoverability, Darwin Core for biological observations, and CAAB and WoRMS for taxonomic interoperability – the products are immediately consumable by DCCEEW's geospatial information systems and compatible with global biodiversity information platforms including OBIS and GBIF. The cloud-optimised Parquet delivery model and the AODN portal's catalogue infrastructure provide the stable, documented endpoints that a functioning national supply chain requires. The project thus demonstrates a replicable model for how research data managed by national facilities can be systematically transformed into assessment-ready information products at scale.

Environmental approval processes

Assessment-ready marine data products have direct utility in environmental approval processes under the EPBC Act, including those for offshore renewable energy development, port expansions, dredging operations and marine infrastructure. The H3 spatial indexing enables rapid extraction of data within proposed development footprints and their surrounding impact zones, while the standardised temporal fields support assessment of seasonal patterns and long-term trends at specific locations. The vessel tracking product is particularly relevant to cumulative impact assessments, providing a consistent pressure layer that can be overlaid with habitat and species distribution data to evaluate interaction risks. The seabird and dugong products support assessments for listed threatened and migratory species, and the seagrass and kelp products inform evaluations of potential impacts on ecologically significant habitats. By providing these data

in documented, reproducible formats with clear provenance, the project reduces the time and cost of evidence gathering for both proponents and regulators and improves the consistency of the evidence base across assessments.

Environmental economic accounts

The preparation of Environmental Economic Accounts requires consistent spatial and temporal data on environmental assets, their extent and their condition. The Project 5.9 products are well suited to this application. The seagrass product, with its national spatial coverage from 1967 to 2024 and standardised presence/absence records at H3 ~1 m² resolution, provides a basis for tracking changes in seagrass extent across Australian Marine Regions. The kelp and reef benthos products offer analogous capabilities for macroalgal and coral habitats. The Australian Marine Region tags embedded in the seagrass product enable direct alignment with the spatial reporting units used in the national accounts framework, eliminating the need for additional spatial processing. As the ABS continues to develop and refine its environmental-economic accounting for the marine domain, the availability of standardised, spatially indexed ecological datasets with long temporal baselines will become increasingly valuable.

Marine Park Management and Threatened Species Assessments

Parks Australia and the Great Barrier Reef Marine Park Authority were represented at the prioritisation workshop and identified specific data needs that the Project 5.9 products address. The seabird product, aggregating observations from 1939 to the present, supports assessments of seabird populations within and adjacent to marine park boundaries. The seagrass and kelp products inform condition assessments for key habitats within the Australian Marine Parks network. The dugong aerial survey database provides a four-decade record of population distribution directly relevant to recovery planning for this listed marine species. The vessel tracking product enables park managers to characterise vessel traffic patterns and assess pressures within managed areas. These applications extend to Australia's reporting obligations under the Kunming-Montreal Global Biodiversity Framework, including progress towards the "30 by 30" marine protection target, and to migratory species assessments under bilateral and multilateral agreements.

Sustainable Ocean Plan

Australia's development of a Sustainable Ocean Plan requires comprehensive evidence based on the distribution and intensity of human activities in the marine environment alongside the ecological assets and values that must be sustained. The vessel tracking product provides a readily accessible, annually updated record of shipping density and vessel type across the Australian Search and Rescue Region, supporting spatial planning and cumulative impact assessments. Overlaying this pressure layer with the habitat and species products enables integrated analyses of the kind required for marine spatial planning – for example, identifying areas where vessel traffic intersects with high-value seagrass habitat or important seabird foraging zones.

6.3. Data gaps and recommendations for future opportunities

The prioritisation workshop (Chapter 3) identified a substantially larger set of candidate datasets than the project could address within its scope and timeline. The datasets that were not selected for development remain strong candidates for future iterations of this work. This section documents the principal gaps and provides recommendations for how they might be addressed.

Sharks and rays

Elasmobranch data were discussed extensively at the workshop, with contributions from acoustic tracking arrays, Global Archive baited remote underwater video (BRUVS) systems and Reef Life Survey visual census. The diversity of data types (tracking telemetry, video-based abundance, visual census counts) and the complexity of data sharing arrangements for satellite tracking data mean that a dedicated scoping exercise would be needed before ETL development could begin. A future product could draw on the growing archive of acoustic detection data managed through the IMOS Animal Tracking Facility.

Marine fish

Long-term underwater visual census data from the Reef Life Survey and BRUVS programmes offer a strong foundation for a national marine fish aggregated product. These data are complementary to the reef benthos product already developed and could share much of the Squidle+ and Darwin Core translation infrastructure. The primary challenge is to maintain the traceability of the image annotations down to the original images.

Marine mammals beyond dugongs

Cetacean and pinniped data were identified as important but fragmented across multiple institutions and tracking programmes. Data sharing and integration challenges remain the principal barrier. The development of a standardised cetacean product would benefit from closer collaboration with the IMOS Animal Tracking Facility and the Australian Marine Mammal Centre.

Mangroves and salt marshes

Geoscience Australia's 10-metre resolution annual mapping product provides a well-maintained national baseline for mangrove extent. The workshop concluded that this product already serves many assessment needs and that Project 5.9's efforts were better directed at datasets where no comparable national product existed. However, integration of the GA mangrove product into the AODN cloud-optimised ecosystem – particularly with H3 spatial indexing and Australian Marine Region tagging – could add value for users requiring consistent spatial querying across habitat types.

Plankton and microbial data

The IMOS Biological Ocean Observer (BOO) has made significant progress in aggregating plankton and microbial observations. These data require different analytical approaches (continuous time series, biogeochemical context) from the occurrence-based products developed under Project 5.9, but the cloud-optimised delivery patterns and H3 spatial indexing framework are directly transferable.

Emerging pressure layers

Workshop participants identified microplastics monitoring, fishing pressure layers (vessel monitoring system data, logbook records), underwater noise and light pollution as emerging pressure datasets of growing policy relevance. These datasets are at varying stages of maturity and standardisation. As monitoring programmes mature and data sharing arrangements are formalised, they represent natural candidates for the ETL infrastructure established by Project 5.9.

Oceanographic time series and ecosystem modelling

Satellite-derived sea-surface temperature, ocean colour, connectivity models and long-term mooring data from IMOS were discussed as complementary capabilities. These are predominantly raster or gridded time-series datasets that would benefit from cloud-optimised formats such as Zarr rather than the tabular Parquet format used for the current products. The architectural patterns (Prefect orchestration, Pydantic validation, S3 delivery) are format-agnostic and could be adapted for gridded products.

6.4. Limitations and future directions

The project has also revealed where the current evidence base remains insufficient. The prioritisation workshop identified several important data domains – sharks and rays, marine fish, marine mammals beyond dugongs, mangroves and salt marshes, plankton, oceanographic time series and emerging pressure layers such as microplastics and underwater noise – that could not be addressed within the project's scope and timeline but represent strong candidates for future iterations. The dugong aerial survey product remains in development pending finalisation of data sharing arrangements with JCU, highlighting the ongoing coordination required to bring custodian-held datasets into national infrastructure.

The workshop recommended that annual follow-up sessions be convened to reassess priorities, review progress and discuss technological developments such as hyperspectral satellite data, expanded eDNA sampling and new machine learning approaches to species detection. The ETL architecture established by Project 5.9 is designed to be extensible: adding new datasets requires developing a bespoke extraction and transformation module that plugs into the shared validation, schema enforcement and cloud delivery infrastructure, rather than rebuilding the pipeline from scratch.

A further area for development is the expansion of aggregated analytical products – maps, summary statistics and trend indicators derived from the assessment-ready datasets – that can serve users who need synthesised information rather than access to the underlying observation records. The training video series lays the methodological groundwork for this by demonstrating H3-based aggregation and time-series extraction workflows that could be formalised into routine indicator production.

6.5. Closing remarks

The central contribution of Project 5.9 is not the data products themselves, but the demonstration that research data can be systematically and sustainably transformed into assessment-ready information within existing national infrastructure. The project has shown that the gap identified by the Samuel Review – between well-managed research data and the information needed for effective environmental decision-making – can be narrowed through targeted investment in data engineering, institutional partnerships and workforce capability. By embedding the ETL pipelines and data products within IMOS/AODN operational infrastructure, the project ensures that its outputs will continue to serve Australia's marine assessment needs as reporting cycles accelerate and information demands grow. The model is replicable, the infrastructure is in place, and the data are ready.

7. References

ABS (2024). *Australian Environmental-Economic Accounts, 2024.* ABS cat. no. 4655.0. Australian Bureau of Statistics, Canberra.

AMSA (2024). About the Automatic Identification System. *Australian Maritime Safety Authority.* Available at: https://www.operations.amsa.gov.au/Spatial/DataServices/DigitalData (Accessed: 15 December 2025).

Apache Software Foundation (2024). *Apache Parquet.* Available at: https://parquet.apache.org/ (Accessed: 15 December 2025).

Apache Arrow Developers. (2026) Apache Arrow: A cross-language development platform for in-memory data (Version 23.0, PyArrow). Apache Software Foundation. Available at https://arrow.apache.org/docs/python/.

ARDC (2024). *Trusted Environmental Data and Information Supply Chains.* Australian Research Data Commons. Available at: https://ardc.edu.au/ (Accessed: 10 January 2026).

Australian Government (2022). *Nature Positive Plan: Better for the Environment, Better for Business.* Department of Climate Change, Energy, the Environment and Water, Canberra. Available at: https://www.dcceew.gov.au/environment/nature-positive-plan (Accessed: 5 January 2026).

Brodsky, I. (2018). H3: Uber's hexagonal hierarchical spatial index. *Uber Engineering Blog.* Available at: https://www.uber.com/en-AU/blog/h3/ (Accessed: 12 December 2025).

CAAB (2024). *Codes for Australian Aquatic Biota.* CSIRO Marine and Atmospheric Research. Available at: https://www.cmar.csiro.au/caab/ (Accessed: 10 January 2026).

CBD (2022). Kunming-Montreal Global Biodiversity Framework. Decision 15/4. *Convention on Biological Diversity.* CBD/COP/DEC/15/4, 19 December 2022.

Cleguer, C. and Marsh, H. 2023. An inventory of dugong aerial surveys in Australia. Report to the National Environmental Science Program. Centre for Tropical Water and Aquatic Ecosystem Research (TropWATER), Report 23/15, James Cook University. pp. 48

Colvin, S., Jolibois, E., Hall, A., Plot, V., & Pydantic contributors. (2026) Pydantic (Version 2.x) [Computer software]. Pydantic. Available at https://docs.pydantic.dev/latest/.

Davies C. H., Everett J. D. and Ord L. (2022) Integrated Marine Observing System (IMOS) Biological Ocean Observer - Shiny APP. v9.3. CSIRO. Service Collection. http://hdl.handle.net/102.100.100/447365?index=1.

Evans, K. (2024). *Identifying priority datasets of relevance to the Gippsland declaration area and pathways for their use in decision making.* Report to the National Environmental Science Program. CSIRO. NESP MaC Hub Project 3.21.

Evans, K., Bax, N. and Smith, D.C. (2017). Marine environment. In: *Australia State of the Environment 2016.* Australian Government Department of the Environment and Energy, Canberra. DOI: 10.4226/94/58b65a4537218.

Friedman, A., Pizarro, O., Williams, S.B. and Johnson-Roberson, M. (2012). Multi-scale measures of rugosity, slope and aspect from benthic stereo image reconstructions. *PLoS ONE,* 7(12), e50440. DOI: 10.1371/journal.pone.0050440.

Friedman A, Monk J, Pizarro O, Lindsay D, Oh E, Thornton B, Carroll A, Przeslawski R and Williams S (2026) Squidle+: a collaborative platform to manage, discover and annotate marine imagery. Front. Mar. Sci. 12:1677103. doi: 10.3389/fmars.2025.1677103.

GBIF (2024). *Global Biodiversity Information Facility.* Available at: https://www.gbif.org (Accessed: 15 December 2025).

Gordon, I.J. and Dunstan, P. (2022). *Scoping Study: Protected Place Management Initiative.* Report to the Reef and Rainforest Research Centre. Cairns, Queensland. NESP MaC Hub Project 1.1.

Hagstrom, H. and Marsh, H. (2022). Aerial survey-based estimates of dugong (*Dugong dugon*) abundance along the Australian coast: a review and recommendations. *Wildlife Research,* 49(8), 659–672. DOI: 10.1071/WR21101.

Hill, N.A., Barrett, N., Lawrence, E., Hulls, J., Dambacher, J.M., Nichol, S., Williams, A. and Hayes, K.R. (2017). Quantifying fish assemblages in large, offshore marine protected areas: an Australian case study. *PLoS ONE,* 9(10), e110831.

IMOS (2023). *IMOS Strategy 2023–2028.* Integrated Marine Observing System, University of Tasmania, Hobart. Available at: https://imos.org.au/ (Accessed: 10 January 2026).

Jordan, A., Klein-Salas, E. and Atkins, N. (2023). *NESP Marine and Coastal Hub Data Management Strategy.* University of Tasmania / IMOS.

Klein E, Yun J, Ghignone L, Miller M, Li X, Hu Y, Fierro Arcos D (2026) GBR Data Management System: A One-Stop Shop to Great Barrier Reef Data. Biodiversity Information Science and Standards 10:e185022. https://doi.org/10.3897/biss.10.185022

Klein-Salas, E. (2025). NESP 5.9 Kick-off Workshop: Making marine environmental data more assessment-ready. Workshop summary, 20 February 2025, Hobart. *NESP MaC Hub Project 5.9 internal report.* University of Tasmania / IMOS.

Lucieer, V., Flukes, E. and Monk, J. (2024). *Seamap Australia – a national benthic marine habitat classification scheme.* Institute for Marine and Antarctic Studies, University of Tasmania. Available at: https://seamapaustralia.org (Accessed: 10 January 2026).

OBIS (2024). *Ocean Biodiversity Information System.* Intergovernmental Oceanographic Commission of UNESCO. Available at: https://obis.org (Accessed: 15 December 2025).

Prefect Technologies, Inc. Prefect: Workflow orchestration for Python. Prefect Technologies, Inc. Available at https://www.prefect.io. (Accessed: 10 January 2026).

Richardson A.J., Eriksen R., Moltman T., Hodgson-Johnston I., Wallis J.R. (2020) State and trends of Australia's ocean report. Integrated Marine Observing System, Hobart.

Roughan, M., Hemming, M., Schiller, A., Moeseneder, C., Lenton, A., Bakker, D. and Sloyan, B. (2019). *IMOS Status and Trends of Australia's Oceans Report (STAR).* Integrated Marine Observing System, Hobart.

Samuel, G. (2020). *Independent Review of the EPBC Act – Final Report.* Department of Agriculture, Water and the Environment, Canberra. Available at: https://epbcactreview.environment.gov.au/ (Accessed: 5 January 2026).

Squidle+ (2024). *Squidle+ Marine Imagery Annotation Platform.* Greybits Engineering / University of Sydney. Available at: https://squidle.org (Accessed: 10 January 2026).

Stoeckl, N., Adams, V., Larson, S., Mahony, T., Steel, R., Allen, S., Emmerling, M., Jarvis, D., Ogier, E., Navarro, M., Chuah, S.H., Langlois, T. and Pecl, G. (2022). *Integrated data requirements for natural resource management.* Report to the Resilient Landscapes Hub of the Australian Government's National Environmental Science Program. University of Tasmania, Hobart. NESP MaC Hub Project 1.17.

Trebilco, R., Fischer, M., Carter, A., et al. (2022). Marine environment. In: *Australia State of the Environment 2021.* Australian Government Department of Climate Change, Energy, the Environment and Water, Canberra. DOI: 10.26194/f1rh-7r05.

Wieczorek, J., Bloom, D., Guralnick, R., Blum, S., Döring, M., Giovanni, R., Robertson, T. and Vieglais, D. (2012). Darwin Core: An evolving community-developed biodiversity data standard. *PLoS ONE,* 7(1), e29715. DOI: 10.1371/journal.pone.0029715.

Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., et al. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data,* 3, 160018. DOI: 10.1038/sdata.2016.18.

WoRMS Editorial Board (2024). *World Register of Marine Species.* Available at: https://www.marinespecies.org (Accessed: 15 December 2025). DOI: 10.14284/170.

Appendix A

ETL: Technical documentation

Detailed documentation of the Extract–Transform–Load data flows is provided in the following pages.

Seagrass Aggregated Data Product

Thomas Galindo - AODN

December 1, 2025 - Version 0.1.1

Contents

1 Background **2**

1.1 Overview . . . 2

1.1.1 Basis of the Seagrass Aggregated Data Product . . . 2

2 Methodology **3**

2.1 Extract . . . 3

2.1.1 Data Source . . . 5

2.1.2 Validation . . . 5

2.1.3 Extraction . . . 6

2.2 Transform . . . 7

2.2.1 Per Dataset Transformation Pattern . . . 7

2.2.2 Post Processing . . . 11

2.3 Load . . . 12

3 Flow Advantages **13**

1 Background

1.1 Overview

1.1.1 Basis of the Seagrass Aggregated Data Product

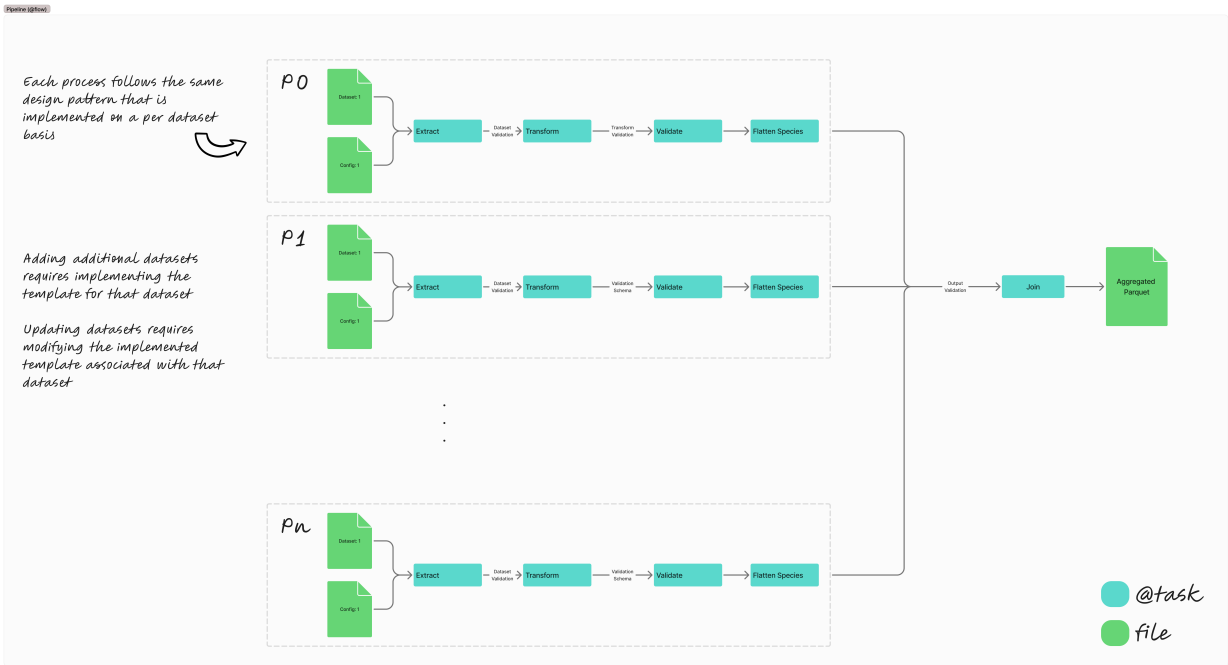
The goal of the aggregated data product is to flatten the disparate seagrass datasets into a single dataset.

2 Methodology

The Seagrass Aggregated Data Product (Data Product) relies on the standard Extract, Transform, and Load (ETL) methodology.

The ETL process to generate the Seagrass Aggregated Data Product is two step:

1. Transform source data to a harmonised intermediary standardised format
2. Transform intermediary harmonised data into singular data product



2.1 Extract

The Seagrass Aggregated Data Product is comprised of the following datasets:

Dataset	Metadata Title
GBR_NESP-TWQ-3.2.1-5.4_JCU_Seagrass_1984-2018_Site-surveys	Seagrass mapping synthesis: A resource for coastal ma
GoC_JCU_Marra-Sea-Country_2021_Seagrass_Deepwater-sites	Benthic habitats of Marra Sea Country - Gulf of Carpe
GoC_JCU_Marra-Sea-Country_2021_Seagrass_Site-Survey	Benthic habitats of Marra Sea Country - Gulf of Carpe
MaC_3_6_Furneaux	Mapping of subtidal temperate seagrass beds in Tayari
NT_JCU_CDU_NESP-MaC-1-12_Yanyuwa-Seagrass-Sites_2021-2022	Benthic habitats of Yanyuwa Sea Country, Barni - Wa
NT_JCU_Yanyuwa-Sea-Country_subtidal-sites_2022-2023	Subtidal benthic habitats of Yanyuwa Sea Country, Ba
SeamapAus_Ext_EOMAP_CocosKeelingMP_2022	Satellite-Derived Bathymetry and Seafloor Habitat Cla
SeamapAus_NSW_estuarine_macrophytes_2009	[SUPERSEDED] An Estuarine Inventory for New Sout
SeamapAus_NSW_estuarine_macrophytes_2022	NSW Estuarine Macrophytes 2023
SeamapAus_NSW_marine_habitats_2002	NSW Marine Habitats 2002
SeamapAus_NSW_nearshore_features_2017	NSW coastal nearshore reef extent 2017
SeamapAus_NT_GunnPt_benthiccomm_2020	Benthic Communities of the Gunn Point Peninsula, 20
SeamapAus_NT_Marra_seagrasss_2021	Benthic habitats of Marra Sea Country - Gulf of Carpe
SeamapAus_NT_Yanyuwa_seagrass_intertidal_2022	Benthic habitats of Yanyuwa Sea Country, Barni - Wa
SeamapAus_NT_Yanyuwa_seagrass_subtidal_2023	Subtidal benthic habitats of Yanyuwa Sea Country, Ba

Dataset	Metadata Title
SeamapAus_NT_greenturtle_habitat_2024	NT mapping of green turtle foraging habitat
SeamapAus_NT_seagrass_ArnhemKakaduGoC_2005	Seagrass meadows of Arnhem Land, Kakadu and Gulf
SeamapAus_QLD_EasternBanks_seagrass_biomass	Multi-temporal mapping of seagrass cover, species and
SeamapAus_QLD_GoldCoast_Broadwater_seagrass_2009	Gold Coast Broadwater Seagrass mapping 2005/08
SeamapAus_QLD_GreatSandyMP_benthic_habitat_2019	Great Sandy Marine Park benthic habitat 2019
SeamapAus_QLD_GreenIsland_seagrass_2020	Seagrass meadow extents derived from field to spacebo
SeamapAus_QLD_HerveyBay_seagrass_1998	Seagrass meadows of Hervey Bay and the Great Sandy
SeamapAus_QLD_LizardIsland_2014	Habitat Map for Lizard Island reef, Australia derived fr
SeamapAus_QLD_LowIsles_seagrass_1997	Seagrass and associated benthic community data deriv
SeamapAus_QLD_MidgePoint_seagrass_2017	Seagrass meadows extents derived from field to spaceb
SeamapAus_QLD_MoretonBay_seagrass_2004	Habitat map of seagrass cover derived from a supervis
SeamapAus_QLD_MoretonBay_seagrass_2011	Habitat map of seagrass cover derived from a supervis
SeamapAus_QLD_MoretonBay_seagrass_2015	Seagrass cover map for the optically complex waters o
SeamapAus_QLD_TorresStrait_GoC_4decadesynth_2022	Four Decades of seagrass meadows data from Torres S
SeamapAus_QLD_TorresStrait_seagrass_consolidation_intertidal_2014	Torres Strait seagrass mapping consolidation 2002-201
SeamapAus_QLD_TorresStrait_seagrass_consolidation_subtidal_2014	Torres Strait seagrass mapping consolidation 2002-201
SeamapAus_QLD_YulePoint_seagrass_2019	Seagrass meadow extents derived from field to spacebo
SeamapAus_QLD_coastal_wetlands_1999	Queensland Coastal Wetlands Resources Mapping Dat
SeamapAus_SA_state_benthic_habitats_2013	South Australia State Marine Benthic Habitats (DEWI
SeamapAus_SA_state_estuarine_habitats_2009	Estuarine Habitats of South Australia
SeamapAus_VIC_statewide_habitats_2023	Victorian Statewide Marine Habitat Map 2023
SeamapAus_WA_CockburnSound_seabed_2004	Mapping of selected areas of Cockburn Sound, Wester
SeamapAus_WA_CockburnSound_seagrass_1967	Changes in seagrass coverage in Cockburn Sound, Wes
SeamapAus_WA_CockburnSound_seagrass_1972	Changes in seagrass coverage in Cockburn Sound, Wes
SeamapAus_WA_CockburnSound_seagrass_1981	Changes in seagrass coverage in Cockburn Sound, Wes
SeamapAus_WA_CockburnSound_seagrass_1995	Changes in seagrass coverage in Cockburn Sound, Wes
SeamapAus_WA_CockburnSound_seagrass_1999	Changes in seagrass coverage in Cockburn Sound, Wes
SeamapAus_WA_EOMAP_AshmoreCartier_2022	Satellite Mapping of Bathymetry and Habitats of Ashr
SeamapAus_WA_IMSA_seafloor_Lancelin_2017	Very High Resolution Seafloor Classification and Satell
SeamapAus_WA_IMSA_seafloor_Swanbourne_2019	Very High Resolution Seafloor Classification and Satell
SeamapAus_WA_JurienBay_benthic_habitat_2018	Jurien Bay benthic habitat mapping
SeamapAus_WA_MarineFutures_biota	WA Marine Futures Project - biota
SeamapAus_WA_Recherche_Archipelago_2003	Characterising the fish habitats of the Recherche Arch
SeamapAus_WA_ReefKIM_habitat_2013	Kimberley reef extent, geomorphology and habitat ma
SeamapAus_WA_Seagrass_Synthesis_2023_Hardy	Hardy Inlet Seagrass Survey - Cover (DWER-112)
SeamapAus_WA_Seagrass_Synthesis_2023_Leschenault	Leschenault Estuary Seagrass Survey - Cover (DWER-
SeamapAus_WA_Seagrass_Synthesis_2023_OysterHarbour	Oyster Harbour Seagrass Survey - Cover (DWER-109)
SeamapAus_WA_Seagrass_Synthesis_2023_PeelHarvey	Bindjareb Djilba (Peel Harvey) Seagrass Survey - Cove
SeamapAus_WA_Seagrass_Synthesis_2023_PrincessRoyal	Princess Royal Harbour Seagrass Survey - Cover (DWE
SeamapAus_WA_Seagrass_Synthesis_2023_Wilson	Wilson Inlet Seagrass Survey - Cover (DWER-114)
SeamapAus_WA_seagrass_synthesis_2013	WA Seagrass Synthesis 2013
TS-GOC_JCU_NESP-MaC-1-13_Seagrass-sites_1983-2022	Four Decades of Seagrass Spatial Data from Torres St
TS_JCU_Dungeness-Reef_2016_Seagrass_Intertidal-Site-Surveys	Dungeness Reef Seagrass Survey, Torres Strait,2016
TS_JCU_Dungeness-Reef_2017_Seagrass_Community-type	Dungeness Reef Seagrass Survey, Torres Strait, 2017 (
TS_JCU_Dungeness-Reef_2017_Seagrass_Subtidal-Site-Surveys	Dungeness Reef Seagrass Survey, Torres Strait, 2017 (
TS_JCU_Dungeness-Reef_2019_Seagrass_Community-type	Dungeness Reef Seagrass Survey, Torres Strait, Februa
TS_JCU_Dungeness-Reef_2019_Seagrass_Site-Survey	Dungeness Reef Seagrass Survey, Torres Strait, Februa

Dataset	Metadata Title
TS_JCU_Dungeness-Reef_2020_Seagrass_Community-type	Dungeness Reef Seagrass Survey, Torres Strait, Februa
TS_JCU_Dungeness-Reef_2020_Seagrass_Site-Survey	Dungeness Reef Seagrass Survey, Torres Strait, Februa
TS_JCU_Dungeness-Reef_2021_Seagrass_Community-type	Dungeness Reef Seagrass Survey, Torres Strait, Februa
TS_JCU_Dungeness-Reef_2021_Seagrass_Site-Survey	Dungeness Reef Seagrass Survey, Torres Strait, Februa
TS_JCU_Eastern-Torres-Strait_2020_Seagrass-community-type	Eastern Torres Strait Seagrass Survey, Torres Strait, 2
TS_JCU_Orman-Reefs_2017_Seagrass_Community-type	Seagrass baseline survey for Orman Reef, Torres Strait
TS_JCU_Orman-Reefs_2017_Seagrass_Site-surveys	Seagrass baseline survey for Orman Reef, Torres Strait
TS_JCU_Orman-Reefs_2018_Seagrass_Community-type	Orman Reef Seagrass Survey, Torres Strait, November
TS_JCU_Orman-Reefs_2018_Seagrass_Site-Surveys	Orman Reef Seagrass Survey, Torres Strait, November
TS_JCU_Orman-Reefs_2019_Seagrass_Community-type	Orman Reef Seagrass Survey, Torres Strait, September
TS_JCU_Orman-Reefs_2019_Seagrass_Site_Surveys	Orman Reef Seagrass Survey, Torres Strait, September
TS_JCU_Western-Torres-Strait_2020_Seagrass-community-type	Western Torres Strait Seagrass Survey, Torres Strait, S
TS_JCU_Western-Torres-Strait_2020_Seagrass_Site-surveys	Western Torres Strait Seagrass Survey, Torres Strait, S

2.1.1 Data Source

The underlying data is typically sourced from geo-servers maintained by:

1. IMAS
2. JCU
3. NESP

Extraction of the original datasets was conducted prior construction of the ETL pipeline.

There are plans to replace the static exports with dynamic exports from source.

2.1.2 Validation

Each dataset was analysed to understand its scheme and is validated against its own custom schema custom schema.

While there are similarities between each dataset, the column names, biological information and geometries have subtle differences that had to be accounted for per dataset.

2.1.2.1 Minimum Required Fields The minimum required fields are the standard biological datasets: when, where and what. | Column | Typical Type | Description | Typical Data Quality | | - | - | - | - | | Geometry | string | Usually a WKT encoding of the biological occurrence | Good | | Date | string | Usually an ISO8601 or similar encoded string | Poor | | Species | string or comma separated string | Either a list of scientific names or one hot encoded scientific names | Good |

This is a minimum example of a categorical encoded seagrass dataset schema:

```
"pyarrow_schema": {
    "fields": [

        // When
        {
            "name": "Date",
            "type": "date32[day]",
            "nullable": false,
            "date_encoding": "start_date"
```

```
        },

        // Where: The wkt geometry encoded in epsg:4326
        {
            "name": "geom",
            "type": "string",
            "nullable": false,
            "projection": 4326,
            "format": "wkt"
        },

        // What: Species present at the geometry
        {
            "name": "SM_HAB_CLS",
            "type": "string",
            "nullable": false,
            "aggregate_encoding": "presence"
        },
    ],
}
```

At runtime, the input data is validated per dataset against its custom schema, ensuring correct type interpretation for CSV data.

2.1.3 Extraction

Extraction is handled via the `common.file.S3Block` class in conjunction with the `common.factory.PyarrowTableFactory` class.

These serializable pydantic classes allow the extraction to live as JSON configuration files, saved in this project as `<dataset>.factory.json` files and ensure type valided arrow is generated from CSV inputs:

```
{
    "file_interface": {
        "block_name": "processing-bucket",
        "path": "seagrass/TS_JCU_Western-Torres-Strait_2020_Seagrass-community-type.csv"
    },
    "format": "csv",
    "pyarrow_schema": {
        "fields": [

            // When
            {
                "name": "Date",
                "type": "date32[day]",
                "nullable": false,
                "date_encoding": "start_date"
            },

            // Where: The wkt geometry encoded in epsg:4326
            {
```

```
                "name": "geom",
                "type": "string",
                "nullable": false,
                "projection": 4326,
                "format": "wkt"
            },

            // What: Species present at the geometry
            {
                "name": "SM_HAB_CLS",
                "type": "string",
                "nullable": false,
                "aggregate_encoding": "presence"
            },
        ],
    }
}
```

2.2 Transform

Because the underlying seagrass data sources have unique time, place and species encodings, the Seagrass Aggregate Data Product transformations are bespoke to each source.

2.2.1 Per Dataset Transformation Pattern

The typical transformation pattern procedure is as follows:

1. Extract geometry
2. Extract date
3. Extract species presence/absence

2.2.1.1 Extract Geometry Because the export data uses a WKT export query paramter, the geometry column almost always contains WKT compliant geometry information. This allows the latitude and longitude to be located.

Where the geometry column is unavailable, the latitude and longitude are used as a substitue.

