Workshop on spatial and space-time models for whale surveys with complex and non-systematic designs

13–15 November 2025

Framsenteret, Tromsø, Norway

DRAFT REPORT

Presented to the 32nd meeting of the Scientific Committee as NAMMCO/SC/32/19

DISCLAIMER:

The content of this report contains the views of the Workshop participants and does not necessarily represent the views of the NAMMCO Scientific Committee or Council.

Cite this report as: NAMMCO (2025). *Report of Name of the meeting* (NAMMCO/SC/WS/2025-01). NAMMCO-North Atlantic Marine Mammal Commission. Tromsø. Norway. 73 pp. https://nammco.no/wp-content/uploads/2026/06/report_modelling-workshop-for-whale-surveys-with-complex-designs_2025-01.pdf

All reports of the Scientific Committee working groups and workshops are available at https://nammco.no/scientific-working-group-reports/

Co-convenors

Martin Biuw, Hans Skaug, Hiroko Solvang

Authors

Ana Cañadas, Deanna Leonard, Frederike Boehm, Guðjón Már Sigurðsson, Hans Skaug, Hiroko Solvang, Jafet Belmont Osuna, Jason Roberts, Len Thomas, Maria Garagouni, Martin Biuw, Olav Nikolai Risdal Breivik, Ricardo de Almeida Mendes, Takai Katsumata, Tiago Marques, Toshihide Kitakado

North Atlantic Marine Mammal Commission

Postbox 6400, N-9294; Visitors: Sykehusveien 21-23, N-9294; Tromsø, Norway

nammco-sec@nammco.org | www.nammco.org

http://www.instagram.com/nammco_org/

1 WELCOME AND PROBLEM STATEMENT

The co-convenors of the Workshop, Martin Biuw, Hans Skaug, and Hiroko Solvang welcomed participants to snowy Tromsø. Biuw outlined the goals of this meeting, namely, to discuss methods, software, and best practice for analysing data from line transect cetacean surveys with complex designs, using spatial and potentially spatial-temporal models. The motivation for organising this meeting stems from certain issues that arose in the latest North Atlantic Sightings Survey (NASS) for cetaceans in general, and the latest cycle of the Norwegian Independent Line-transect Survey (NILS), for primarily minke whales. During NASS 2024, some pre-determined survey blocks and transect lines were not covered, due to consistently poor weather conditions in key regions. Additionally, while some vessels made use of the independent observation (IO) protocol by using two observation platforms, and followed a transect design optimised for whale surveys, others used a single platform and followed fixed transects not designed for cetacean surveys. For geopolitical reasons, key regions in the eastern part of the Barents Sea could not be covered during the NILS 2020-2024 survey cycle.

2 METHODS / SOFTWARE PACKAGES

2.1 DENSITY SURFACE MODELLING

Jason Roberts introduced the density surface modelling (DSM) approach, as well as an example of its application to sperm whales in the western North Atlantic.

Summary

DSM is a two-stage modelling process used to predict species density (as well as associated uncertainty) within a region, combining species observations with oceanographic features in that area. The first stage of the process is similar to design-based abundance estimation, in that the detectability of the study species is modelled using distance sampling. The detection function is fitted to observed perpendicular distances, also considering other factors that could influence detectability, e.g., sea state, weather, group size, etc. With the right data, detection probability can also be corrected for perception and availability biases. The second stage of the process incorporates explicit spatial covariates in a generalised additive modelling (GAM) framework. Survey transects are split into segments and each segment is then assigned the values of spatial covariates at that location (e.g., segment centroid). The detection function selected in stage one is then predicted for each segment or each observation (ideally the former). Finally, the observations are modelled as a function of the environmental covariates, model fit is checked and the most appropriate covariates are retained, using which, predictions of species density can be made for a given area.

Appealing traits of DSM include that: it is an established approach supported by an active community and training resources; it does not require statistical expertise to run; it can tolerate heterogeneous sampling. The GAM component is advantageous because it permits non-linear relationships and (some) ecological interpretation of environmental influences; but it can also be replaced by different methods if needed. However, it can be challenging to run a two-stage approach like this, for instance when it comes to propagating the detection function variance from the first stage to the final output of the second stage. The method is also not ideal for combining different data sources, such as visual, acoustic, and telemetry data, into a single model.

Discussion

DSM is a two-stage modelling process, i.e., the detection probability is estimated separately from the spatial modelling process. As these stages are independent from each other, it is entirely possible to use the same covariates for both, or entirely distinct ones—as long as they are actually relevant for each.

2.2 INLABRU

Jafet Osuna presented the R packages Inla and Inlabru and their capabilities.

Summary
INLA is a fast and accurate framework for Bayesian inference in latent Gaussian models. inlabru is an R package that implements INLA using a flexible and user-friendly interface, facilitating the specification of hierarchical Bayesian models with spatial and spatiotemporal dependencies. The package also provides a log-Gaussian Cox process likelihood for modeling univariate and spatial point processes derived from ecological survey data, including distance sampling. Unlike traditional density surface models, inlabru treats observed sightings as realisations of a point process with a continuously varying spatial intensity, using a single-step approach that propagates uncertainty from the observation process into the underlying ecological process.

Transect-level covariates can be incorporated into the detection function as fixed or random effects, and the specification of multiple observation models enables the integration of multiple data streams to infer a shared latent ecological process. Furthermore, inlabru supports marked point process models, allowing mark attributes such as species identity or group size to be explicitly included in the analysis.

Despite its flexibility and computational efficiency, careful model specification in inlabru is needed. In particular, users must pay close attention to the choice of prior, ensuring they are appropriate for the scale and interpretation of model parameters. Spatial components should be defined using a consistent and correct coordinate reference system (CRS), and the spatial units of the mesh, covariates, and observation data must be compatible. Covariates should be meaningfully scaled and interpreted relative to these spatial units, and prior assumptions should be evaluated considering both ecological knowledge and the spatial resolution of the data.

Discussion
It was mentioned that, when penalising a prior, there were different approaches of specifying it. In the discussion, it was questioned how an approach should be picked. Jafet responded that penalizing a prior was not the same as setting a Bayesian prior and that it was more like informing a model of how likely the prior was to shift. Therefore, if the existing problem was not understood, a conservative approach could be taken, [Something about setting a probability of 50 and fine tuning it, depending on how the model runs. Alternatively, the probability could be set to be unconstrained.]

An example of the use of Inlabru was given from a case study in the Gulf of Mexico, based on a fishery survey, where dolphin sightings were collected. A comment was made on how the approach used seem to be discretising the data. Jafet recognized it as being sort of the case, and that the intention was to avoid using infinite computing time.

The total number of groups estimated for the entire Gulf of Mexico by the modelled results was questioned by one of the participants, as this was, approximately, three times higher than the numbers on the transect strips. This was considered as being too low and that it could indicate a possible issue with the units used.

There was a discussion about the appropriateness of extrapolation, as this is of interest to Norway´s case. It was questioned how Inlabru should be set up and how it would work. Interpolation would be much easier to work with than extrapolation and would provide significantly smaller uncertainty.

It was discussed how density would be sampled in extrapolation. Sampling would be done directly from the density within the boundary.

Another point that was brought forward was how the method would hold up, if extrapolation was applied to the whole Gulf of Mexico, from US data only. The extrapolation would work well, as long as there was a massive assumption that all the environmental covariates are the same across the entire area. Without covariate information, the expected behaviour of any model should be that it would default to the mean with ever increasing variance, so extrapolation would not be an issue.

2.3 MULTIPLE SCALE LINE TRANSECT ANALYSIS WITH TEMPLATE MODEL BUILDER

Olav Breivik presented the recently published Multiple Scale Line Transect (MSLT) package (Breivik et al. 2025). The package makes use of Template Model Builder (TMB), and expands on point process modelling with additional variation for predetermined high- and low-density areas.

Summary

A spatial model implemented in TMB was presented. The model was recently published in JASA (Breivik et al., 2025). It treats whale sightings as arising from a thinned Cox process, with intensity driven by a latent Gaussian Markov random field (using the SPDE approach) and a two-state Markov-modulated Poisson process (MMPP). The SPDE component is intended to capture long-range spatial structures, wheras the MMPP accounts for short-range structure.

It was demonstrated that including the MMPP component typically increases the spatial correlation length of the SPDE component. This implies that the model can propagate the SPDE process further away from the transects when the MMPP is successfully included. The model jointly estimates the detection function, group intensity, and group size, ensuring that uncertainty is propagated automatically through the delta method implemented in TMB.

Discussion

Some advantages of the TMB approach include: i) the ability to penalise priors, including the prior for spatial scale; ii) that variance propagation is automatic, meaning that the model will calculate accordingly if the user determines that there is covariance, e.g., in group size and encounter rate; and iii) that residual error is reduced by incorporating the MMPP. It was noted that the model including MMPP reverts to the mean more slowly than the model that does not include it, resulting in a higher overall estimate.

Theoretically, it would be possible to include multiple SPDE setups at different scales (as a substitute for the MMPP), although that would require a much larger number of integration points, and would thus being computationally unfeasible.

3 METHODOLOGICAL CONSIDERATIONS

3.1 ENVIRONMENTAL COVARIATES

Ana Cañadas presented an overview of species density modelling (SDM) with examples from several species and different areas of the Atlantic.

Environmental Covariates in Species Density Modelling

Species density modelling links animal density to environmental features, providing spatially explicit estimates of distribution and abundance. Unlike traditional distance sampling, SDMs incorporate environmental realism by revealing *why* species occur where they do. They combine dynamic and static environmental covariates to model ecological processes, allowing prediction across unsurveyed regions, hypothesis testing, and integration into conservation and management tools, such as marine protected area (MPA) design and risk mapping.

Environmental covariates are central to SDMs—they bridge survey data and ecological processes. Choosing appropriate covariates determines model realism and predictive strength. Covariates can be:

- **Static** (unchanging): bathymetry, slope, seabed type, or distance to coast—representing persistent habitat features.
- **Dynamic** (time-varying): sea surface temperature (SST), chlorophyll-a, salinity, and currents—capturing temporal changes in productivity and prey availability.

Temporal resolution is critical. Models must match ecological and data scales, accounting for whether to use contemporaneous data (showing fine temporal variability but prone to missing data) or climatologies (averaged data that smooth short-term variation but reduce computational demand).

Analyses may test sensitivity to temporal aggregation and consider lagged biological responses to environmental change.

Challenges include dealing with missing pixels (e.g., from cloud cover), aligning datasets of different spatial or temporal resolutions, and ensuring consistent coordinate reference systems and formats. Transparent and reproducible data preprocessing—reprojecting, resampling, and masking to the study area—is essential.

When predicting, covariates must undergo the same transformations as used during model fitting. For dynamic models, monthly or seasonal prediction surfaces allow visualization of temporal patterns, while climatologies yield average distribution maps. Caution is needed when extrapolating beyond sampled environmental conditions (the extent of extrapolation should be explored); *Winsorizing* extreme values helps minimize errors.

Ultimately, effective use of environmental covariates enhances understanding of species–habitat relationships, supports management decisions, and strengthens ecological inference by explicitly linking animal distributions to the physical and biological environment.

3.2 GENERAL CONSIDERATIONS FOR ALL METHODS

The primary consideration of any spatial modelling project should be to **define the objectives**. The project objectives influence the appropriate spatio-temporal scale of inference, which in turn influences the appropriate spatio-temporal scale of covariates. Any predictions from a model must be made at the same scale as the one at which the model was fit, so it is important to define this scale *a priori*. For example, for the purposes of an ecological impact analysis or risk assessment, the objective may be to predict animal density at small time scales (days or weeks) and small spatial scales (kilometres or tens of kilomtres). On the other hand, for the purposes of a stock assessment, the objective may be to predict density at large time scales (annual) and large spatial scales (entire stock/population)—and spatial-temporal modelling would be used in place of a design-based abundance estimate if the available data were not generated from a well-designed randomised survey. In such a case, we may choose not to try to use covariates to explain ephemeral or small-spatial-scale variation since we are not interested in predicting using this information; instead we may model such variation as a random process and then average over it when making predictions.

When selecting a modelling approach, there are multiple **one- or two-stage methods** to choose from. Traditionally, a two-stage process, such as DSM, allows for modelling the detection function first, potentially using specialised software, and then feeding that into the second stage, spatial encounter rate modeling. Information about perpendicular distances is not informative about larger-scale spatial density patterns, so there is little two-way feedback between approaches, meaning nothing is lost in terms of information with this approach. One difficulty, though, is propagating uncertainty from stage 1 into stage 2. Methods have been developed to address this but are not fully satisfactory (e.g., don't fully cover animal-level covariates). A one-stage process makes uncertainty propagation easy (as it is automatically integrated into the modelling), but less flexible detection function tools are available, model selection may be harder, and fitting takes longer to run, which discourages experimentation with detection function models.

A potential issue with unevenly distributed effort relative to whale density is increased uncertainty in the resulting estimates. For example, in NASS 2024, the ice cover in several survey blocks near Greenland meant that only a narrow strip was surveyed, in an area where whales were likely concentrated against the ice edge waiting for the ice to recede (in contrast with their sparser distribution in other parts of the NASS region). The resulting imbalance would increase the variance in areas with a lower concentration of sightings.

4 CASE STUDIES

4.1 NORTH ATLANTIC SIGHTINGS SURVEYS – 2015

Data summary

The original dataset from Iceland and the Faroe Islands included data from both the 2015 and 2024 NASS surveys. Due to completeness and more species being available in the 2015 dataset, the trial runs of the different modelling methods were used on the 2015 data. The 2015 survey was conducted in June/July 2015 and covered a large area of the northern North Atlantic. The Icelandic and Faroese ship survey component of the NASS covered the area between the Faroe Islands and East Greenland from latitude 52° to 72° N. The survey used three vessels, two in Iceland and one in the Faroes. In Iceland, the surveying was done partly by a joint fisheries survey, were redfish or mackerel was surveyed alongside the cetaceans. In all cases independent double-platform configuration was used, with each platform staffed by a minimum of 2 observers. The process and estimates from the traditional distance sampling based on this data is described in detail in Pike et al. (2019).

Applying DSM to NASS 2015 data

During the workshop DSM was applied to the NASS 2015 data in a preliminary study (see Appendix 3 for details). To the extent that time allowed, model selection and diagnostic checking were performed and a plot of the whale density surface produced, with a corresponding estimate of 40,337.54 fin whales. The fitted density surface is relatively smooth, being partly constrained by the number of basis functions used. The estimated abundance also corresponds well with a previous estimate of 36,773 (CV = 0.17, 95% CI: 25,811–52,392) in Faroese and Icelandic waters and 3,729 (CV=0.44, 95% CI: 1,531–9,081) in Norwegian waters, obtained using a standard design-based approach (Pike et al., 2019).

Applying TMB to NASS 2015 data

During the workshop, data were converted to the input format used by TMB. Then TMB was run but produced a much more variable density surface than DSM, when using the SPDE-only option of TMB. This is likely due to constraints imposed in the DSM analyses, in terms of the number of knots controlling the smoothness. The abundance estimates produced by TMB were substantially lower than from DSM, and in the discussion that followed it was speculated what the reason for this was. One difference between the two analyses was that in DSM different functional forms of the detection function were used for the Norwegian and Icelandic parts of the surface. The current implementation of TMB does not allow different functional forms (half-normal and hazard rate) for the detection function in different parts of the survey region. This is, however, unlikely to fully explain the difference between the two estimates, and further investigation is needed. Different truncation distances were also used in the DSM analysis for each of the survey components, thus including a higher number of sightings than TMB.

4.2 COMMON DOLPHIN SURVEYS IN PORTUGAL

This case study combines information from a structured line transect distance sampling survey and whale watching data to make inferences about spatio-temporal distribution of short-beaked common dolphins (*Delphinus delphis*) off mainland Portugal. In that sense, it was the only case study that would allow to implement data integration approaches. The distance sampling survey data, which was used for modelling during the workshop, corresponded to a line transect survey conducted along mainland Portugal, comprising a total of 30 observations of groups of dolphins. The case study's data is well described in the preprint by Klaassen et al. (2025), where it is used as an illustration of data integration. A fully worked out analysis of the case study using inlabru was provided to participants and can be found as an appendix to this report.

Klaassen et al. (2025) considered modelling the intensity of groups, not the intensity of animals, which is usually of higher interest, the work developed during the workshop addressed the incorporation of group size in the model. This was implemented in the inlabru framework. Given that the focus was not

understanding which covariates were relevant to model spatial density, but only to try to include group size in the spatial model, we opted by using only the (depth) slope in the spatial model, allowing the group size to be described by a Gaussian random field sharing spatial information with the density of groups. While the approach seems to have worked, inlabru does not currently allow for the inclusion of group size affecting the detection function, which is often the case and therefore remains a hurdle to the widespread use of the method for real life case studies. Additionally, this case study did not include double observers to estimate perception bias, that is, the probability of detecting a group on the transect line. As for including group size as a covariate influencing detectability, in general it would be good to incorporate double observer methods in the inlabru framework. While that might be possible by allowing a different observation model, that remains to be implemented in inlabru.

4.3 MINKE WHALE SURVEYS – EASTERN BARENTS SEA

Data summary

The Northeast Atlantic common minke whale, a species hunted commercially in Norway, is managed based on the International Whaling Commission's Revised Management Procedure (RMP). From 2014 to 2019, annual surveys were conducted covering the entire northeast Atlantic, from the North Sea (southern boundary 52˚N) to the ice edge, and from the Greenland Sea in the west to the Barents Sea in the east. The data covered in this workshop are online-transect sighting data from the small management area EB surveyed in 2017. Minke whales were searched by naked eye by two independent observers standing on two platforms at different heights. All observation records were assigned GPS time and location information, and relevant covariate such as weather, visibility, Beaufort wind force etc. were also recorded. For further details on the data one is referred to Solvang et al. (2021). For spatial abundance estimation, the DSM and inlabru applications utilise initial observations from the upper platform (typically using a barrel on the mast).

Background to the presented case study

The dedicated survey conducted from 2014 to 2019 covered all relevant management areas, excluding the Russian exclusive economic zone (EEZ) in the Barents Sea management area. The absolute abundance estimates for the survey periods 2002–2007 in the Barents Sea area showed an increasing trend, and the most recent estimates for 2014–2019 accounted for over 50% of the entire eastern sea area subject to catch quota calculations in Norway. Although RMP guidelines stipulate that abundance estimates for areas without dedicated surveys should be set to "none", we attempt to extrapolate the abundance estimates for these areas using spatial abundance models, incorporating greater uncertainty to provide estimates for uncovered area. Therefore, the case study took 35 initial observations in covered areas, named EB2a and EB3a and extrapolated the relative abundance estimates in the uncovered area, named EB2b and 3b.

Using DSM

As the first step in DSM, detection function models were applied to the line transect observations for EB2a and EB3a. The hazard rate detection function was judged by AIC to be a better model than the half-normal model. In models including covariates, the hazard rate detection function model incorporating visibility and Beaufort wind force proved better than models without covariate/with other covariates combinations. In the second step of the DSM, the statistical properties of the response variable, counted number, within the GAM were considered. As the expected value of counted number was close to the variance of counted number, the link functions defined in GAM were applied to Poisson distribution, a negative binomial distribution with dispersion of 0.1, and the Tweedie distribution, known for its robustness when the data are zero-inflated. GAMs were applied to EB2a and 3a incorporating a spatial smoothing term alongside static covariates such as water depth, distance from coastline, slope, and standard deviation of slope. The BIC selected GAM including a spatial smoothing term and the smoothed covariates for distance to coastline as the optimal model. The estimated interceptions were largely identical across the three link functions setting, the p-values for the two smoothing terms were significant, whereas they were not significant for the negative

binomial distribution. However, the two smoothing terms showed a consistent pattern across all models.

Each spatial abundance estimate for covered and uncovered areas was similar among the settings of Poisson, Tweedie, and negative Binomial. For the link function in GAM using restrict maximum likelihood estimation (REML), the default setting is quasi-Poisson distribution; however, the case indicated unidealistic large abundance estimates values in covered and uncovered areas. If a smaller number of base for the thin plate smoothing functions were selected by model selection based on the Laplace approximation of quasi log-likelihood and effect of degree of freedom values (that was manually calculated), the spatial abundance was similar number as the three distribution settings in covered/uncovered area.

During the discussion, the following points were proposed:

- Estimation of detection function using data encompassing broader regions, the MRDS method (Miller et al. 2021), and hpm method (Skaug, et al. 2004).
- Shrinkage for covariate in GAM (Mara and Wood 2011), which are expected better prediction.
- The ratio of model-based abundances to design-based abundance (Becker et al. 2010).

Using INLABRU

Code incorporating spatial covariates and detection function with transect covariates is available. As this method is based on Bayesian modelling, technical challenges exist that do not directly correspond to the frequentist framework seen in DSM. First, setting an appropriate prior for hazard rate detection function and treating spatial covariates as random effects is required. Plugging the detection function identified by other methods into inlabru appears difficult. Furthermore, extrapolation to uncovered area should be conducted using a model applied to the entire area encompassing both covered and uncovered areas, differing from the DSM application presented in this workshop. Considering these points, we plan to examine priors suited to the data characteristics and calculate the spatial abundance estimates.

5 POINTS FOR FURTHER CONSIDERATION, BEST PRACTICE, APPLICABILITY, AND CAVEATS FOR DIFFERENT METHODS ON DIFFERENT DATA TYPES

Several points were raised throughout the course of the workshop, which warrant further exploration and discussion beyond the scope of this meeting.

In the context of Bayesian versus frequentist (maximum likelihood) inference, no MCMC-based Bayesian approach was presented at this workshop, possibly because such models tend to be slow. Model selection in a Bayesian framework may also be considered less straightforward than in an ML framework. The role of Bayesian priors can be essentially fulfilled by penalisation in the ML approach.

Some issues with distance sampling were brought up, including:

- The importance of adequately dealing with spatial variation in detectability via detection function covariates—and adequate sample size of detections—and of flexible detection function modelling in general, for robust inferences on density
- Dealing with g0<1, preferably in an integrated way
- Modelling group size.

For spatial modelling approaches in general, different tools exist that can account for different issues; the most appropriate one should be selected carefully. One important consideration is how to deal with residual spatial autocorrelation (something GAMs are not traditionally designed for). One approach to deal with this is SPDE, while MMPP allows a further layer of spatial auto-dependence, allowing the user to separate small-scale variation in density that may not be of interest for prediction from larger-scale variation—even if the larger-scale variation is not directly of interest, modelling

autocorrelation at two scales may produce a better-fitting model). Another consideration is taking account of matrix sparseness, for which point-process models may be more suitable. Finally, including spatio-temporal models of group *size* in the same models as group *density* could potentially allow the two to covary.

The importance of carefully selecting model covariates was reiterated several times. For temporal models, it is necessary to distinguish between finer-scale models (e.g., attempting to account for variation between seasons) and longer-term (e.g., annual) trend models. Furthermore, while it is possible to extrapolate to future scenarios, it is imperative to consider that the environmental conditions of the future may not have been encountered in the measured (and modelled) covariates heretofore—both predictions and interpretations must therefore be treated with extreme caution. Similar caution should be used for spatial extrapolation, in selecting both the correct covariates and the appropriate way to deal with data gaps. Finally, care should be taken when propagating uncertainty in the covariate values.

Incorporating multiple types of survey data was touched upon at the present meeting, but should be the subject of a follow-up NAMMCO workshop. Many types of data could be informative about absolute and/or relative density, and several up-and-coming new data collection methods could be explored.

REFERENCES

Breivik, O. N., Skaug, H. J., Jullum, M., & Biuw, M. (2025). Spatial Variation on Multiple Scales in Line Transect Data; the Case of Antarctic Fin Whales. *Journal of the American Statistical Association*, 1–13. https://doi.org/10.1080/01621459.2025.2566422

Klaassen M., Fernandez M., Lindgren F., Morera-Pujol V., Thomas L., Oliveira N., Castro J., Martinho F., Magalhaes S., Rodrigues A., Martins M., Alves F., Marques T. A. 2025. Spatiotemporal data integration for marine megafauna SDMs in dynamic environments: A point process approach. Bioarxiv. DOI: https://www.biorxiv.org/content/10.1101/2025.11.07.687170v1

Pike, D. G., Gunnlaugsson, T., Mikkelsen, B., Halldórsson, S. D., & Víkingsson, G. (2019). Estimates of the Abundance of Cetaceans in the Central North Atlantic based on the NASS Icelandic and Faroese Shipboard Surveys Conducted in 2015. *NAMMCO Scientific Publications*, *11*. https://doi.org/10.7557/3.4941

Solvang, H.K., Skaug, H. and Øien N., Abundance of common minke whales in the Northeast Atlantic based on survey data collected over the period 2014-2019, SC/68C/ASI/04, 2021.