2.2.1.2 Extract Date Date information is usually ISO8601 compliant, but in many cases is not.

The dates are manually audited and mapped where the date encoding is non ISO8601 compliant and follows inconsistent patterns, eg:

```
start_year = dates.map(
    {
        "16/12/2020": 2020,
        "Dec-19": 2019,
        "2015-2016": 2015,
        "10,17,19,27/5/22": 2022,
        "December-2010, November-2011": 2010,
        "Jun-21": 2021,
        "May/June-2012": 2012,
        "2-4/11/2020": 2020,
        "May-2020": 2020,
```

```
        "14-18/11/22": 2022,
        "October, 2017": 2017,
        "Aug-20": 2020,
        "9/02/2022,16/02/2022": 2022,
        "October-2010": 2010,
        "2005,2008": 2005,
        "27-28/10/2021": 2021,
        "21/07/2022": 2022,
        "22-24/09/2020": 2020,
        "Nov-19": 2019,
        ...
    }
)

start_month = dates.map(
    {
        "16/12/2020": 12,
        "Dec-19": 12,
        "2015-2016": 0,
        "10,17,19,27/5/22": 5,
        "December-2010, November-2011": 12,
        "Jun-21": 6,
        "May/June-2012": 5,
        "2-4/11/2020": 11,
        "May-2020": 5,
        "14-18/11/22": 11,
        "October, 2017": 10,
        "Aug-20": 8,
        "9/02/2022,16/02/2022": 2,
        "October-2010": 10,
        "2005,2008": 1,
        "27-28/10/2021": 10,
        "21/07/2022": 7,
        "22-24/09/2020": 9,
        "Nov-19": 11,
        ...
    }
)

start_day = dates.map(
    {
        "16/12/2020": 16,
        "Dec-19": 0,
        "2015-2016": 0,
        "10,17,19,27/5/22": 10,
        "December-2010, November-2011": 0,
        "Jun-21": 0,
        "May/June-2012": 0,
        "2-4/11/2020": 2,
```

```
        "May-2020": 0,
        "14-18/11/22": 14,
        "October, 2017": 0,
        "Aug-20": 0,
        "9/02/2022,16/02/2022": 9,
        "October-2010": 0,
        "2005,2008": 0,
        "27-28/10/2021": 27,
        "21/07/2022": 21,
        "22-24/09/2020": 22,
        "Nov-19": 0,
    }
)
```


2.2.1.3 Extract Species Presence/Absence

The typical species data transforms are to:

1. Re-encode delimited string columns
2. Re-encode one hot encoded species
3. Map species to the Codes for Australian Aquatic Biota controlled vocabulary (CAAB)
4. Disambiguation of species

2.2.1.3.1 Delimited String Species

The presence/absence of species are commonly encoded in a delimited string column.

In these cases, the column values were audited and mapped to the CAAB controlled vocabulary.

eg:

Species
"HO, HA, AO"
"HO, HA"
"AO"

2.2.1.3.2 One Hot Encoded Species

The presence/absence of species are also commonly one hot encoded.

In these cases, the column names were audited and mapped to the CAAB controlled vocabulary.

eg:

HO	HA	AO
TRUE	TRUE	TRUE
TRUE	TRUE	FALSE
FALSE	FALSE	TRUE

2.2.1.3.3 Species Mapping

The species were mapped to the CAAB controlled vocabulary. Details of this vocabulary are found in the enumeration.py.

2.2.1.3.4 Species Disambiguation

Another issue commonly encountered is the usage of `sp.` or `spp.`. In such instances, the scientific name is rolled up to the nearest parent.

2.2.1.4 Per Dataset Transformation Validation Once a dataset it is validated against an intermediary schema:

name	type	nullable
source	string	false
source_id	string	false
metadata_link	string	false
*When *	-	-
start_year	int16	false
start_month	int8	true
start_day	int8	true
end_year	int16	true
end_month	int8	true
end_day	int8	true
*Where *	-	-
geometry	string	false
What	-	-
seagrass_present	bool	false
thalassia_present	bool	true
thalassia_hemprichii_present	bool	true
hydrocharitaceae_present	bool	true
enhalus_present	bool	true
enhalus_acoroides_present	bool	true
halophila_present	bool	true
Species continues...	-	-

SeagrassSampleFlat	(Species Present)	(Species Biomass)	(Spe
- uuid: str - source: str - start_year: int - start_month: int - start_day: int - end_year: int - end_month: int - end_day: int - seagrass_present: bool - seagrass_biomass: float - seagrass_coverage: float - geometry: str (wkt) - centroid: float - lat: float - lon: float - caab_codes: str (rollup, | separated)	- hydrocharitaceae_present: bool - enhalus_spp__present: bool - enhalus_acoroides_present: bool - halophila_spp__present: bool - halophila_australis_present: bool - halophila_decipiens_present: bool - halophila_minor_present: bool - halophila_ovalis_present: bool - halophila_sp__present: bool - halophila_spinulosa_present: bool - halophila_tricostata_present: bool - cymodoceaceae_present: bool - amphibolis_spp__present: bool - amphibolis_antarctica_present: bool - amphibolis_griffithii_present: bool - cymodocea_spp__present: bool - cymodocea_angustata_present: bool - cymodocea_rotundata_present: bool - cymodocea_serrulata_present: bool - halodule_spp__present: bool - halodule_pinifolia_present: bool - halodule_uninervis_present: bool - syringodium_spp__present: bool - syringodium_isoetifolium_present: bool - thalassodendron_spp__present: bool - thalassodendron_ciliatum_present: bool - thalassodendron_pachyrhizum_present: bool - posidoniaceae_present: bool - posidonia_spp__present: bool - posidonia_angustifolia_present: bool - posidonia_australis_present: bool - posidonia_coriacea_present: bool - posidonia_denhartogii_present: bool - posidonia_kirkmanii_present: bool - posidonia_ostenfeldii_present: bool - posidonia_robertsoniae_present: bool - posidonia_sinuosa_present: bool - zosteraceae_present: bool - heterozostera_spp__present: bool - heterozostera_tasmanica_present: bool - zostera_spp__present: bool - zostera_capricorni_present: bool - zostera_mucronata_present: bool - zostera_muelleri_present: bool	- hydrocharitaceae_biomass: float - enhalus_spp__biomass: float - enhalus_acoroides_biomass: float - halophila_spp__biomass: float - halophila_australis_biomass: float - halophila_decipiens_biomass: float - halophila_minor_biomass: float - halophila_ovalis_biomass: float - halophila_sp__biomass: float - halophila_spinulosa_biomass: float - halophila_tricostata_biomass: float - cymodoceaceae_biomass: float - amphibolis_spp__biomass: float - amphibolis_antarctica_biomass: float - amphibolis_griffithii_biomass: float - cymodocea_spp__biomass: float - cymodocea_angustata_biomass: float - cymodocea_rotundata_biomass: float - cymodocea_serrulata_biomass: float - halodule_spp__biomass: float - halodule_pinifolia_biomass: float - halodule_uninervis_biomass: float - syringodium_spp__biomass: float - syringodium_isoetifolium_biomass: float - thalassodendron_spp__biomass: float - thalassodendron_ciliatum_biomass: float - thalassodendron_pachyrhizum_biomass: float - posidoniaceae_biomass: float - posidonia_spp__biomass: float - posidonia_angustifolia_biomass: float - posidonia_australis_biomass: float - posidonia_coriacea_biomass: float - posidonia_denhartogii_biomass: float - posidonia_kirkmanii_biomass: float - posidonia_ostenfeldii_biomass: float - posidonia_robertsoniae_biomass: float - posidonia_sinuosa_biomass: float - zosteraceae_biomass: float - heterozostera_spp__biomass: float - heterozostera_tasmanica_biomass: float - zostera_spp__biomass: float - zostera_capricorni_biomass: float - zostera_mucronata_biomass: float - zostera_muelleri_biomass: float	- hydrocharitaceae_co - enhalus_spp__covera - enhalus_acoroides_co - halophila_spp__cove - halophila_australis_ - halophila_decipiens_ - halophila_minor_cov - halophila_ovalis_co - halophila_sp__covera - halophila_spinulosa_ - halophila_tricostata - cymodoceaceae_covera - amphibolis_spp__cov - amphibolis_antarctic - amphibolis_griffith - cymodocea_spp__cove - cymodocea_angustata_ - cymodocea_rotundata_ - cymodocea_serrulata_ - halodule_spp__covera - halodule_pinifolia_c - halodule_uninervis_c - syringodium_spp__co - syringodium_isoetifo - thalassodendron_spp_ - thalassodendron_cil - thalassodendron_pach - posidoniaceae_covera - posidonia_spp__cove - posidonia_angustifo - posidonia_australis_ - posidonia_coriacea_c - posidonia_denhartog - posidonia_kirkmanii_ - posidonia_ostenfeld - posidonia_robertson - posidonia_sinuosa_c - zosteraceae_coverag - heterozostera_spp__c - heterozostera_tasma - zostera_spp__covera - zostera_capricorni_c - zostera_mucronata_c - zostera_muelleri_co

The full output schema is shown below:

2.2.2 Post Processing

Once extract transform is complete per dataset, the harmonised data is stacked vertically ready for post processing.

2.2.2.1 Wide to Long Pivot The intermediary transformed data is kept in wide format to simplify translation of species.

The final dataset however is pivoted to long format to improve usability.

2.2.2.2 Output Validation The long format output table is validated against the schema below:

name	type	nullable	is_list	metadata.description
source	string	false	false	The source dataset name
source_id	string	false	false	The source dataset row id
metadata_link	string	false	false	The source dataset metadata link
start_year	int16	false	false	The start year of the measurement
start_month	int8	true	false	The start month of the measurement
start_day	int8	true	false	The start day of the measurement
end_year	int16	true	false	The end year of the measurement (if applicable)
end_month	int8	true	false	The end month of the measurement (if applicable)
end_day	int8	true	false	The end day of the measurement (if applicable)
geometry	string	false	false	The WKT encoded geometry of the measurement
lat	float32	false	false	The latitude of the measurement, taken from the centroid of the
lon	float32	false	false	The longitude of the measurement, taken from the centroid of th
h3_index	string	false	false	A hexadecimal string representing a H3 polygon at resolution 5,
australian_marine_regions_tags	string	true	false	The australian marine regions tags found applicable to the meas

name	type	nullable	is_list	metadata.description
present	boolean	false	false	The presence of the species of the measurement
biomass	float32	true	false	The biomass of the species of the measurement (if applicable, m
scientific_name	string	false	false	The scientific name of the measurement, as per WoRMS catalog
aphia_id	int64	false	false	The scientific name aphia id of the measurement, as per WoRMS
rank	string	false	false	The taxon rank of the measurement, as per WoRMS catalogue
status	string	false	false	The taxon acceptance of the measurement, as per WoRMS catal

2.3 Load

Finally, the transformed Parquet is loaded to the **[XXX BUCKET PATH]** bucket as a Parquet file.

3 Flow Advantages

Reproducibility The original Squidle+ relational schema requires deep database expertise to join and flatten. This ETL procedure provides a reliable, replicable path to a flat Darwin Core standard.

Harmonisation the original seagrass data have unique schemes and data encodings which require per dataset effort to harmonise. The pre-harmonisation of this data into an anlysis read product allows scientists to focus on the analysis instead of data wrangling.

Space Efficiency static geo server exports are reduced from 2gb+ to ~**230MB** with modern compression and file formats (Parquet), optimising storage and access

Time Efficiency scheduled updates allow users to access up-to-date aggregated data without re-implementing the aggregation steps and validation

Cloud-native workflows reduce I/O overhead

Seabird Aggregated Data Product

Thomas Galindo - AODN

December 1, 2025 - Version 0.1.1

Contents

1 Background **2**
- 1.1 Overview . . . 2
  - 1.1.1 Basis of the Seabird Aggregated Data Product . . . 2

2 Methodology **3**
- 2.1 Extract . . . 3
  - 2.1.1 Data Source . . . 3
  - 2.1.2 Extraction . . . 3
  - 2.1.3 Darwin Core to Parquet Translation . . . 3
  - 2.1.4 Validation . . . 5
  - 2.1.5 Metadata . . . 6
- 2.2 Transform . . . 6
  - 2.2.1 Aggregation . . . 7
  - 2.2.2 Pre-Filtering . . . 7
  - 2.2.3 Extended Measurement or Fact Pivot . . . 8
  - 2.2.4 Extended Measurement or Fact Value Merging . . . 9
  - 2.2.5 Occurrence Table (Core Table) . . . 9
  - 2.2.6 Geospatial Enrichment . . . 9
- 2.3 Load . . . 9

3 Flow Advantages **10**

1 Background

There are many seabird datasets available in the OBIS Australia Node.

Like many biological datasets, Australian Seabird datasets are disparate.

1.1 Overview

1.1.1 Basis of the Seabird Aggregated Data Product

The goal of the aggregated data product is to flatten the OBIS Australia Node seabird datasets into a single aggregated data product ready for scientific analysis.

2 Methodology

The Seabird Aggregated Data Product (Data Product) relies on the standard Extract, Transform, and Load (ETL) methodology.

2.1 Extract

2.1.1 Data Source

The underlying datasets are sourced from the OBIS Australia Node.

Additionally, the relevant WoRMS and life stage vocabularies were extracted from theWoRMS marine species website and GBIF vocabularies website, respectively.

2.1.2 Extraction

Darwin core files are archive files containing:

1. An `eml.xml`, a metadata file
2. A `meta.xml`, an explanation file
3. One or more core tables; `event`, `occurrence`
4. Optional tables; `measurementOrFact`, `extendedMeasurementOrFact`

See more details here.

2.1.3 Darwin Core to Parquet Translation

The original darwin core data is validated against the following schemas before serialisation to intermediary parquet files:

2.1.3.1 Occurrence

name	type	nullable
id	string	true
modified	string	true
bibliographicCitation	string	true
references	string	true
institutionCode	string	true
collectionCode	string	true
basisOfRecord	string	true
occurrenceID	string	**false**
catalogNumber	string	true
recordNumber	string	true
recordedBy	string	true
individualCount	int64	true
organismQuantity	string	true
organismQuantityType	string	true
sex	string	true
lifeStage	string	true
occurrenceStatus	string	true
associatedOccurrences	string	true
associatedReferences	string	true

name	type	nullable
otherCatalogNumbers	string	true
occurrenceRemarks	string	true
organismID	string	true
preparations	string	true
materialSampleID	string	true
eventID	string	true
fieldNumber	string	true
eventDate	string	**false**
eventTime	string	true
year	string	true
month	string	true
verbatimIdentification	string	true
verbatimEventDate	string	true
samplingProtocol	string	true
sampleSizeValue	string	true
sampleSizeUnit	string	true
waterBody	string	true
country	string	true
stateProvince	string	true
county	string	true
locality	string	true
minimumDepthInMeters	int64	true
maximumDepthInMeters	int64	true
decimalLatitude	float64	true
decimalLongitude	float64	true
coordinateUncertaintyInMeters	int64	true
coordinatePrecision	float64	true
footprintWKT	string	true
identificationQualifier	string	true
typeStatus	string	true
identifiedBy	string	true
dateIdentified	string	true
identificationReferences	string	true
identificationRemarks	string	true
scientificNameID	string	**false**
taxonConceptID	string	true
scientificName	string	**false**
kingdom	string	true
phylum	string	true
class	string	true
family	string	true
genus	string	true
subgenus	string	true
specificEpithet	string	true
infraspecificEpithet	string	true
taxonRank	string	true
scientificNameAuthorship	string	true

name	type	nullable
vernacularName	string	true

Note that rows missing `decimalLatitude` or `decimalLongitude` are dropped

2.1.3.2 ExtendedMeasurementOrFact

name	type	nullable
eventID	string	true
occurrenceID	string	true
habitat	string	true
sampleSizeUnit	string	true
sampleSizeValue	string	true
samplingEffort	string	true
samplingProtocol	string	true
behavior	string	true
individualCount	string	true
lifeStage	string	true
organismQuantity	string	true
organismQuantityType	string	true
sex	string	true
type	string	true
measurementAccuracy	string	true
measurementDeterminedBy	string	true
measurementDeterminedDate	string	true
measurementID	string	true
measurementMethod	string	true
measurementRemarks	string	true
measurementType	string	true
measurementTypeID	string	true
measurementUnit	string	true
measurementUnitID	string	true
measurementValue	string	true
measurementValueID	string	true

Note that rows missing both eventID and occurrenceID are dropped

2.1.4 Validation

Being sourced from an OBIS Node, the data is of high quality and subject to stringent dataset structures.

This goes a long way to simplifying the extraction procedure.

That being said, we conduct the following validation to ensure data quality.

2.1.4.1 Naive Timestamp Check The `eventDate` recommended best practice is *"... to use a date that conforms to ISO 8601-1:2019."*

This means the string formated dates need to be pattern matched to ensure downstream interpretation is correct:

`^\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}Z$` eg `2012-11-08T12:31:00Z`

2.1.4.2 Scientific Name and ID Match The `verbatimIdentification`, `scientificName` and `scientificNameID` darwin core columns are used in conjunction to ensure the identified species is interoperable between datasets.

We ensure the `scientificName` is up to date with the `scientificNameId` data found in the World Register of Marine Speices (WoRMS).

2.1.5 Metadata

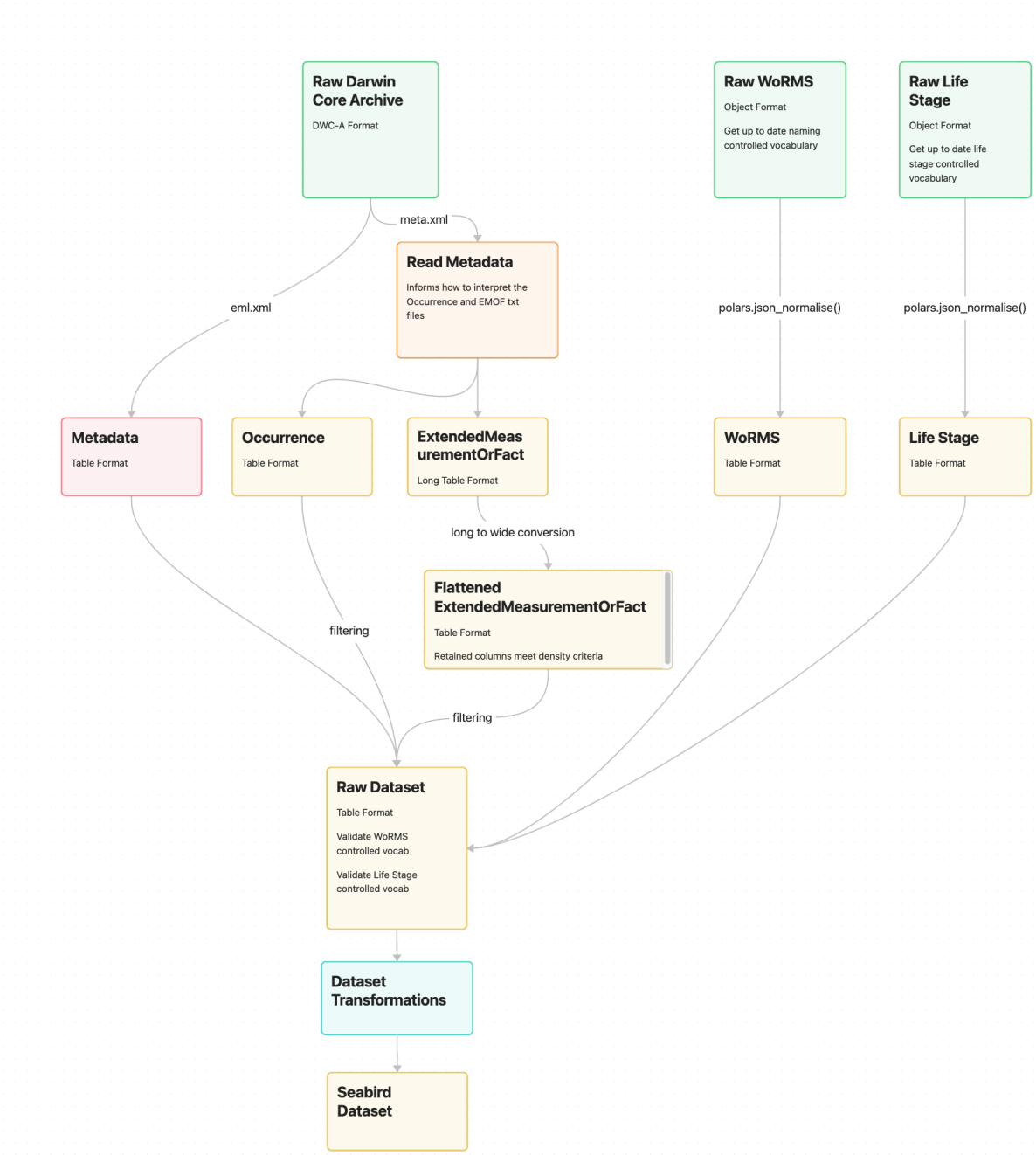
Darwin Core archives must have a `eml.xml` file, which contains traditional dataset metadata.

We pull the following from the `eml.xml`:

name	description
dataset_name	the dataset long name
start	the dataset start date
end	the dataset end date
geographic_bounds	the geographic bounding coordinates
contacts	the dataset conctacts
abstract	the dataset abstract
pub_date	the dataset publication date
version	the dataset publication version

2.2 Transform

The transform step synthesises the darwin core, WoRMS and GBIF Life Stages tables:



2.2.1 Aggregation

The individual Darwin Core archive parquet are aggregated vertically:

2.2.2 Pre-Filtering

Pre filtering steps are carried out to ensure valid transformations.

2.2.2.1 Drop incomplete extended measurement or fact

Valid extended measurement or fact tables must point to either an occurrence or event row. Rows pointing to neither are dropped.

2.2.2.2 Drop all null columns

Columns completely devoid of information are dropped.

2.2.2.3 Drop missing latitude and longitude

Rows missing `decimalLatitude` or `decimalLongitude` are dropped.

2.2.2.4 Drop missing quantity

Rows missing `organismQuantity` are dropped.

2.2.3 Extended Measurement or Fact Pivot

The extended measurement or fact table is a long table with a m:1 relation to the event and occurrence tables.

The extended measurement or fact (EMOF) table therefore must be pivoted to a wide format in order to produce a single parquet output. This involves a trade off between table width and data retention. The measurements retained were arbitrarily cut off at 10,000, which is where a value is present for ~10% of the events/occurrences:

measurementType	count
Sea state	27894
Sea surface temperature	26444
Depth (m)	25186
Salinity (psu)	21772
Wind direction (deg)	20768
Air Pressure (hPa)	18892
Wind Speed (knt)	18262
Wind Force (Beaufort scale)	17292
Cloud cover	17229
Atmospheric temperature (deg C)	16944
Cloud cover (oktas)	13744
CUTOFF	-----
Visibility	9363
Bird behaviour	3397
ship speed	2557
Wind direction	93
Air temperature	93
Depth	88
ship activity	75
Cetacean behaviour	62
precipitation	42
Culman Height	26
Animal Mass	26
Tag Type	26
Culman Length	26
LifeStage	13
Seal behaviour	8
sun glare	4

2.2.4 Extended Measurement or Fact Value Merging

Many measurements were found. Careful analysis of values allowed merging of the following `measurementType` was valid with minimal value transform:

2.2.4.1 Wind Speed `Wind Speed (knt)`

`Wind Force (Beaufort scale)`

2.2.4.2 Wind Direction `Wind direction (deg)`

`Wind direction`

2.2.4.3 Air Temperature `Air temperature`

`Atmospheric temperature (deg C)`

2.2.4.4 Depth `Depth (m)`

`Depth`

As the validity of any value merges is extremely sensitive to the value data itself, the ETL process reports any mutation of the underlying measurement data to AODN for further investigation.

2.2.5 Occurrence Table (Core Table)

The Seabird datasets were found to only contain Occurrence Core tables.

1. Update empty strings to null
2. Join EMOF table
3. Join metadata table
4. Construct and validate all source urls using download link and version
5. Life stage values are validated against the GBIF Life Stage Vocabulary
6. Timestamps are validated and updated to dates
7. WoRMS taxon hierarchy are mapped via the `scientificNameID`
8. Update the output table metadata with the ETL timestamp

2.2.6 Geospatial Enrichment

The output dataset is enriched with the following geospatial columns:

- `h3Index` - A hexadecimal string representing a H3 polygon at resolution 4, derived from the `decimalLatitude` and `decimalLongitude`
- `australianMarineRegionsTags` - A | separated tag column featuring common Australian marine regions

2.3 Load

Finally, the transformed Parquet is loaded to the **[XXX BUCKET PATH]** bucket as a Parquet file.

3 Flow Advantages

Reproducibility the original seabird darwin core files require relational database understanding and specialised software to load and join tables. The ETL procedure makes this process reliable and replicatable

Space Efficiency raw original data is reduced to ~**5MB** with modern compression and file formats (Parquet), optimising storage and access

Time Efficiency scheduled updates allow users to access up-to-date aggregated data without re-implementing the aggregation steps and validation

Cloud-native workflows reduce I/O overhead

Kelp Squidle+ Annotations Data Product

Thomas Galindo - AODN

January 6, 2026 - Version 0.1.1

Contents

1 Background **2**
- 1.1 Squidle+ . . . 2
- 1.2 Darwin Core Mapping . . . 2
- 1.3 Basis of the Kelp Squidle Annotations Data Product . . . 2

2 Methodology **4**
- 2.1 Extract . . . 4
  - 2.1.1 Data Source . . . 4
  - 2.1.2 Filtering and Label Schemes . . . 4
  - 2.1.3 Extraction . . . 4
- 2.2 Transform . . . 4
  - 2.2.1 Summary . . . 4
  - 2.2.2 Cleaning . . . 4
  - 2.2.3 DWC Translation . . . 5
  - 2.2.4 Temporal-Spatial . . . 5
  - 2.2.5 Validation . . . 5

3 Load **6**

4 Flow Advantages **7**

5 Appendix **8**
- 5.1 Kelp Label Allow List . . . 8

1 Background

The Kelp Squidle+ Annotations Data Product extracts kelp annotations from Squidle and translates them to Darwin Core.

1.1 Squidle+

Squidle+ is a centralised marine image data management, discovery, and annotation platform with vocabulary translation.

Scientists and the general public use Squidle+ to host and annotate photo based marine surveys.

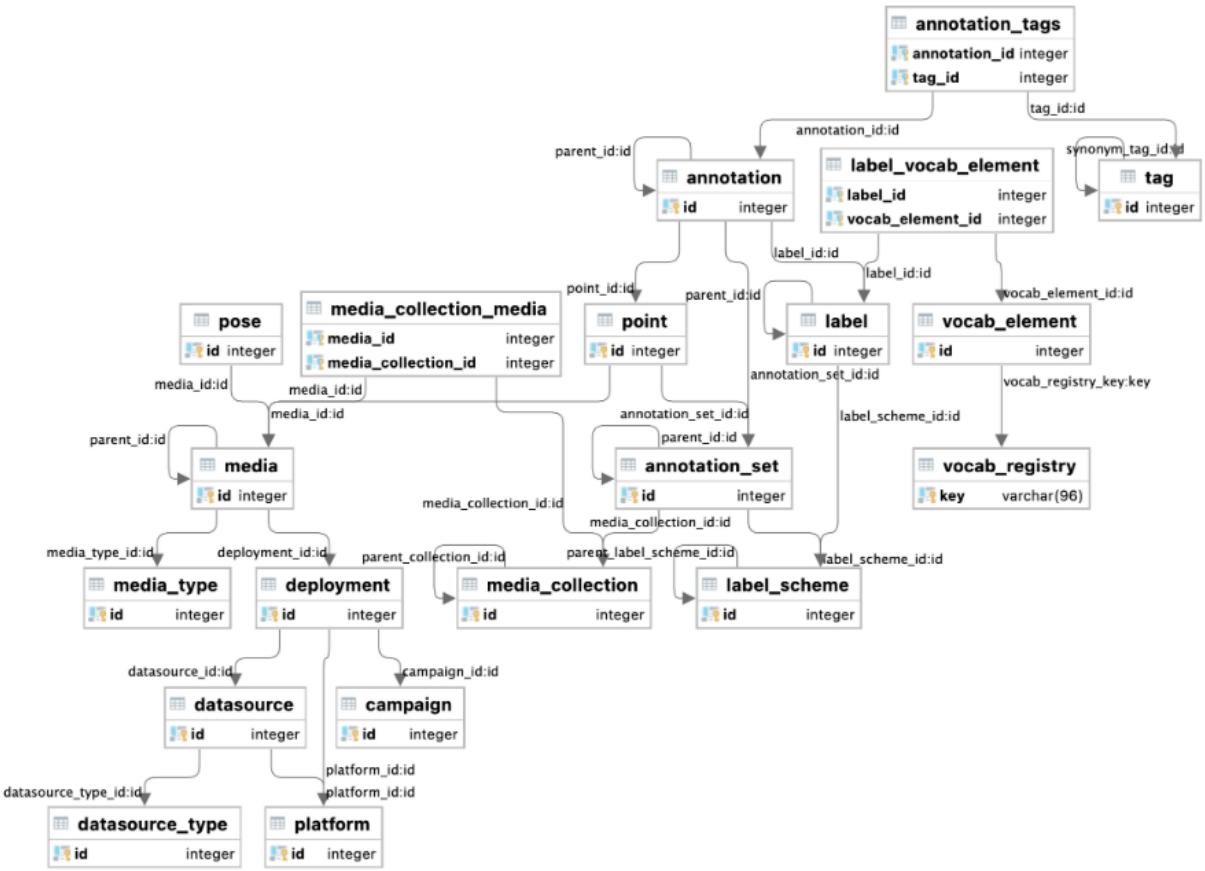
1.2 Darwin Core Mapping

The Kelp Annotations are mapped to the Darwin Core scheme to increase interoperability of Squidle+ data with existing Darwin Core tooling and systems (eg OBIS).

1.3 Basis of the Kelp Squidle Annotations Data Product

Bulk exportation of Squidle+ data requires experience working with REST APIs (see API reference here) as well as a good understanding of relational databases.

The complexity of the underlying Squidle+ relational database further complicates the extraction and aggregation of Squidle+ data into aggregated data products suitable for scientific analysis, as exemplified by the simplified Entity-Relationship Model (ERM):



The goal of the aggregated data product is to flatten the Kelp survey photo annotations into a single aggregated data product that is ready for scientific analysis and interoperable with key marine biological data systems.

2 Methodology

The Kelp Squidle Annotations Data Product (Data Product) relies on the standard Extract, Transform, and Load (ETL) methodology.

2.1 Extract

2.1.1 Data Source

The underlying data are sourced from public Squidle+ annotations and label schemes.

Additionally, the relevant WoRMS vocabulary was extracted from the WoRMS marine species website.

2.1.2 Filtering and Label Schemes

Squidle+ annotations are labeled with user specified controlled vocabularies.

The user controlled aspect of these vocabularies makes identifying kelp by text difficult as the label text evolves over time as per user modifications.

To reduce potential inclusion of non-kelp labels, the following ID list was generated to filter Squidle+ annotations by the Kelp Label Allow List

2.1.3 Extraction

Annotations and related translation metadata are exported from Squidle+ using the Squidle+ API.

This is a three step process:

1. Filter to kelp containing public annotation sets using the Kelp Label Allow List
2. Export all kelp containing annotation sets
3. Export all vocab elements for the kelp labels

The raw annotations, related WoRMS taxonomic controlled vocabulary elements and a mapping between Squidle+ labels and WoRMS labels is passed on to the transform stage.

2.2 Transform

2.2.1 Summary

Phase	Task	Description
Setup	Workspace Prep	Ensures the temporary directory `transform_dir` exists for ephemeral data.
Cleaning	Data Refinement	Filters for kelp-specific records and resolves overlapping or conflicting media
DWC Translation	Standardisation	Maps the local schema to the global **Darwin Core** standard using WoRMS t
Temporal-Spatial	Time Pruning & Geofencing	Filters out all records occurring before **September 28, 2007** and constrains
Validation	Quality Control	Validates the final Polars DataFrame and exports it as a **PyArrow Table**.

2.2.2 Cleaning

The cleaning phase prunes annotations to kelp and resolves label conflicts:

2.2.2.1 Kelp Pruning Filters the dataset to retain only records classified as **Kelp**, removing unrelated biological observations.

2.2.2.2 Media Conflict Resolution Squidle+ allows overlapping annotation layers on a single image. The pipeline identifies media with multiple annotation sets and **rejects** any media where labels are contradictory, ensuring high data integrity.

2.2.3 DWC Translation

This step converts raw annotations into a standardized occurrence format through several key operations:

2.2.3.1 Taxonomic Alignment The local `label.id` is joined with the **World Register of Marine Species (WoRMS)** vocabulary. This enriches the data with authoritative taxonomic hierarchies (Kingdom through Genus) and provides a globally unique `scientificNameID` (LSID).

2.2.3.2 Identifier Generation A unique `occurrenceID` is constructed by concatenating the internal annotation ID with the deployment ID, ensuring traceability back to the original survey.

2.2.3.3 Temporal Normalization Diverse time formats are parsed into a unified `eventDate` (GMT timestamp).

2.2.3.4 Spatial Normalization Raw pose data (latitude/longitude) is renamed to DwC standard `decimalLatitude` and `decimalLongitude`.

2.2.3.5 Observation Metadata Sets the `basisOfRecord` to `MachineObservation` and hardcodes the `occurrenceStatus` to `present`, as the input consists of photographs and positive labels, respectively.

2.2.3.6 Media Linkage Constructs a direct URL for `associatedMedia` by formatting the Squidle+ media ID into an API-accessible iframe link for visual verification.

2.2.4 Temporal-Spatial

Filters out all records occurring before **September 28, 2007** and constrains data to the Australian region.

2.2.5 Validation

Validates the final Polars DataFrame conforms to the expected output schema and confirms null constraints are valid.

3 Load

Finally, the transformed Parquet is loaded to the **[XXX BUCKET PATH]** bucket as a Parquet file.

4 Flow Advantages

Reproducibility the original seabird darwin core files require relational database understanding and specialised software to load and join tables. The ETL procedure makes this process reliable and replicable.