APPENDIX 1: AGENDA

1. **Welcome and problem statement**
2. **Case studies**
   - 2.1. NASS 2024
   - 2.2. Norwegian minke whale survey in Eastern Barents Sea 2017
   - 2.3. Integrating different data types: common dolphins in Portugal
   - 2.4. Sperm whales in the Gulf of Mexico
3. **Methods**
   - 3.1. DSM
   - 3.2. Inlabru
   - 3.3. TMB
4. **Applications**
   - 4.1. Species distribution modelling
   - 4.2. Extrapolation to unsurveyed areas
   - 4.3. Hands-on work
5. **Plenary discussion**
6. **Outcomes and conclusions**
7. **Workshop end**

APPENDIX 2: LIST OF PARTICIPANTS

NAMMCO MEMBER COUNTRIES

Deanna Leonard (NO)
Institute of Marine Research

Hiroko Solvang (NO)
Institute of Marine Research
hiroko.solvang@hi.no

Frederike Boehm (NO)
Institute of Marine Research
frederike.boehm@hi.no

Martin Biuw (SC, NO)
Institute of Marine Research
martin.biuw@hi.no

Gudjón Már Sigurdsson (SC, IS)
Marine and Freshwater Research Institute
gudjon.mar.sigurdsson@hafogvatn.is

Olav Nikolai Breivik (NO)
Norwegian Computing Centre
olavbr@nr.no

Hans Skaug (NO)
University of Bergen
hans.skaug@uib.no

INVITED EXPERTS

Ana Cañadas
Duke University
anacanadas65@gmail.com

Taiki Katsumata
Institute of Cetacean Research
katsumata@cetacean.jp

Jafet Belmont Osuna
University of Glasgow
jafet.belmontosuna@glasgow.ac.uk

Tiago Marques
University of St Andrews
tiago.marques@st-andrews.ac.uk

Jason Roberts
Duke University
jason.roberts@duke.edu

Toshihide Kitakado
Tokyo University
kitakado@kaiyodai.ac.jp

Len Thomas
University of St Andrews
len.thomas@st-andrews.ac.uk

NAMMCO SECRETARIAT

Maria Garagouni
Deputy Secretary
maria@nammco.org

Ricardo de Almeida Mendes
Intern
intern@nammco.org

NASS 2015 Prototype Density Surface Model

Jason J. Roberts Ana Canadas Deanna Leonard Guðjón Már Sigurðsson
Maria Garagouni Ricardo de Almeida Mendes

2025-11-14

1 Introduction

This notebook develops a prototype density surface model (Miller et al. 2013) for fin whales (and minke and pilot whales, if we have time) from 2015 shipboard NASS surveys from Norway, Iceland, and the Faroe Islands. This example was prepared during working sessions at the workshop *"Spatial and space-time models for whale surveys with complex and non-systematic designs"* held in Tromsø, Norway on 13–15 November 2025, convened by Hans Skaug, Martin Biuw, and Hiroko Solvang.

The main purpose of this notebook is to assist the workshop attendees in understanding how to apply DSM modeling techniques and R code to NASS data, and to see what the results might look like. The results developed here are not intended to provide credible abundance estimates, detection functions, or density maps, and should not be used by anyone for any scientific or management purpose without advice from the authors.

1.1 Requirements

To run this notebook, you need R plus R Studio or similar software capable of rendering a Quarto markdown document. This notebook was originally developed with R 4.5.1 and packages that were current as of November 2025. You also need access to the data files that accompany this notebook's .qmd file. This writeup assumes you have expertise in R programming, statistical modeling, line transect survey methods, and marine mammal ecology.

2 R Packages

This analysis makes use of numerous packages. We're loading them in alphabetical order and suppressing their startup messages to keep the rendered document clean, but you should examine the messages if you are concerned about them masking each other's functions.

```
suppressPackageStartupMessages({
  library(Distance)     # To fit detection functions
  library(dsm)          # To fit density surface models as GAMs from mgcv
  library(gratia)       # For nice, ggplot-style plots of GAMs
  library(kableExtra)   # To style tables for PDF rendering
  library(patchwork)    # To glue plots together into a single figure
  library(plotrix)      # For the color.legend() function
  library(RColorBrewer) # To access color scales
  library(sf)           # To manipulate vector-based GIS data
  library(stars)        # To convert rasters to objects tmap can handle
```

```
  library(terra)        # To manipulate raster-based GIS data
  library(this.path)    # To conveniently form paths to this example's data files
  library(tidyverse)    # For data processing and plotting
  library(tmap)         # To make maps
  library(viridisLite)  # To access color scales
})
```

3 Load the data

Deanna provided a CSV of observations and of segments containing data for all surveys.

```
segments <- file.path(this.path::here(), "..", "segments_with_env_2015_xy.csv") |>
  read_csv(show_col_types = FALSE) |>
  tibble() |>
  # Limit to NASS surveys, which were only in 6,7,8
  filter(Month %in% c(6,7,8)) |>
  # Set column types
  mutate(
    Survey = factor(if_else(Vessel_Code == "FTR", "Norway", "Ice_Far")),
    Vessel_Code = factor(Vessel_Code),
    TransectID = factor(TransectID),
    #DateTime = as.POSIXct(DateTime, format = "%d/%m/%Y %H:%M", tz = "UTC"),
    #Date = as.Date(DateTime, format = "%d/%m/%Y"),
    Month = factor(Month, levels = c(6, 7, 8)),
    Beaufort = factor(Beaufort),
    Weather = factor(Weather),
    Platform = factor(Platform)
  ) |>
  # Derived columns
  mutate(
    Survey_Type = if_else(Vessel_Code %in% c(74, 75), "Fisheries", "Dedicated") |>
      factor()
  )

obs <- file.path(this.path::here(), "..", "NASS_2015_sightings_combined.csv") |>
  read_csv(show_col_types = FALSE) |>
  tibble() |>
  # Limit to NASS surveys, which were only in 6,7,8
  filter(Month %in% c(6,7,8)) |>
  # Set column types
  mutate(
    Survey = factor(Survey),
    Vessel_Code = factor(Vessel_Code),
    TransectID = factor(TransectID),
    #DateTime = as.POSIXct(DateTime, format = "%d/%m/%Y %H:%M", tz = "UTC"),
    #Date = as.Date(DateTime, format = "%d/%m/%Y"),
    Month = factor(Month, levels = c(6, 7, 8)),
    Latin.name = factor(Latin.name),
    English.name = factor(English.name),
    Species = factor(Species),
    Side = factor(Side),
```

```
    Beaufort = factor(Beaufort, levels = levels(segments$Beaufort)),
    Weather = factor(Weather, levels = levels(segments$Weather)),
    Block = factor(Block),
    Platform = factor(Platform)
  ) |>
  # Derived columns
  mutate(
    Survey_Type = if_else(Vessel_Code %in% c(74, 75), "Fisheries", "Dedicated") |>
      factor()
  )
```

```
pred_grid <- file.path(this.path::here(), "..", "prediction_grid_25km_3413_xy.gpkg") |>
  st_read()
## Reading layer `prediction_grid_25km_3413_xy' from data source
##    `C:\jjr8\DenMod\Collaborations\Norway_workshop_2025\Case study - NASS data\NASS_2015\prediction_
##    using driver `GPKG'
## Simple feature collection with 6709 features and 23 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 193106 ymin: -4051362 xmax: 2343106 ymax: -551362.3
## Projected CRS: WGS 84 / NSIDC Sea Ice Polar Stereographic North

segment_lines <- file.path(this.path::here(), "..", "segments_with_env_2015.gpkg") |>
  st_read() |>
  st_transform(3413)
## Reading layer `segments_with_env_2015' from data source
##    `C:\jjr8\DenMod\Collaborations\Norway_workshop_2025\Case study - NASS data\NASS_2015\segments_wi
##    using driver `GPKG'
## Simple feature collection with 2469 features and 47 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -44.31356 ymin: 52.11126 xmax: 27.97675 ymax: 73.98309
## Geodetic CRS:  WGS 84
```

4 Detection Modeling

We decided to build separate detection functions for each combination of species and NASS organization (Iceland/Faroe and Norway). If data were not sufficient for this, we will pool surveys across the two organizations or across species, as determined by expert opinion of the members of our working group.

4.1 Fin whales

4.1.1 Icelandic/Faroese surveys

4.1.1.1 Filter the segments

```
obs_fin_ice <- obs |>
  filter(English.name == "fin whale",
         Survey == "Ice_Far",
         Beaufort %in% c(1,2,3,4,5,6))    # Not enough effort in Beaufort > 6
```

4.1.1.2 Choose a truncation distance

To choose the truncation distance, we built a hazard rate detection function with no adjustments or covariates and then visually selected a distance at which detection probability dropped to about 0.1, following advice in Buckland et al. (2001) and L. Thomas.

```
# If the ds() function reports Warning: Unknown or uninitialised column: `distbegin`
# we can ignore it.

dfunc <- ds(obs_fin_ice, truncation = max(obs_fin_ice$distance), key = "hr", adjustment = NULL)
## Warning: Unknown or uninitialised column: `distbegin`.
## Warning: Unknown or uninitialised column: `distend`.
## Warning: Unknown or uninitialised column: `distbegin`.
## Fitting hazard-rate key function
## AIC= 10750.156
## No survey area information supplied, only estimating detection function.
plot(dfunc)

fin_ice_trunc_dist <- 5000
obs_fin_ice_retained <- obs_fin_ice[obs_fin_ice$distance < fin_ice_trunc_dist, ]
print(sprintf("Observations truncated: %d (%.1f %%)",
              nrow(obs_fin_ice) - nrow(obs_fin_ice_retained),
              (nrow(obs_fin_ice) - nrow(obs_fin_ice_retained)) / nrow(obs_fin_ice) * 100))
## [1] "Observations truncated: 15 (2.3 %)"
```

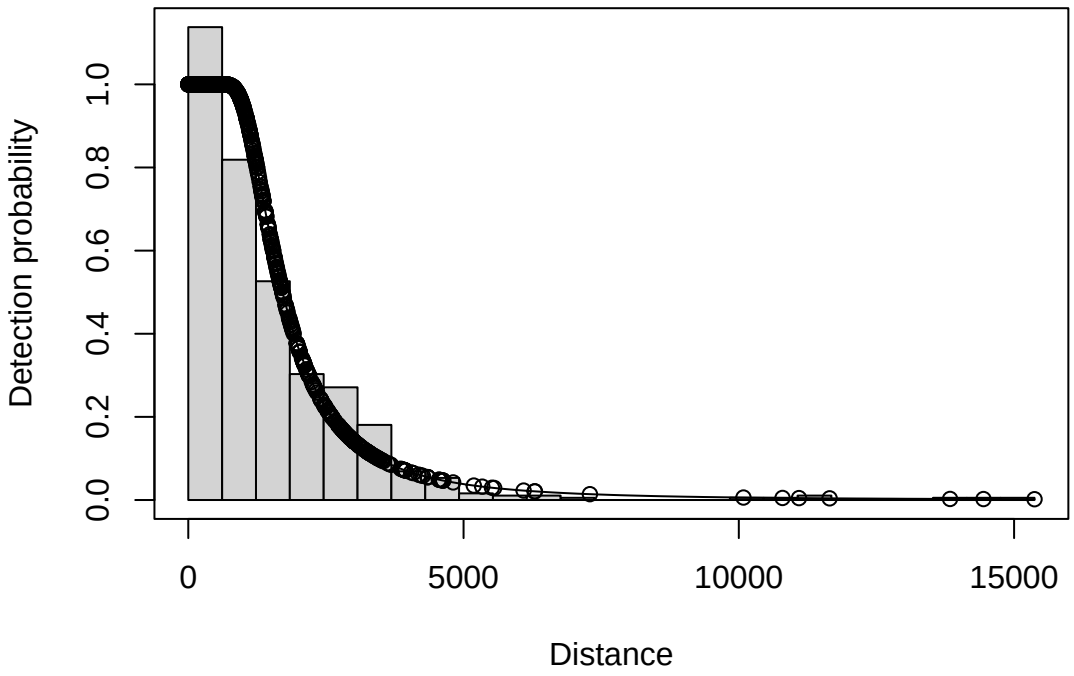


Figure 1: For Icelandic/Faroese surveys, detection function used to select the fin whale truncation distance.