Space Efficiency raw original data is reduced to ~**2MB** with modern compression and file formats (Parquet), optimising storage and access

Time Efficiency scheduled updates allow users to access up-to-date aggregated data without re-implementing the aggregation steps and validation

Cloud-native workflows reduce I/O overhead

5 Appendix

5.1 Kelp Label Allow List


```
[
    {
        "id": 7327,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Phyllospora comosa"
    },
    {
        "id": 29052,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Seirococcus axillaris"
    },
    {
        "id": 14983,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Cystophora spp"
    },
    {
        "id": 29051,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Perithalia caudata"
    },
    {
        "id": 12678,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Durvillaea potatorum"
    },
    {
        "id": 7325,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Brown Other Canopy Forming spp"
    },
    {
        "id": 7326,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Ecklonia radiata"
    },
    {
        "id": 28751,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Scaberia agardhii"
    },
    {
        "id": 583,
        "catalogue": "Australian_Morphospecies_Catalogue",
        "name": "Brown"
    },
```

```
{
    "id": 13379,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Carpoglossum confluens"
},
{
    "id": 28462,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Xiphophora gladiata"
},
{
    "id": 13407,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Scytothalia dorycarpa"
},
{
    "id": 12680,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Lessonia corrugata"
},
{
    "id": 28465,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Macrocystis pyrifera"
},
{
    "id": 28883,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Carpophyllum spp"
},
{
    "id": 29037,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Platythalia angustifolia"
},
{
    "id": 29038,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Platythalia quercifolia"
},
{
    "id": 29036,
    "catalogue": "Australian_Morphospecies_Catalogue",
    "name": "Caulocystis spp"
},
{
    "id": 28945,
    "catalogue": "Benthic_habitat_and_relief_schema_for_BRUV__BOSS_and_DOV_Imagery",
    "name": "Scytothalia dorycarpa"
```

```
},
{
    "id": 28943,
    "catalogue": "Benthic_habitat_and_relief_schema_for_BRUV__BOSS_and_DOV_Imagery",
    "name": "Brown"
},
{
    "id": 28946,
    "catalogue": "Benthic_habitat_and_relief_schema_for_BRUV__BOSS_and_DOV_Imagery",
    "name": "Phyllospora comosa"
},
{
    "id": 28944,
    "catalogue": "Benthic_habitat_and_relief_schema_for_BRUV__BOSS_and_DOV_Imagery",
    "name": "Ecklonia radiata"
},
{
    "id": 583,
    "catalogue": "Catami_1_4",
    "name": "Brown"
},
{
    "id": 10905,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Desmarestia and Himantothallus"
},
{
    "id": 10901,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Durvillaea"
},
{
    "id": 10906,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Large brown laminarian kelps"
},
{
    "id": 28227,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Lessonia corrugata"
},
{
    "id": 10903,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Phyllospora"
},
{
    "id": 14948,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
```

```
    "name": "Sargassum"
},
{
    "id": 13553,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Carpoglossum"
},
{
    "id": 10904,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Macrocystis"
},
{
    "id": 14950,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Xiphophora"
},
{
    "id": 14947,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Cystophora"
},
{
    "id": 28201,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Perithalia caudata"
},
{
    "id": 10902,
    "catalogue": "Hunter_-_Extended_RLS_Scheme",
    "name": "Ecklonia radiata"
},
{
    "id": 10904,
    "catalogue": "RLS_Australian_Coral_Species_List",
    "name": "Macrocystis"
},
{
    "id": 10903,
    "catalogue": "RLS_Australian_Coral_Species_List",
    "name": "Phyllospora"
},
{
    "id": 10905,
    "catalogue": "RLS_Australian_Coral_Species_List",
    "name": "Desmarestia and Himantothallus"
},
{
    "id": 10906,
```

```
        "catalogue": "RLS_Australian_Coral_Species_List",
        "name": "Large brown laminarian kelps"
    },
    {
        "id": 10902,
        "catalogue": "RLS_Australian_Coral_Species_List",
        "name": "Ecklonia radiata"
    },
    {
        "id": 10901,
        "catalogue": "RLS_Australian_Coral_Species_List",
        "name": "Durvillaea"
    },
    {
        "id": 10904,
        "catalogue": "RLS_Catalogue",
        "name": "Macrocystis"
    },
    {
        "id": 10903,
        "catalogue": "RLS_Catalogue",
        "name": "Phyllospora"
    },
    {
        "id": 10901,
        "catalogue": "RLS_Catalogue",
        "name": "Durvillaea"
    },
    {
        "id": 10906,
        "catalogue": "RLS_Catalogue",
        "name": "Large brown laminarian kelps"
    },
    {
        "id": 10902,
        "catalogue": "RLS_Catalogue",
        "name": "Ecklonia radiata"
    },
    {
        "id": 10905,
        "catalogue": "RLS_Catalogue",
        "name": "Desmarestia and Himantothallus"
    },
    {
        "id": 285,
        "catalogue": "SQUIDLE_1_0",
        "name": "Ecklonia radiata"
    },
    {
```

```
        "id": 583,
        "catalogue": "SQUIDLE_1_0",
        "name": "Brown"
    }
]
```

NRMN Photo Quadrat Aggregated Data Product Technical Document

Thomas Galindo - AODN

January 8, 2025 - Version 1.0.0

Contents

1 Background **2**
- 1.1 Overview . . . 2
  - 1.1.1 National Reef Monitoring Network (an IMOS sub-facility) . . . 2
  - 1.1.2 Reef Life Survey Photo Quadrats . . . 2
- 1.2 Squidle+ . . . 2
  - 1.2.1 Basis of the NRMN Photo Quadrat Aggregated Data Product . . . 2
  - 1.2.2 Normalised Annotations . . . 3

2 Methodology **4**
- 2.1 Extract . . . 4
  - 2.1.1 Data Sources . . . 4
  - 2.1.2 Static Export . . . 4
  - 2.1.3 Squidle+ Extraction . . . 4
  - 2.1.4 Validation . . . 6
- 2.2 Transform . . . 6
  - 2.2.1 Static Export Transformation . . . 6
  - 2.2.2 Squidle+ Export Transformation . . . 7
  - 2.2.3 Transform Validation . . . 9
  - 2.2.4 Geospatial Enrichment . . . 9
- 2.3 Load . . . 9
  - 2.3.1 Private Survey Table . . . 9
  - 2.3.2 Public Survey Table . . . 10

3 NRMN Flow Advantages **11**

1 Background

1.1 Overview

1.1.1 National Reef Monitoring Network (an IMOS sub-facility)

The National Reef Monitoring Network (NRMN) is an IMOS sub-facility that brings together shallow reef surveys conducted around Australia (and globally) into a centralised database.

The IMOS National Reef Monitoring Network sub-facility collates, cleans, stores, and makes this data rapidly available from contributors, including Reef Life Survey.

1.1.2 Reef Life Survey Photo Quadrats

A common Reef Life Survey method is Method 13, also known as the photo quadrats method.

In this method, the target groups are:

- all algae
- sessile invertebrates (including corals)
- substratum categories (e.g., sand)

Digital photo-quadrats are taken directly downwards from approximately 50 cm above the seabed in a line. A typical photo-quadrat survey is comprised of 20 images taken in a line.

The percentage cover of different macroalgal, coral, sponge, and other attached invertebrate species in photo-quadrats is later assessed using Squidle+ annotation, and is therefore not part of the initial field survey itself.

1.2 Squidle+

Squidle+ is a centralised marine image data management, discovery, and annotation platform with vocabulary translation.

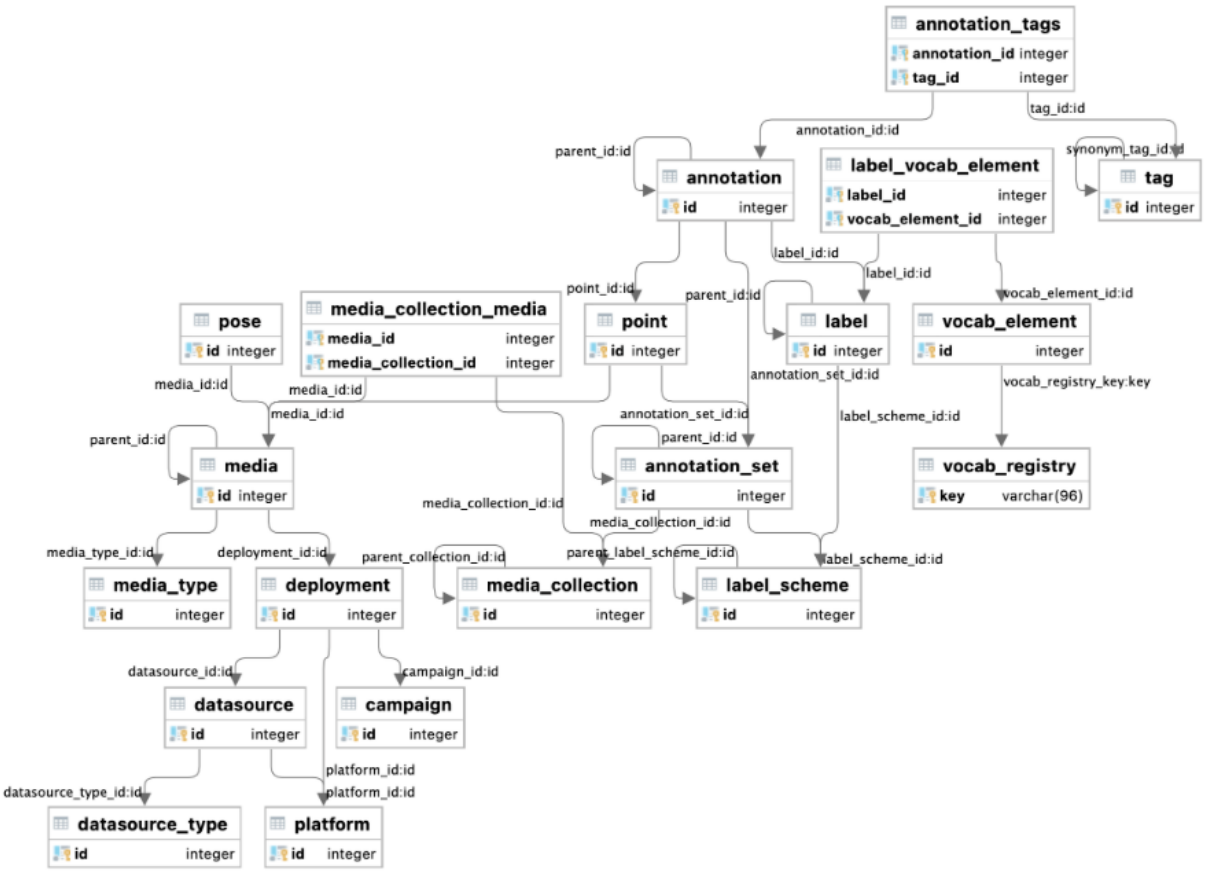
Reef Life Survey uses Squidle+ to:

- Store photo-quadrat surveys image data
- Annotate (and store annotations of) photo-quadrats

1.2.1 Basis of the NRMN Photo Quadrat Aggregated Data Product

Bulk exportation of Squidle+ data requires experience working with REST APIs (see API reference here) as well as a good understanding of relational databases.

The complexity of the underlying Squidle+ relational database further complicates the extraction and aggregation of Squidle+ data into aggregated data products suitable for scientific analysis, as exemplified by the simplified Entity-Relationship Model (ERM):



The goal of the aggregated data product is to flatten the NRMN Photo Quadrat annotations into a single aggregated data product ready for scientific analysis.

1.2.2 Normalised Annotations

Annotated Photo Quadrat surveys use different labelling schemes.

While these schemes are similar, they have subtle differences in their depth and hierarchical structure, which complicates survey analysis.

Squidle+ facilitates translation of label schemes, allowing surveys annotated with different label schemes to be easily compared.

The labels in this dataset are translated to:

1. Catami 1.4
2. RLS Catalogue.

Both label schemes are available under the Public Schemas section at Squidle+.

2 Methodology

The NRMN Photo Quadrat Aggregated Data Product (Data Product) relies on the standard Extract, Transform, and Load (ETL) methodology.

2.1 Extract

2.1.1 Data Sources

There are two data sources for the NRMN Photo Quadrat Aggregated Data Product:

1. A static export of survey aggregated data from the legacy database
2. A dynamic export of raw annotations (non-survey aggregated data) from Squidle+

The Data Product harmonises the data from these sources.

2.1.2 Static Export

The static export is securely cached in S3 buckets available for in AODN Non-Production and Production Amazon tenants.

2.1.3 Squidle+ Extraction

Extraction from Squidle+ is performed using the Squidle+ API.

2.1.3.1 Data Access Control While Squidle+ is a public, free tool, users may selectively share their data with other users rather than allow it to be fully public.

The NRMN data in Squidle+ is mostly private. A curated subset of NRMN's quality-controlled survey data is shared with the `aodn@nrmn.org.au` Squidle+ account.

2.1.3.2 Annotations, Annotation Sets and Labels There are three separate export methods required to extract NRMN Photo Quadrat Annotations from Squidle+:

1. Export all `Annotation Sets` that are in the `RLS` platform.
2. Export all `Annotation` that are in those annotation sets.
3. Export all `Label` and their translations to the desired label schemes.

2.1.3.3 Annotation Set Export Annotation Sets are exported in JSON format and normalised to a table structure. An excerpt of a typical JSON export for an `Annotation Set` is shown below, illustrating the nested structure that requires flattening:

```
{
    "metadata": {
        "id": 800,
        "data": {
            "type": "quincunx",
            "params": {"grid": "show"},
            "allow_add": true,
            "allow_wholeframe": false,
            "labels_per_point": 1,
            "labels_per_frame": 1
        },
```

```
        "name": "***",
        "description": "RLS HC species",
        "created_at": "***",
        "annotation_count": 50,
        "usergroup_count": 1,
        "media_count": 43,
        "point_count": 210,
        "unannotated_point_count": 160,
        "label_scheme": {"id": 11, "name": "RLS Australian Coral Species List"},
        "user": {"id": 166, "username": "***", "first_name": "***", "last_name": "***"},
        "media_collection": {"id": 872, "name": "***", "current_user_can_view": true}
    },
    "objects": [
        {
            "point": {
                "media": {
                    "deployment": {
                        "id": 7108,
                        "campaign": {"id": 376, "name": "RLS_Whitsundays", "key": "RLS_Whitsundays"},
                        "name": "GBR",
                        "key": "GBR"
                    },
                    "id": 5426153,
                    "key": "image_0.jpg"
                },
                "pose": {
                    "id": 5426153,
                    "lon": 149.13078510189496,
                    "lat": -19.8099775,
                    "dep": 5.0,
                    "timestamp": "***",
                    "data": {},
                    "timestamp_local": "***"
                },
                "is_targeted": false,
                "y": 0.8333333333333334,
                "x": 0.16666666666666666,
                "has_xy": true,
                "data": {},
                "id": 7770846
            },
            "user": {"affiliations_names": "***", "username": "***", "full_name": "***"},
            "updated_at": "***",
            "tag_names": "",
            "likelihood": 1.0,
            "needs_review": false,
            "id": 7878480
        },
        ...
```

```
    ]
}
```

These exports are aggregated into label translated survey data in the transform step.

2.1.4 Validation

Squidle+ JSON data is validated using Pydantic models to ensure data integrity.

2.2 Transform

Transform is required for both data sources.

2.2.1 Static Export Transformation

The static export is already aggregated to the survey and requires little transformation:

2.2.1.1 Add Scalars The following scalar fields are added to each record:

```
{
    "database": "NRMN",
    "method": "full biodiversity",
    "program": "RLS",
    "sq_annotation_set_id": null,
    "sq_annotation_set_owner": null
}
```

2.2.1.2 Construct Dataset ID A unique Dataset ID is constructed from the `survey_id` and `resolution`.

2.2.1.3 Update the RLS Lineage The static export does not have the full RLS lineage. This is added using the Squidle+ Label Scheme translation exports, for example: `Crustose coralline algae` is translated to `Biota > Macroalgae > Crustose coralline algae`.

2.2.1.4 Translate Bleached and Dead Coral Columns to Tags The indicators for bleached and dead coral are columns. The parquet format allows these columns to be translated to a single `tags` column in line with the Squidle+ exports:

Bleached	Dead	->	Tags
True	False		["bleached"]
False	True		["dead"]
False	False		[]

2.2.1.5 Column Renaming

Old Column Name	New Column Name
survey_date	date
resolution	label_scheme
original_category	original_label
rls_category	rls_label

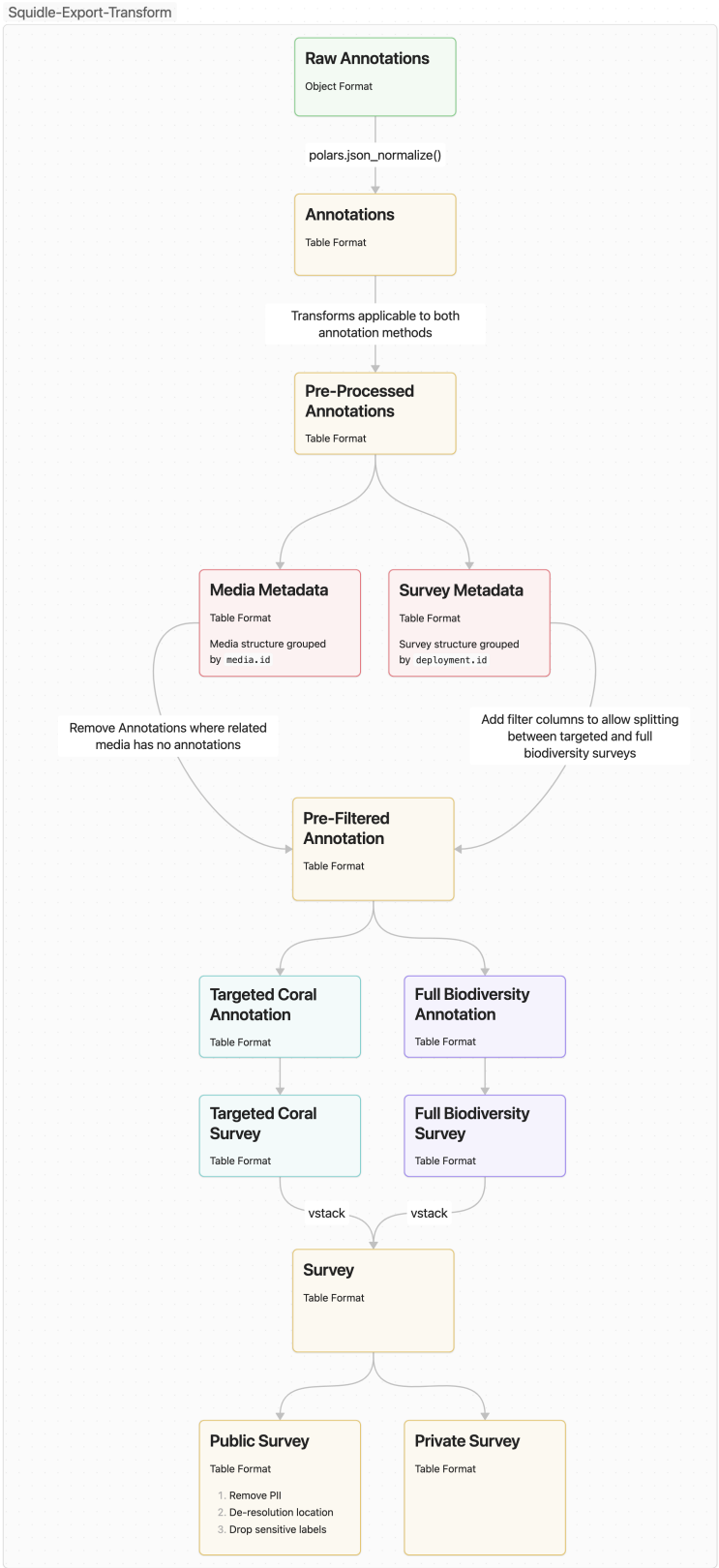
Old Column Name	New Column Name
catami	catami_label
num_points	label_count
total_points	dataset_total_label_count

2.2.2 Squidle+ Export Transformation

Squidle+ data comes as raw annotations records as opposed to aggregated survey coverage information. It requires much more transformation than the static export transform.

Additionally, there are two kinds of annotation method: (1) Full biodiversity and (2) Targeted Coral.

These methods require different processes to correctly aggregate raw annotations into surveys.



2.2.2.1 Diagram

2.2.3 Transform Validation

The transform is validated against the schema:

name	type	nullable
dataset_id	string	false
database	string	false
program	string	false
country	string	false
area	string	false
location	string	false
site_code	string	false
site_name	string	false
latitude	float64	false
longitude	float64	false
date	date32[day]	false
hour	string	true
depth	float64	false
visibility	float64	true
label_scheme	string	false
original_label	string	false
rls_label	string	false
rls_lineage	string	true
catami_label	string	false
catami_lineage	string	false
tags	string	true
label_count	float64	true
dataset_total_label_count	float64	true
percent_cover	float64	false
sq_annotation_set_id	float64	true
sq_annotation_set_owner	string	true
method	string	true
h3_index	string	false
australian_marine_regions_tags	string	true

2.2.4 Geospatial Enrichment

The occurrence output dataset is enriched with the following geospatial columns:

- `h3_index` - A hexadecimal string representing a H3 polygon at resolution 5, derived from the `latitude` and `longitude`
- `australian_marine_regions_tags` - A | separated tag column featuring common Australian marine regions

2.3 Load

Finally, the transformed Parquet is loaded in its private and public formats to AWS S3.

2.3.1 Private Survey Table

The private survey Table is stored for internal use by NRMN as a Parquet file.

2.3.2 Public Survey Table

The public survey table is stored for external use as:

1. A Parquet file
2. A CSV file

3 NRMN Flow Advantages

Reproducibility the original NRMN Squidle+ export scripts were not version controlled or hosted in cloud ETL infrastructure, making maintenance difficult.

Space Efficiency raw static file and Squidle+ exports were reduced to ~**1MB** with modern compression and file formats (Parquet), optimising storage and access.

Time Efficiency scheduled updates allow users to access up-to-date data in just a fraction of the time (Squidle+ exports take at least 2 hours to complete).

Cloud-native workflows reduce I/O overhead.

Contents

1 Dugong Optimised Data Product **2**

1.1 1. Introduction . . . 2

1.2 2. Source Data Characteristics . . . 2

1.2.1 2.1 Delivery structure . . . 2

1.2.2 2.2 Common inconsistencies accounted for . . . 2

1.3 3. Output Data Products . . . 2

1.3.1 3.1 Validated dataset (Master Parquet) . . . 2

1.3.2 3.2 Rejected dataset (DLQ Parquet) . . . 3

1.4 4. Processing Method (ETL) . . . 3

1.4.1 4.1 Extract . . . 3

1.4.2 4.2 Transform . . . 3

1.5 5. Validation Rules (Waterfall) . . . 4

1.6 6. Load Strategy . . . 4

1.6.1 6.1 Validated Master Load . . . 4

1.6.2 6.2 Rejected Output Load . . . 4

1.7 7. Output Schema (Summary) . . . 4

1.8 8. Configuration . . . 5

1.8.1 8.1 Source Configuration . . . 5

1.8.2 8.2 Schema Configuration . . . 5

1.9 9. Dependencies . . . 5

1.10 10. Running the Pipeline . . . 5

1.10.1 Prerequisites . . . 5

1.10.2 Execution . . . 6

1.10.3 Cloud Deployment . . . 6

1 Dugong Optimised Data Product

1.1 1. Introduction

The Dugong data pipeline consolidates historical and ongoing survey data from multiple distinct sources into a single, unified, analytics-ready Parquet dataset. The raw deliveries are provided as a mix of historical Data Management System (DMS) records and James Cook University (JCU) geospatial shapefile archives.

This work converts those diverse deliveries into a consistent dataset aligned with the **Darwin Core (DwC)** standard, featuring:

- cleaned and standardised coordinates
- unrolled biological counts (exploding aggregated group counts into discrete adult and calf occurrence records)
- derived `eventID` and `occurrenceID` keys
- an H3 spatial index for fast spatial grouping and joins
- Australian Marine Region spatial tags
- a separate rejected-records output with explicit rejection reasons

---

1.2 2. Source Data Characteristics

1.2.1 2.1 Delivery structure

The pipeline ingests data from two primary formats:

- **DMS Data**: Provided as a flat `.parquet` file containing historically aggregated sightings.
- **JCU Data**: Provided as zipped archives (e.g., `nGBR_2023.zip`) containing multiple Shapefiles (`.shp` + `.dbf` + `.shx` + `.prj`). These archives typically split data into two primary tables: **Sightings** and **Herds**.

1.2.2 2.2 Common inconsistencies accounted for

To keep the pipeline robust across historical files, different survey teams, and future variations, the processing logic includes defensive handling for scenarios such as:

- drastically different column naming conventions (e.g., `number_in_group` vs `n_in_gp`, `calves` vs `n_calves`)
- variable region naming casing (e.g., "Northern Great Barrier Reef" vs "northern great barrier reef")
- dates formatted as raw strings vs native datetimes
- conflicting schema types across different shapefiles (e.g., numeric fields in one shapefile presented as strings in another)

1.3 3. Output Data Products

Two outputs are generated per pipeline execution:

1.3.1 3.1 Validated dataset (Master Parquet)

A single unified Parquet file (`master_dugong_dataset.parquet`) is produced. It contains:

- validated, Darwin Core standardised records
- exploded biological counts (individual rows for distinct life stages)
- `h3Index` (configured at resolution 4)
- `australianMarineRegionsTags`

1.3.2 3.2 Rejected dataset (DLQ Parquet)

Records that fail validation are written separately:

- written as `master_dugong_dataset_rejected.parquet`
- includes `RejectionReason`
- used for traceability and diagnostics to ensure zero data loss during extraction
- preserves native types for stable fields (e.g. dates, coordinates, counts) while stringifying highly variable/raw-source fields to avoid schema merge conflicts between heterogeneous shapefile structures

---

1.4 4. Processing Method (ETL)

The processing is implemented as an Extract → Transform → Load pipeline orchestrated with Prefect.

1.4.1 4.1 Extract

Objective: read disparate sources, apply base validation, unroll counts, and map to a uniform Darwin Core structure.

Steps:

1. **DMS Extraction:** Read parquet directly from S3 into memory (`BytesIO`).
2. **JCU Extraction:** Extract ZIP archives to a temporary directory and read `.shp` files using `pyogrio.read_arrow()`, separating tables into *Sightings* and *Herds*.
3. **Region & Event ID Generation (JCU):** - Lowercase the raw region string and map to a standard abbreviation via a strict dictionary (e.g., `northern great barrier reef` → `nGBR`).
   - Construct the `eventID` using the pattern `{region_abbr}-{YYYY}-{MM}`.
4. **Calf/Adult Unrolling (Explode Logic):** - Apply row-level math to separate total group counts into specific life stages.
   - Calculate Adults: `Total Group Count - Calves`.
   - Generate individual row structs for `adult` and `calves` types.
   - Filter out zero-counts and `.explode()` the structs into discrete occurrence records.
5. **Schema Mapping:** Select and rename columns to target Darwin Core fields (e.g., `lat` → `decimalLatitude`, `turbidity` → `turbidityScale`).

Extract output: A tuple containing `(valid_dataframe, rejected_dataframe)` for each processed source.

1.4.2 4.2 Transform

Objective: enrich the validated, concatenated master dataframe with spatial analytics properties.

1.4.2.1 A) Spatial Indexing

- Compute `h3Index` at resolution 4 using `polars_h3.latlng_to_cell` based on the validated `decimalLatitude` and `decimalLongitude`.

1.4.2.2 B) Regional Tagging

- Execute point-in-polygon spatial joins to append `australianMarineRegionsTags` using the `spatial.australian_marine_regions` package.

1.5 5. Validation Rules (Waterfall)

Validation is applied at the point of extraction. Bad records are tagged, separated, and routed to the error bucket.

1. **Missing Coordinates**
   - rule: `LATITUDE` and `LONGITUDE` (or `lat` / `long`) must be non-null
   - reason: `Missing Coordinates`
2. **Missing Time/Date**
   - rule: `start_time` or `datetime` must be non-null
   - reason: `Missing Date`

Note: Records missing group counts are filtered out during the explode phase (zero-count rows are dropped) rather than rejected explicitly.

1.6 6. Load Strategy

1.6.1 6.1 Validated Master Load

The final, enriched dataset is written as a Parquet (Snappy) file to the Optimised bucket:

- Path: `optimised-bucket/dugong/master_dugong_dataset.parquet`

1.6.2 6.2 Rejected Output Load

Records that failed the validation rules are concatenated and written as a Parquet file to the Error bucket:

- Path: `error-bucket/dugong/master_dugong_dataset_rejected.parquet`

1.7 7. Output Schema (Summary)

The master dataset adheres to Darwin Core (DwC) standards. Key fields include:

Identifiers & Taxonomy:

- `eventID`, `occurrenceID`
- `occurrenceType`
- `scientificName` (Hardcoded: *Dugong dugon*)
- `scientificNameID` (Hardcoded: *urn:lsid:marinespecies.org:taxname:220227*)

Spatial & Temporal:

- `eventDate`
- `decimalLatitude`, `decimalLongitude`
- `locationID`, `locationName`
- `geometry` (WKB Binary)

Biological Data:

- `organismQuantity`
- `organismQuantityType`

Observation Data:

- `recordedBy`

Environmental Data:

- `beaufortScale`
- `turbidityScale`

Enrichments:

- `h3Index`
- `australianMarineRegionsTags`

1.8 8. Configuration

The pipeline is configuration-driven using JSON files in the `config/` directory.

1.8.1 8.1 Source Configuration

File	Description
`dms.pyarrow_table_factory.json`	DMS parquet source path and S3 block name
`jcu_shapefile.file_interface.json`	JCU shapefile archive paths and S3 block name

1.8.2 8.2 Schema Configuration

File	Description
`extract.schema.json`	Defines exact datatypes and nullability rules for the extract output. A strict PyArrow validation guaran

---

1.9 9. Dependencies

Key libraries used in this pipeline:

Library	Purpose
`polars`	DataFrame operations and transformations
`polars_h3`	H3 spatial indexing
`pyogrio`	Shapefile reading (Arrow-native)
`pyarrow`	Schema validation and Parquet I/O
`prefect`	Workflow orchestration
`pydantic`	Configuration validation

1.10 10. Running the Pipeline

1.10.1 Prerequisites

1. AWS credentials configured (for S3 access)
2. Prefect server running (optional, for UI)

1.10.2 Execution

```
# From the repository root
cd src/datauplift/dugong
python -m datauplift.dugong.etl
```

Or via the CLI:

```
python main.py run-etl
```

1.10.3 Cloud Deployment

For Prefect cloud deployment, use:

```
python cloud.py
```

Contents

1 AMSA Vessel Tracking Optimised Data Product **2**

1.1 1. Introduction . . . 2

1.1.1 Data source . . . 2

1.2 2. Source Data Characteristics . . . 2

1.2.1 2.1 Delivery structure . . . 2

1.2.2 2.2 Common inconsistencies accounted for . . . 2

1.3 3. Output Data Products . . . 2

1.3.1 3.1 Validated dataset (Yearly master Parquet) . . . 2

1.3.2 3.2 Rejected dataset (Per source file) . . . 3

1.4 4. Processing Method (ETL) . . . 3

1.4.1 4.1 Extract . . . 3

1.4.2 4.2 Transform . . . 3

1.5 5. Validation Rules (Waterfall) . . . 4

1.6 6. Schema Enforcement . . . 4

1.7 7. Load Strategy . . . 5

1.7.1 7.1 Rejected output . . . 5

1.7.2 7.2 Yearly master upsert . . . 5

1.8 8. Output Schema (Summary) . . . 5

1.9 9. Configuration . . . 5

1.9.1 Global config (`amsa_config.json`) . . . 5

1 AMSA Vessel Tracking Optimised Data Product

1.1 1. Introduction

AMSA publishes monthly Vessel Tracking spatial datasets as downloadable archives. The raw deliveries are provided as **nested ZIP files** containing **Shapefiles**. This work converts those monthly Shapefile deliveries into a consistent, analytics-ready Parquet dataset with:

- cleaned and standardised coordinates
- consistently typed UTC timestamps
- a strict output schema
- an H3 spatial index for fast spatial grouping and joins
- a separate rejected-records output with explicit rejection reasons

1.1.1 Data source

`https://www.operations.amsa.gov.au/spatial/DataServices/DigitalData`

The source package also includes a metadata PDF describing the dataset fields and spatial reference details. That metadata is used as the reference for schema and validation expectations.

1.2 2. Source Data Characteristics

1.2.1 2.1 Delivery structure

Monthly deliveries follow this structure:

- **Outer ZIP** (monthly package)
  - contains an **Inner ZIP**
    - contains **Shapefiles** (`.shp` + `.dbf` + `.shx` + `.prj`, etc.)

1.2.2 2.2 Common inconsistencies accounted for

To keep the pipeline robust across historical files and any future variations, the processing logic includes defensive handling for scenarios that are robust to future changes in monthly geospatial files:

- archive naming conventions may vary (month abbreviation vs full month name, ordering of month/year)
- timestamps provided in different string formats
- coordinate columns may be present (`LAT`, `LON`) or coordinates may need to be derived from geometry
- duplicate records appearing in the source extracts

1.3 3. Output Data Products

Two outputs are generated:

1.3.1 3.1 Validated dataset (Yearly master Parquet)

A single Parquet file per year is produced. It contains:

- validated, standardised records
- `H3_INDEX` (configured resolution)
- audit fields (`SOURCE_FILE_NAME`, `PROCESSED_DATE`)

1.3.2 3.2 Rejected dataset (Per source file)

Records that fail validation are written separately:

- written per processed source file
- includes `REJECTION_REASON`
- used for traceability and diagnostics

1.4 4. Processing Method (ETL)

The processing is implemented as an Extract → Transform → Load pipeline orchestrated with Prefect.

1.4.1 4.1 Extract

Objective: read monthly archives and convert Shapefile content into Arrow tables.

Steps:

1. Download outer ZIP from S3 into memory (`BytesIO`)
2. Parse Year/Month from the outer ZIP filename using resilient regex patterns
3. Extract the inner ZIP to a temporary directory
4. Extract Shapefile contents using:
   - standard `zipfile` extraction first
   - fallback `stream_unzip` for archives that fail standard extraction
5. Read `*.shp` using `pyogrio.read_dataframe(..., use_arrow=True)`
6. Convert geometry to WKB for safe transport and downstream processing
7. Convert to `pyarrow.Table` and return extracted items

Extract output:
`(shapefile_name, arrow_table, year, month)` for each Shapefile found.

1.4.2 4.2 Transform

Objective: standardise columns, recover coordinates, normalise time, enrich, validate, and enforce a strict schema.

1.4.2.1 A) Column standardisation

- uppercase all column names
- map known misspellings to standard names, e.g.:
  - `TMESTAMP`, `TIMETAMP`, `TIMESAMP` → `TIMESTAMP`
  - `LATITUDE` → `LAT`
  - `LONGITUDE` → `LON`

1.4.2.2 B) Coordinate recovery

- if `LAT/LON` exist, use them
- if missing or null, derive from geometry:
  - decode WKB geometry to X/Y (LON/LAT)
- round coordinates to 5 decimal places

1.4.2.3 C) Timestamp normalisation

- parse `TIMESTAMP` using a coalesced multi-format strategy:
  - date-only
  - 24-hour datetime
  - 12-hour datetime with AM/PM
- normalise to `timestamp[ms, UTC]`

1.4.2.4 D) Enrichment

- `SOURCE_FILE_NAME` (lineage key)
- `PROCESSED_DATE` (UTC processing timestamp)
- `H3_INDEX` computed using configured H3 resolution
- `AUSTRALIAN_MARINE_REGIONS_TAGS` containing relevant Australian marine region classifications

---

1.5 5. Validation Rules (Waterfall)

Validation is applied sequentially. Each rejected set is preserved and labelled.

1. **Duplicate detection**
   - rule: full-row first-distinct check
   - reason: `Duplicate`
2. **Missing coordinates**
   - rule: `LAT` and `LON` must be non-null
   - reason: `Missing Coordinates`
3. **Invalid global coordinate ranges**
   - rule: `-90 <= LAT <= 90` and `-180 <= LON <= 180`
   - reason: `Invalid Global Coords`
4. **Configured region bounds (SAR bounding box)**
   - rule: coordinates must be within configured bounds
   - reason: `Outside SAR Region`
5. **Non-marine location detection**
   - rule: coordinates identified as land using global_land_mask. This check is applied on:
     - Australia Land : `-10 < LAT < -45` and `110 < LON < 156`
     - Antartica Land : `LAT < -68`
   - reason: `Non-Marine Locations`

Rejected rows from all stages are concatenated into a single rejected DataFrame and written to S3.

1.6 6. Schema Enforcement

A strict output schema is defined via a Pydantic wrapper around a PyArrow schema. Before load:

- missing expected columns are added as nulls
- columns are ordered consistently

- the dataset is cast to the target Arrow schema
- the final typed table is rehydrated back to Polars for writing

This ensures stable typing across months, years, and reruns.

1.7 7. Load Strategy

1.7.1 7.1 Rejected output

Rejected records are written as Parquet (Snappy) files per source:

- `.../{year}/{source_stem}_rejected.parquet`

1.7.2 7.2 Yearly master upsert

The validated dataset is loaded into a yearly master file using an idempotent upsert:

1. Load existing yearly master if present
2. Remove existing rows matching the current `SOURCE_FILE_NAME`
3. Append new validated rows
4. Overwrite the yearly master Parquet

This allows safe reprocessing without duplication.

1.8 8. Output Schema (Summary)

Validated master includes:

- `CRAFT_ID`
- `LAT`, `LON`
- `TIMESTAMP` (UTC)
- `COURSE`, `SPEED`
- `TYPE`, `SUBTYPE`
- `LENGTH`, `BEAM`, `DRAUGHT`
- `H3_INDEX`
- `AUSTRALIAN_MARINE_REGIONS_TAGS`
- `SOURCE_FILE_NAME`
- `PROCESSED_DATE`

Rejected output includes:

- `REJECTION_REASON`

1.9 9. Configuration

The pipeline is configuration-driven and validated using Pydantic.

1.9.1 Global config (`amsa_config.json`)

Controls:

- input/output/rejected S3 locations
- bounding box for region validation
- H3 resolution

Example:

```
{
  "input":   { "block_name": "processing-bucket", "path": "karishma.khanna/AMSA/Processing/Extracted" }
  "output":  { "block_name": "optimised-bucket",  "path": "karishma.khanna/AMSA/Optimised" },
  "rejected":{ "block_name": "error-bucket",      "path": "karishma.khanna/AMSA/Rejected" },
  "geospatial": {
    "lon_min": 75.0,
    "lon_max": 163.0,
    "lat_min": -86.6,
    "lat_max": -2.0,
    "h3_resolution": 8
  }
}
```

Appendix B

Notebooks

The notebooks used in the training videos are provided as a separate set of PDF files.

The original code is publicly available in https://github.com/aodn/imos-user-code-library repository.

Seagrass R Notebook

Thomas Galindo, Denisse Fierro Arcos

2026-02-10

- Background
  - Relevant packages
  - Accessing NESP datasets
    - Loading dataset as data frame
    - Subsetting dataset before loading
  - Observation Effort, Organism Quantity and Richness
  - Visualising seagrass dataset
  - Data aggregation and mapping with `H3`
    - `H3` indexing
  - Australian Marine Regions
  - Exploring Available Marine Regions Tags
  - Filtering to a Particular Marine Region
  - Identifying suitable trend sites
    - Per Region
    - Per Dataset
    - Mapping the Spatial Extent of a Particular Dataset
    - Finding Available Datasets
  - Finding trend sites with `H3`

Background

This R notebook outlines the packages needed to access and analyse NESP Data Products, as well as how to check their content structure. This notebook was originally written in Python by Thomas Galindo. It was "translated" into R by Denisse Fierro Arcos. It should be noted that although both notebooks cover the same topic, the content may vary between these notebooks as features may not be available in both Python and R.

This notebook includes the following sections:
- Relevant R packages needed to access NESP Data Products - How to access NESP Data Products - How aggregate and map NESP datasets using the H3 package - Identifying suitable sampling sites where trend analysis can be carried out

Note: The `H3` package was originally developed in C, but there are at least three packages that make this package accessible in R. We recommend you install the `h3-r` package, which published in GitHub (https://github.com/crazycapivara/h3-r/) because this package will automatically install the original `H3` package written in C (unlike the other two options available as of February 2026). More information about the `h3-r` package, as well as a short tutorial can be found in their documentation (https://crazycapivara.github.io/h3-r/articles/h3.html).

Relevant packages

```
# Connects to datasets remotely
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.5.2
```

```
##
## Attaching package: 'arrow'
```

```
## The following object is masked from 'package:utils':
##
##     timestamp
```

```
# Indexing system that uses hexagons
library(h3)
# Represents simple spatial features
library(sf)
```

```
## Linking to GEOS 3.13.1, GDAL 3.11.0, PROJ 9.6.0; sf_use_s2() is TRUE
```

```
# Data manipulation
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(stringr)
# Plotting
library(ggplot2)
# Interactive maps
library(leaflet)
```

```
## Warning: package 'leaflet' was built under R version 4.5.2
```

Accessing NESP datasets

NESP 5.9 datasets can be accessed via the `arrow` package for R. First, we will need to create a connection with the parent folder containing NESP data as follows:

```
conn <- s3_bucket("s3://data-uplift-public/", anonymous = TRUE,
                  region = "ap-southeast-2")
```

Once a connection has been established, then we can select a specific dataset that interest us. In this case, we will be accessing a dataset about seagrasses. By using the `open_dataset` function from `arrow` we can load either the dataset on its entirety or a subset.

```
url <- "s3://data-uplift-public/stored/datauplift/seagrass/seagrass.parquet"
ds <- open_dataset(sources = url, filesystem = conn)
ds
```

```
## FileSystemDataset with 1 Parquet file
## 21 columns
## datasetName: string not null
## sourceID: string not null
## metadataLink: string not null
## startYear: int16 not null
## startMonth: int8
## startDay: int8
## endYear: int16
## endMonth: int8
## endDay: int8
## footprintWKT: string not null
## decimalLatitude: float not null
## decimalLongitude: float not null
## h3Index: string not null
## australianMarineRegionsTags: string
## occurrenceStatus: string not null
## organismQuantity: float
## organismQuantityType: string
## scientificName: string not null
## scientificNameID: int64 not null
## taxonRank: string not null
## ...
## 1 more columns
## Use `schema()` to see entire schema
##
## See $metadata for additional Schema metadata
```

The seagrasses dataset is partitioned (i.e., separated into parquet fragments), but it is really a single file. We can check the number of files included in a dataset as follows:

```
ds$files
```

```
## [1] "data-uplift-public/stored/datauplift/seagrass/seagrass.parquet"
```

We can also check the file structure (i.e., schema) of the seagrass dataset. This will give us the name of the columns contained in the dataset, which can help us filter the data before accessing.

```
ds$schema
```

```
## Schema
## datasetName: string not null
## sourceID: string not null
## metadataLink: string not null
## startYear: int16 not null
## startMonth: int8
## startDay: int8
## endYear: int16
## endMonth: int8
## endDay: int8
## footprintWKT: string not null
## decimalLatitude: float not null
## decimalLongitude: float not null
## h3Index: string not null
## australianMarineRegionsTags: string
## occurrenceStatus: string not null
## organismQuantity: float
## organismQuantityType: string
## scientificName: string not null
## scientificNameID: int64 not null
## taxonRank: string not null
## taxonomicStatus: string not null
##
## See $metadata for additional Schema metadata
```

From here, we can see there are 21 columns, but we can check how many records in total are included in the dataset as follows:

```
dim(ds)
```

```
## [1] 3286273      21
```

This dataset has over 3 million records, which may be too much for some computers to handle, but `arrow` allows us to lazily load the dataset. This means that the data is only load into your memory when you request it, instead of loading all 3 million records at once. This could be especially useful if you have limited computing resources, so you can subset or summarise the data as you need before loading it to your computer to avoid overloading the computer's memory.

Loading dataset as data frame

If the dataset you are interested in is relatively small, we recommend you simply load it into memory usin the `collect` function from `arrow`.

```
# Note that running this code may take a while, so only run it if you have enough memory available
df <- ds |>
  collect()

# We can check the first few rows in this dataset
head(df)
```

```
## # A tibble: 6 × 21
##   datasetName        sourceID metadataLink startYear startMonth startDay endYear
##   <chr>              <chr>    <chr>            <int>      <int>    <int>   <int>
## 1 Seagrass mapping … GBR_NES… unknown           2012          6       NA      NA
## 2 Seagrass mapping … GBR_NES… unknown           2012          6       NA      NA
## 3 Seagrass mapping … GBR_NES… unknown           2012          6       NA      NA
## 4 Seagrass mapping … GBR_NES… unknown           2012          6       NA      NA
## 5 Seagrass mapping … GBR_NES… unknown           2012          6       NA      NA
## 6 Seagrass mapping … GBR_NES… unknown           2012          6       NA      NA
## # ℹ 14 more variables: endMonth <int>, endDay <int>, footprintWKT <chr>,
## #   decimalLatitude <dbl>, decimalLongitude <dbl>, h3Index <chr>,
## #   australianMarineRegionsTags <chr>, occurrenceStatus <chr>,
## #   organismQuantity <dbl>, organismQuantityType <chr>, scientificName <chr>,
## #   scientificNameID <int>, taxonRank <chr>, taxonomicStatus <chr>
```

Subsetting dataset before loading

If the dataset of your interest is really large, then it is probably a better idea to subset it before loading to memory. For example, we will keep records when `startYear` is 2025.

```
ds_2025 <- ds |>
  filter(startYear == 2025) |>
  collect()

dim(ds_2025)
```

```
## [1] 25 21
```

This is a much smaller version of the seagrass dataset containing on 25 records, which is much easier for a computer to handle.

Remember: Use the `collect` function once you are ready to load the dataset of your interest.

Observation Effort, Organism Quantity and Richness

Observation Effort is the number of occurrence records (e.g., sampling campaigns), while **Organism Quantity** is the number of organisms of a species recorded at each occurrence.

Observation Effort must be taken into account when analysing trends; fluctuations in observation effort tend to distort the reported occurrences.

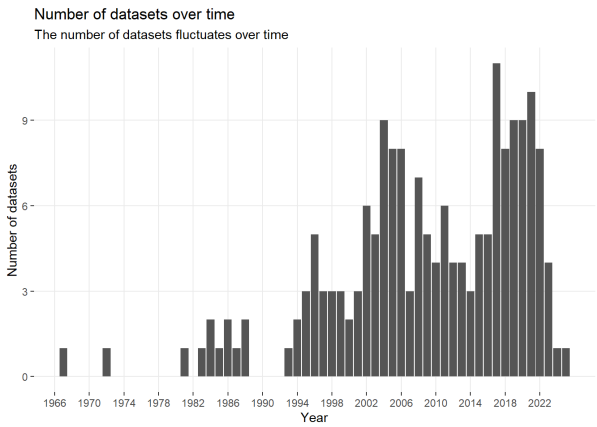
Species **Richness** refers to the number of different species present in a particular area. While **Organism Quantity** tells us "how many" individuals there are, **Richness** tells us "how many types" of organisms coexist.

Typically the **Richness** of an area correlates to the number of studies carried out in a given area, but eventually plateaus and this plateau is an indication that the studied region is well sampled.

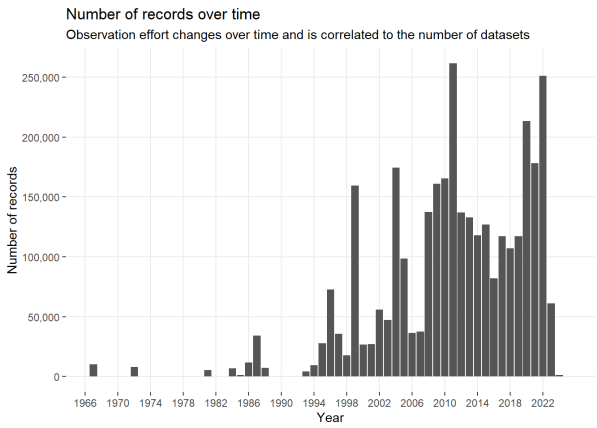
Visualising seagrass dataset

We can create a few plots to better understand the structure of this dataset.

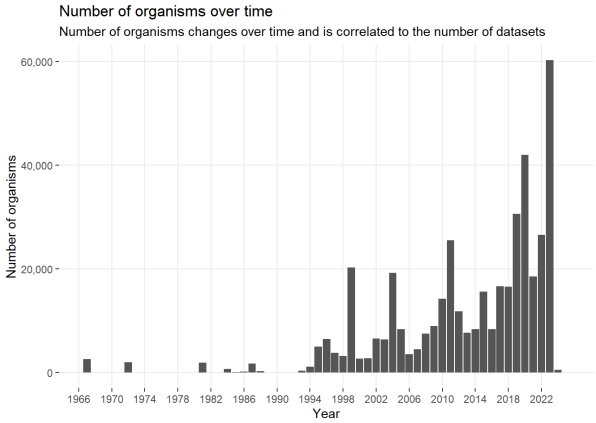
```
# Number of unique datasets per year
df |>
  distinct(startYear, datasetName) |>
  group_by(startYear) |>
  count() |>
  ggplot(aes(startYear, n))+
  geom_bar(stat = "identity")+
  scale_x_continuous(breaks = seq(min(df$startYear, na.rm = T)-1,
                                  max(df$startYear, na.rm = T), by = 4))+
  theme_bw()+
  labs(x = "Year", y = "Number of datasets",
       title = "Number of datasets over time",
       subtitle = "The number of datasets fluctuates over time")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```



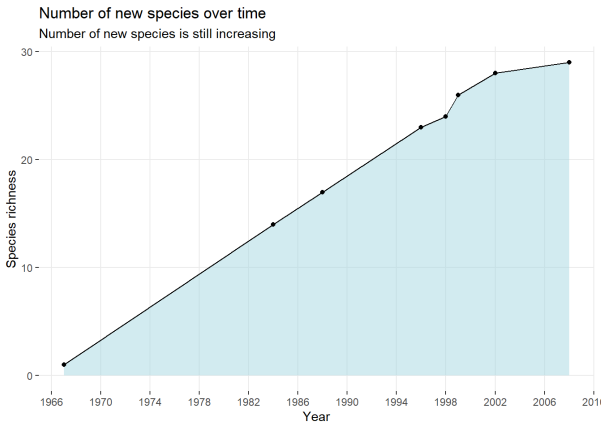

```
# Number of records per year
df |>
  group_by(startYear) |>
  count() |>
  ggplot(aes(startYear, n))+
  geom_bar(stat = "identity")+
  scale_x_continuous(breaks = seq(min(df$startYear, na.rm = T)-1,
                                  max(df$startYear, na.rm = T), by = 4))+
  scale_y_continuous(labels = scales::label_comma(),
                     breaks = seq(0, 250000, by = 50000))+
  theme_bw()+
  labs(x = "Year", y = "Number of records",
       title = "Number of records over time",
       subtitle = "Observation effort changes over time and is correlated to the number of datasets")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```




```
# Number of observations per year
df |>
  select(startYear, occurrenceStatus, organismQuantity) |>
  mutate(organismQuantity = case_when((is.na(organismQuantity) &
                                         occurrenceStatus == "absent") ~ 0,
                                      (is.na(organismQuantity) &
                                         occurrenceStatus == "present") ~ 1,
                                      T ~ organismQuantity)) |>
  group_by(startYear) |>
  summarise(tot_obs = sum(organismQuantity, na.rm = T)) |>
  ggplot(aes(startYear, tot_obs))+
  geom_bar(stat = "identity")+
  scale_x_continuous(breaks = seq(min(df$startYear, na.rm = T)-1,
                                  max(df$startYear, na.rm = T), by = 4))+
  scale_y_continuous(labels = scales::label_comma())+
  theme_bw()+
  labs(x = "Year", y = "Number of organisms",
       title = "Number of organisms over time",
       subtitle = "Number of organisms changes over time and is correlated to the number of datasets")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```




```
# Number of unique new species per year
df |>
  distinct(startYear, scientificNameID) |>
  arrange(startYear) |>
  distinct(scientificNameID, .keep_all = T) |>
  group_by(startYear) |>
  count() |>
  ungroup() |>
  mutate(tot_sp = cumsum(n)) |>
  ggplot(aes(startYear, tot_sp))+
  geom_density(stat = "identity", fill = "lightblue", alpha = 0.5)+
  geom_point()+
  scale_x_continuous(breaks = seq(min(df$startYear, na.rm = T)-1,
                                  max(df$startYear, na.rm = T), by = 4))+
  theme_bw()+
  labs(x = "Year", y = "Species richness",
       title = "Number of new species over time",
       subtitle = "Number of new species is still increasing")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```

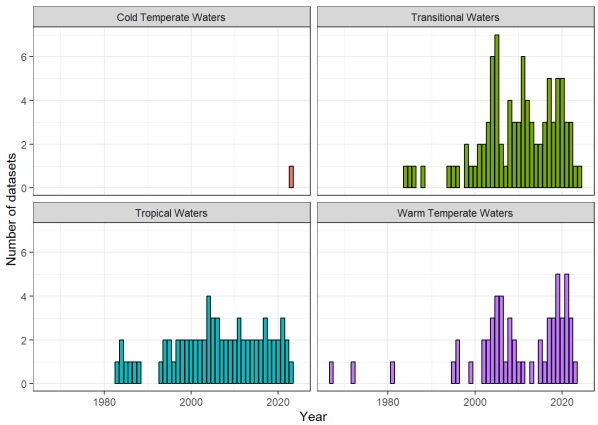


We can also explore the information in this table further, by extracting information about the "Water Type" of the sites sampled.

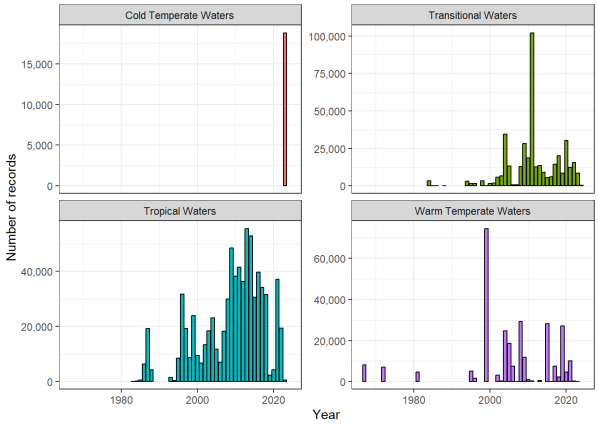
```
df_water_type <- df |>
  mutate(short = str_extract(australianMarineRegionsTags,
                             "\\|(IMCRA:Mesoscale:Water Type.*Waters)\\|",
                             group = 1),
         watertype = str_extract(str_split_i(short, "\\|", i = 1), "Type:(.*)",
                             group = 1)) |>
  select(!short) |>
  drop_na(watertype)
```

Now we can check the number of unique datasets available over time per "Water Type".

```
df_water_type |>
  distinct(startYear, datasetName, watertype) |>
  group_by(startYear, watertype) |>
  count() |>
  ggplot(aes(startYear, n))+
  geom_bar(stat = "identity", aes(fill = watertype), colour = "black")+
  facet_wrap(~watertype)+
  labs(y = "Number of datasets", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```

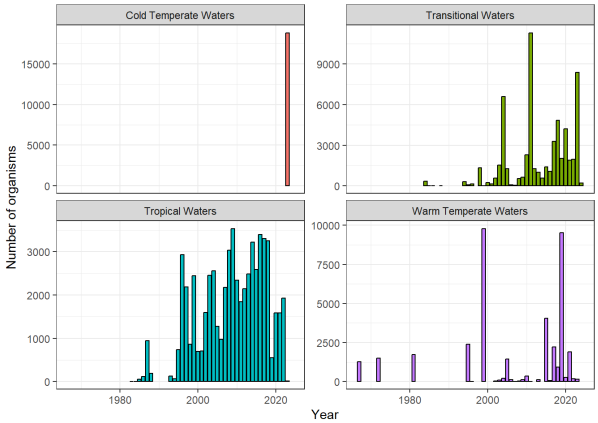


```
df_water_type |>
  group_by(startYear, watertype) |>
  count() |>
  ggplot(aes(startYear, n))+
  geom_bar(stat = "identity", aes(fill = watertype), colour = "black")+
  facet_wrap(~watertype, scales = "free_y")+
  scale_y_continuous(labels = scales::label_comma())+
  labs(y = "Number of records", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```

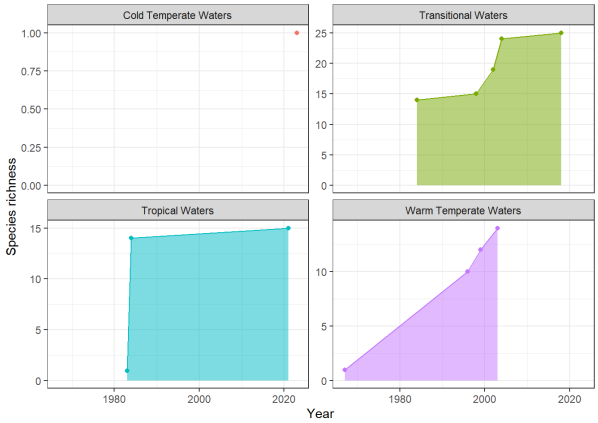



```
df_water_type |>
  mutate(organismQuantity = case_when((is.na(organismQuantity) &
                                          occurrenceStatus == "absent") ~ 0,
                                      (is.na(organismQuantity) &
                                          occurrenceStatus == "present") ~ 1,
                                      T ~ organismQuantity)) |>
  group_by(startYear, watertype) |>
  summarise(tot_obs = sum(organismQuantity, na.rm = T)) |>
  ggplot(aes(startYear, tot_obs))+
  geom_bar(stat = "identity", aes(fill = watertype), colour = "black")+
  facet_wrap(~watertype, scales = "free_y")+
  labs(y = "Number of organisms", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```

```
## `summarise()` has grouped output by 'startYear'. You can override using the
## `.groups` argument.
```




```
df_water_type |>
  distinct(startYear, watertype, scientificNameID) |>
  arrange(startYear) |>
  distinct(scientificNameID, watertype, .keep_all = T) |>
  group_by(startYear, watertype) |>
  count() |>
  group_by(watertype) |>
  mutate(tot_sp = cumsum(n)) |>
  ggplot(aes(startYear, tot_sp, fill = watertype, colour = watertype))+
  geom_density(stat = "identity", alpha = 0.5)+
  geom_point()+
  facet_wrap(~watertype, scales = "free_y")+
  labs(y = "Species richness", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```



Data aggregation and mapping with `H3`

The NESP 5.9 datasets are aggregated data products that cover large regions and temporal periods. This aggregation is useful as it helps us understand the spatio-temporal bounds of our dataset.
We use the `H3` package in this notebook because it is an open source package that partitions the world into hexagonal cells, allowing for faster data access.

`H3` indexing

The `H3` package allows us to spatially index datasets at different resolutions (https://h3geo.org/docs/core-library/restable/#average-area-in-km2) (i.e., zoom levels) relative to the recorded latitude and longitude. If you go through the dataset schema, you will notice that it has an `h3Index` column, which contains unique identifiers for each hexagon in our dataset.

```
# Getting number of records for each polygon ID (h3Index)
h3_layers <- df |>
  group_by(h3Index) |>
  summarise(number_records = n())

# Now we get information about datasets associated to each polygon ID
h3_layers <- df |>
  distinct(h3Index, datasetName) |>
  group_by(h3Index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(h3_layers, by = "h3Index") |>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3Index)) |>
  ungroup()

# We can check the results
head(h3_layers)
```

```
## # A tibble: 6 × 4
##   h3Index         datasets                          number_records index$h3_index
##   <chr>           <chr>                                      <int> <chr>
## 1 85941537fffffff Satellite Mapping of Bathymetry…          27022 85941537fffff…
## 2 85941583fffffff Satellite Mapping of Bathymetry…           7794 85941583fffff…
## 3 85941587fffffff Satellite Mapping of Bathymetry…          30799 85941587fffff…
## 4 8594158ffffffff Satellite Mapping of Bathymetry…          13380 8594158fffff…
## 5 859415b3fffffff Satellite Mapping of Bathymetry…            183 859415b3fffff…
## 6 859415bbfffffff Satellite Mapping of Bathymetry…           8672 859415bbfffff…
## # ℹ 1 more variable: index$geometry <POLYGON [°]>
```

We can now create an interactive map as shown below. Note that you can interact with the map, that is zoom in and out, and hover over each title to get information about the number of records and datasets available at each polygon.

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = h3_layers$number_records,
                                         n = 10)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets:", h3_layers$datasets),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```

Screenshot of the map above generated by leaflet.

Australian Marine Regions

Each occurrence in the seagrass dataset that we have been exploring so far has been classified using the following Australian Marine Regions (see table below). Note that the classification is not 100% accurate at the region boundaries; it is accurate to ~6km.

Note that the Great Barrier Reef has been included from the World Heritage Areas map.

Marine Region Type	Map Explore URL	Metadata URL (REST/About)
Commonwealth Marine Regions	Explore (https://fed.dcceew.gov.au/datasets/commonwealth-marine-regions/explore)	Metadata (https://gis.environment.gov.au/gispubmap/rest/services/ogc_services/Commonwealth_Marine_F
World Heritage Areas	Explore (https://fed.dcceew.gov.au/datasets/erin::australia-world-heritage-areas/explore)	Metadata (https://gis.environment.gov.au/gispubmap/rest/services/ogc_services/World_Heritage_

Marine Region Type	Map Explore URL	Metadata URL (REST/About)
Marine Parks	Explore (https://fed.dcceew.gov.au/datasets/2b3eb1d42b8d4319900cf4777f0a83b9_0/explore)	Metadata (https://gis.environment.gov.au/gispubmap/rest/services/ogc_services/Australian_Marine_Parks/l
IMCRA Provincial	Explore (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation-of-australia-imcra-v4-0-provincial-bioregions/explore)	Metadata (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation v4-0-provincial-bioregions/about)
IMCRA Mesoscale	Explore (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation-of-australia-imcra-v4-0-meso-scale-bioregions/explore)	Metadata (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation v4-0-meso-scale-bioregions/about)

Follow the Explore links to explore the Australian Marine Regions. The Commonwealth Marine Regions can be seen in the following link: https://fed.dcceew.gov.au/datasets/commonwealth-marine-regions/explore?location=-33.665700%2C140.517300%2C4 (https://fed.dcceew.gov.au/datasets/commonwealth-marine-regions/explore?location=-33.665700%2C140.517300%2C4)

Exploring Available Marine Regions Tags

Australian Marine Regions Tags have the following syntax:

TAG	TYPE	OPTIONAL SUPER REGIONS	REGION
`Commonwealth Marine Region: South-east`	`Commonwealth Marine Region`		`South-east`
`Marine Park:National Park Zone:Freycinet`	`Marine Park`	`National Park Zone`	`Freycinet`
`IMCRA:Provincial:Water Type:Cold Temperate Waters`	`IMCRA`	`Provincial, Water Type`	`Cold Temperate Waters`

Extracting the available Australian Marine Regions Tags requires splitting the delimited column:

```
australian_marine_regions_tags <- df |>
  select(australianMarineRegionsTags) |>
  drop_na() |>
  separate_longer_delim(australianMarineRegionsTags, delim = "|") |>
  group_by(australianMarineRegionsTags) |>
  summarise(count = n()) |>
  arrange(desc(count))

# Printing the first 10 tags
head(australian_marine_regions_tags, n = 10)
```

```
## # A tibble: 10 × 2
##    australianMarineRegionsTags                          count
##    <chr>                                                <int>
##  1 IMCRA:Provincial:Water Type:Tropical Waters       1495922
##  2 IMCRA:Mesoscale:Water Type:Tropical Waters        1377704
##  3 World Heritage Area:Great Barrier Reef            1222090
##  4 IMCRA:Provincial:Name:Northeast Shelf Province     970257
##  5 IMCRA:Mesoscale:Water Type:Transitional Waters     794033
##  6 IMCRA:Provincial:Water Type:Transitional Waters    794033
##  7 IMCRA:Mesoscale:Water Type:Warm Temperate Waters   517106
##  8 IMCRA:Provincial:Water Type:Warm Temperate Waters  517106
##  9 IMCRA:Mesoscale:Name:Shoalwater Coast              491652
## 10 IMCRA:Provincial:Name:Northeast Shelf Transition   431041
```

```
# Getting number of records for each polygon ID (h3Index)
marine_reg <- df |>
  drop_na(australianMarineRegionsTags) |>
  group_by(h3Index) |>
  summarise(number_records = n())

# Now we get information about datasets associated to each polygon ID
marine_reg <- df |>
  drop_na(australianMarineRegionsTags) |>
  distinct(h3Index, datasetName) |>
  group_by(h3Index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(marine_reg, by = "h3Index")

# Finally we get information about the marine tags
marine_reg <- df |>
  select(h3Index, australianMarineRegionsTags) |>
  drop_na(australianMarineRegionsTags) |>
  distinct() |>
  separate_longer_delim(australianMarineRegionsTags, delim = "|") |>
  distinct() |>
  group_by(h3Index) |>
  summarise(tags = paste(australianMarineRegionsTags, collapse = ", ")) |>
  right_join(marine_reg, by = "h3Index") |>
  mutate(index = h3_to_geo_boundary_sf(marine_reg$h3Index)) |>
  ungroup()

# Checking results
head(marine_reg)
```

```
## # A tibble: 6 × 5
##   h3Index         tags                     datasets number_records index$h3_index
##   <chr>           <chr>                    <chr>             <int> <chr>
## 1 85941537ffffff IMCRA:Provincial:Name:… Satelli…          27022 85941537fffff…
## 2 85941583ffffff Marine Park:Sanctuary … Satelli…           7794 85941583fffff…
## 3 85941587ffffff IMCRA:Provincial:Name:… Satelli…          30799 85941587fffff…
## 4 8594158fffffff Marine Park:Sanctuary … Satelli…          13380 8594158fffff…
## 5 859415b3ffffff IMCRA:Provincial:Name:… Satelli…            183 859415b3fffff…
## 6 859415bbffffff IMCRA:Provincial:Water… Satelli…           8672 859415bbfffff…
## # ℹ 1 more variable: index$geometry <POLYGON [°]>
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = marine_reg$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = marine_reg$number_records,
                                         n = 10)(marine_reg$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", marine_reg$number_records,
                             "\nDatasets: ", marine_reg$datasets,
                             "\nTags: ", marine_reg$tags),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```

Screenshot of the map above generated by leaflet.

Filtering to a Particular Marine Region

We could also create a map after selecting (filtering) for a particular tag as follows:

```
tropical_waters_df <- df |>
  select(!h3Index) |>
  filter(str_detect(australianMarineRegionsTags,
                    "IMCRA:Provincial:Water Type:Tropical Waters")) |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)

tropical_waters_df <- tropical_waters_df |>
  mutate(h3index = geo_to_h3(tropical_waters_df, 7))

# Getting number of records for each polygon ID (h3Index)
h3_layers <- tropical_waters_df |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

# Now we get information about datasets associated to each polygon ID
h3_layers <- tropical_waters_df |>
  st_drop_geometry() |>
  distinct(h3index, datasetName) |>
  group_by(h3index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(h3_layers, by = "h3index") |>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3index)) |>
  ungroup()
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_records),
                                         n = 10)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets: ", h3_layers$datasets),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```

Screenshot of the map above generated by leaflet.