We chose a truncation distance of 5000 m, which resulted in the truncation of 2.3% of the sightings.

4.1.1.3 Explore detection covariates

4.1.1.3.1 Beaufort

```
p1 <- obs_fin_ice_retained |>
  ggplot(aes(x = Beaufort)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_ice_retained |>
  ggplot(aes(x = Beaufort, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

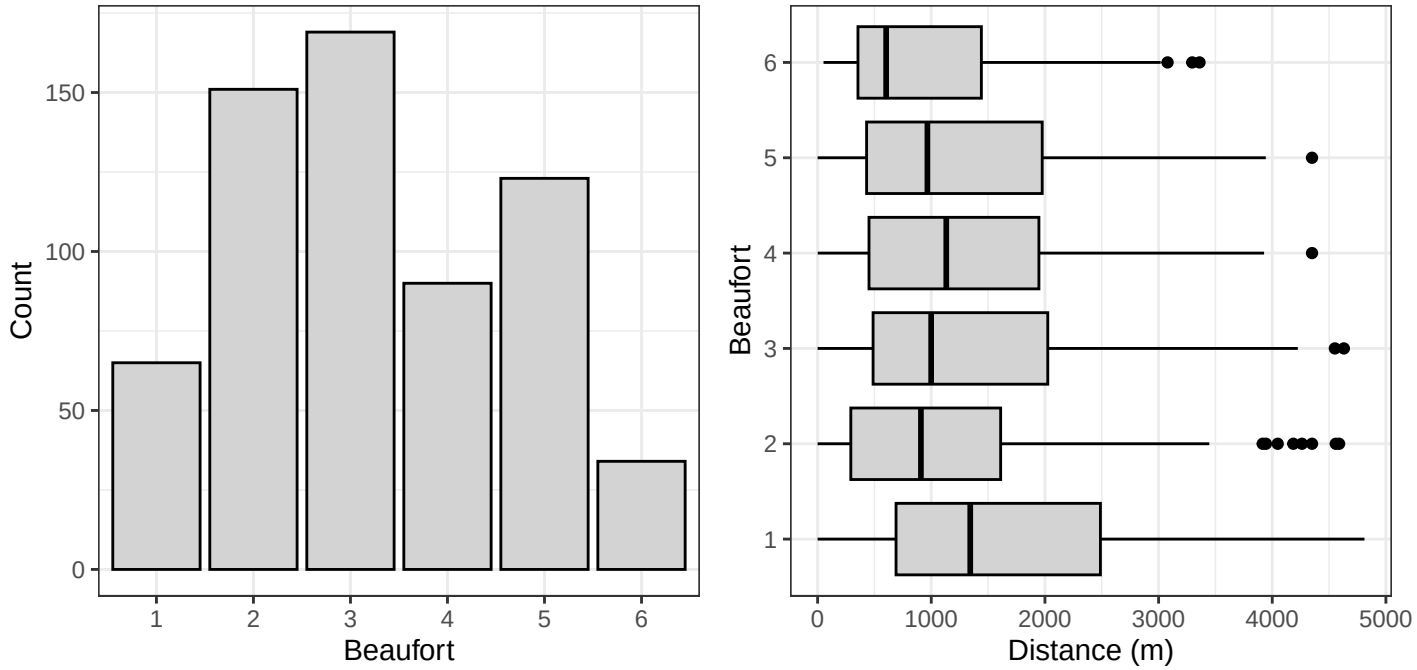


Figure 2: For Icelandic/Faroese surveys, count of fin whale observations reported by Beaufort (left), and box plots of distance by Beaufort (right).

This shows an unusual pattern of long detection distances at Beaufort 1, then dropping at Beaufort 2, rising a bit back to Beaufort 4, and finally falling again at higher sea states. Participants speculated as to why this would happen and whether levels should be pooled, but with limited time we decided to leave them as they are and investigate this later if time permits.

4.1.1.3.2 Vessel_Code

```
p1 <- obs_fin_ice_retained |>
  ggplot(aes(x = Vessel_Code)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_ice_retained |>
  ggplot(aes(x = Vessel_Code, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

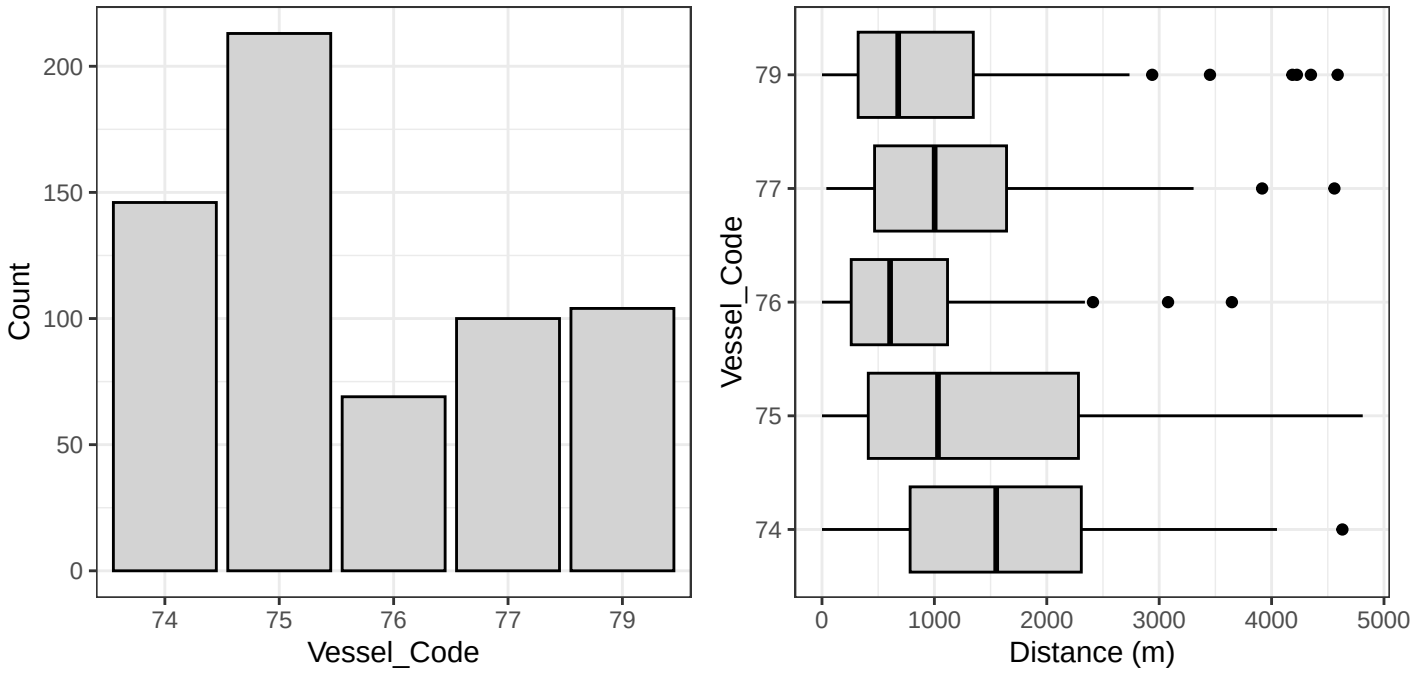


Figure 3: For Icelandic/Faroese surveys, count of fin whale observations reported by Vessel_Code (left), and box plots of distance by Vessel_Code (right).

We noted that vessels 74 and 75 were fisheries surveys, while the others were dedicated marine mammal surveys. Given that, we decided to create a second covariate that grouped these two survey types.

4.1.1.3.3 Survey_Type

```
p1 <- obs_fin_ice_retained |>
  ggplot(aes(x = Survey_Type)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_ice_retained |>
```

```
  ggplot(aes(x = Survey_Type, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

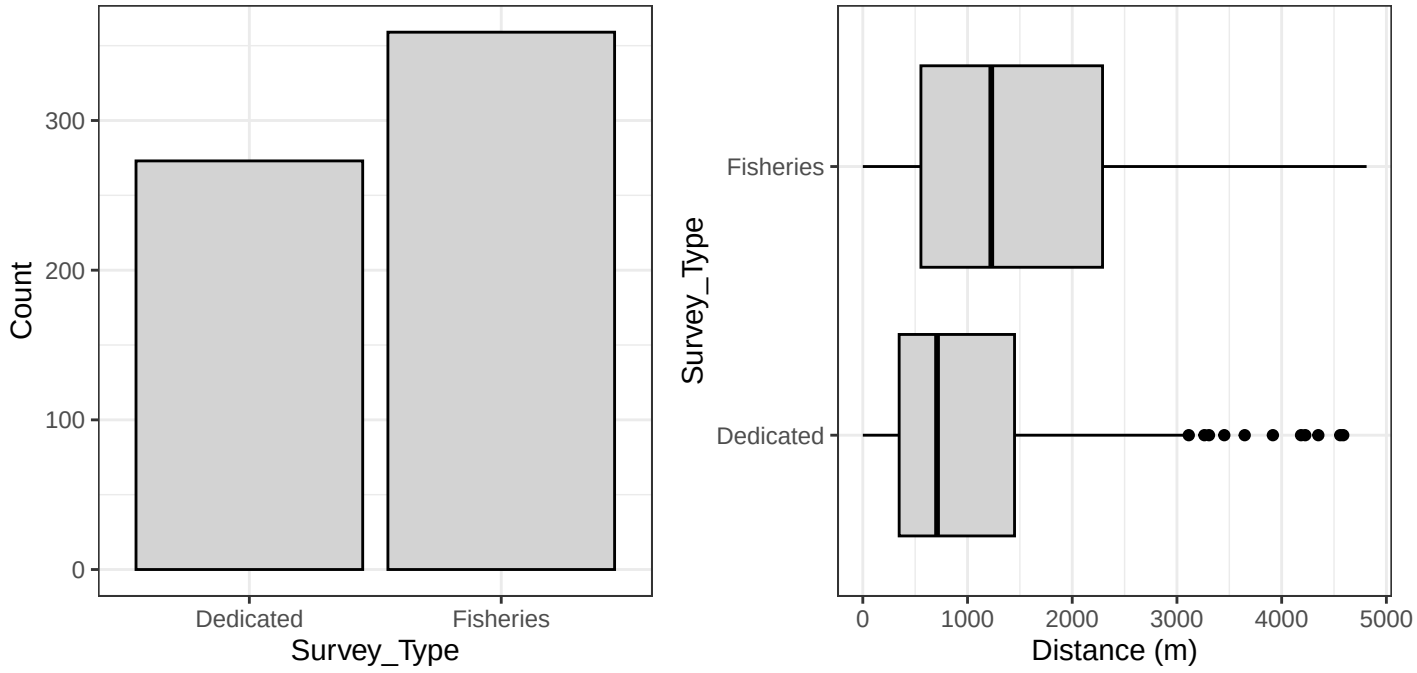


Figure 4: For Icelandic/Faroese surveys, count of fin whale observations reported by Survey_Type (left), and box plots of distance by Survey_Type (right).

As expected, there was a strong difference in detectability between the two types of surveys.

4.1.1.3.4 Month

```
p1 <- obs_fin_ice_retained |>
  ggplot(aes(x = Month)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_ice_retained |>
  ggplot(aes(x = Month, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