Identifying suitable trend sites

In the original map that includes all data, we can see there are some sites with a high density of records, while others have very limited observations. The high density polygons are the most ideal sites for trend analysis.

Per Region

Regions with a high density of records are typically good candidates for trend analysis. See above for subsetting and plotting by Australian Marine Regions.

Per Dataset

Additionally, the singular datasets that comprise the aggregated data product are typically good candidates for trend analysis.

Mapping the Spatial Extent of a Particular Dataset

Similarly to mapping Australian Marine Regions, we may map particular datasets by applying a string constraint.

Finding Available Datasets

We may find available datasets by aggregating by the `datasetName` column.

```
data_count <- df |>
  group_by(datasetName) |>
  summarise(count = n()) |>
  arrange(count)

data_count
```

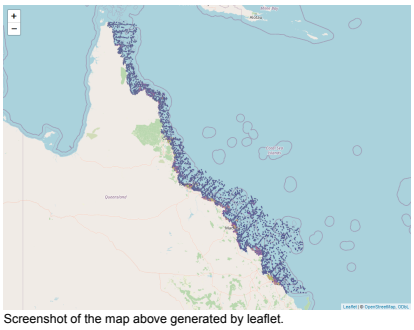
```
## # A tibble: 43 × 2
##    datasetName                                                          count
##    <chr>                                                                <int>
##  1 "Seagrass meadows extents derived from field to spaceborne earth obser…    72
##  2 "Dungeness Reef Seagrass Survey, Torres Strait, 2017 (Community Type)"     88
##  3 "Eastern Torres Strait Seagrass Survey, Torres Strait, 2020 "             162
##  4 "Seagrass meadow extents derived from field to spaceborne earth observ…   488
##  5 "Seagrass and associated benthic community data derived from field sur…   515
##  6 "Bindjareb Djilba (Peel Harvey) Seagrass Survey - Cover (DWER-128)"       539
##  7 "Orman Reef Seagrass Survey, Torres Strait, November 2018 (TropWATER, …   882
##  8 "Dungeness Reef Seagrass Survey, Torres Strait, 2017 (Site-Surveys)"      910
##  9 "Mapping of selected areas of Cockburn Sound, Western Australia"          960
## 10 "Dungeness Reef Seagrass Survey, Torres Strait,2016"                     1113
## # ℹ 33 more rows
```

```
data_int <- data_count |>
  filter(count == max(count)) |>
  pull(datasetName)

#Check name of dataset with the most observations
data_int
```

```
## [1] "Seagrass mapping synthesis: A resource for coastal management in the Great Barrier Reef (NESP TWQ 3.2.1
and NESP TWQ 5.4, TropWATER, James Cook University)"
```

```
trop_water_jcu_df <- df |>
  select(!h3Index) |>
  filter(datasetName == data_int) |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)

trop_water_jcu_df  <- trop_water_jcu_df  |>
  mutate(h3index = geo_to_h3(trop_water_jcu_df, 6))

# Getting number of records for each polygon ID (h3Index)
h3_layers <- trop_water_jcu_df  |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

# Now we get information about datasets associated to each polygon ID
h3_layers <- trop_water_jcu_df  |>
  st_drop_geometry() |>
  distinct(h3index, datasetName) |>
  group_by(h3index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(h3_layers, by = "h3index") |>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3index)) |>
  ungroup()
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_records),
                                         n = 10)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets: ", h3_layers$datasets),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.

Finding trend sites with `H3`

We may also identify trend sites by finding `H3` cells associated with data over a year range.

```r
df_h3 <- df |>
  select(!h3Index) |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)

df_h3 <- df_h3|>
  mutate(h3index = geo_to_h3(df_h3, 7))

# Getting number of records for each polygon ID (h3Index)
n_rec <- df_h3 |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

n_dat <- df_h3 |>
  st_drop_geometry() |>
  distinct(h3index, datasetName) |>
  group_by(h3index) |>
  summarise(number_datasets = n(),
            datasets = paste(datasetName, collapse = ", "))

n_yr <- df_h3 |>
  st_drop_geometry() |>
  distinct(h3index, startYear) |>
  group_by(h3index) |>
  summarise(number_years = n(),
            years = paste(startYear, collapse = "|"))

# Now we get information about datasets associated to each polygon ID
h3_layers <- n_rec |>
  right_join(n_yr, by = "h3index") |>
  right_join(n_dat, by = "h3index") |>
  filter(number_years >= 2)

h3_layers <- h3_layers|>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3index)) |>
  ungroup()
```

```r
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygons with
  # a higher number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_years),
                                         n = 5)(h3_layers$number_years),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets: ", h3_layers$datasets,
                             "\nYears: ", h3_layers$years),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```

Screenshot of the map above generated by leaflet.

NESP 5.9 Seagrass Notebook

Summary by Sub-Area	
Author:	Thomas Mathews
Added:	2016-03-30
Affiliations:	AUSRIVERS
e-mail:	thomas.mathews@ausrivers.com.au
Last Updated:	2020-10-29
Date of last update:	2020-10-29

Background

This is a Jupyter notebook outlining the Marine Data and Content structure for NESP Data Products.

It addresses:

1. Corresponding to NESP 5.9 datasets
2. 5.9 Aggregation and Mapping
3. Identifying Hand Data

NESP 5.9 Datasets

Dataset name	Description	Metadata	GA URL
Index	NESP Index Template	link	
Hand Data	Hand Data Template	link	
Habitat	Habitat Mapping Data	link	
Seagrass	Seagrass Observations and Counting	link	
Transects	Seagrass Surveys	link	

5.9 Details

NESP 5.9 datasets are currently scattered in DM and partially available as per the following table:

Dataset	Site	Port Hedland
dpaw:vpn:gis:public	/mnt/datahub/share/DPaw/spatial/services/services/parcelgeo	☑
dpaw:vpn:gis:secure	/mnt/datahub/share/DPaw/SDE/parcelgeo	☑
dpaw:vpn:gis:sde	/mnt/datahub/share/DPaw/SDE/parcelgeo	☑
dpaw:vpn:gis:spatial	/mnt/datahub/share/DPaw/spatial/parcelgeo	☑
dpaw:vpn:gis:services	/mnt/datahub/share/DPaw/services/services/parcelgeo	☑

Required Python & Packages

The required packages are found in the requirements.txt file in our project root.

5.9 Installation

It is highly recommended to manage python version, environment and projects with pyenv.

```
$ cmd$ $ pyenv install 3.6.5
$ pyenv virtualenv 3.6.5 nesp-5.9
```

5.9 Installation

Otherwise, native python tooling can be used to manage the python environment and project.

Ensure that python3.6 or greater is required

```
python3 -m venv .venv $ source .venv/bin/activate $ pip install -r
```

Running Notebooks in Jupyter or IDE

The notebook can be manually run by launching jupyter notebook from the console

```
$ jupyter notebook
```

Otherwise, if working in an IDE, select the menu number 1,select, as the kernel.

Notebook Stack

5.9

Corresponding to the delivery proposal and since standard library tooling in the python ecosystem, offering exceptional performance, documentation and usability.

So, this Jupyter Notebook stack is based on Python3 and Jupyter Notebook.

Configuration

.gitignore is fast replicating gitignore for the primary python dataframe IDE. It is more highly integrated with vimrc and pyenv but, has exceptional performance, documentation and usability. Jupyter notebook dependencies, a more consistent and comprehensive IDE, SLAM scripting and many other advantages.

Mapping

.gitignore is a python mapping file. GDAL/OGR, (GIPS) powered framework for massively parallel geospatial data analysis of large datasets. It is optimized for Jupyter Notebook and Python3 environments.

Standard mapping tools like TOPOJSON , Kepler.gl bundle the number of mapped shapes, so we have this more optimised GEOJSON library.

Connecting to NESP 5.9 Datasets

NESP 5.9 datasets can be accessible via the scripts: DATALOAD.sh.

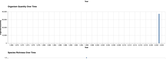
The DATALOAD.sh manages the connections to the NESP 5.9 datasets stored in S3, resulting in a .geojsonlines.gzipped file. Sandia that we may use for further processing and analysis, in the future.

Note that a subset of the datasets are provided for all jurisdictions into corporate postgres/postgresql.

This step also makes the dataset schema available and the structure available allowing for any filtering without downloading the entire dataset.

Finally, we automate the process, allowing for the processing of the dataset. If the dataset is small, we may use a .GEOJSONLINE wrapped, using .GEOJSONLINE.gz dataset to load large, we must join aggregate and or filter using a .CSVLATLONG before loading the dataset into memory.

```
# Connecting to NESP 5.9 Datasets
# Author: Thomas Mathews
# Date: 2020-10-29

# Variables
# ------------------------------------------------------------------------
# Variables for connecting to the postgres database
# ------------------------------------------------------------------------
# Port for the Postgres/PostGIS DB
PGPORT=5432

# The hostname of the Postgres/PostGIS DB
PGHOST=localhost

# The name of the Postgres/PostGIS database
PGDB=nesp_5_9_test

# The Postgres/PostGIS DB user
PGUSER=nesp_5_9_test

# The Postgres/PostGIS DB user's password
PGPASSWORD=nesp_5_9_test

# The schema in the Postgres/PostGIS database
PGSCHEMA=public,nesp_5_9_test

# The Postgres/PostGIS DB connection string
PGCONNSTRING="host=$PGHOST port=$PGPORT dbname=$PGDB user=$PGUSER password=$PGPASSWORD"

# The Amazon S3 bucket name
S3BUCKET=s3-ausrivers-fsi-sandia-datasets-nesp

# The Amazon S3 bucket path
S3PATH=nesp_5_9_test

# The AWS profile name
AWSPROFILE=default

# The AWS region name
AWSREGION=ap-southeast-2

# The AWS access key id
AWSACCESSKEYID=AKIAXXXXXXXXXXXXXXXX

# The AWS secret access key
AWSSECRETACCESSKEY=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

# The AWS output format
AWSOUTPUT=text

# The AWS output delimiter
AWSDELIM=,

# The AWS output query
AWSQUERY="Contents[].{Key:Key,ETag:ETag,LastModified:LastModified,Size:Size,StorageClass:StorageClass}"

# The AWS output command
AWSOUTCMD="aws --profile $AWSPROFILE --region $AWSREGION --output $AWSOUTPUT --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/ --recursive"

# The AWS output command for the JMESPATH query for the JSON output
AWSOUTCMDJSON="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/ --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name
AWSOUTCMDJSONDS="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .geojsonlines.gz file
AWSOUTCMDJSONDSGZ="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.geojsonlines.gz --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .CSVLATLONG file
AWSOUTCMDJSONDSCSV="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.CSVLATLONG --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE file
AWSOUTCMDJSONDSGEO="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file
AWSOUTCMDJSONDSGEOGZ="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE.gz --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file
AWSOUTCMDJSONDSGEOGZCSV="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE.gz --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file
AWSOUTCMDJSONDSGEOGZCSVGEO="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file
AWSOUTCMDJSONDSGEOGZCSVGEOGZ="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE.gz --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSV="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.CSVLATLONG --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSVGEO="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSVGEOGZ="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE.gz --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSVGEOGZCSV="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.CSVLATLONG --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSVGEOGZCSVGEO="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSVGEOGZCSVGEOGZ="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE.gz --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSVGEOGZCSVGEOGZCSV="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.CSVLATLONG --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file
AWSOUTCMDJSONDSGEOGZCSVGEOGZCSVGEOGZCSVGEOGZCSVGEO="aws --profile $AWSPROFILE --region $AWSREGION --output JSON --query $AWSQUERY s3 ls s3://$S3BUCKET/$S3PATH/dataset/*.GEOJSONLINE --recursive"

# The AWS output command for the JMESPATH query for the JSON output, with the dataset name, and the .GEOJSONLINE.gz file, and the .CSVLATLONG file, and the .GEOJSONLINE file
```



H3 Aggregation and Mapping

The NMP H3 dataset can aggregated data products from cover large regions and time periods.

It is useful to first aggregate the datasets so that we may understand their temporal spatial bounds.

H3 Indexing

The 3D Library allows us to spatially index datasets at different resolutions (coarse levels according to the second latitude and longitude, medium levels, like `pyarrow_h3` library offers a pyarrow stack interface for faster ingesting with `pyarrow.duckdb` module).

```
# Add a H3 column to our DataFrame with latitude and longitude columns
df['h3_index'] = df.apply(
    lambda row: h3.latlng_to_cell(row['latitude'], row['longitude'],
    resolution=3),
    axis=1
)

# Display first few rows to verify
print("DataFrame with H3 indexes:")
print(df[['latitude', 'longitude', 'h3_index']].head())

# Count unique H3 cells
print(f"\nNumber of unique H3 cells: {df['h3_index'].nunique()}")
```

```
DataFrame with H3 indexes:
   latitude  longitude  h3_index
0 -33.8688  151.2093  ...
1 -33.8688  151.2093  ...
2 -33.8688  151.2093  ...
3 -33.8688  151.2093  ...
4 -33.8688  151.2093  ...

Number of unique H3 cells: 1
```

H3 Mapping

The `pyarrow.h3` module offers a-level `h3.latlng_to_cell` layer that greatly reduces complexity in mapping H3 shapes.

```
# Create a new table with the H3 index
df = df.drop(columns=['h3_index'])

# Convert back to pyarrow table
table = pa.Table.from_pandas(df)

# Add the H3 index using pyarrow.h3 functions
table = table.append_column('h3_index', pc.h3.latlng_to_cell(
    table['latitude'],
    table['longitude'],
    resolution=3
)
)

# Verify the new table
print("Table with H3 indexes:")
print(table.select(['latitude', 'longitude', 'h3_index']).to_pandas().head())
```

```
Table with H3 indexes:
   latitude  longitude  h3_index
0 -33.8688  151.2093  ...
1 -33.8688  151.2093  ...
2 -33.8688  151.2093  ...
3 -33.8688  151.2093  ...
4 -33.8688  151.2093  ...
```

Then a-level with a `GeoPolygon`:

```
# Convert back to pandas for easier viewing
df = table.to_pandas()

# Display the first few rows
print("Final DataFrame:")
print(df.head())
```

```
Final DataFrame:
   latitude  longitude  ...  h3_index
0 -33.8688  151.2093  ...  ...
1 -33.8688  151.2093  ...  ...
2 -33.8688  151.2093  ...  ...
3 -33.8688  151.2093  ...  ...
4 -33.8688  151.2093  ...  ...

[5 rows x 7 columns]
```

Australian Marine Regions

Each coordinate has been tagged for the following Australian Marine Regions (see table below). Note that the tagged is not 100% accurate at this region classification. It is accurate to 4km.

Note that the Great Barrier Reef has been excluded from the World Heritage Areas map.

Marine Region Type	**Abbreviation**	**Abbreviation URL (SKOS Concept)**	**Area (km²)**	**UI Colour**
Commonwealth Marine Reserve	Expansion	Marine	Sea (km²)	
Commonwealth Marine Reserve	Existing	Marine	Sea (km²)	
Marine Parks	Expansion	Marine	Sea (km²)	
Marine Parks	Existing	Marine	Sea (km²)	
IMCRA Provincial	Expansion	Marine	Sea (km²)	
IMCRA Mesoscale	Expansion	Marine	Sea (km²)	

Follow the Explore links to explore the Australian Marine Regions. The Commonwealth Marine Reserve explore map has been embedded below.

```
# Create the map centered on Marine Regions
# Using the Commonwealth Marine Reserves
# Note: This is a specific layer
m = folium.Map(location=[-25.0, 135.0], zoom_start=4,
               tiles='https://{s}.tile.openstreetmap.fr/hot/{z}/{x}/{y}.png?lang=en',
               attr='Marine Regions')

# Add the WFS layer for Commonwealth Marine Reserves
wms_layer = folium.WMS(
    url='https://geoserver.imas.utas.edu.au/geoserver/ows?service=WFS&version=1.0.0&request=GetFeature&typeName=CLIMAS:CMR_2021_PROTECTED&outputFormat=application/geojson',
    layers='CLIMAS:CMR_2021_PROTECTED',
    name='Marine Reserves',
    fmt='image/jpeg',
    transparent=True,
    overlay=True,
    control=True,
    version='1.0.0'
)
wms_layer.add_to(m)

# Add layer control
folium.LayerControl().add_to(m)

# Save the map
m.save('marine_regions_map.html')

# Display the map in the notebook
m
```

Exploring Australian Marine Regions Tags

Australian Marine Regions Tags have the following tables:

TAG	TYPE	OPERATIONAL SUPPORT REGIONS	REGION
COMMONWEALTH MARINE RESERVE	SOUTH-EAST	COMMONWEALTH MARINE RESERVE	SOUTH-EAST
NATIONAL TIDAL BASELINE	TIDAL STRATA	NATIONAL TIDAL BASELINE	COASTLINE
IMCRA PROVINCIAL(IMCRA4.0)	TEMP. TEMPERATE	PROVINCIAL(IMCRA4.0)	LOCAL TEMPERATE WATERS

Extracting the available Australian Marine Regions. Tags regions outline the extended coastal.

```
# Exploring available tags
print("Available tags:")
print(df['tag'].value_counts())

# Exploring tag types
print("\nTag types:")
print(df.groupby('tag')['type'].nunique())

# Exploring tag regions
print("\nTag regions:")
print(df.groupby('tag')['region'].nunique())
```

```
Available tags:
tag
COMMONWEALTH MARINE RESERVE    ...
NATIONAL TIDAL BASELINE        ...
IMCRA PROVINCIAL(IMCRA4.0)     ...
Name: count, dtype: int64

Tag types:
tag
COMMONWEALTH MARINE RESERVE    1
NATIONAL TIDAL BASELINE        1
IMCRA PROVINCIAL(IMCRA4.0)     1
Name: type, dtype: int64

Tag regions:
tag
COMMONWEALTH MARINE RESERVE    ...
NATIONAL TIDAL BASELINE        ...
IMCRA PROVINCIAL(IMCRA4.0)     ...
Name: region, dtype: int64
```

Filtering to a Particular Marine Region

We map filter to a particular region using `df[df.isin(['value'])]` expression:

```
# Filter for the Commonwealth Marine Reserves
tags_to_keep = ['COMMONWEALTH MARINE RESERVE']
df = df[df['tag'].isin(tags_to_keep)]
print("\nAfter filtering for Commonwealth Marine Reserves:")
print(df['tag'].value_counts())

# Alternatively, you can also filter by multiple tags
tags_to_keep = ['COMMONWEALTH MARINE RESERVE', 'NATIONAL TIDAL BASELINE']
df = df[df['tag'].isin(tags_to_keep)]
print("\nAfter filtering for multiple tags:")
print(df['tag'].value_counts())
```

```
After filtering for Commonwealth Marine Reserves:
tag
COMMONWEALTH MARINE RESERVE    ...
Name: count, dtype: int64

After filtering for multiple tags:
tag
COMMONWEALTH MARINE RESERVE    ...
NATIONAL TIDAL BASELINE        ...
Name: count, dtype: int64
```

Mapping the Spatial Extent of a Particular Dataset

Similar to mapping Australian Marine Regions, we may map particular datasets by applying a string contraint.

Finding Australian Datasets

We may find available datasets by aggregating by the `datasetName` column.

```
# Count unique datasets
print("Available datasets:")
print(df['datasetName'].value_counts(dropna=False))
```

```
Available datasets:
datasetName
Tasmania Mapping Synthesis: A resource for coastal management in the Great Barrier Reef (NERP TWQ Hub)    ...
Tasmanian Seagrass Survey, Tasmania, Tasmania, Tasmania
```

```
)

# Add years to the tooltip
tooltip = util.TOOLTIP.copy()
tooltip["html"] = tooltip["html"].replace(
    "</div>",
    "<br/><b>Years:</b></br>{years}</div>",
)


# Run the map
deck = pydeck.Deck(
    layers=h3_layers,
    map_style=pydeck.map_styles.LIGHT_NO_LABELS,
    tooltip=tooltip,
    initial_view_state=util.generate_view_state(df)
)
deck.show()
```

Out[12]:

Seabird

Thomas Galindo, Denisse Fierro Arcos

2026-02-18

- Background
  - Relevant packages
  - Accessing NESP datasets
    - Loading dataset as data frame
  - Observation Effort, Organism Quantity and Richness
  - Visualising seagrass dataset
  - Data aggregation and mapping with `H3`
    - `H3` indexing
  - Australian Marine Regions
  - Exploring Available Marine Regions Tags
  - Filtering to a Particular Marine Region
  - Identifying suitable trend sites
    - Per Region
    - Per Dataset
    - Mapping the Spatial Extent of a Particular Dataset
    - Finding Available Datasets
  - Wind Farm Regions
    - H3 Index Joins

Background

This R notebook outlines the packages needed to access and analyse NESP Data Products, as well as how to check their content structure. This notebook was originally written in Python by Thomas Galindo. It was "translated" into R by Denisse Fierro Arcos. It should be noted that although both notebooks cover the same topic, the content may vary between these notebooks as features may not be available in both Python and R.

This notebook includes the following sections:
- Relevant R packages needed to access NESP Data Products - How to access NESP Data Products - How aggregate and map NESP datasets using the H3 package - Identifying suitable sampling sites where trend analysis can be carried out

Note: The `H3` package was originally developed in C, but there are at least three packages that make this package accessible in R. We recommend you install the `h3-r` package, which published in GitHub (https://github.com/crazycapivara/h3-r/) because this package will automatically install the original `H3` package written in C (unlike the other two options available as of February 2026). More information about the `h3-r` package, as well as a short tutorial can be found in their documentation (https://crazycapivara.github.io/h3-r/articles/h3.html).

Relevant packages

```
# Connects to datasets remotely
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.5.2
```

```
##
## Attaching package: 'arrow'
```

```
## The following object is masked from 'package:utils':
##
##     timestamp
```

```
# Indexing system that uses hexagons
library(h3)
# Represents simple spatial features
library(sf)
```

```
## Linking to GEOS 3.13.1, GDAL 3.11.0, PROJ 9.6.0; sf_use_s2() is TRUE
```

```
# Data manipulation
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:arrow':
##
##     duration
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(tidyr)
library(stringr)
# Plotting
library(ggplot2)
# Interactive maps
library(leaflet)
```

```
## Warning: package 'leaflet' was built under R version 4.5.2
```

```
library(htmlwidgets)
library(webshot)
```

```
## Warning: package 'webshot' was built under R version 4.5.2
```

Accessing NESP datasets

NESP 5.9 datasets can be accessed via the `arrow` package for R. First, we will need to create a connection with the parent folder containing NESP data as follows:

```
conn <- s3_bucket("s3://data-uplift-public/", anonymous = TRUE,
                  region = "ap-southeast-2")
```

Once a connection has been established, then we can select a specific dataset that interest us. In this case, we will be accessing a dataset about seagrasses. By using the `open_dataset` function from `arrow` we can load either the dataset on its entirety or a subset.

```
url <- "s3://data-uplift-public/stored/datauplift/seabird/seabird.parquet"
ds <- open_dataset(sources = url, filesystem = conn)
ds
```

```
## FileSystemDataset with 1 Parquet file
## 56 columns
## resource_link: string not null
## download_link: string not null
## datasetName: string not null
## datasetVersion: string not null
## id: string not null
## modified: timestamp[ms] not null
## bibliographicCitation: string
## institutionCode: string
## collectionCode: string
## basisOfRecord: string not null
## occurrenceID: string not null
## catalogNumber: string
## recordedBy: string
## organismQuantity: string
## organismQuantityType: string
## sex: string
## lifeStage: string
## occurrenceStatus: string
## occurrenceRemarks: string
## organismID: string
## ...
## 36 more columns
## Use `schema()` to see entire schema
##
## See $metadata for additional Schema metadata
```

We can check the file structure (i.e., schema) of the seabird dataset. This will give us the name of the columns contained in the dataset, which can help us filter the data before accessing.

```
ds$schema
```

```
## Schema
## resource_link: string not null
## download_link: string not null
## datasetName: string not null
## datasetVersion: string not null
## id: string not null
## modified: timestamp[ms] not null
## bibliographicCitation: string
## institutionCode: string
## collectionCode: string
## basisOfRecord: string not null
## occurrenceID: string not null
## catalogNumber: string
## recordedBy: string
## organismQuantity: string
## organismQuantityType: string
## sex: string
## lifeStage: string
## occurrenceStatus: string
## occurrenceRemarks: string
## organismID: string
## fieldNumber: string
## eventDate: date32[day] not null
## country: string
## stateProvince: string
## locality: string
## minimumDepthInMeters: int64
## maximumDepthInMeters: int64
## decimalLatitude: double not null
## decimalLongitude: double not null
## coordinateUncertaintyInMeters: int64
## footprintWKT: string
## identifiedBy: string
## scientificNameID: string not null
## scientificName: string not null
## valid_authority: string
## kingdom: string
## phylum: string
## class: string
## order: string
## family: string
## genus: string
## specificEpithet: string
## taxonRank: string
## scientificNameAuthorship: string
## vernacularName: string
## surveyType: string not null
## windDirection: double
## airTemperature: double
## windSpeed: double
## depth: double
## airPressure: double
## seaState: string
## salinity: double
## cloudCover: string
## h3Index: string not null
## australianMarineRegionsTags: string
##
## See $metadata for additional Schema metadata
```

From here, we can see there are 56 columns, but we can check how many records in total are included in the dataset as follows:

```
dim(ds)
```

```
## [1] 122396     56
```

This dataset has over 120,000 records, which may be too much for some computers to handle, but `arrow` allows us to lazily load the dataset. This means that the data is only load into your memory when you request it, instead of loading all records at once. This could be especially useful if you have limited computing resources, so you can subset or summarise the data as you need before loading it to your computer to avoid overloading the computer's memory.

Loading dataset as data frame

If the dataset you are interested in is relatively small, we recommend you simply load it into memory usin the `collect` function from `arrow`.

```
# Note that running this code may take a while, so only run it if you have enough memory available
df <- ds |>
  collect()

# We can check the first few rows in this dataset
head(df)
```

```
## # A tibble: 6 × 56
##   resource_link                   download_link datasetName datasetVersion id
##   <chr>                           <chr>         <chr>       <chr>          <chr>
## 1 https://www.marine.csiro.au/ip… https://www.… aae_201314… 1.6            AAE-…
## 2 https://www.marine.csiro.au/ip… https://www.… aae_201314… 1.6            AAE-…
## 3 https://www.marine.csiro.au/ip… https://www.… aae_201314… 1.6            AAE-…
## 4 https://www.marine.csiro.au/ip… https://www.… aae_201314… 1.6            AAE-…
## 5 https://www.marine.csiro.au/ip… https://www.… aae_201314… 1.6            AAE-…
## 6 https://www.marine.csiro.au/ip… https://www.… aae_201314… 1.6            AAE-…
## # ℹ 51 more variables: modified <dttm>, bibliographicCitation <chr>,
## #   institutionCode <chr>, collectionCode <chr>, basisOfRecord <chr>,
## #   occurrenceID <chr>, catalogNumber <chr>, recordedBy <chr>,
## #   organismQuantity <chr>, organismQuantityType <chr>, sex <chr>,
## #   lifeStage <chr>, occurrenceStatus <chr>, occurrenceRemarks <chr>,
## #   organismID <chr>, fieldNumber <chr>, eventDate <date>, country <chr>,
## #   stateProvince <chr>, locality <chr>, minimumDepthInMeters <int>, …
```

Observation Effort, Organism Quantity and Richness

Observation Effort is the number of occurrence records (e.g., sampling campaigns), while **Organism Quantity** is the number of organisms of a species recorded at each occurrence.

Observation Effort must be taken into account when analysing trends; fluctuations in observation effort tend to distort the reported occurrences.

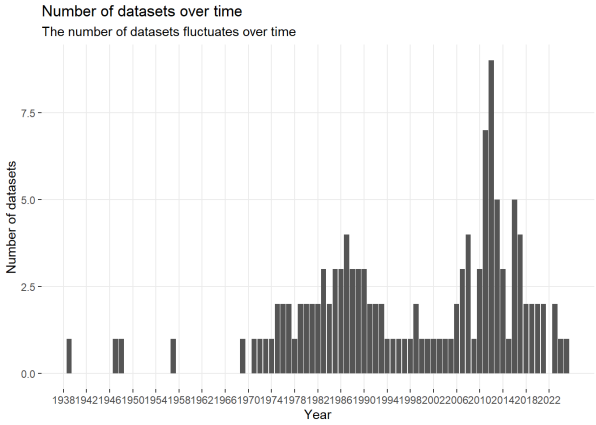
Species **Richness** refers to the number of different species present in a particular area. While **Organism Quantity** tells us "how many" individuals there are, **Richness** tells us "how many types" of organisms coexist.

Typically the **Richness** of an area correlates to the number of studies carried out in a given area, but eventually plateaus and this plateau is an indication that the studied region is well sampled.

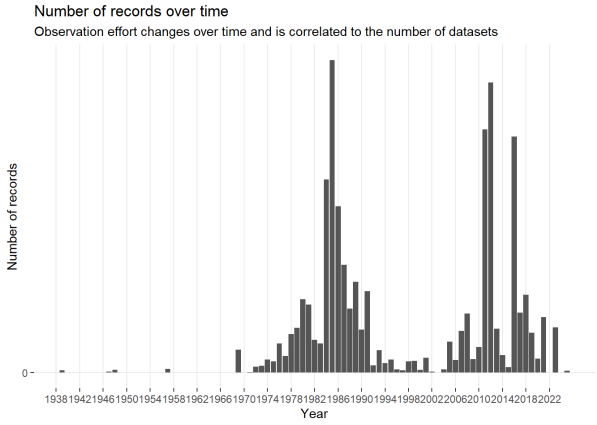
Visualising seagrass dataset

We can create a few plots to better understand the structure of this dataset.

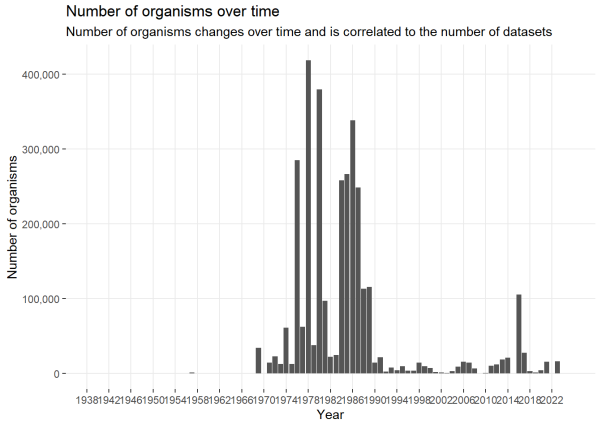
```
# Number of unique datasets per year - We will add a "year" column to the
# dataset
df <- df |>
  mutate(year = year(eventDate))

df |>
  distinct(year, datasetName) |>
  group_by(year) |>
  count() |>
  ggplot(aes(year, n))+
  geom_bar(stat = "identity")+
  scale_x_continuous(breaks = seq(min(df$year, na.rm = T)-1,
                                  max(df$year, na.rm = T), by = 4))+
  theme_bw()+
  labs(x = "Year", y = "Number of datasets",
       title = "Number of datasets over time",
       subtitle = "The number of datasets fluctuates over time")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```



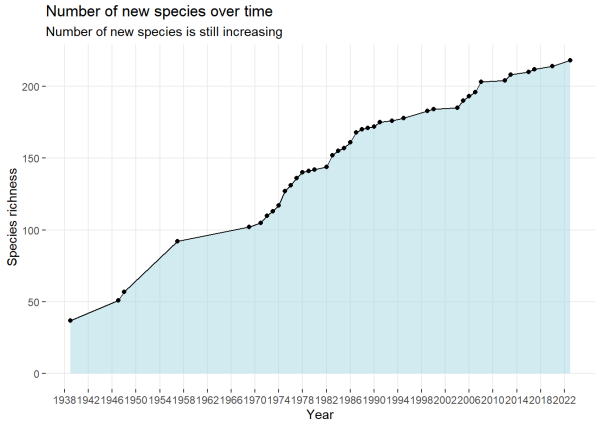

```
# Number of records per year
df |>
  group_by(year) |>
  count() |>
  ggplot(aes(year, n))+
  geom_bar(stat = "identity")+
  scale_x_continuous(breaks = seq(min(df$year, na.rm = T)-1,
                                  max(df$year, na.rm = T), by = 4))+
  scale_y_continuous(labels = scales::label_comma(),
                     breaks = seq(0, 250000, by = 50000))+
  theme_bw()+
  labs(x = "Year", y = "Number of records",
       title = "Number of records over time",
       subtitle = "Observation effort changes over time and is correlated to the number of datasets")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```




```
# Number of observations per year
df |>
  select(year, occurrenceStatus, organismQuantity) |>
  mutate(organismQuantity = case_when(organismQuantity == "present" ~ "1",
                                      T ~ organismQuantity),
         organismQuantity = as.numeric(organismQuantity)) |>
  group_by(year) |>
  summarise(tot_obs = sum(organismQuantity, na.rm = T)) |>
  ggplot(aes(year, tot_obs))+
  geom_bar(stat = "identity")+
  scale_x_continuous(breaks = seq(min(df$year, na.rm = T)-1,
                                  max(df$year, na.rm = T), by = 4))+
  scale_y_continuous(labels = scales::label_comma())+
  theme_bw()+
  labs(x = "Year", y = "Number of organisms",
       title = "Number of organisms over time",
       subtitle = "Number of organisms changes over time and is correlated to the number of datasets")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```




```
# Number of unique new species per year
df |>
  distinct(year, scientificNameID) |>
  arrange(year) |>
  distinct(scientificNameID, .keep_all = T) |>
  group_by(year) |>
  count() |>
  ungroup() |>
  mutate(tot_sp = cumsum(n)) |>
  ggplot(aes(year, tot_sp))+
  geom_density(stat = "identity", fill = "lightblue", alpha = 0.5)+
  geom_point()+
  scale_x_continuous(breaks = seq(min(df$year, na.rm = T)-1,
                                  max(df$year, na.rm = T), by = 4))+
  theme_bw()+
  labs(x = "Year", y = "Species richness",
       title = "Number of new species over time",
       subtitle = "Number of new species is still increasing")+
  theme(panel.grid.minor = element_blank(), panel.border = element_blank())
```

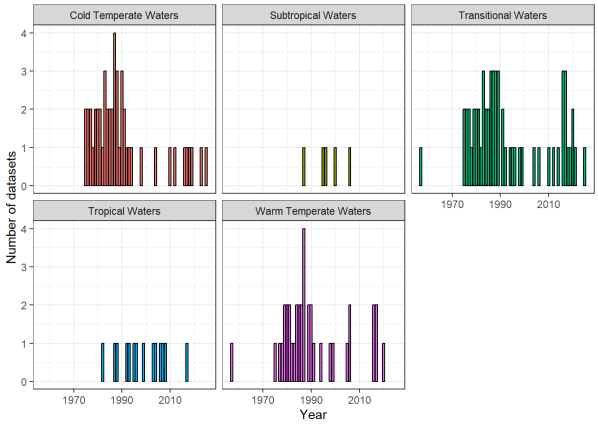


We can also explore the information in this table further, by extracting information about the “Water Type” of the sites sampled.

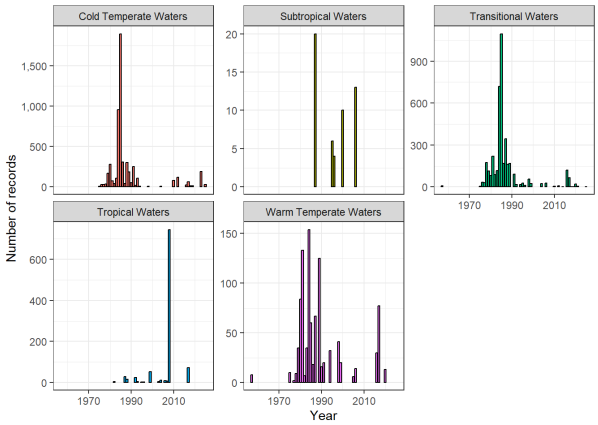
```
df_water_type <- df |>
  mutate(short = str_extract(australianMarineRegionsTags,
                             "\\|(IMCRA:Mesoscale:Water Type.*Waters)\\|",
                             group = 1),
         watertype = str_extract(str_split_i(short, "\\|", i = 1), "Type:(.*)",
                             group = 1)) |>
  select(!short) |>
  drop_na(watertype)
```

Now we can check the number of unique datasets available over time per “Water Type”.

```
df_water_type |>
  distinct(year, datasetName, watertype) |>
  group_by(year, watertype) |>
  count() |>
  ggplot(aes(year, n))+
  geom_bar(stat = "identity", aes(fill = watertype), colour = "black")+
  facet_wrap(~watertype)+
  labs(y = "Number of datasets", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```

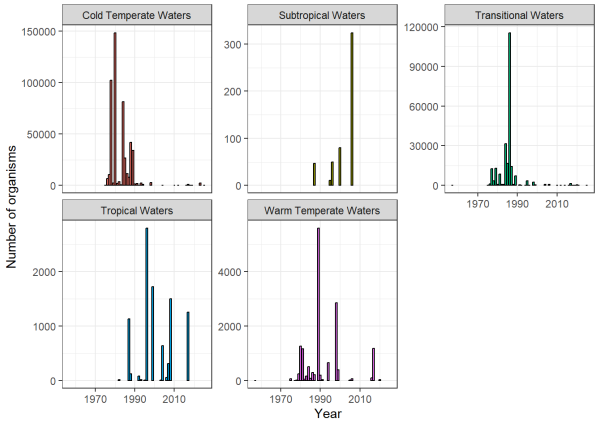



```
df_water_type |>
  group_by(year, watertype) |>
  count() |>
  ggplot(aes(year, n))+
  geom_bar(stat = "identity", aes(fill = watertype), colour = "black")+
  facet_wrap(~watertype, scales = "free_y")+
  scale_y_continuous(labels = scales::label_comma())+
  labs(y = "Number of records", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```

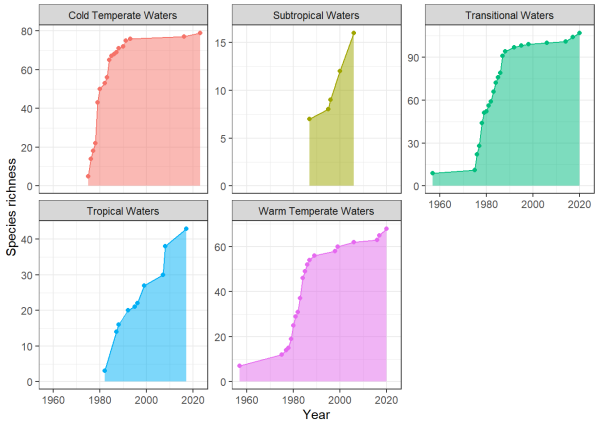



```
df_water_type |>
  mutate(organismQuantity = case_when(organismQuantity == "present" ~ "1",
                                      T ~ organismQuantity),
         organismQuantity = as.numeric(organismQuantity)) |>
  group_by(year, watertype) |>
  summarise(tot_obs = sum(organismQuantity, na.rm = T)) |>
  ggplot(aes(year, tot_obs))+
  geom_bar(stat = "identity", aes(fill = watertype), colour = "black")+
  facet_wrap(~watertype, scales = "free_y")+
  labs(y = "Number of organisms", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```




```
df_water_type |>
  distinct(year, watertype, scientificNameID) |>
  arrange(year) |>
  distinct(scientificNameID, watertype, .keep_all = T) |>
  group_by(year, watertype) |>
  count() |>
  group_by(watertype) |>
  mutate(tot_sp = cumsum(n)) |>
  ggplot(aes(year, tot_sp, fill = watertype, colour = watertype))+
  geom_density(stat = "identity", alpha = 0.5)+
  geom_point()+
  facet_wrap(~watertype, scales = "free_y")+
  labs(y = "Species richness", x = "Year")+
  theme_bw()+
  theme(legend.position = "none")
```



Data aggregation and mapping with `H3`

The NESP 5.9 datasets are aggregated data products that cover large regions and temporal periods. This aggregation is useful as it helps us understand the spatio-temporal bounds of our dataset.
We use the `H3` package in this notebook because it is an open source package that partitions the world into hexagonal cells, allowing for faster data access.

`H3` indexing

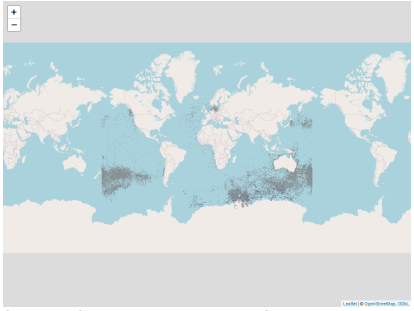
The `H3` package allows us to spatially index datasets at different resolutions (https://h3geo.org/docs/core-library/restable/#average-area-in-km2) (i.e., zoom levels) relative to the recorded latitude and longitude. If you go through the dataset schema, you will notice that it has an `h3Index` column, which contains unique identifiers for each hexagon in our dataset.

```
# Getting number of records for each polygon ID (h3Index)
h3_layers <- df |>
  group_by(h3Index) |>
  summarise(number_records = n())

# Now we get information about datasets associated to each polygon ID
h3_layers <- df |>
  distinct(h3Index, datasetName) |>
  group_by(h3Index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(h3_layers, by = "h3Index") |>
  ungroup()

h3_cross <- h3_to_geo_boundary_sf(h3_layers$h3Index) |>
  st_wrap_dateline()

h3_layers <- h3_layers |>
  mutate(index = h3_cross)

# We can check the results
head(h3_layers)
```

```
## # A tibble: 6 × 4
##   h3Index         datasets                      number_records index$h3_index
##   <chr>           <chr>                                  <int> <chr>
## 1 8408967ffffffff arctic_tern_tracking_2008_2015             1 8408967ffffffff
## 2 8408b01ffffffff arctic_tern_tracking_2008_2015             1 8408b01ffffffff
## 3 8408b07ffffffff arctic_tern_tracking_2008_2015             1 8408b07ffffffff
## 4 8408b0dffffffff arctic_tern_tracking_2008_2015             1 8408b0dffffffff
## 5 8408b21ffffffff arctic_tern_tracking_2008_2015             1 8408b21ffffffff
## 6 8408b27ffffffff arctic_tern_tracking_2008_2015             4 8408b27ffffffff
## # ℹ 1 more variable: index$geometry <POLYGON [°]>
```

We can now create an interactive map as shown below. Note that you can interact with the map, that is zoom in and out, and hover over each title to get information about the number of records and datasets available at each polygon.

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_records),
                                         n = 10)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets:", h3_layers$datasets),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.

Australian Marine Regions

Each occurrence in the seabird dataset that we have been exploring so far has been classified using the following Australian Marine Regions (see table below). Note that the classification is not 100% accurate at the region boundaries; it is accurate to ~6km.

Note that the Great Barrier Reef has been included from the World Heritage Areas map.

Marine Region Type	Map Explore URL	Metadata URL (REST/About)
Commonwealth Marine Regions	Explore (https://fed.dcceew.gov.au/datasets/commonwealth-marine-regions/explore)	Metadata (https://gis.environment.gov.au/gispubmap/rest/services/ogc_services/Commonwealth_Marine_R
World Heritage Areas	Explore (https://fed.dcceew.gov.au/datasets/erin::australia-world-heritage-areas/explore)	Metadata (https://gis.environment.gov.au/gispubmap/rest/services/ogc_services/World_Heritage_
Marine Parks	Explore (https://fed.dcceew.gov.au/datasets/2b3eb1d42b8d4319900cf4777f0a83b9_0/explore)	Metadata (https://gis.environment.gov.au/gispubmap/rest/services/ogc_services/Australian_Marine_Parks/l
IMCRA Provincial	Explore (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation-of-australia-imcra-v4-0-provincial-bioregions/explore)	Metadata (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation v4-0-provincial-bioregions/about)
IMCRA Mesoscale	Explore (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation-of-australia-imcra-v4-0-meso-scale-bioregions/explore)	Metadata (https://fed.dcceew.gov.au/datasets/erin::integrated-marine-and-coastal-regionalisation v4-0-meso-scale-bioregions/about)

Follow the Explore links to explore the Australian Marine Regions. The Commonwealth Marine Regions can be seen in the following link: https://fed.dcceew.gov.au/datasets/commonwealth-marine-regions/explore?location=-33.665700%2C140.517300%2C4 (https://fed.dcceew.gov.au/datasets/commonwealth-marine-regions/explore?location=-33.665700%2C140.517300%2C4)

Exploring Available Marine Regions Tags

Australian Marine Regions Tags have the following syntax:

TAG	TYPE	OPTIONAL SUPER REGIONS	REGION
`Commonwealth Marine Region: South-east`	`Commonwealth Marine Region`		`South-east`
`Marine Park:National Park Zone:Freycinet`	`Marine Park`	`National Park Zone`	`Freycinet`
`IMCRA:Provincial:Water Type:Cold Temperate Waters`	`IMCRA`	`Provincial , Water Type`	`Cold Temperate Waters`

Extracting the available Australian Marine Regions Tags requires splitting the delimited column:

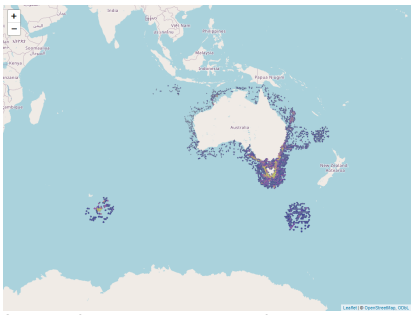
```
australian_marine_regions_tags <- df |>
  select(australianMarineRegionsTags) |>
  drop_na() |>
  separate_longer_delim(australianMarineRegionsTags, delim = "|") |>
  group_by(australianMarineRegionsTags) |>
  summarise(count = n()) |>
  arrange(desc(count))

# Printing the first 10 tags
head(australian_marine_regions_tags, n = 10)
```

```
## # A tibble: 10 × 2
##    australianMarineRegionsTags                       count
##    <chr>                                             <int>
##  1 Commonwealth Marine Region:South-east             33615
##  2 IMCRA:Provincial:Water Type:Cold Temperate Waters 20013
##  3 IMCRA:Provincial:Water Type:Transitional Waters   18014
##  4 IMCRA:Provincial:Name:Tasmania Province           12478
##  5 IMCRA:Mesoscale:Water Type:Cold Temperate Waters   8949
##  6 IMCRA:Provincial:Name:Tasmanian Shelf Province     7105
##  7 IMCRA:Provincial:Water Type:Warm Temperate Waters  6745
##  8 IMCRA:Mesoscale:Water Type:Transitional Waters     6602
##  9 Commonwealth Marine Region:Temperate East          5955
## 10 IMCRA:Provincial:Name:Southeast Transition         5938
```

```
# Getting number of records for each polygon ID (h3Index)
marine_reg <- df |>
  drop_na(australianMarineRegionsTags) |>
  group_by(h3Index) |>
  summarise(number_records = n())

# Now we get information about datasets associated to each polygon ID
marine_reg <- df |>
  drop_na(australianMarineRegionsTags) |>
  distinct(h3Index, datasetName) |>
  group_by(h3Index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(marine_reg, by = "h3Index")

# Finally we get information about the marine tags
marine_reg <- df |>
  select(h3Index, australianMarineRegionsTags) |>
  drop_na(australianMarineRegionsTags) |>
  distinct() |>
  separate_longer_delim(australianMarineRegionsTags, delim = "|") |>
  distinct() |>
  group_by(h3Index) |>
  summarise(tags = paste(australianMarineRegionsTags, collapse = ", ")) |>
  right_join(marine_reg, by = "h3Index") |>
  mutate(index = h3_to_geo_boundary_sf(marine_reg$h3Index)) |>
  ungroup()

# Checking results
head(marine_reg)
```

```
## # A tibble: 6 × 5
##   h3Index         tags                    datasets number_records index$h3_index
##   <chr>           <chr>                   <chr>            <int> <chr>
## 1 848c047fffffff IMCRA:Provincial:Water… seabird…             16 848c047ffffff…
## 2 848c0cbfffffff IMCRA:Provincial:Name:… seabird…              3 848c0cbffffff…
## 3 848c201fffffff IMCRA:Provincial:Water… seabird…              1 848c201ffffff…
## 4 848c203fffffff Marine Park:National P… seabird…              8 848c203ffffff…
## 5 848c207fffffff IMCRA:Provincial:Water… seabird…              4 848c207ffffff…
## 6 848c209fffffff Marine Park:National P… seabird…              1 848c209ffffff…
## # ℹ 1 more variable: index$geometry <POLYGON [°]>
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = marine_reg$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(marine_reg$number_records),
                                         n = 10)(marine_reg$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", marine_reg$number_records,
                             "\nDatasets: ", marine_reg$datasets,
                             "\nTags: ", marine_reg$tags),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



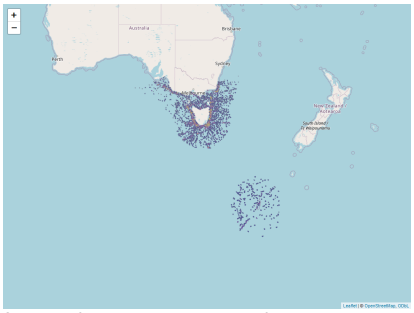
Screenshot of the map above generated by leaflet.

Filtering to a Particular Marine Region

We could also create a map after selecting (filtering) for a particular tag as follows:

```
south_east_commonwealth_marine_region_df <- df |>
  select(!h3Index) |>
  filter(str_detect(australianMarineRegionsTags,
                    "Commonwealth Marine Region:South-east")) |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)

south_east_commonwealth_marine_region_df <-
  south_east_commonwealth_marine_region_df  |>
  mutate(h3index = geo_to_h3(south_east_commonwealth_marine_region_df , 5))

# Getting number of records for each polygon ID (h3Index)
h3_layers <- south_east_commonwealth_marine_region_df  |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

h3_years <- south_east_commonwealth_marine_region_df |>
  st_drop_geometry() |>
  distinct(h3index, year) |>
  group_by(h3index) |>
  summarise(years = paste(year, collapse = ", "))

# Now we get information about datasets associated to each polygon ID
h3_layers <- south_east_commonwealth_marine_region_df  |>
  st_drop_geometry() |>
  distinct(h3index, datasetName) |>
  group_by(h3index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(h3_layers, by = "h3index") |>
  right_join(h3_years, by = "h3index") |>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3index)) |>
  ungroup()
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_records),
                                         n = 10)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets: ", h3_layers$datasets,
                             "\nYears: ", h3_layers$years),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.

Identifying suitable trend sites

In the original map that includes all data, we can see there are some sites with a high density of records, while others have very limited observations. The high density polygons are the most ideal sites for trend analysis.

Per Region

Regions with a high density of records are typically good candidates for trend analysis. See above for subsetting and plotting by Australian Marine Regions.

Per Dataset

Additionally, the singular datasets that comprise the aggregated data product are typically good candidates for trend analysis.

Mapping the Spatial Extent of a Particular Dataset

Similarly to mapping Australian Marine Regions, we may map particular datasets by applying a string constraint.

Finding Available Datasets

We may find available datasets by aggregating by the `datasetName` column.

```
data_count <- df |>
  group_by(datasetName) |>
  summarise(count = n()) |>
  arrange(count)

data_count
```

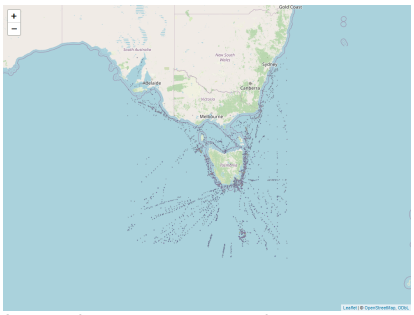
```
## # A tibble: 39 × 2
##    datasetName           count
##    <chr>                 <int>
##  1 in2024_v04_wov            4
##  2 balleny_wildlife_2014    31
##  3 seabirds_coral_2023      63
##  4 so198502_wov             66
##  5 in2025_v01_wov           93
##  6 in2016_v04_wov          108
##  7 eades_macca_seabirds    154
##  8 seabird_log_1957        160
##  9 seabirds_ejw_1983       217
## 10 fr198702_wov            230
## # ℹ 29 more rows
```

```
data_int <- data_count |>
  filter(count == max(count)) |>
  pull(datasetName)

#Check name of dataset with the most observations
data_int
```

```
## [1] "seabird_atlas_southeast_australia"
```

```
seabird_atlas_sea_df <- df |>
  select(!h3Index) |>
  filter(datasetName == data_int) |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)

seabird_atlas_sea_df  <- seabird_atlas_sea_df  |>
  mutate(h3index = geo_to_h3(seabird_atlas_sea_df, 6))

# Getting number of records for each polygon ID (h3Index)
h3_layers <- seabird_atlas_sea_df  |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

# Now we get information about datasets associated to each polygon ID
h3_layers <- seabird_atlas_sea_df  |>
  st_drop_geometry() |>
  distinct(h3index, datasetName) |>
  group_by(h3index) |>
  summarise(datasets = paste(datasetName, collapse = ", ")) |>
  right_join(h3_layers, by = "h3index") |>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3index)) |>
  ungroup()
```

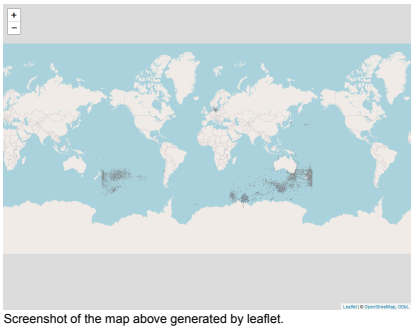
```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_records),
                                         n = 10)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets: ", h3_layers$datasets),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.

```
df_h3 <- df |>
  select(!h3Index) |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)

df_h3 <- df_h3|>
  mutate(h3index = geo_to_h3(df_h3, 4))

# Getting number of records for each polygon ID (h3Index)
n_rec <- df_h3 |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

n_dat <- df_h3 |>
  st_drop_geometry() |>
  distinct(h3index, datasetName) |>
  group_by(h3index) |>
  summarise(number_datasets = n(),
            datasets = paste(datasetName, collapse = ", "))

n_yr <- df_h3 |>
  st_drop_geometry() |>
  distinct(h3index, year) |>
  group_by(h3index) |>
  summarise(number_years = n(),
            years = paste(year, collapse = "|"))

# Now we get information about datasets associated to each polygon ID
h3_layers <- n_rec |>
  right_join(n_yr, by = "h3index") |>
  right_join(n_dat, by = "h3index") |>
  filter(number_years >= 2)

h3_cross <- h3_to_geo_boundary_sf(h3_layers$h3index) |>
  st_wrap_dateline()

h3_layers <- h3_layers |>
  mutate(index = h3_cross)
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygons with
  # a higher number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_years),
                                         n = 5)(h3_layers$number_years),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets: ", h3_layers$datasets,
                             "\nYears: ", h3_layers$years),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.

Wind Farm Regions

Bird and vessel traffic are of significant interest to offshore wind farms; generally, it is desirable to minimise the impact of offshore wind farms on human and wildlife activities.

You may explore more details on the designated wind farm areas here (https://www.dcceew.gov.au/energy/renewable/offshore-wind/areas/gippsland#toc_3).

H3 Index Joins

Additional to pre-processing, H3 indexes may be applied to new geometries.

Note that re-indexing to the same H3 resolution may be required (and is so in the following case).

```r
# Check if shapefile has been uncompressed. If not, do so.
if(!file.exists("OffshoreRenewable_Energy_Infrastructure_Regions.shp")){
  unzip("OffshoreRenewable_Energy_Infrastructure_Regions.zip")
}

gdf <- read_sf("Offshore_Renewable_Energy_Infrastructure_Regions.shp")
```

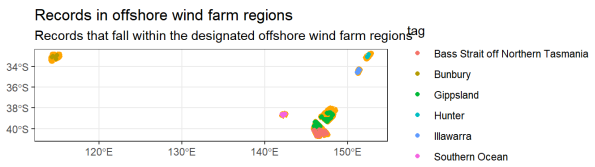
```r
offshore_wind_gdf <- gdf |>
  filter(Status == "Declared") |>
  select(Region, geometry) |>
  rename(tag = Region) |>
  group_by(tag) |>
  summarise(geometry = st_union(geometry)) |>
  ungroup() |>
  st_transform(crs = 4326)
offshore_wind_gdf
```

```
## Simple feature collection with 6 features and 1 field
## Geometry type: GEOMETRY
## Dimension:     XY
## Bounding box:  xmin: 114.1909 ymin: -40.80711 xmax: 152.8645 ymax: -32.70123
## Geodetic CRS:  WGS 84
## # A tibble: 6 × 2
##   tag                                                                  geometry
## * <chr>                                                            <GEOMETRY [°]>
## 1 Bass Strait off Northern Tasmania MULTIPOLYGON (((146.9727 -40.7004, 147.1299…
## 2 Bunbury                           MULTIPOLYGON (((114.9591 -32.79194, 114.893…
## 3 Gippsland                         MULTIPOLYGON (((148.4013 -38.0818, 147.9416…
## 4 Hunter                            POLYGON ((152.3021 -32.93767, 152.2995 -32..…
## 5 Illawarra                         POLYGON ((151.1339 -34.60615, 151.1872 -34..…
## 6 Southern Ocean                    POLYGON ((142.3769 -38.84842, 142.5748 -38..…
```

```r
offshore_wind_df <- df |>
  filter(str_detect(australianMarineRegionsTags, "South-east|Temperate East|South-west")) |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)
offshore_wind_df
```

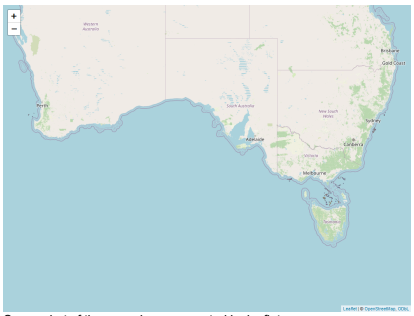
```
## Simple feature collection with 40688 features and 55 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 111.0667 ymin: -58.4 xmax: 171.5 ymax: -24.4742
## Geodetic CRS:  WGS 84
## # A tibble: 40,688 × 56
##    resource_link                  download_link datasetName datasetVersion id
##  * <chr>                          <chr>         <chr>       <chr>          <chr>
##  1 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  2 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  3 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  4 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  5 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  6 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  7 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  8 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
##  9 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
## 10 https://www.marine.csiro.au/i… https://www.… aae_201314… 1.6            AAE-…
## # ℹ 40,678 more rows
## # ℹ 51 more variables: modified <dttm>, bibliographicCitation <chr>,
## #   institutionCode <chr>, collectionCode <chr>, basisOfRecord <chr>,
## #   occurrenceID <chr>, catalogNumber <chr>, recordedBy <chr>,
## #   organismQuantity <chr>, organismQuantityType <chr>, sex <chr>,
## #   lifeStage <chr>, occurrenceStatus <chr>, occurrenceRemarks <chr>,
## #   organismID <chr>, fieldNumber <chr>, eventDate <date>, country <chr>, …
```

```r
overlapping_pts <- st_join(offshore_wind_df, offshore_wind_gdf) |>
   filter(!is.na(tag))

ggplot()+
  geom_sf(data = offshore_wind_gdf, fill = "lightblue", colour = "orange",
          linewidth = 2)+
  geom_sf(data = overlapping_pts, aes(colour = tag))+
  theme_bw()+
  labs(title = "Records in offshore wind farm regions",
       subtitle = "Records that fall within the designated offshore wind farm regions")
```



```
overlapping_pts <- overlapping_pts |>
  mutate(h3index = geo_to_h3(overlapping_pts, 6)) |>
  select(!h3Index)

n_recs <- overlapping_pts |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

n_dat <- overlapping_pts |>
  st_drop_geometry() |>
  distinct(h3index, datasetName) |>
  group_by(h3index) |>
  summarise(number_datasets = n(),
            datasets = paste(datasetName, collapse = ", "))

h3_layers <- overlapping_pts |>
  right_join(n_dat, by = "h3index") |>
  right_join(n_recs, by = "h3index")

h3_layers <- h3_layers |>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3index)) |>
  ungroup()
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygons with
  # a higher number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_records),
                                         n = 5)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records,
                             "\nDatasets: ", h3_layers$datasets),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.

NESP 5.9 Seabird Notebook

Author	Thomas Mustine
Updated	2020-06-30
attributes	AUTHOR,NESP
e-mail	thomas.mustine@adelaide.edu.au
Date of creation	2019-10-30
Date of last update	2020-06-30

Background

This is a reproducible notebook collecting the library stack and standard workflow for NESP Data Products.

It contains:

- Connecting to NESP 5.9 datasets
- 5.9 Aggregations and Mapping
- Identifying Trend Sites

NESP 5.9 Datasets

Dataset name	Description	Extensions	CC-BY
Seabird	Seabird Survey Tracking	.csv	No
Dolphin	Dolphin Sightings	.csv	No
Turtle	Turtle Sightings	.csv	No
Dugong	Dugong Sightings	.csv	No
Seagrass	Seagrass Transect Data Tracking	.csv	No
Seagrass	Seagrass Surveys	.csv	No

5.9 Details

NESP 5.9 datasets are currently stored in S3 and are publically available as per the following table:

Name	Path
dolphin-sp2019-v20210	s3://dev-data-lake/data/5.9/dolphin/dolphin-sp2019-v20210.parquet/
dolphin-sp2019-v20210	s3://dev-data-lake/data/5.9/dolphin/dolphin-sp2019-v20210.parquet/
dugong-sp2019-v20210	s3://dev-data-lake/data/5.9/dugong/dugong-sp2019-v20210.parquet/
dugong-sp2019-v20210	s3://dev-data-lake/data/5.9/dugong/dugong-sp2019-v20210.parquet/
seabird-sp2019-v20210	s3://dev-data-lake/data/5.9/seabird/seabird-sp2019-v20210.parquet/
seabird-sp2019-v20210	s3://dev-data-lake/data/5.9/seabird/seabird-sp2019-v20210.parquet/

Required Python & Packages

The required packages are found in the requirejs.txt file:

1) Installation

It is highly recommended to manage python versions, environment and projects with pyenv.

```
pyenv install 3.6.8
pyenv virtualenv 3.6.8 nesp5.9-3.6.8
pyenv local nesp5.9-3.6.8
```

2) Installation

Otherwise, install python binary (as per above) to manage the python environment and project.

Note that pandoc 2.0 or greater is required.

```
python -m venv .venv ; source .venv/bin/activate ; pip install -r
```

Running Notebooks in Jupyter or IDE

The notebooks can be executed via the kernel/environment created from the commands:

```
jupyter notebook
ipykernel install --name=nesp5.9-3.6.8 --display-name="nesp5.9-3.6.8"
```

Connecting to NESP 5.9 Datasets

NESP 5.9 Datasets can be connected to via a generic S3FS-DATASTORE.

S3FS-DATASTORE manages the connections to the NESP 5.9 datasets (stored in S3), mounting in a local DATA-DIRECTORY. Symlinks then may be used to point to specific versions of datasets.

Note that it is advised that no consideration of a large prepared file is performed (into separate parquet files/partitions).

This step also creates the relevant environment variables which other processes may use (filtering without downloading the entire dataset).

Finally, one may optionally mirror a remote dataset out by navigating into the dataset's bin/ and running the command: mirror-data-from-s3. Alternatively, one may also upload changes back to the remote dataset with the command: mirror-data-to-s3 (assuming one has write permissions).

Notebook Stack

IO

DATATABLE is the defacto parquet and arrow standard library binding in the python arrow ecosystem, offering exceptional performance, documentation and stability.

PYARROW is the official library for the parquet/arrow specification, maintained by the Apache Foundation.

Computation

DASK is a native threading by default, distributed library with a pandas like API. It has been highly integrated with arrow (and parquet), has exceptional performance, greater than memory dataset manipulation, a more consistent and declarative API, built in plotting and many other advantages.

Mapping

GEOPANDAS is a pandas binding for SHAPELY, a GPS coordinate framework for visual expressions; data analysis of large datasets. It is optimized for coupled with pandas and has a very intuitive API.

Standard mapping libs like FOLIUM, Kepler & Kepler.gl are available, as we have (like some optimized JUPYTER library extensions).

Connecting to NESP 5.9 Datasets

NESP 5.9 Datasets can be connected to via a generic S3FS-DATASTORE.

S3FS-DATASTORE manages the connections to the NESP 5.9 datasets (stored in S3), mounting in a local DATA-DIRECTORY. Symlinks then may use for specific versions of datasets.

Note that it advised that no consideration of a large prepared file is performed (into separate parquet fragments/files).

This step also creates the relevant environment variables (NESP5.9_HOME) which other processes may use (filtering without downloading the entire dataset).

Finally, one may optionally mirror a remote dataset out by navigating into the dataset's bin/ and running the command: mirror-data-from-s3. Alternatively, one may also upload changes back to the remote dataset with the command: mirror-data-to-s3 (assuming one has write permissions).

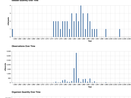
```
export DATA-DIRECTORY=$(pwd)/.local/share/nesp5.9-data
export S3FS-DATASTORE=nesp5.9-3.6.8
export NESP5.9_HOME=$(pwd)/.local/share/nesp5.9-data/nesp5.9-3.6.8
mkdir -p $NESP5.9_HOME
cd $NESP5.9_HOME
ln -s -f -T $(ls -d -1 | sort -r | head -1) current
export PATH=$(pwd)/current/bin:$PATH
export PYTHONPATH=$(pwd)/current/lib:$PYTHONPATH
export LD_LIBRARY_PATH=$(pwd)/current/lib:$LD_LIBRARY_PATH
export JUPYTER_PATH=$(pwd)/current/share/jupyter:$JUPYTER_PATH
export JUPYTER_CONFIG_DIR=$(pwd)/current/etc/jupyter:$JUPYTER_CONFIG_DIR
export JUPYTER_DATA_DIR=$(pwd)/current/share/jupyter:$JUPYTER_DATA_DIR
export JUPYTER_RUNTIME_DIR=$(pwd)/current/var/run/jupyter:$JUPYTER_RUNTIME_DIR
export JUPYTERLAB_DIR=$(pwd)/current/share/jupyter/lab:$JUPYTERLAB_DIR
export JUPYTERLAB_SETTINGS_DIR=$(pwd)/current/etc/jupyter/lab/settings:$JUPYTERLAB_SETTINGS_DIR
export JUPYTERLAB_WORKSPACES_DIR=$(pwd)/current/etc/jupyter/lab/workspaces:$JUPYTERLAB_WORKSPACES_DIR
export NODE_PATH=$(pwd)/current/share/jupyter/lab/staging/node_modules:$NODE_PATH
export PATH=$(pwd)/current/share/jupyter/lab/staging/node_modules/.bin:$PATH
export S3FS_DEBUG=1
export MOUNTPOINT=$(pwd)/.local/share/s3fs-fuse.$S3FS-DATASTORE
mkdir -p $MOUNTPOINT
s3fs $S3FS-DATASTORE $MOUNTPOINT -ocurldbg -o dbglevel=info -f -o curl_ca_bundle=/etc/ssl/certs/ca-bundle.crt
export DATA-DIRECTORY=$MOUNTPOINT
cd $DATA-DIRECTORY
export -f mirror-data-from-s3
export -f mirror-data-to-s3
export -f s3-ls
export -f s3-du
export -f s3-rm
export -f s3-put
export -f s3-get
export -f s3-mv
export -f s3-mkdir-p
export -f s3-touch
export -f s3-ln
export -f s3-ln-fzf
export -f s3-ln-rm
export -f s3-ln-rm-all
export -f s3-ln-list
export -f s3-ln-clean
export -f s3-ln-clean-all
export -f s3-ln-update
export -f s3-ln-update-all
export -f s3-ln-fsck
export -f s3-ln-fsck-all
export -f s3-ln-vacuum
export -f s3-ln-vacuum-all
export -f s3-ln-find
export -f s3-ln-find-broken
export -f s3-ln-find-broken-all
export -f s3-ln-find-broken-clean
export -f s3-ln-find-broken-clean-all
export -f s3-ln-find-broken-clean-force
export -f s3-ln-find-broken-clean-force-all
export -f s3-ln-find-broken-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-all
export -f s3-ln-find-broken-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-all
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean
export -f s3-ln-find-broken-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list-clean-force-dry-run-list
```

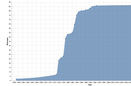
```
[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Cold Temperate Waters',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Subtropical Waters',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Transitional Waters',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Tropical Waters',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Warm Temperate Waters',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Polar Waters',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Coastal Waters',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Open Ocean',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Upwelling',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Coastal Upwelling',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')

[ ]: ax = plot_ecosystem_data_year_start_plot(ecosystem,
                                              'Open Ocean Upwelling',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent Year',
                                              'Ecosystem Extent')
```

Cold Temperate Waters

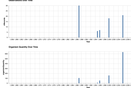
Subtropical Waters

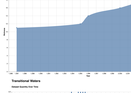
Transitional Waters

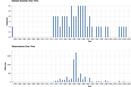




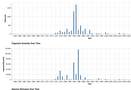
Tropical Waters

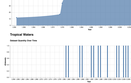


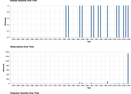




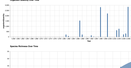
Warm Temperate Waters

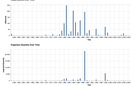






Polar Waters





Coastal Waters

Open Ocean

Upwelling

Coastal Upwelling

Open Ocean Upwelling

H3 Aggregation and Mapping

The WMP H3 datasets are aggregated datasets which provide both coarse regions and time periods.

It is useful to first aggregate the datasets so that we may understand their temporal spatial bounds.

H3 Indexing

The H3 library allows us to quickly index datasets at different resolution coarse levels according to the source latitude and longitude.

Additionally, the `polyfill` function offers a quick way to index the frame regions with `polyfill(GeoJSONDict)`.