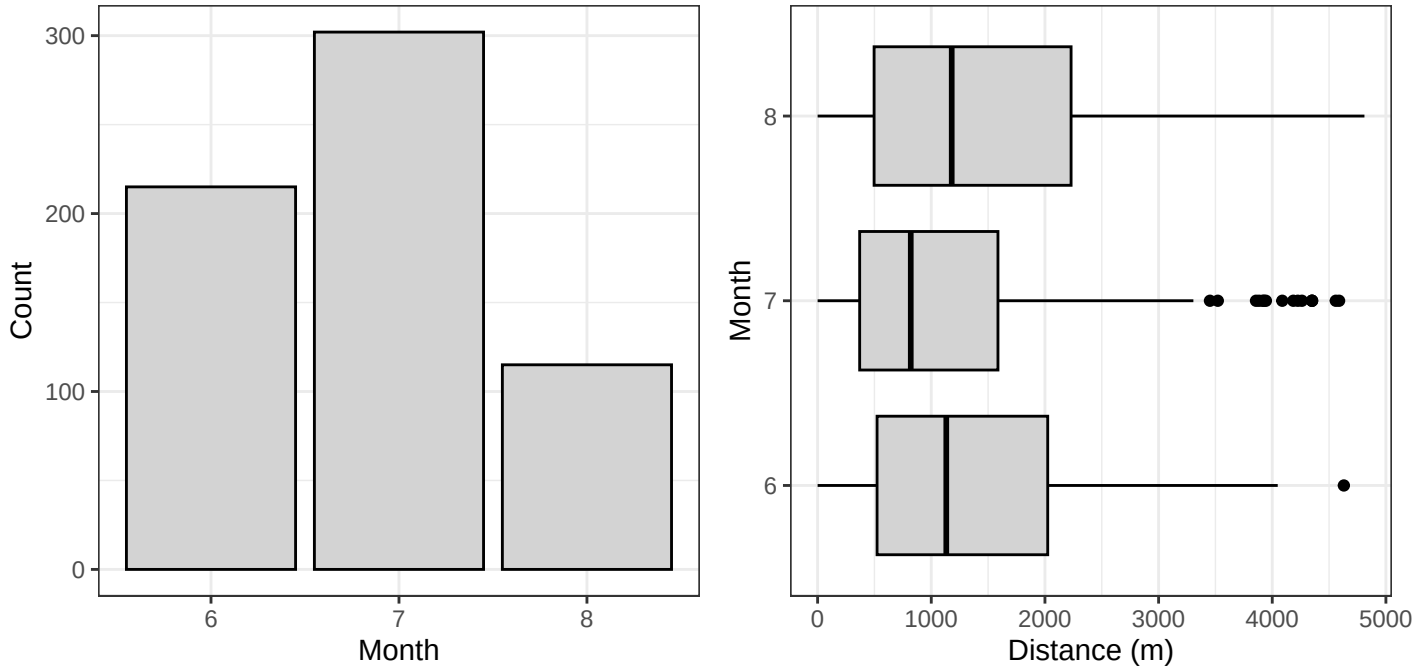


Figure 5: For Icelandic/Faroese surveys, count of fin whale observations reported by Month (left), and box plots of distance by Month (right).

We had no a priori reason to suspect the month as being important, but Ana suggested it as another covariate that was easy to test and that could be relevant based on weather, species behaviors, etc.

4.1.1.4 Fit candidate detection functions

Preliminary investigation suggested that half-normal (hn) detection functions would not fit nearly as well as hazard-rate (hr) functions, so we only developed hr candidates.

```
# Construct a tibble of the detection functions to fit.

dfuncs_fin_ice <- tribble(
  ~Key, ~Adjustment, ~NAdj, ~Formula,
  "hr", NULL,        NULL,  ~1,
  "hr", "cos",       1,     ~1,
  "hr", "cos",       2,     ~1,
  "hr", "cos",       3,     ~1,
  "hr", "herm",      1,     ~1,
  "hr", "herm",      2,     ~1,
  "hr", "herm",      3,     ~1,
  "hr", "poly",      1,     ~1,
  "hr", "poly",      2,     ~1,
  "hr", "poly",      3,     ~1,
  "hr", NULL,        NULL,  ~Beaufort,
  "hr", NULL,        NULL,  ~Vessel_Code,
  "hr", NULL,        NULL,  ~Survey_Type,
  "hr", NULL,        NULL,  ~Month,
  # To save time, we are only trying univariate detection functions, so I'm
  # commenting out the multivariate functions
```

```
  #"hr", NULL,        NULL,  ~Beaufort + Vessel_Code,
  #"hr", NULL,        NULL,  ~Beaufort + Survey_Type,
  #"hr", NULL,        NULL,  ~Beaufort + Month,
  #"hr", NULL,        NULL,  ~Vessel_Code + Month,
  #"hr", NULL,        NULL,  ~Survey_Type + Month,
  #"hr", NULL,        NULL,  ~Beaufort + Vessel_Code + Month,
  #"hr", NULL,        NULL,  ~Beaufort + Survey_Type + Month,
)

#dfuncs <- dfuncs |>
#  bind_rows(dfuncs |> mutate(Key = "hn"))   # hn does not perform well, so not doing it

# Fit the detection functions with a tidyverse pipeline, using quietly() to
# capture any errors or warnings. You can replace purrr::pmap() with furrr's
# future_pmap() to do parallel processing on multiple CPUs, but it can make
# debugging particularly challenging.

dfuncs_fin_ice2 <- dfuncs_fin_ice |>
  mutate(
    res = purrr::pmap(
      list(key = Key, adj = Adjustment, nadj = NAdj, form = Formula),
      function(key, adj, nadj, form) {
        quietly(ds)(obs_fin_ice_retained,
                    truncation = fin_ice_trunc_dist,
                    key = key,
                    adjustment = adj,
                    nadj = nadj,
                    formula = form)
      }),
    DFunc = map(res, "result"),
    AIC = map_dbl(DFunc, ~ if (length(.x) > 0) {AIC(.x)$AIC} else {NA_real_}),
    Errors = map_chr(res, ~ paste(.x$errors, collapse = "; ")),
    Warnings = map_chr(res, ~ paste(.x$warnings, collapse = "; "))
  ) |>
  dplyr::select(-res)

# Delete warnings we don't care about, calculate delta AIC, and sort by AIC.

dfuncs_fin_ice3 <- dfuncs_fin_ice2 |>
  mutate(
    Warnings = str_replace(Warnings, "Unknown or uninitialised column: `distbegin`.; Unknown or unini
  ) |>
  arrange(AIC) |>
  mutate(DeltaAIC = AIC - min(AIC)) |>
  relocate(DeltaAIC, .after = "AIC")

# Calculate the Cramer-von Mises goodness of fit test.

dfuncs_fin_ice3 <- dfuncs_fin_ice3 |>
  mutate(
    res = map(DFunc, ~ if (length(.x) > 0) {quietly(gof_ds)(.x, plot = FALSE)} else {NULL}),
    CvM = map_dbl(res, ~ if (length(.x) > 0) {.x$result$dsgof$CvM$W} else {NA_real_}),
```

```
    CvMp = map_dbl(res, ~ if (length(.x) > 0) {.x$result$dsgof$CvM$p} else {NA_real_})
  ) |>
  relocate(c(CvM, CvMp), .before = "Errors") |>
  dplyr::select(-res)

# Make a kable table.

dfuncs_fin_ice3 |>
  # Remove dfunc object, also Errors (there were none)
  dplyr::select(-c(DFunc, Errors)) |>
  # Change NAs to empty strings
  mutate(across(everything(), ~ ifelse(lengths(.x) == 0 | is.na(.x), "", .x))) |>
  # Escape underscores in Formulas, so latex won't fail
  mutate(
    Formula = map_chr(Formula, ~ deparse(.x) |> paste(collapse = " ")),
    Formula = str_replace_all(Formula, "_", "\\\\_")
  ) |>
  kbl(col.names = c("Key", "Adj.", "\\# Adj.", "Covariate", "AIC", "$\\Delta$AIC",
                    "CvM", "CvM p", "Warnings"),
      digits = c(NA, NA, 0, NA, 2, 2, 3, 3, NA),
      escape = FALSE,
      booktabs = TRUE,
      longtable = TRUE,
      na = "",
      linesep = "\\addlinespace") |>
  column_spec(9, width = "2.5in") |>
  kable_styling(latex_options = c("hold_position", "repeat_header"))
```

Table 1: Candidate detection functions for fin whale for Icelandic/Faroese surveys.

Key	Adj.	# Adj.	Covariate	AIC	ΔAIC	CvM	CvM p	Warnings
hr			Vessel_Code	10265.24	0.00	0.108	0.546	
hr			Survey_Type	10274.42	9.18	0.172	0.329	
hr	poly	1	1	10294.78	29.54	0.023	0.994	
hr	herm	1	1	10295.04	29.80	0.022	0.995	Estimated hazard-rate scale parameter close to 0 (on log scale). Possible problem in data (e.g., spike near zero distance).; Estimated hazard-rate scale parameter close to 0 (on log scale). Possible problem in data (e.g., spike near zero distance).
hr	poly	2	1	10296.76	31.52	0.022	0.995	
hr	herm	2	1	10296.76	31.53	0.020	0.997	
hr	cos	2	1	10297.16	31.92	0.031	0.971	
hr	herm	3	1	10298.77	33.53	0.021	0.996	
hr	poly	3	1	10298.82	33.58	0.023	0.994	
hr			Month	10298.92	33.68	0.121	0.493	

(continued)

Key	Adj.	# Adj.	Covariate	AIC	ΔAIC	CvM	CvM p	Warnings
hr	cos	3	1	10299.16	33.92	0.030	0.975	
hr			Beaufort	10305.82	40.58	0.125	0.475	
hr			1	10306.74	41.50	0.107	0.554	
hr	cos	1	1	10308.74	43.50	0.106	0.554	

4.1.1.5 Final detection function

We don't have time to explore the detection functions fully, so we'll just pick the highest-ranked one and proceed with spatial modeling.

```
source(file.path(this.path::here(), "plot.ds.color.R"), local = TRUE)

dfunc_fin_ice_final <- dfuncs_fin_ice3$DFunc[[1]]

summary(dfunc_fin_ice_final)
##
## Summary for distance analysis
## Number of observations :  632
## Distance range         :  0  -  5000
##
## Model       : Hazard-rate key function
## AIC         :  10265.24
## Optimisation:  mrds (nlminb)
##
## Detection function parameters
## Scale coefficient(s):
##                 estimate        se
## (Intercept)    7.6692030 0.1239950
## Vessel_Code75 -0.3736727 0.1489441
## Vessel_Code76 -1.0881619 0.1942174
## Vessel_Code77 -0.5470176 0.1750836
## Vessel_Code79 -0.9431866 0.1754077
##
## Shape coefficient(s):
##              estimate         se
## (Intercept) 0.7674652 0.08796915
##
##                          Estimate           SE         CV
## Average p               0.3645798   0.01877492 0.05149742
## N in covered region 1733.5026334 105.67803311 0.06096214
par(mfcol = c(1, 2))
plot.ds.color(dfunc_fin_ice_final)
gof_ds(dfunc_fin_ice_final, main=NULL)
##
## Goodness of fit results for ddf object
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.108169 p-value = 0.546295
```

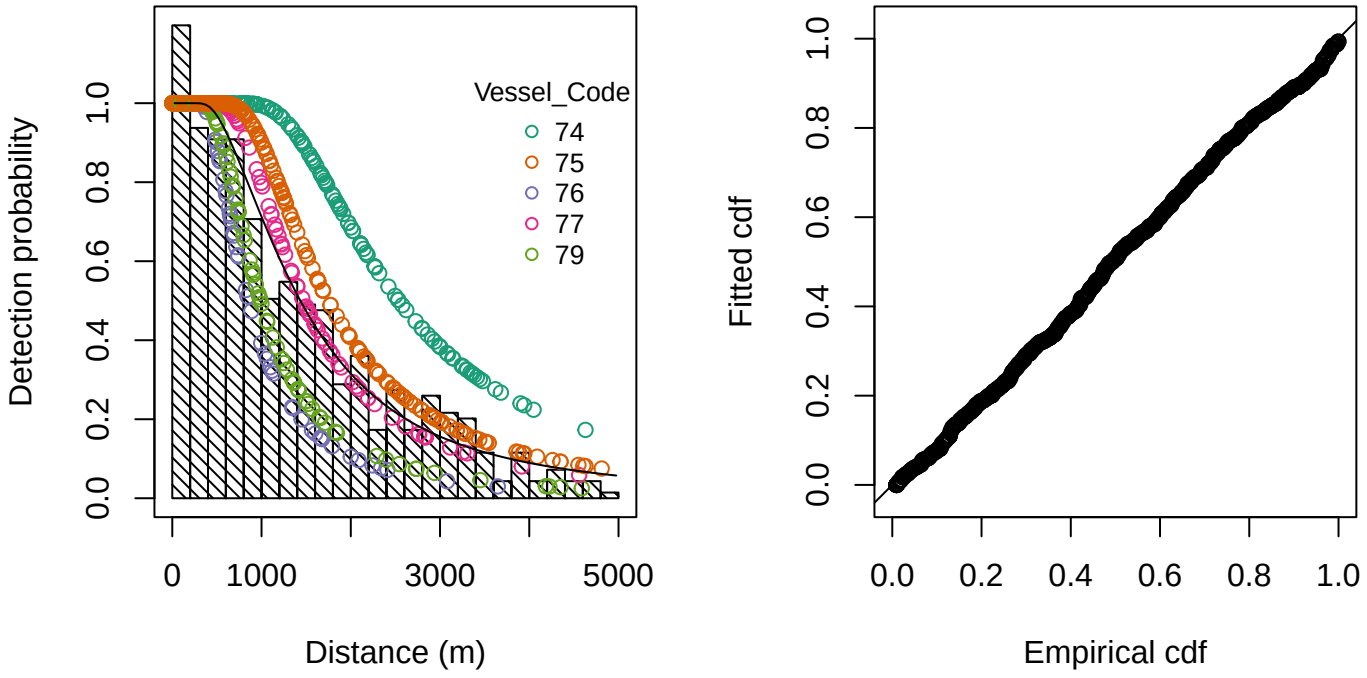


Figure 6: Final fin whale detection function for Icelandic/Faroese surveys.