```
# Generate H3 indexes for each coarse level
# of all using_polyfill
[ ]: for res in range(0, 6):
     # Iterate through the dataset for series_id namespace
     for series_id, df in ecosystem:
         # Apply the initial and temporal values
         df[f'polyfill_{res}'] = df[['geometry',
         'begin_year', 'end_year']].\
             apply(lambda row: polyfill_wrapper(
             row['geometry'],
             row['begin_year'],
             row['end_year'],
             res), axis=1)
         # Save the resolution
         ecosystem[series_id] = df
         # Update the index mirror type, not the 'polyfill_0000' or 'polyfill_String'
         return_df[series_id] = df.copy
     # Print the columns
     print(return_df.columns)
```

H3 Mapping

The `folium` library offers a quick `GeoJsonLayer()` layer type that greatly reduces complexity in mapping H3 Shapes:

```
# Generate a H3 hexagon layer visualization on a global map
folium.f3.GeoJsonLayer(
    # Specify this is a H3 Hexagon Layer
    h3_hexagons=True,
    # Unfortunately an entire dataframe exports, so just the 'polyfill_Initial' column
    data=ecosystem['Cold Temperate Waters'][['polyfill_0']],
    # Specify the hexagon
    get_hexagon="polyfill_0",
    # Add a random color
    get_fill_color="random",
    # Add some opacity
    get_fill_opacity=0.5,
    # Add a popup to show the value
    popup=True,
    # Add to the map
    fill_map,
    name="Cold Temperate Waters",
    show=False,
    overlay=True,
    control=True
)
```

```
[ ]: # Aggregated dataframe with dataset and number of records attributes
     agg_df = pd.
         concat(
             [df.
             assign(
             dataset=k,
             records=len(df))
             for k, df in
             ecosystem.items()
             ], ignore_index=True,
             sort=False
             )
     agg_df.sample(5)
```

```
[ ]: # Aggregated dataframe with dataset and number of records attributes
     agg_df = pd.
         concat(
             [df.
             assign(
             dataset=k,
             records=len(df))
             for k, df in
             return_dict.items()
             ], ignore_index=True,
             sort=False
             )
     agg_df.sample(5)
```

```
[ ]: # Aggregated dataframe with dataset and number of records attributes
     agg_df = pd.
         concat(
             [df.
             assign(
             dataset=k,
             records=len(df))
             for k, df in
             ecosystem.items()
             ], ignore_index=True,
             sort=False
             )
     agg_df.sample(5)
```

```
[ ]: # Aggregated dataframe with dataset and number of records attributes
     agg_df = pd.
         concat(
             [df.
             assign(
             dataset=k,
             records=len(df))
             for k, df in
             return_dict.items()
             ], ignore_index=True,
             sort=False
             )
     agg_df.sample(5)
```

```
[ ]: import folium

     # Generate and generate H3 layers
     for layer, df in generate_polyfill_hexagon_layers:
         # Generate
         # H3 Hexagonal dataframe can exceed and number of records attributes
         df[f'{layer}_count'] = df.
             groupby(
             f'{layer}').
             count()[
             f'{layer}'].
             reset_index(drop=True).
             sort_values(
             ascending=False)[
             f'{layer}_count']
         # Add the polyfill layer to the map
         folium.f3.GeoJsonLayer(
             h3_hexagons=True,
             data=df[[f'{layer}', f'{layer}_count']],
             get_hexagon=f'{layer}',
             get_fill_color="random",
             get_fill_opacity=0.5,
             popup=True,
             fill_map,
             name=f'{layer}',
             show=False,
             overlay=True,
             control=True
             )
     folium.LayerControl().add_to(fill_map)
     fill_map
```

```
[ ]: # Add the layer control
     folium.LayerControl().add_to(fill_map)
     fill_map
```

```
[ ]: import geopandas as gpd
     import matplotlib.pyplot as plt

     # Load the world boundaries
     world = gpd.
         read_file(
         gpd.datasets.
         get_path('naturalearth_lowres'))

     # Create figure and axis
     fig, ax = plt.
         subplots(
         1, 1,
         figure=False,
         ncols=1,
         nrows=1,
         dpi=300,
         facecolor='w',
         edgecolor='k')

     # Plot the world boundaries
     world.
         plot(
         ax=ax,
         color='lightgrey')

     # Plot the polyfill_0 layer
     df[df['polyfill_0'].
         isin(
         df['polyfill_0'].
         sample(
         10,
         random_state=1
         )
         )
         ].
         plot(
         ax=ax,
         cmap='Set1',
         alpha=0.5,
         edgecolor='k')

     plt.
         show()
```


Australian Marine Regions

Open Australian Marine Regions by the Marine Biodiversity Hotspots Data Wiki Website. Note that the tagging in the 1000 accounts as the regions are mapped to the 1000 regions.

Note that the Open Australian Marine Regions contain the World Heritage Areas map.

Marine Region Type	Map Explorer URL	Metadata URL (WKT/Shfiles)	Data/SDM GIT Endpoint
Commonwealth Marine Reserves	Explore	Metadata	Data/SDM
World Heritage Areas	Explore	Metadata	Data/SDM
Key Biodiversity Areas	Explore	Metadata	Data/SDM
IMCRA Provinces	Explore	Metadata	Data/SDM
IMCRA Mesoscale	Explore	Metadata	Data/SDM

Follow the **Explore** links to explore the Australian Marine Regions. The Commonwealth Marine Reserve Regions map has been embedded below.

```
[ ]: import IFrame.display
     print("Commonwealth Marine Reserves")
     display.display(
     IFrame(
     "https://www.marinebiodiversity.org/#/search/-148.0,-24.0,-103.0,-4.0?q=%7B%22filters%22%3A%7B%22IMCRA_Provinces%22%3A%5B%5D%2C%22Commonwealth_Marine_Reserves%22%3A%5B%5D%2C%22World_Heritage_Areas%22%3A%5B%5D%2C%22Key_Biodiversity_Areas%22%3A%5B%5D%2C%22IMCRA_Mesoscale%22%3A%5B%5D%2C%22Ecosystem_Extent%22%3A%5B%5D%2C%22Ecosystem_Risk%22%3A%5B%5D%2C%22Ecosystem_Recovery%22%3A%5B%5D%2C%22Species_Richness%22%3A%5B%5D%2C%22Species_Extent%22%3A%5B%5D%2C%22Species_Risk%22%3A%5B%5D%2C%22Species_Recovery%22%3A%5B%5D%2C%22Species_Protection%22%3A%5B%5D%2C%22Species_Management%22%3A%5B%5D%2C%22Species_Habitat%22%3A%5B%5D%2C%22Species_Occurrence%22%3A%5B%5D%2C%22Species_Occurrence_Buffer%22%3A%5B%5D%2C%22Species_Habitat_Buffer%22%3A%5B%5D%2C%22Species_Protection_Buffer%22%3A%5B%5D%2C%22Species_Management_Buffer%22%3A%5B%5D%2C%22Species_Richness_Buffer%22%3A%5B%5D%2C%22Species_Extent_Buffer%22%3A%5B%5D%2C%22Species_Risk_Buffer%22%3A%5B%5D%2C%22Species_Recovery_Buffer%22%3A%5B%5D%2C%22Ecosystem_Extent_Buffer%22%3A%5B%5D%2C%22Ecosystem_Risk_Buffer%22%3A%5B%5D%2C%22Ecosystem_Recovery_Buffer
```

"NEW Protected Marine Type Transitional Waters"	5978
"NEW Protected Marine Type Transitional Waters"	5979
"NEW Ordinance Marine Type Coastal Territorial Waters"	6966
"NEW Ordinance Marine Type Coastal Territorial Waters"	6967
"NEW Protected Marine Type Marine Temperate Waters"	6968
"NEW Ordinance Marine Type Transitional Waters"	6972
"Commonwealth Marine Region Temperate East"	10965
"NEW Protected Marine Southern Transition"	10968
"Marine Park National Park Zone Western Kangaroo Island"	2
"NEW Ordinance Marine Parks"	3
"Marine Park National Park Zone Coral Grounds"	5
"Marine Park National Park Zone Blue Corner"	6
"Marine Park Special Purpose Zone Great Australian Bight"	7
"Marine Park National Park Zone Gulf of Carpentaria"	8
"Marine Park Habitat Protection Zone Carnarvon Canyon"	9
"Marine Park Multiple Use Zone Roebuck"	10
"Marine Park Special Purpose Zone (Trawl) Argo-Rowley Terrace"	11
"Marine Park National Park Zone Montague Sound"	12

Filtering to All Data within Australian Marine Regions

We map filter to a particular marine regions a using `st_buffer(0)` constraint

```
# Filter to the commonwealth marine region
australian_marine_regions_df = df filter(
  st_intersects(st_buffer(australian_marine_regions$geometry, 0), aus_shp))

# Aggregate and generate id layer
h3_layers = until_generate_geohex_hexagon_layers(
  df=df,
  australian_marine_regions_df
  )

  # Additional id column at a higher resolution
  with_columns(
    geohex_id_5=h3_fn_to_int(
      h3_geohex_to_h3(st_geohex(geometry,
      level=5,keep_geometry=FALSE)
      )
      return_dtype=pl$n_bytes,
      )alias("h3Index"),

    group_by(
      pull(h3_id_5)
      )
    ,
    .by=NULL,
    .keep=FALSE,
    .order_by=NULL,
    .with_rowid=FALSE,
    .maintain_order=TRUE,
    .name_sep="_",
    .lazy=FALSE,
    .sort_by=NULL,
    .descending=FALSE,
    .nulls_last=FALSE,
    .multiple=TRUE,
    .over=NULL
    )
    ,
    .name="count",
    .weight=NULL,
    .ddof=1
    
```

AMSA Vessel Tracking: Comprehensive Strategic Analysis (2012–2025)

Author:	Karishma Khanna
Edited:	2026-01-08
Affiliation:	AODN/IMOS
e-mail:	karishma.khanna@utas.edu.au
Date of creation:	2026-01-08
Date of last update:	2026-01-08

Background

This is a template notebook outlining the library stack and content structure for NESP Data Products.

It outlines:

1. Connecting to NESP 5.9 Datasets
2. H3 Aggregation and Mapping
3. Identifying Trend Sites

NESP 5.9 Datasets

Dataset name	S3 URL
AMSA	link
Kelp	link
NRMN	link
Seabird	link
Seagrass	link

S3 Details

NESP 5.9 datasets are currently stored in S3 and are publicly available as per the following table:

Bucket	Key	Partitioned
`data-uplift-public`	`stored/datauplift/amsa/year=*/source=*/*.parquet`	[x]
`data-uplift-public`	`stored/datauplift/kelp/kelp.parquet`	[ ]
`data-uplift-public`	`stored/datauplift/nrmn/nrmn.parquet`	[ ]
`data-uplift-public`	`stored/datauplift/seabird/seabird.parquet`	[ ]
`data-uplift-public`	`stored/datauplift/seagrass/seagrass.parquet`	[ ]

Required Python & Packages

The required packages are found in the pyproject.toml under `dependencies`.

`uv` installation

It is highly recommended to manage python version, environment and projects with uv.

```
uv venv && source .venv/bin/activate && uv pip install .
```

`pip` installation

Otherwise, native python tooling can be used to manage the python environment and project.

Note that python3.12 or greater is required.

```
python -m venv .venv && source .venv/bin/activate && pip install .
```

Running Notebooks in Jupyter or IDE

The notebook can be manually run by launching jupyter notebook from the console:

```
jupyter notebook
```

Otherwise, if running in an IDE, select the newly created `.venv` as the kernal.

Notebook Stack

IO

`pyarrow` is the defacto parquet and arrow standard library tooling in the python eco-system, offering exceptional performance, documentation and stability.

Its `fs` class allows connecting to parquet datasets in AWS S3, which is perfect for our use case.

Computation

`polars` is fast replacing pandas as the primary python dataframe API. It is more tightly integrated with arrow (and pyarrow), has exceptional performance, greater than memory dataset manipulation, a more consistent and declarative API, built in plotting and many other advantages.

Mapping

`pydeck` is a python binding for `deck.gl`, a GPU-powered framework for visual exploratory data analysis of large datasets. It is optimised for Jupyter notebooks and highly customisable.

Standard mapping tools like `folium` struggle to handle the number of mapped shapes, so we favor the more optimised `pydeck` library.

Plotting

`seaborn` and `matplotlib` are used for statistical data visualization. `seaborn` provides a high-level interface for drawing attractive and informative statistical graphics, while `matplotlib` manages the figure layouts and axis formatting.

Rich

`rich` provides colorful, formatted console output and tables for Python, improving readability of logs and data in both terminals and Jupyter notebooks.

```
# Set up rich console
import rich
import rich.console
console = rich.console.Console()
```

Connecting to NESP 5.9 Datasets

NESP 5.9 datasets can be connected to via a pyarrow `S3FileSystem`.

The `S3FileSystem` manages the connection to the NESP 5.9 datasets stored in S3, resulting in a `pyarrow.Dataset` handle that we may use for efficient loading and subsetting of the dataset.

Note that a dataset can be comprised of a single parquet file or partitioned (into separate parquet fragments/files).

This step also makes the dataset schema available and file structure available allowing for pre-filtering without downloading the entire dataset.

Finally, we may estimate the in memory dataset size by sampling the dataset. If the dataset is small, we may use a `DataFrame` straight away. Otherwise if the dataset is too large, we must pre-aggregate and or filter using a `LazyFrame` before loading the dataset into memory.

```
import pyarrow
import pyarrow.fs
import pyarrow.dataset
import util

# --- FileSystem Configuration ---
# Set up a connection to Amazon S3 storage in the 'ap-southeast-2' region.
# The 'anonymous=True' option means no login or password is needed, as the data is public.

FILE_SYSTEM = pyarrow.fs.S3FileSystem(
    region="ap-southeast-2",
    anonymous=True,
)

# --- Dataset Connection ---
# Connect to the AMSA dataset stored in S3.
# This does NOT download any data yet; it just sets up a link so we can explore or load data later.
ds = pyarrow.dataset.dataset(
    source="data-uplift-public/stored/datauplift/amsa/dataset",
    filesystem=FILE_SYSTEM,
```

```
)

# --- Schema Inspection ---
console.log("📂 Inspecting AMSA Archive...")
# Print the number of files found in the dataset.
rich.print('Total Inspected files: ',len(list(ds.get_fragments())))

# Display the dataset's schema (column names, definitions, and units) in a nice table.
schema_rich_table = util.print_schema_rich_table(
    schema=ds.schema,
    metadata_keys=[
        "definition",
        "unit"
    ],
)

console.print(schema_rich_table)
```

```
[11:52:52] 📂 Inspecting AMSA Archive...                                  2523287152.py:24
Total Inspected files:  160
```

name	type	definition	unit
craftID	int64	Unique identifier for each vessel	
decimalLatitude	double	The geographic latitude (in decimal degrees, using the spatial reference system given in dwc:geodeticDatum) of the geographic center of a dcterms:Location. Positive values are north of the Equator, negative values are south of it. Legal values lie between -90 and 90, inclusive.	
decimalLongitude	double	The geographic longitude (in decimal degrees, using the spatial reference system given in dwc:geodeticDatum) of the geographic center of a dcterms:Location. Positive values are east of the Greenwich Meridian, negative values are west of it. Legal values lie between -180 and 180, inclusive.	
course	double	Course over ground in decimal degrees	decimal degrees
speed	double	Speed over ground in knots	knots
type	string	Primary vessel classification	
subType	string	Specific vessel sub-category	
length	int16	Vessel length measured in metres	m
beam	int16	Vessel beam (width) measured in metres	m
draught	double	Vertical distance between the waterline and the bottom of the hull in metres	m
eventDate	timestamp	The date-time or interval during which a dwc:Event occurred. For occurrences, this is the date-time when the dwc:Event was recorded. Not suitable for a time in a geological context.	
h3Index	string	A hexadecimal string representing a H3 polygon, derived from the `decimalLatitude` and `decimalLongitude`.	
australianMarineRegionsTags	string	A `|` separated tag column featuring common australian marine regions	

None

Loading Data

The dataset handle is our connection to data in S3. We can use this handle to load data directly into a `DataFrame` or `LazyFrame`

DataFrame

For smaller datasets the `DataFrame` is preferrable. This loads the entire dataset into memory ready for analysis.

LazyFrame

For larger datasets the `DataFrame` may not fit into memory. In these cases the `LazyFrame` is ideal as it allows the dataset to be filtered and or aggregated prior to loading the dataset, allowing it to fit into memory, ready for analysis.

> **For AMSA:** Given the dataset exceeds available RAM, we initialize a `LazyFrame`.

```
import polars as pl
import polars_h3
import polars
import os

# --- Initialize LazyFrame ---
# Initialize a lazy connection to the AMSA dataset (no data loaded yet)
# Data will be read only when an action like .collect() is executed
amsa_lf = pl.scan_pyarrow_dataset(ds)

print("✅ LazyFrame initialized.")
```

```
✅ LazyFrame initialized.
```

H3 Aggregation and Mapping

The NESP 5.9 datasets are aggregated data products that cover large regions and time periods. It is useful to first aggregate the datasets so that we may understand their temporal-spatial bounds.

H3 Indexing

The `h3` library allows us to spatially index datasets at different resolutions (zoom levels) according to the record latitude and longitude.

Additionally, the `polars_h3` library offers a polars style interface for faster tagging with `polars.DataFrames`.

H3 Mapping

The `pydeck` library offers a built in `H3HexagonLayer` layer type that greatly reduces complexity in mapping H3 Shapes.

```
console.log("🚀 Starting H3 Aggregation...")

# --- H3 Aggregation ---
# Aggregate records by H3 resolution 8 cell and year
# Streaming collect runs the computation and loads only the results
h3_trend_df = (
    amsa_lf.group_by(
        polars.col("h3Index").alias("h3IndexResolution8"),
        polars.col("eventDate").dt.year().alias("year"),
    )
    .agg(
        polars.len().alias("n_records"),
    )
    # The collect statement applies the above aggregation and streams
    # only the required data to the client (us)
    .collect(engine="streaming")
)

console.log(f"✅ Aggregation Complete.")
```

```
          🚀 Starting H3 Aggregation...                                  1025018184.py:1
[11:25:04] ✅ Aggregation Complete.                                      1025018184.py:19
```

Visualising the H3 Aggregation

With the data aggregated and the global scale determined, we can now render the spatial density map using `pydeck`.

The `generate_comparative_map` function below handles:

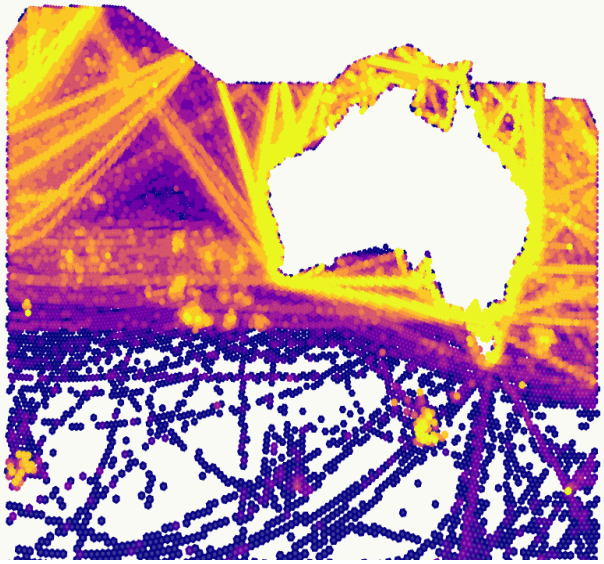
1. **Filtering:** Selecting the specific year from the aggregated dataset.
2. **Normalization:** Mapping the Log-Score to a fixed color palette using the `GLOBAL_MAX_LOG` calculated previously.
3. **Rendering:** Generating the 3D Hexagon Layer.

```
import pydeck

def generate_comparative_map(dataframe, target_year):
    """
    Generates a pydeck H3 Hexagon map for a specific year using a fixed global color scale.

    This function filters a Polars DataFrame for a specific year, bins the data into
    discrete color buckets based on a global maximum value to ensure visual consistency
    across different time slices, and renders the result using pydeck.

    :param dataframe: The input data containing H3 indices and record counts.
    :type dataframe: polars.LazyFrame or polars.DataFrame
    :param target_year: The specific year to filter and visualize.
    :type target_year: int
```

```
    :return: A pydeck Deck object rendered in the current environment.
    :rtype: pydeck.bindings.deck.Deck
    """

    console.log(f"🎨 Generating Map for {target_year}...")

    # --- Filter Data ---
    # Filter the LazyFrame for the target year.
    year_df = dataframe.filter(
        polars.col("year").eq(target_year),
    )

    # # --- Color Palette ---
    # # Generate a 10-step discrete color palette ('plasma') for high contrast.
    n_quantiles = 10

    # --- Layer Construction ---
    # Create a separate H3HexagonLayer for each color bucket.
    # This is required because pydeck layers typically take a single fill color.
    layers = util.generate_pydeck_hexagon_layers(
        df=year_df,
        aggregate_column_name="n_records",
        hexagon_index_column_name="h3IndexResolution8",
        n_quantiles=n_quantiles,
        color_palette="plasma",
    )

    # --- View State & Deck ---
    # Dynamically center view as per data
    view_state = util.generate_view_state(
        df=year_df.with_columns(
            polars_h3.cell_to_lat(polars.col("h3IndexResolution8")).alias("decimalLatitude"),
            polars_h3.cell_to_lng(polars.col("h3IndexResolution8")).alias("decimalLongitude"),
        ),
    )

    r = pydeck.Deck(
        layers=layers,
        initial_view_state=view_state,
        map_style=pydeck.map_styles.CARTO_LIGHT, # White background
        tooltip={
            "html": "<b>H3:</b> {h3IndexResolution8}<br/><b>Pings:</b> {n_records}<br/>"
        }
    )

    return r.show()
```

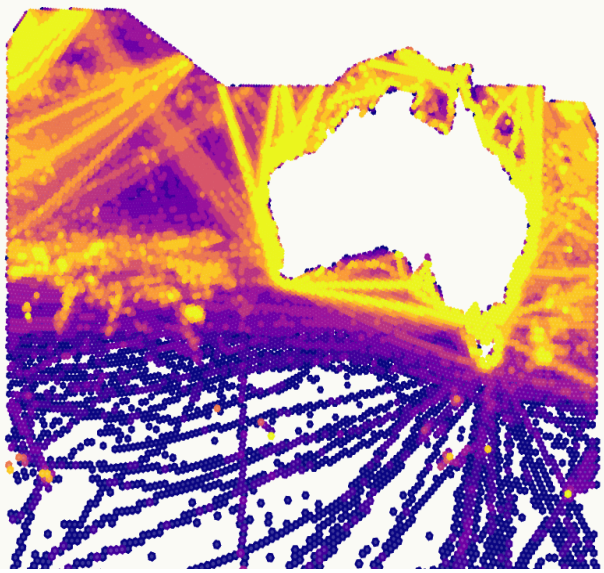
```
# The map for year 2013
print("--- 2013 Vessel Density ---")
generate_comparative_map(h3_trend_df, 2013)
```

```
--- 2013 Vessel Density ---
[11:25:05] 🎨 Generating Map for 2013...                                    2217516252.py:19
```



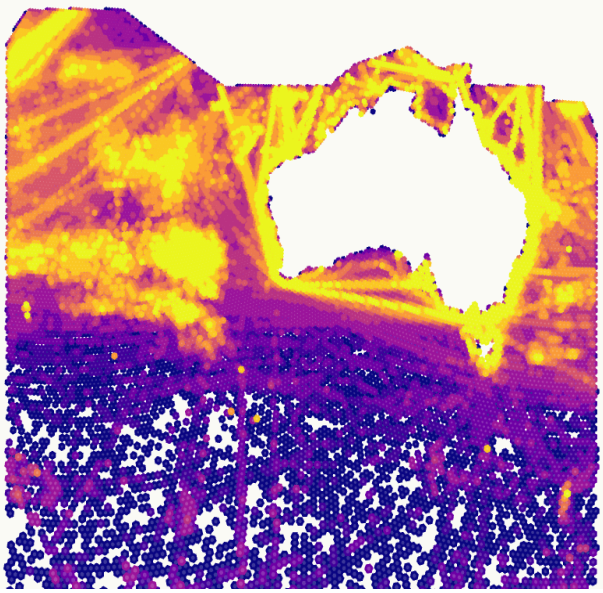
```
# The map for year 2019
print("--- 2019 Vessel Density ---")
generate_comparative_map(h3_trend_df, 2019)
```

```
--- 2019 Vessel Density ---
[11:25:06] 🎨 Generating Map for 2019...                    2217516252.py:19
```



```
# The map for year 2024
print("--- 2024 Vessel Density ---")
generate_comparative_map(h3_trend_df, 2024)
```

```
--- 2024 Vessel Density ---
          🎨 Generating Map for 2024...                     2217516252.py:19
```



Regional Analysis: Sydney Harbour

While the global map uses a coarse resolution (Resolution 8, approx 0.7km²) suitable for open ocean shipping lanes, complex waterways like Sydney Harbour require finer detail.

To analyze this region effectively, we perform a **Spatial Deep Dive**:

1. **Spatial Filter:** We restrict the query to the Sydney bounding box.
2. **Re-Indexing:** We calculate a new H3 Index at **Resolution 9** (approx 0.1km²). This allows us to resolve individual ferry routes and harbour traffic that would otherwise be merged into a single block.

```
# To minimise memory footprint, load a where and when df
temporal_spatial_df = amsa_lf.select(
    polars.col("eventDate"),
    polars.col("decimalLatitude"),
    polars.col("decimalLongitude"),
).collect()
temporal_spatial_df.estimated_size(unit="gb")
temporal_spatial_df.describe()
```

shape: (9, 4)

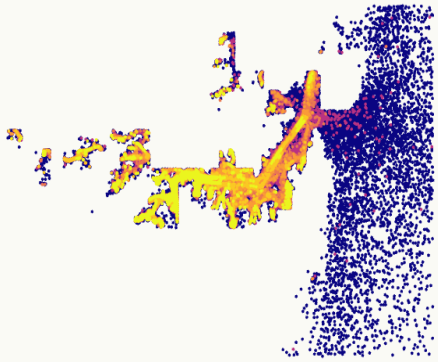
statistic	eventDate	decimalLatitude	decimalLongitude
str	str	f64	f64
"count"	"199364428"	1.99364428e8	1.99364428e8
"null_count"	"0"	0.0	0.0
"mean"	"2020-06-12 03:10:48.815000+00:…	-24.551805	133.849296
"std"	null	8.952974	21.919716
"min"	"2012-09-01 00:00:00+00:00"	-81.42667	75.0
"25%"	"2017-06-20 14:28:10+00:00"	-32.25354	116.6988
"50%"	"2021-01-11 08:33:00+00:00"	-23.77358	145.48195
"75%"	"2023-09-05 15:22:28+00:00"	-19.25459	151.48203
"max"	"2025-12-31 23:59:59+00:00"	-2.0	163.0

```
def generate_high_res_map(
    df: polars.DataFrame,
    target_year: int,
    h3_resolution: int = 9,
    max_lat: float = -33.78,
    min_lat: float = -33.93,
    max_lon: float = 151.35,
    min_lon: float = 151.10,
):
    """
    Generate a high resolution h3 map.mro

    Defaults set up for Sydney.
    """

    console.log(f"⚓ Generating High-Res Map (Res {h3_resolution}) for {target_year}...")