The summary and diagnostic plots look good for this detection function. The standard errors of the scale coefficients suggest that vessel 76 and 79 are not significantly different from each other. This makes them candidates for pooling, but because we have limited time and because this detection function is already ranked best, we decided not to discuss whether we should pool these vessels.

4.1.2 Norwegian surveys

4.1.2.1 Filter the segments

```
obs_fin_nor <- obs |>
  filter(English.name == "Fin whale",
         Survey == "Norway",
         Beaufort %in% c(1,2,3,4,5,6))    # Not enough effort in Beaufort > 6
```

4.1.2.2 Choose a truncation distance

To choose the truncation distance, we built a hazard rate detection function with no adjustments or covariates and then visually selected a distance at which detection probability dropped to about 0.1, following advice in Buckland et al. (2001) and L. Thomas.

```
# If the ds() function reports Warning: Unknown or uninitialised column: `distbegin`
# we can ignore it.

dfunc <- ds(obs_fin_nor, truncation = max(obs_fin_nor$distance), key = "hr", adjustment = NULL)
```

```
## Warning: Unknown or uninitialised column: `distbegin`.
## Warning: Unknown or uninitialised column: `distend`.
## Warning: Unknown or uninitialised column: `distbegin`.
## Fitting hazard-rate key function
## AIC= 1065.808
## No survey area information supplied, only estimating detection function.
plot(dfunc)

fin_nor_trunc_dist <- 3100
obs_fin_nor_retained <- obs_fin_nor[obs_fin_nor$distance < fin_nor_trunc_dist, ]
print(sprintf("Observations truncated: %d (%.1f %%)",
              nrow(obs_fin_nor) - nrow(obs_fin_nor_retained),
              (nrow(obs_fin_nor) - nrow(obs_fin_nor_retained)) / nrow(obs_fin_nor) * 100))
## [1] "Observations truncated: 1 (1.5 %)"
```

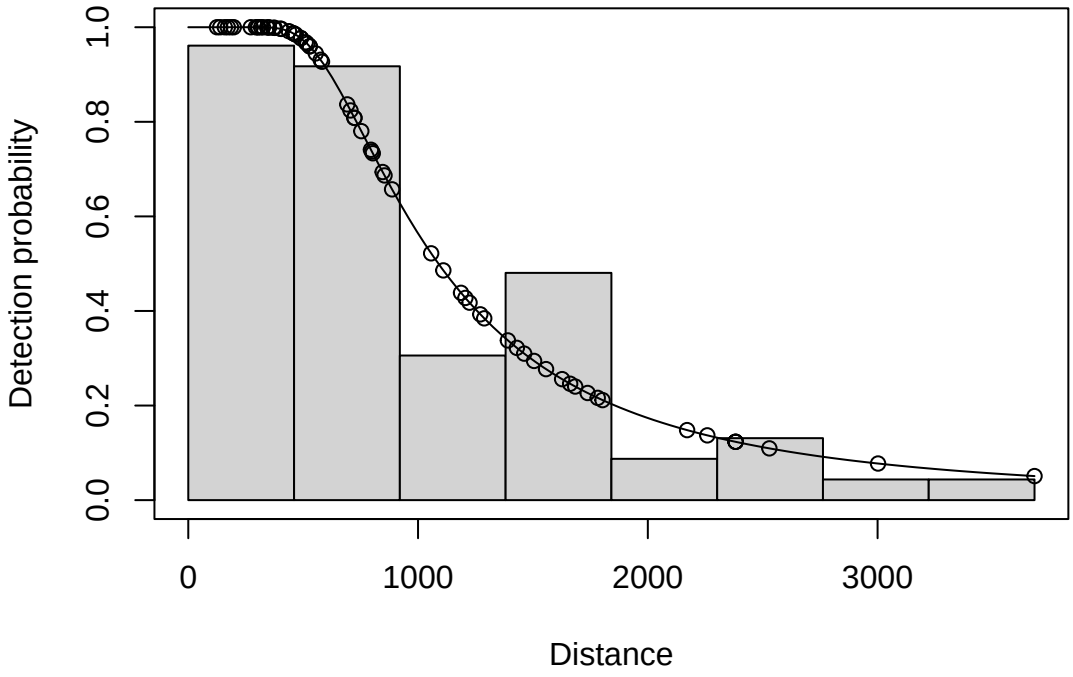


Figure 7: For Norwegian surveys, detection function used to select the fin whale truncation distance.

We chose a truncation distance of 3100 m. This only resulted in the truncation of 1 sighting, but because we only had 68 to start with, we preferred to retain as many as possible so that the spatial model could better calibrate species-habitat relationships in waters covered by these surveys.

4.1.2.3 Explore detection covariates

4.1.2.3.1 Beaufort

```
p1 <- obs_fin_nor_retained |>
  ggplot(aes(x = Beaufort)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_nor_retained |>
  ggplot(aes(x = Beaufort, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

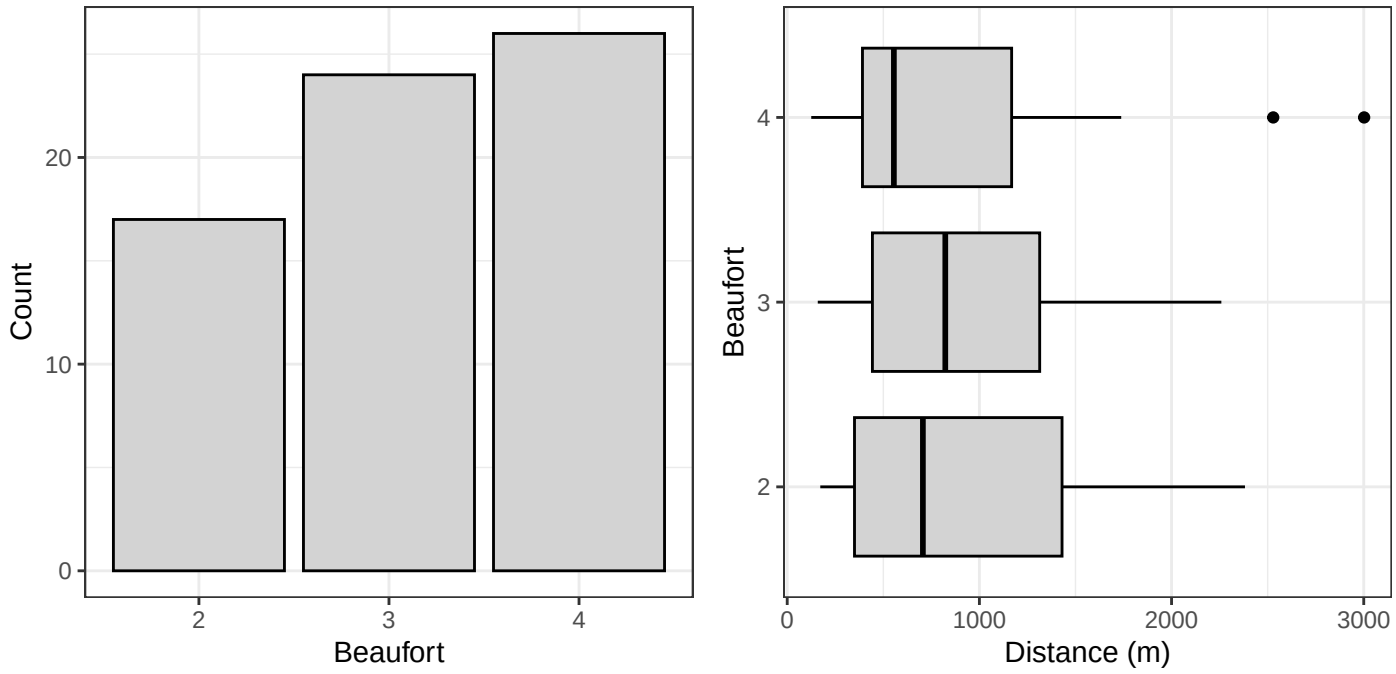


Figure 8: For Norwegian surveys, count of fin whale observations reported by Beaufort (left), and box plots of distance by Beaufort (right).

There is an even spread of sightings across three Beaufort levels: 2, 3, and 4. If Beaufort is selected as our covariate, we'll have to limit the segments to Beaufort 4 and treat Beaufort 0 and 1 as Beaufort 2.

4.1.2.3.2 Platform

```
p1 <- obs_fin_nor_retained |>
  ggplot(aes(x = Platform)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_nor_retained |>
```

```
  ggplot(aes(x = Platform, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

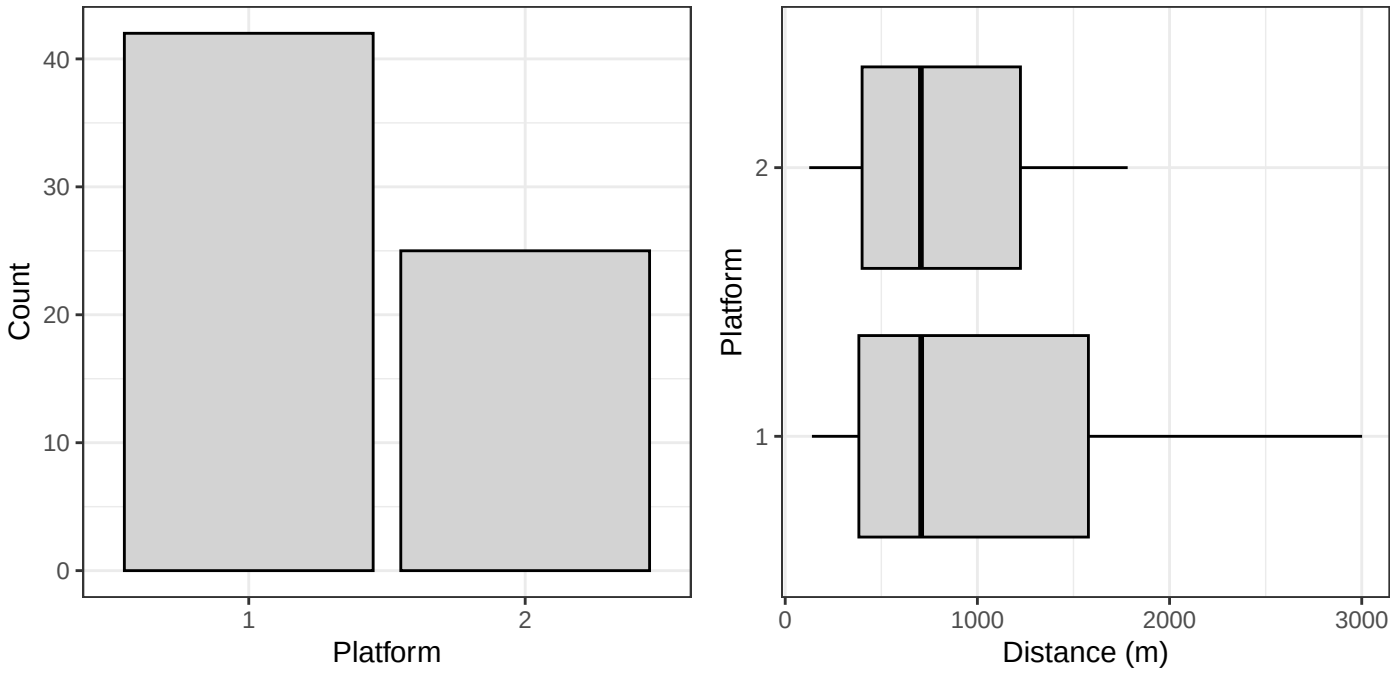


Figure 9: For Norwegian surveys, count of fin whale observations reported by Platform (left), and box plots of distance by Platform (right).