    # Error check the bounds
    if max_lat <= min_lat:
        raise ValueError("max_lat must be greater than min_lat: found `{max_lat}` <= `{min_lat}`")
    if max_lon <= min_lon:
        raise ValueError("max_lon must be greater than min_lat: found `{max_lon}` <= `{min_lon}`")

    # --- Filter & Re-Index (Resolution 9) ---
    # We filter the main LazyFrame ('amsa_lf') to the specific region and year.
    # We calculate a new 'h3Index' at the specified new h3 resolution.
    map_df = (
        df
        .filter(
            polars.col("decimalLatitude").is_between(min_lat, max_lat),
            polars.col("decimalLongitude").is_between(min_lon, max_lon),
            polars.col("eventDate").dt.year().eq(target_year),
        )
        .select(["decimalLatitude", "decimalLongitude"])
        .with_columns(
            polars_h3.latlng_to_cell(
                lat=polars.col("decimalLatitude"),
                lng=polars.col("decimalLongitude"),
                resolution=h3_resolution,
                return_dtype=polars.String
            ).alias("h3Index")
        )
        .group_by("h3Index")
        .agg(polars.len().alias("n_records"))
    )

    # --- Validation ---
    if map_df.height == 0:
        console.log("⚠️ No data found for this year/region.")
        return

    n_quantiles = 10
```

```
    layers = util.generate_pydeck_hexagon_layers(
        df=map_df,
        aggregate_column_name="n_records",
        hexagon_index_column_name="h3Index",
        n_quantiles=n_quantiles,
        color_palette="plasma",
    )

    # --- View State & Deck ---
    # Dynamically center view as per data
    view_state = util.generate_view_state(
        df=map_df.with_columns(
            polars_h3.cell_to_lat(polars.col("h3Index")).alias("decimalLatitude"),
            polars_h3.cell_to_lng(polars.col("h3Index")).alias("decimalLongitude"),
        ),
    )

    r = pydeck.Deck(
        layers=layers,
        initial_view_state=view_state,
        map_style=pydeck.map_styles.CARTO_LIGHT,
        tooltip={"text": "n_records: {n_records}"}
    )

    return r.show()
```

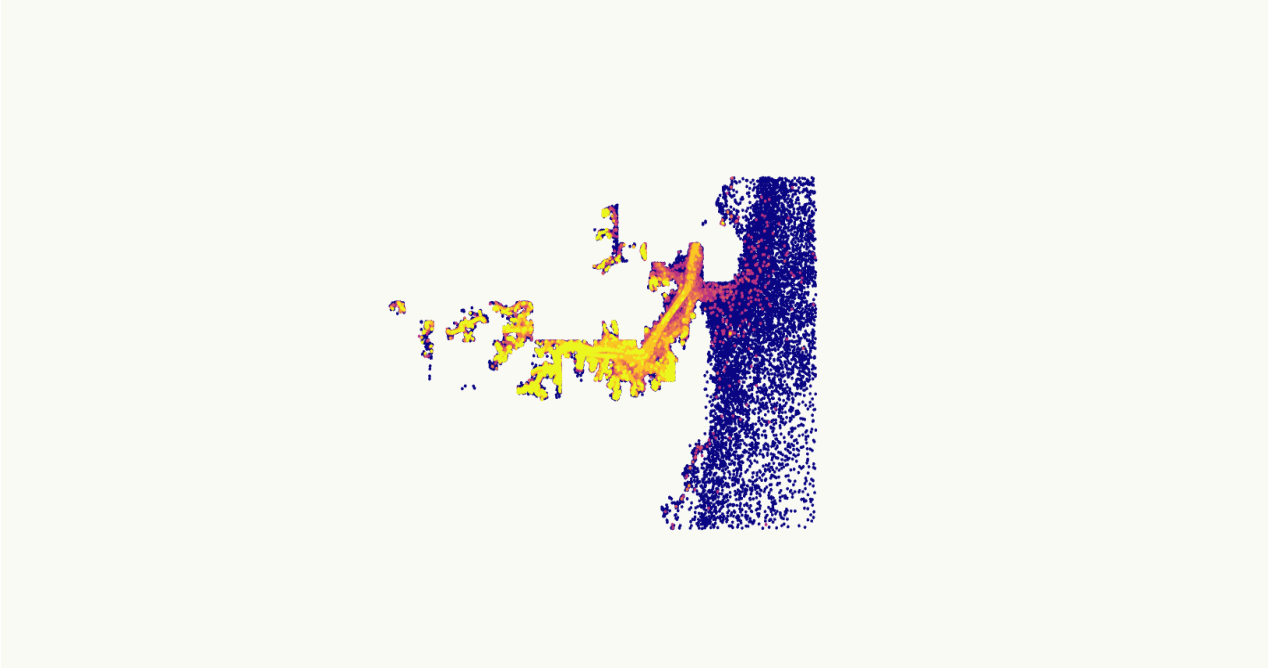
```
# The map for year 2016
print("--- 2016 Vessel Density Map of Sydney Harbour---")
generate_high_res_map(temporal_spatial_df, 2016, h3_resolution=12)
```

```
--- 2016 Vessel Density Map of Sydney Harbour---
[11:26:06] ⚓ Generating High-Res Map (Res 12) for 2016...                    3135006617.py:16
```



```
# The map for year 2024
print("--- 2024 Vessel Density Map of Sydney Harbour---")
generate_high_res_map(temporal_spatial_df, 2024, h3_resolution=12)
```

```
--- 2024 Vessel Density Map of Sydney Harbour---
[11:26:12] ⚓ Generating High-Res Map (Res 12) for 2024...                    3135006617.py:16
```



Longitudinal Data Health

This section characterizes the dataset's **temporal stability**.

Long-term maritime datasets are frequently influenced by **"Structural Breaks"**—step-changes in data volume caused by upgrades to surveillance infrastructure (e.g., new satellite launches). The following analysis distinguishes between **actual fleet growth** (increased traffic) and **systematic growth** (improved sensor sensitivity).

Methodology

To quantify these trends, we aggregate the data annually and derive three core metrics:

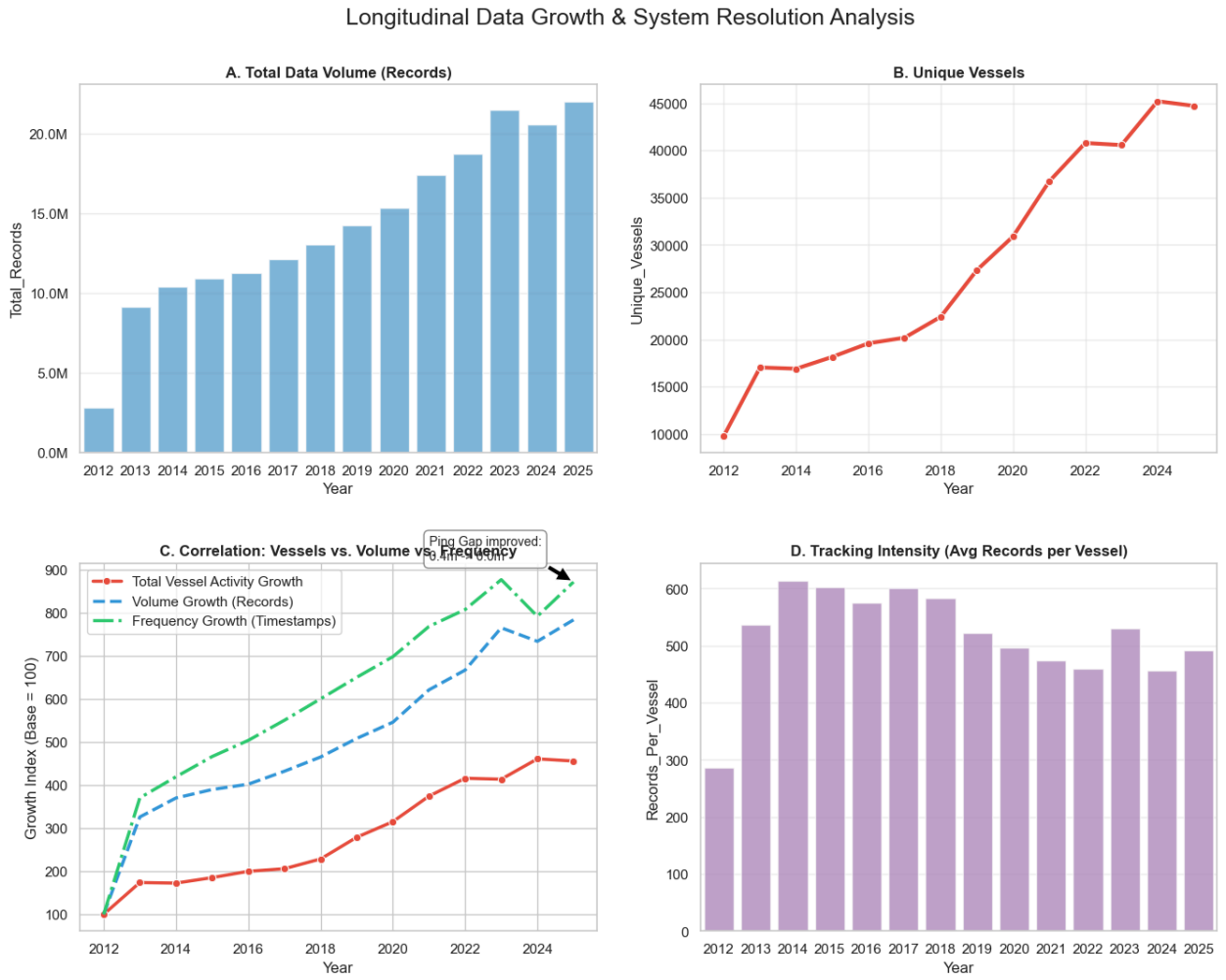
1. **Volume:** The total count of AIS transmission records (Data Size).
2. **Fleet:** The count of unique vessels identified by `craftID` (Traffic Size).
3. **Intensity:** The average number of records per vessel (Sensor Resolution).

> **Normalization Strategy:** To compare trends across different scales (thousands of ships vs. millions of records), all metrics are indexed to their **2012 baseline (2012 = 100)**. This allows for a direct correlation analysis of relative growth rates.

```
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.ticker import FuncFormatter

# --- Aggregation ---
# We reuse the 'amsa_lf' LazyFrame initialized above and group by Year to extract high-level statistics without loading the

console.log("🚀 Calculating Annual Statistics (Streaming Mode)...")

yearly_stats = (
    amsa_lf.select(["eventDate", "craftID"])
    .with_columns(pl.col("eventDate").dt.year().alias("Year"))
    .group_by("Year")
    .agg([
        pl.col("craftID").n_unique().alias("Unique_Vessels"),      # Fleet Size
        pl.len().alias("Total_Records"),                           # Data Volume
        pl.col("eventDate").n_unique().alias("Unique_Timestamps")  # Temporal Granularity
    ])
    .sort("Year")
    .collect(streaming=True)    # Trigger computation
)

# --- Normalization (Base 100) ---
# To compare "Vessels" (thousands) vs "Records" (millions), we index them to their 2012 baseline. (2012 value = 100).

base_vessels = yearly_stats["Unique_Vessels"][0]
base_records = yearly_stats["Total_Records"][0]
```

```
base_timestamps = yearly_stats["Unique_Timestamps"][0]

yearly_stats = yearly_stats.with_columns([
    (pl.col("Total_Records") / pl.col("Unique_Vessels")).alias("Records_Per_Vessel"),
    (pl.col("Unique_Vessels") / base_vessels * 100).alias("Idx_Vessels"),
    (pl.col("Total_Records") / base_records * 100).alias("Idx_Records"),
    (pl.col("Unique_Timestamps") / base_timestamps * 100).alias("Idx_Timestamps"),
    # Calculate Ping Gap (Inverse of Frequency) for annotation
    (365 * 24 * 60 / pl.col("Unique_Timestamps")).alias("Avg_Gap_Minutes")
])

# --- Visualization ---
# We use Matplotlib to create a 2x2 Grid, and Seaborn to paint the data.

sns.set_theme(style="whitegrid")
df_plot = yearly_stats.to_pandas()  # Convert small agg stats to Pandas for plotting

# Function to format Y-axis as Millions
def millions(x, pos):
    return '%1.1fM' % (x * 1e-6)

sns.set_theme(style="whitegrid")
fig, axes = plt.subplots(2, 2, figsize=(16, 12))
plt.subplots_adjust(hspace=0.3)

# A. Volume (Bar Chart)
sns.barplot(data=df_plot, x="Year", y="Total_Records", ax=axes[0,0], color="#3498db", alpha=0.7)
axes[0,0].yaxis.set_major_formatter(FuncFormatter(millions))
axes[0,0].set_title("A. Total Data Volume (Records)", fontweight='bold')
axes[0,0].grid(axis='y', alpha=0.3)

# B. Fleet Size (Line Chart)
sns.lineplot(data=df_plot, x="Year", y="Unique_Vessels", marker="o", ax=axes[0,1], color="#e74c3c", linewidth=3)
axes[0,1].set_title("B. Unique Vessels", fontweight='bold')
axes[0,1].grid(True, alpha=0.3)

# C. Growth Correlation (Multi-Line)
# This graph reveals if data grew because *ships* increased or *sensors* improved.
ax_c = axes[1,0]
sns.lineplot(data=df_plot, x="Year", y="Idx_Vessels", ax=ax_c, color="#e74c3c", label="Total Vessel Activity Growth", linew
sns.lineplot(data=df_plot, x="Year", y="Idx_Records", ax=ax_c, color="#3498db", label="Volume Growth (Records)", linewidth=
sns.lineplot(data=df_plot, x="Year", y="Idx_Timestamps", ax=ax_c, color="#2ecc71", label="Frequency Growth (Timestamps)", l

ax_c.set_ylabel("Growth Index (Base = 100)")
ax_c.set_title("C. Correlation: Vessels vs. Volume vs. Frequency", fontweight='bold')
ax_c.legend(loc='upper left')

# Add "Ping Gap" Annotation
if len(df_plot) > 0:
    gap_start = df_plot["Avg_Gap_Minutes"].iloc[0]
    gap_end = df_plot["Avg_Gap_Minutes"].iloc[-1]
    last_year = df_plot["Year"].iloc[-1]
    last_val = df_plot["Idx_Timestamps"].iloc[-1]

    ax_c.annotate(f"Ping Gap improved:\n{gap_start:.1f}m -> {gap_end:.1f}m",
                  xy=(last_year, last_val),
                  xytext=(last_year - 4, last_val + 50),
                  arrowprops=dict(facecolor='black', shrink=0.05),
                  fontsize=10, bbox=dict(boxstyle="round,pad=0.4", fc="white", ec="gray", alpha=0.9))

# D. Intensity (Bar Chart)
sns.barplot(data=df_plot, x="Year", y="Records_Per_Vessel", ax=axes[1,1], color="#9b59b6", alpha=0.6)
axes[1,1].set_title("D. Tracking Intensity (Avg Records per Vessel)", fontweight='bold')
axes[1,1].grid(axis='y', alpha=0.3)

plt.suptitle("Longitudinal Data Growth & System Resolution Analysis", fontsize=18, y=0.95)
plt.show()
```

```
[12:45:46] 🚀 Calculating Annual Statistics (Streaming Mode)...                    1766868370.py:8
C:\Users\khannak\AppData\Local\Temp\ipykernel_20976\1766868370.py:20: DeprecationWarning: the `streaming` parameter was de
  .collect(streaming=True)    # Trigger computation
```



Australian Marine Regions

Each occurrence in the dataset is tagged with its corresponding **Australian Marine Region**. These tags allow us to filter and analyze traffic within specific legal boundaries (e.g., Marine Parks, Heritage Areas).

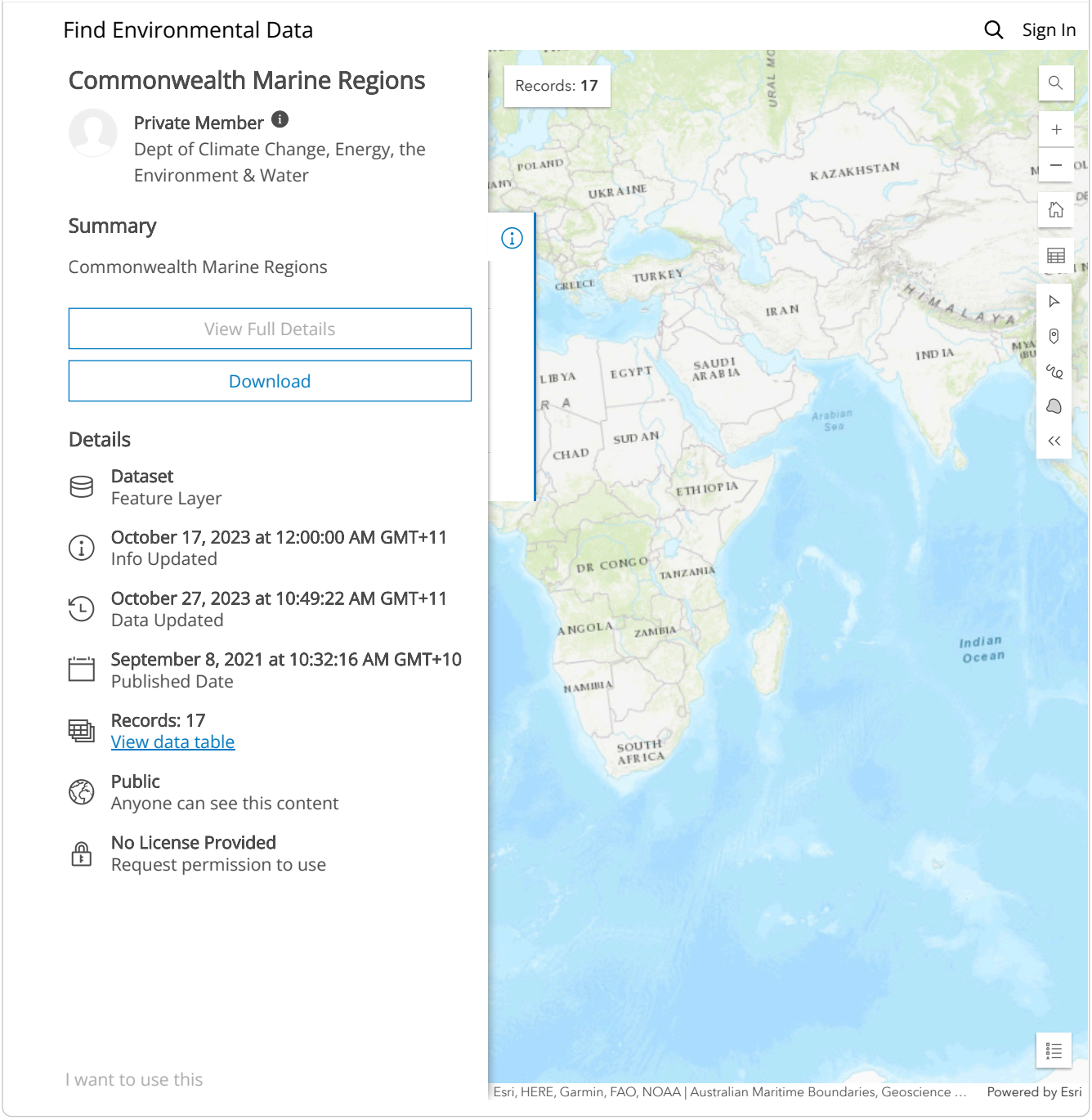
> **Note:** The tagging is accurate to ~6km at region boundaries.

Interactive Explorer

Use the map below to explore the official boundaries of Commonwealth Marine Regions.

```
import IPython.display

# Display the Commonwealth Marine Regions
# This allows us to visually explore the boundaries before analyzing them.
IPython.display.IFrame(
    src="https://fed.dcceew.gov.au/datasets/commonwealth-marine-regions/explore?location=-33.665700%2C140.517300%2C4",
    width="100%",
```

```
    height="1000px",
)
```



7Exploring Available Marine Regions Tags

The `australianMarineRegionsTags` column contains pipe-delimited values (e.g., `Commonwealth Marine Region: South-east|Marine Park: Freycinet`). To understand the traffic distribution, we split these tags and count the occurrences of each unique region.

```
import polars as pl

marine_lf = (
    amsa_lf
    .filter(
        pl.col("australianMarineRegionsTags").str.contains(pattern="Commonwealth Marine Region:South-east", literal=True)
    )
    .group_by(
        pl.col("h3Index"),
        pl.col("eventDate").dt.year().alias("Year")
            )
    .agg(
        pl.len().alias("n_records")
    )
    .collect()
```

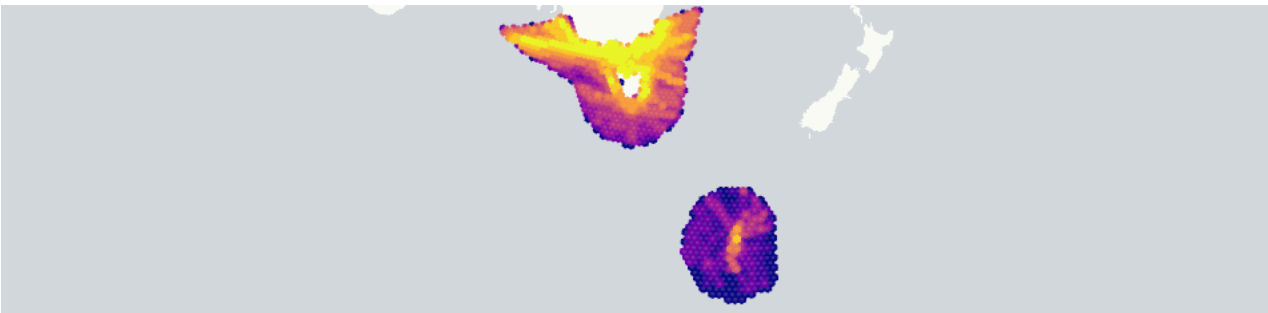
```
)
```

```
print(f"✅ Data Collected. Rows: {len(marine_lf)}")
```

```
✅ Data Collected. Rows: 11477
```

```
# --- 2. Template Prep (Fixed) ---
map_plot = (
    marine_lf
    .group_by("h3Index")
    .agg([
        pl.sum("n_records").alias("n_records"),
        pl.lit("South-east Region").alias("datasets")
    ])
    .with_columns(
        polars_h3.cell_to_latlng("h3Index").alias("coords")
    )

    .with_columns(
        pl.col("coords").list.get(0).alias("decimalLatitude"),
        pl.col("coords").list.get(1).alias("decimalLongitude")
    )
    .drop("coords") # Clean up
)

# --- 3. Visualization ---
h3_layers = util.generate_pydeck_hexagon_layers(
    df=map_plot,
    aggregate_column_name="n_records",
    n_quantiles=10,
    color_palette="plasma",
)
```

```
import pydeck
deck = pydeck.Deck(
    layers=h3_layers,
    map_style=pydeck.map_styles.LIGHT_NO_LABELS,
    tooltip=util.TOOLTIP,
    initial_view_state=util.generate_view_state(
        df=map_plot,
        view_proportion=0.9
    )
)

deck.show()
```



AMSA

Karishma Khanna, Denisse Fierro Arcos

2026-02-18

- Background
  - Relevant packages
  - Accessing NESP datasets
  - `H3` Aggregation and Mapping
    - `H3` Indexing
    - Visualization Strategy
  - Visualising the `H3` Aggregation
  - Regional Analysis: Sydney Harbour

Background

This R notebook outlines the packages needed to access and analyse NESP Data Products, as well as how to check their content structure. This notebook was originally written in Python by Karishma Kahnna It was “translated” into R by Denisse Fierro Arcos. It should be noted that although both notebooks cover the same topic, the content may vary between these notebooks as features may not be available in both Python and R.

This notebook includes the following sections:
- Relevant R packages needed to access NESP Data Products - How to access NESP Data Products - How aggregate and map NESP datasets using the H3 package - Identifying suitable sampling sites where trend analysis can be carried out

Note: The `H3` package was originally developed in C, but there are at least three packages that make this package accessible in R. We recommend you install the `h3-r` package, which published in GitHub (https://github.com/crazycapivara/h3-r/) because this package will automatically install the original `H3` package written in C (unlike the other two options available as of February 2026). More information about the `h3-r` package, as well as a short tutorial can be found in their documentation (https://crazycapivara.github.io/h3-r/articles/h3.html).

Relevant packages

```
# Connects to datasets remotely
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.5.2
```

```
##
## Attaching package: 'arrow'
```

```
## The following object is masked from 'package:utils':
##
##     timestamp
```

```
# Indexing system that uses hexagons
library(h3)
# Represents simple spatial features
library(sf)
```

```
## Linking to GEOS 3.13.1, GDAL 3.11.0, PROJ 9.6.0; sf_use_s2() is TRUE
```

```
# Data manipulation
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:arrow':
##
##     duration
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(tidyr)
library(stringr)
# Plotting
library(ggplot2)
# Interactive maps
library(leaflet)
```

```
## Warning: package 'leaflet' was built under R version 4.5.2
```

Accessing NESP datasets

NESP 5.9 datasets can be accessed via the `arrow` package for R. First, we will need to create a connection with the parent folder containing NESP data as follows:

```
conn <- s3_bucket("s3://data-uplift-public/", anonymous = TRUE,
                  region = "ap-southeast-2")
```

Once a connection has been established, then we can select a specific dataset that interest us. In this case, we will be accessing a dataset about seagrasses. By using the `open_dataset` function from `arrow` we can load either the dataset on its entirety or a subset.

```
url <- "s3://data-uplift-public/stored/datauplift/amsa/dataset"
ds <- open_dataset(sources = url, filesystem = conn,
                   partitioning = schema(year = int16(), source = utf8())
```

The AMSA dataset is partitioned (i.e., separated into parquet fragments), but it is really a single file. We can check the number of files included in a dataset as follows:

```
ds$files
```

We can check the file structure (i.e., schema) of the seabird dataset. This will give us the name of the columns contained in the dataset, which can help us filter the data before accessing.

```
ds$schema
```

From here, we can see there are 16 columns, but we can check how many records in total are included in the dataset as follows:

```
dim(ds)
```

This dataset has a few million records, which may be too much for some computers to handle, but `arrow` allows us to lazily load the dataset. This means that the data is only load into your memory when you request it, instead of loading all records at once. This could be especially useful if you have limited computing resources, so you can subset or summarise the data as you need before loading it to your computer to avoid overloading the computer's memory.

`H3` Aggregation and Mapping

The NESP 5.9 datasets are aggregated data products that cover large regions and time periods. It is useful to first aggregate the datasets so that we may understand their temporal-spatial bounds.

`H3` Indexing

The `H3` library allows us to spatially index datasets at different resolutions (zoom levels) according to the record latitude and longitude.

Visualization Strategy

- **Log-Scaling:** Maritime traffic density varies wildly (Power Law). We apply a transformation to make both ports and open oceans visible.
- **Global Max:** We calculate the maximum density across all years to create a fixed color scale. This ensures that "Red" in 2013 represents the same density as "Red" in 2024, allowing for valid comparisons.

```
# --- H3 Aggregation ---
# Let's check the years available in the dataset
yrs <- ds |>
  mutate(year = year(eventDate)) |>
  distinct(year) |>
  collect()

#Check results
yrs
```

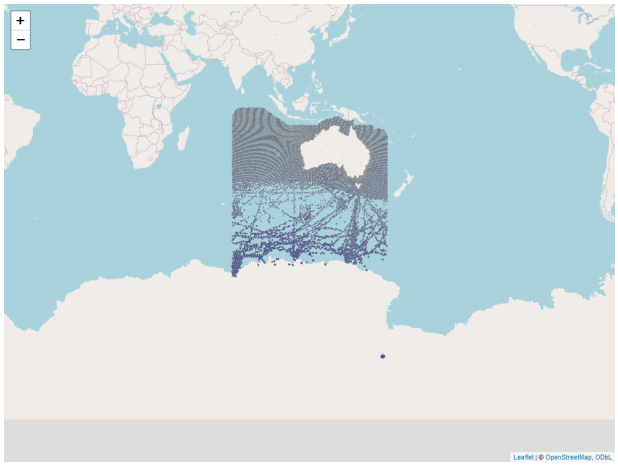
Visualising the `H3` Aggregation

With the data aggregated and the global scale determined, we can now render the spatial density map. We will complete the following steps: - **Filtering:** Selecting the specific year from the aggregated dataset. - **Normalization:** Mapping the Log-Score to a fixed color palette using the GLOBAL_MAX_LOG calculated previously. - **Rendering**: Generating the 3D Hexagon Layer.

```
df_short <- ds |>
  filter(eventDate >= as_date("2013-01-01") & eventDate <= as_date("2013-12-31")) |>
  group_by(h3Index) |>
  summarise(n_records = n()) |>
  collect()

df_short <- df_short |>
  mutate(index = h3_to_geo_boundary_sf(df_short$h3Index)) |>
  ungroup()

df_short
```

```
## # A tibble: 17,741 × 3
##    h3Index         n_records index$h3_index                            $geometry
##    <chr>               <int> <chr>                                <POLYGON [°]>
##  1 84a6115ffffffff      1538 84a6115ffffffff ((113.5193 -21.89119, 113.4917 -22…
##  2 84a613bffffffff      1758 84a613bffffffff ((113.3844 -21.56532, 113.3569 -21…
##  3 84a6131ffffffff     11025 84a6131ffffffff ((113.6025 -21.34837, 113.5748 -21…
##  4 84a6135ffffffff     18038 84a6135ffffffff ((113.8216 -21.12954, 113.7936 -21…
##  5 84a6ac9ffffffff      1334 84a6ac9ffffffff ((114.0416 -20.90882, 114.0134 -21…
##  6 84a6127ffffffff       960 84a6127ffffffff ((114.399 -21.0177, 114.3706 -21.2…
##  7 84a6a11ffffffff      1379 84a6a11ffffffff ((114.8446 -20.56957, 114.8158 -20…
##  8 84a6a15ffffffff      5545 84a6a15ffffffff ((115.0688 -20.34263, 115.0399 -20…
##  9 84a6a39ffffffff      3999 84a6a39ffffffff ((115.294 -20.11377, 115.2648 -20.…
## 10 84a6a67ffffffff    112416 84a6a67ffffffff ((116.6238 -20.1999, 116.5936 -20.…
## # ℹ 17,731 more rows
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = df_short$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(df_short$n_records),
                                         n = 10)(df_short$n_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", df_short$n_records),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.

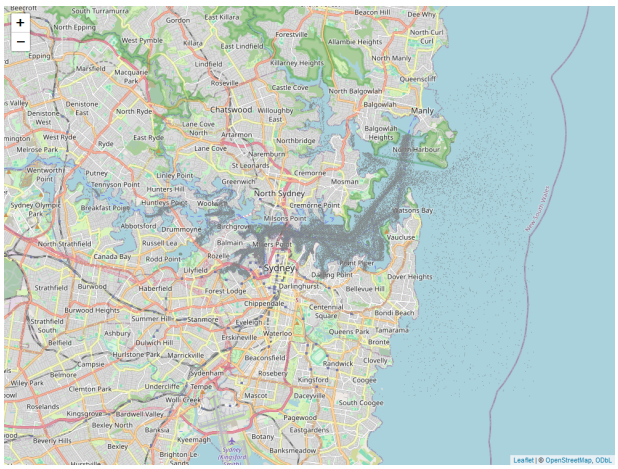
Regional Analysis: Sydney Harbour

While the global map uses a coarse resolution (Resolution 8, approx 0.7km²) suitable for open ocean shipping lanes, complex waterways like Sydney Harbour require finer detail.

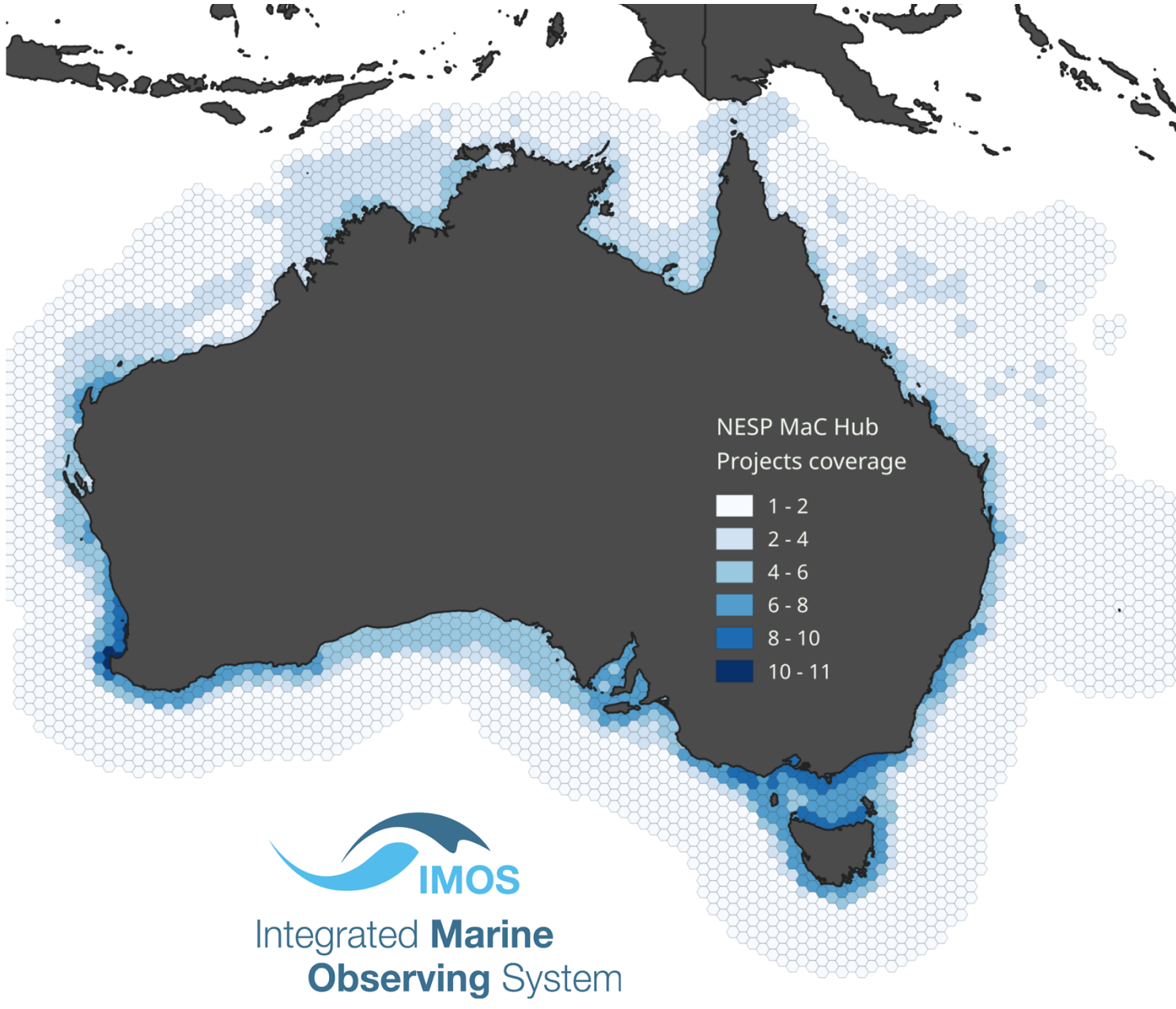
To analyze this region effectively, we perform a Spatial Deep Dive:
1. **Spatial Filter:** We restrict the query to the Sydney bounding box.
2. **Re-Indexing:** We calculate a new H3 Index at Resolution 9 (approx 0.1km²). This allows us to resolve individual ferry routes and harbour traffic that would otherwise be merged into a single block.

```
map_df <- ds |>
  filter(decimalLatitude >= -33.93 & decimalLatitude <= -33.78 &
           decimalLongitude >= 151.1 & decimalLongitude <= 151.35) |>
  filter(eventDate >= as_date("2016-01-01") & eventDate <= as_date("2016-12-31")) |>
  select(!h3Index) |>
  collect()

map_df <- map_df |>
  st_as_sf(coords = c("decimalLongitude", "decimalLatitude"), crs = 4326)

map_df <- map_df |>
  mutate(h3index = geo_to_h3(map_df, 12))

h3_layers <- map_df  |>
  st_drop_geometry() |>
  group_by(h3index) |>
  summarise(number_records = n())

h3_layers <- h3_layers |>
  mutate(index = h3_to_geo_boundary_sf(h3_layers$h3index)) |>
  ungroup()
```

```
m <- leaflet() |>
  addTiles() |>
  # Note that we use quantiles so it is easier to distinguish polygon with
  # a high number of observations
  addPolygons(data = h3_layers$index,
              fillColor = ~colorQuantile("plasma",
                                         domain = unique(h3_layers$number_records),
                                         n = 10)(h3_layers$number_records),
              fillOpacity = 0.7, color = "grey", weight = 1,
              label = paste0("Records: ", h3_layers$number_records),
              labelOptions = labelOptions(style = list("max-width" = "750px",
                                                       "white-space" = "normal")))
```



Screenshot of the map above generated by leaflet.



Marine and Coastal
National Environmental Science Program

CONTACT

Eduardo Klein-Salas
eduardo.kleinsalas@utas.edu.au

imos.org.au

nespmarinecoastal.edu.au

National Environmental Science Program

This project is supported with funding from the Australian Government under the National Environmental Science Program.