It appears that Platform 1 reported more distant sightings. We'll try it as a covariate. However, Platform is a per-sighting covariate, not a per-segment covariate. Both platforms operate at the same time, and we can't predict a detection function with Platform on the segments. So if we select a detection function that includes Platform, we'll have to use a Horvitz-Thomson formulation of density, rather than a traditional formulation.

4.1.2.3.3 Weather

```
p1 <- obs_fin_nor_retained |>
  ggplot(aes(x = Weather)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_nor_retained |>
  ggplot(aes(x = Weather, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()
```

```
p1 + p2
```

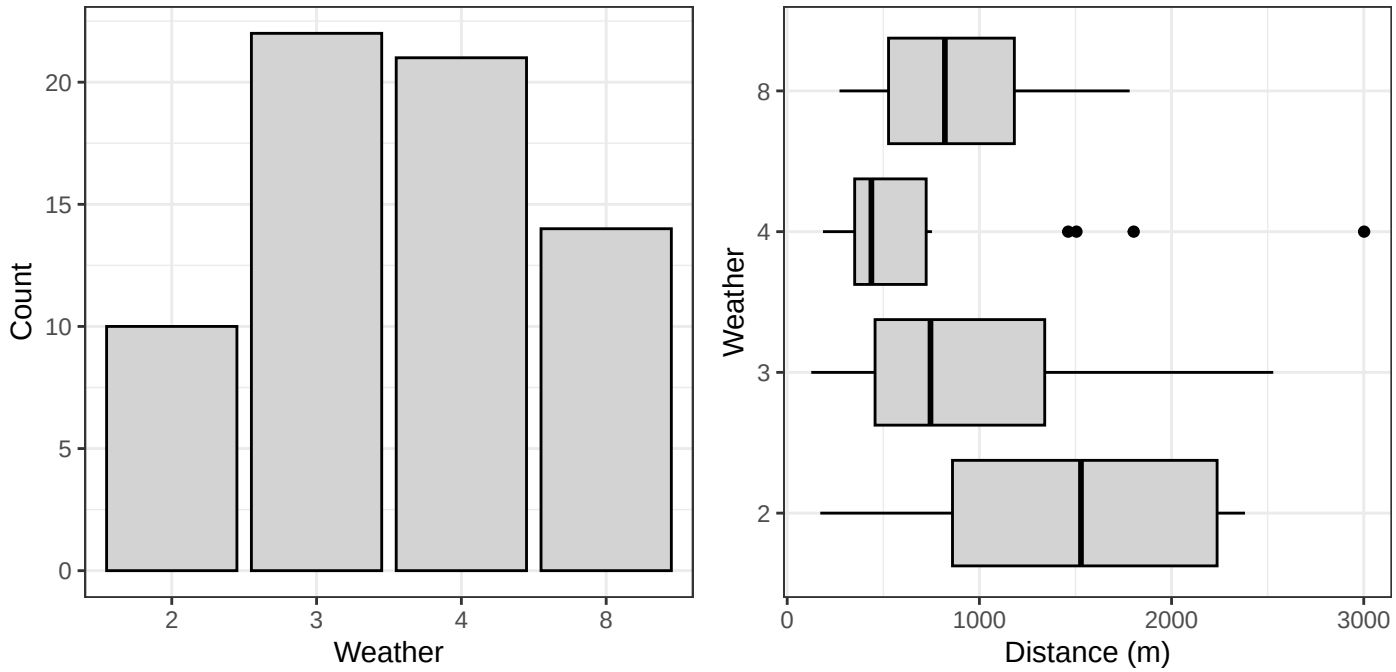


Figure 10: For Norwegian surveys, count of fin whale observations reported by Weather (left), and box plots of distance by Weather (right).

Weather codes 2 and 4 appear to have a strong effect, although code 4 has some outliers, including at the farthest distance. Codes 2 and 8 have relatively few sightings. It's hard to say how this covariate will perform.

4.1.2.3.4 Month

```
p1 <- obs_fin_nor_retained |>
  ggplot(aes(x = Month)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_nor_retained |>
  ggplot(aes(x = Month, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

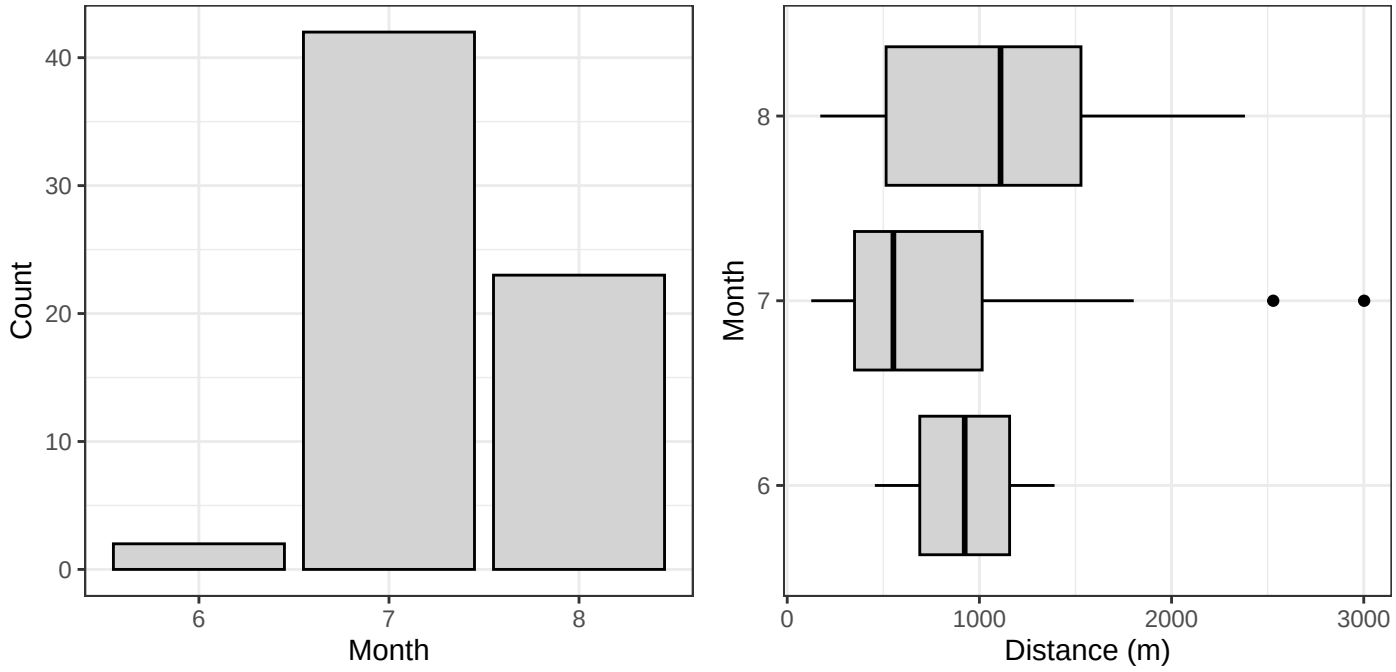


Figure 11: For Norwegian surveys, count of fin whale observations reported by Month (left), and box plots of distance by Month (right).

This shows the same pattern as the Icelandic/Faroese surveys. However, the data are much sparser. If we use this survey, we'll have to pool Month 6 with another Month. I will tentatively pool it with 7, based on it being the closest in time.

```
obs_fin_nor_retained <- obs_fin_nor_retained |>
  mutate(Month2 = factor(if_else(Month %in% c(6,7), "6,7", "8")))

p1 <- obs_fin_nor_retained |>
  ggplot(aes(x = Month2)) +
  geom_bar(color = "black", fill = "lightgray") +
  labs(y = "Count") +
  theme_bw()

p2 <- obs_fin_nor_retained |>
  ggplot(aes(x = Month2, y = distance)) +
  geom_boxplot(color = "black", fill = "lightgray") +
  coord_flip() +
  labs(y = "Distance (m)") +
  theme_bw()

p1 + p2
```

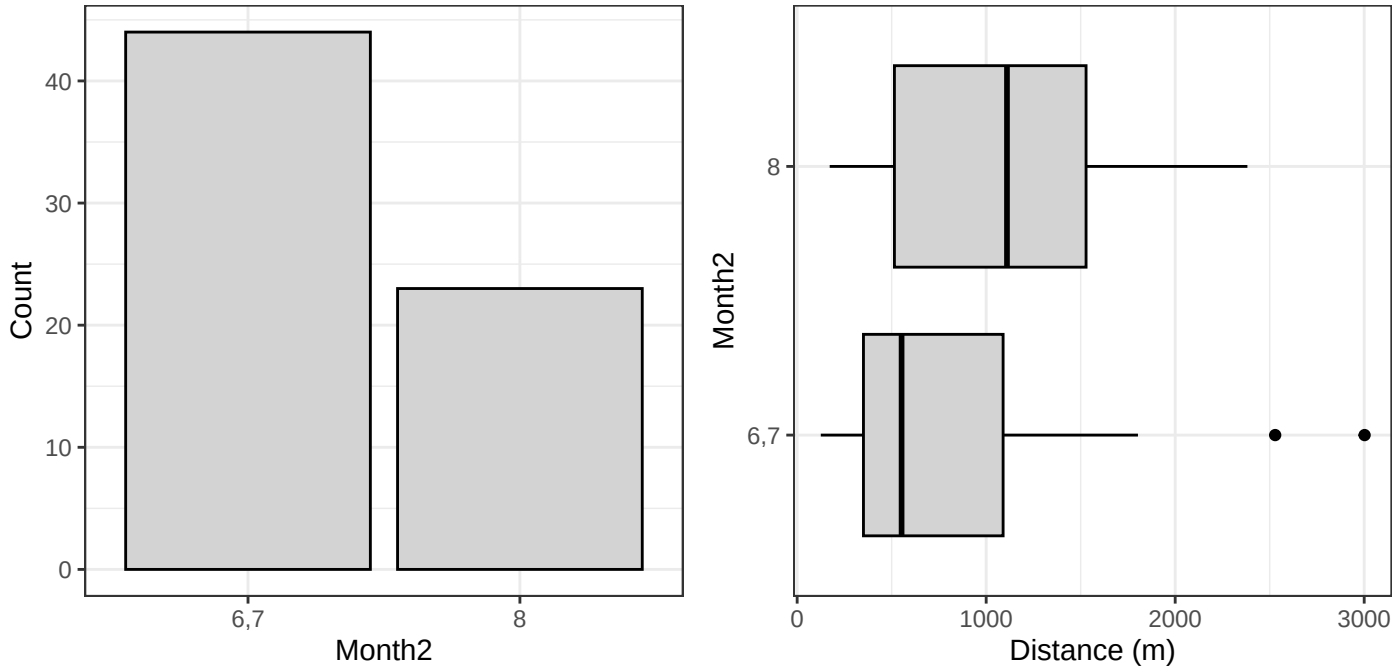


Figure 12: For Norwegian surveys, count of fin whale observations reported by Month2 (left), and box plots of distance by Month2 (right).

4.1.2.4 Fit candidate detection functions

```
# Construct a tibble of the detection functions to fit.

dfuncs_fin_nor <- tribble(
  ~Key, ~Adjustment, ~NAdj, ~Formula,
  "hr", NULL,        NULL,  ~1,
  "hr", "cos",       1,     ~1,
  "hr", "cos",       2,     ~1,
  "hr", "cos",       3,     ~1,
  "hr", "herm",      1,     ~1,
  "hr", "herm",      2,     ~1,
  "hr", "herm",      3,     ~1,
  "hr", "poly",      1,     ~1,
  "hr", "poly",      2,     ~1,
  "hr", "poly",      3,     ~1,
  "hr", NULL,        NULL,  ~Beaufort,
  "hr", NULL,        NULL,  ~Platform,
  "hr", NULL,        NULL,  ~Weather,
  "hr", NULL,        NULL,  ~Month2,
  # To save time, we are only trying univariate detection functions, so I'm
  # commenting out the multivariate functions
  #"hr", NULL,        NULL,  ~Beaufort + Platform,
  #"hr", NULL,        NULL,  ~Beaufort + Weather,
  #"hr", NULL,        NULL,  ~Beaufort + Month2,
  #"hr", NULL,        NULL,  ~Platform + Weather,
  #"hr", NULL,        NULL,  ~Platform + Month2,
  #"hr", NULL,        NULL,  ~Weather + Month2,
```

```
  #"hr", NULL,        NULL,  ~Beaufort + Platform + Month2,
  #"hr", NULL,        NULL,  ~Beaufort + Weather + Month2,
)

# Duplicate them, but with an hn key.

dfuncs_fin_nor <- dfuncs_fin_nor |>
  bind_rows(dfuncs_fin_nor |> mutate(Key = "hn"))

# Fit the detection functions with a tidyverse pipeline, using quietly() to
# capture any errors or warnings. You can replace purrr::pmap() with furrr's
# future_pmap() to do parallel processing on multiple CPUs, but it can make
# debugging particularly challenging.

dfuncs_fin_nor2 <- dfuncs_fin_nor |>
  mutate(
    res = purrr::pmap(
      list(key = Key, adj = Adjustment, nadj = NAdj, form = Formula),
      function(key, adj, nadj, form) {
        quietly(ds)(obs_fin_nor_retained,
                    truncation = fin_nor_trunc_dist,
                    key = key,
                    adjustment = adj,
                    nadj = nadj,
                    formula = form)
      }),
    DFunc = map(res, "result"),
    AIC = map_dbl(DFunc, ~ if (length(.x) > 0) {AIC(.x)$AIC} else {NA_real_}),
    Errors = map_chr(res, ~ paste(.x$errors, collapse = "; ")),
    Warnings = map_chr(res, ~ paste(.x$warnings, collapse = "; "))
  ) |>
  dplyr::select(-res)

# Delete warnings we don't care about, calculate delta AIC, and sort by AIC.

dfuncs_fin_nor3 <- dfuncs_fin_nor2 |>
  mutate(
    Warnings = str_replace(Warnings, "Unknown or uninitialised column: `distbegin`.; Unknown or unini
  ) |>
  arrange(AIC) |>
  mutate(DeltaAIC = AIC - min(AIC)) |>
  relocate(DeltaAIC, .after = "AIC")

# Calculate the Cramer-von Mises goodness of fit test.

dfuncs_fin_nor3 <- dfuncs_fin_nor3 |>
  mutate(
    res = map(DFunc, ~ if (length(.x) > 0) {quietly(gof_ds)(.x, plot = FALSE)} else {NULL}),
    CvM = map_dbl(res, ~ if (length(.x) > 0) {.x$result$dsgof$CvM$W} else {NA_real_}),
    CvMp = map_dbl(res, ~ if (length(.x) > 0) {.x$result$dsgof$CvM$p} else {NA_real_})
  ) |>
  relocate(c(CvM, CvMp), .before = "Errors") |>
```

```
  dplyr::select(-res)

# Make a kable table.

dfuncs_fin_nor3 |>
  # Remove dfunc object, also Errors (there were none)
  dplyr::select(-c(DFunc, Errors)) |>
  # Change NAs to empty strings
  mutate(across(everything(), ~ ifelse(lengths(.x) == 0 | is.na(.x), "", .x))) |>
  # Escape underscores in Formulas, so latex won't fail
  mutate(
    Formula = map_chr(Formula, ~ deparse(.x) |> paste(collapse = " ")),
    Formula = str_replace_all(Formula, "_", "\\\\_")
  ) |>
  kbl(col.names = c("Key", "Adj.", "\\# Adj.", "Covariate", "AIC", "$\\Delta$AIC",
                    "CvM", "CvM p", "Warnings"),
      digits = c(NA, NA, 0, NA, 2, 2, 3, 3, NA),
      escape = FALSE,
      booktabs = TRUE,
      longtable = TRUE,
      na = "",
      linesep = "\\addlinespace") |>
  column_spec(9, width = "2.5in") |>
  kable_styling(latex_options = c("hold_position", "repeat_header"))
```

Table 2: Candidate detection functions for fin whale for Norwegian surveys.

Key	Adj.	# Adj.	Covariate	AIC	ΔAIC	CvM	CvM p	Warnings
hr			Weather	1036.62	0.00	0.186	0.295	
hr			Month2	1039.76	3.14	0.130	0.455	
hn			Weather	1039.82	3.20	0.128	0.463	
hn			Platform	1040.54	3.91	0.142	0.416	
hn			Month2	1041.05	4.42	0.147	0.399	
hn			1	1041.29	4.67	0.150	0.390	
hr			1	1041.81	5.19	0.127	0.468	
hn	cos	1	1	1042.62	6.00	0.135	0.437	
hn	poly	1	1	1042.75	6.12	0.134	0.443	
hn	herm	1	1	1043.28	6.65	0.149	0.391	
hr	poly	1	1	1043.51	6.89	0.125	0.475	
hr	herm	1	1	1043.69	7.06	0.127	0.466	
hr			Platform	1043.79	7.17	0.125	0.475	
hr	cos	1	1	1043.81	7.19	0.127	0.468	
hn	herm	2	1	1044.62	8.00	0.134	0.442	
hn	cos	2	1	1044.62	8.00	0.136	0.436	
hn	poly	2	1	1044.65	8.03	0.133	0.445	

(continued)

Key	Adj.	# Adj.	Covariate	AIC	ΔAIC	CvM	CvM p	Warnings
hn			Beaufort	1044.89	8.27	0.141	0.417	
hr	cos	2	1	1045.04	8.41	0.128	0.463	
hr			Beaufort	1045.22	8.59	0.114	0.519	
hr	herm	2	1	1045.51	8.88	0.126	0.473	
hr	poly	2	1	1045.52	8.89	0.126	0.472	
hn	cos	3	1	1046.49	9.87	0.122	0.488	
hn	herm	3	1	1046.62	10.00	0.134	0.441	
hn	poly	3	1	1046.63	10.00	0.134	0.443	
hr	cos	3	1	1047.02	10.39	0.128	0.464	
hr	poly	3	1	1047.31	10.69	0.125	0.474	
hr	herm	3	1	1047.50	10.88	0.126	0.473	

4.1.2.5 Final detection function

We don't have time to explore the detection functions fully, so we'll just pick the highest-ranked one and proceed with spatial modeling.

```
dfunc_fin_nor_final <- dfuncs_fin_nor3$DFunc[[1]]

summary(dfunc_fin_nor_final)
##
## Summary for distance analysis
## Number of observations :  67
## Distance range         :  0  -  3100
##
## Model       : Hazard-rate key function
## AIC         :  1036.624
## Optimisation:  mrds (nlminb)
##
## Detection function parameters
## Scale coefficient(s):
##                estimate        se
## (Intercept)  7.9771829 0.7221986
## Weather3    -0.9539215 0.7716419
## Weather4    -1.4892013 0.8073463
## Weather8    -0.9945836 0.7945026
##
## Shape coefficient(s):
##              estimate        se
## (Intercept) 0.9963624 0.2627187
##
##                         Estimate         SE        CV
## Average p              0.4212971  0.0594516 0.1411156
## N in covered region 159.0326510 27.2892346 0.1715952
par(mfcol = c(1, 2))
```

```
plot.ds.color(dfunc_fin_nor_final)
gof_ds(dfunc_fin_nor_final, main=NULL)
##
## Goodness of fit results for ddf object
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.186381 p-value = 0.295484
```

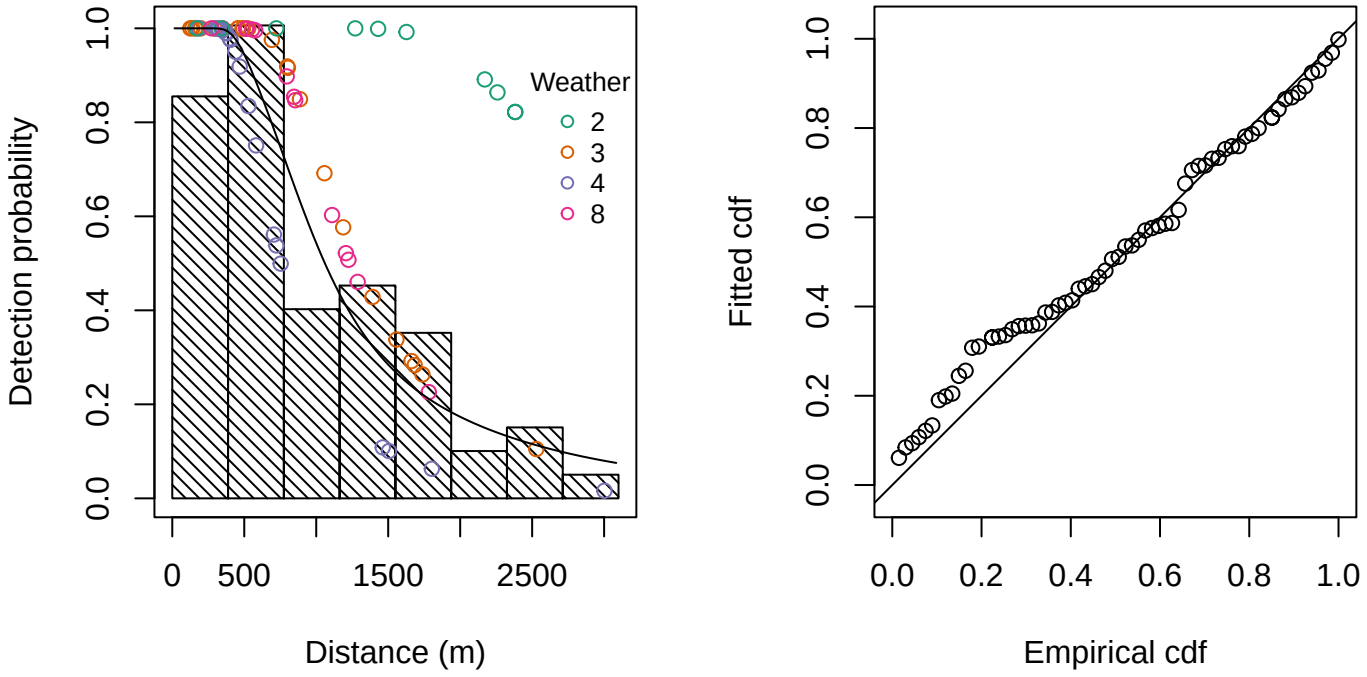


Figure 13: Final fin whale detection function for Norwegian surveys.

The summary indicates that although weather codes 3, 4, and 8 differ significantly from weather code 2, they do not differ from each other significantly. We could potentially pool them. The magnitude of their scale coefficient estimates indicate that code 2 has much higher detection probability, which we can see in the plot.

The observers in our working group reviewed the meanings of these codes and concluded that these relationships were logical. Codes 2, 3, and 4 represent increasing cloud cover and they judged it logical that they would yield degrading detection ranges. Code 8 represented fog patches, and it was seen as reasonable that it would be similar to code 3.

A further issue is that the effort data contained additional codes for which there were no sightings:

```
summary(segments |> filter(Vessel_Code == "FTR") |> pull(Weather) |> factor())
##   1   2   3   4   5   6   7   8  12
##  23  60 342 121   1  14   5  84   4
```

This means we either have to pool these missing codes with ones that had sightings, or to exclude those segments. After discussing this, the Norway experts decided to:

- Pool 1 with 2 because 1 represents even lower cloud cover than 2.
- Drop 5 (rain), 6 (drizzle), and 12 (snow) because these were bad conditions.
- Pool 7 with 8 because they are both types of fog.

For comparison, let's look at the second ranked detection function:

```
dfunc_fin_nor_final2 <- dfuncs_fin_nor3$DFunc[[2]]

summary(dfunc_fin_nor_final2)
##
## Summary for distance analysis
## Number of observations :  67
## Distance range         :  0  -  3100
##
## Model       : Hazard-rate key function
## AIC         :  1039.761
## Optimisation:  mrds (nlminb)
##
## Detection function parameters
## Scale coefficient(s):
##               estimate        se
## (Intercept) 6.6653956 0.2846881
## Month28     0.6700137 0.3666219
##
## Shape coefficient(s):
##               estimate        se
## (Intercept) 0.8197638 0.2923153
##
##                         Estimate          SE        CV
## Average p              0.4297194  0.06563707 0.1527440
## N in covered region 155.9157032 28.01190800 0.1796606
par(mfcol = c(1, 2))
plot.ds.color(dfunc_fin_nor_final2)
## Warning in brewer.pal(length(covVals), "Dark2"): minimal value for n is 3, returning requested pal
gof_ds(dfunc_fin_nor_final2, main=NULL)
##
## Goodness of fit results for ddf object
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.130422 p-value = 0.455155
```

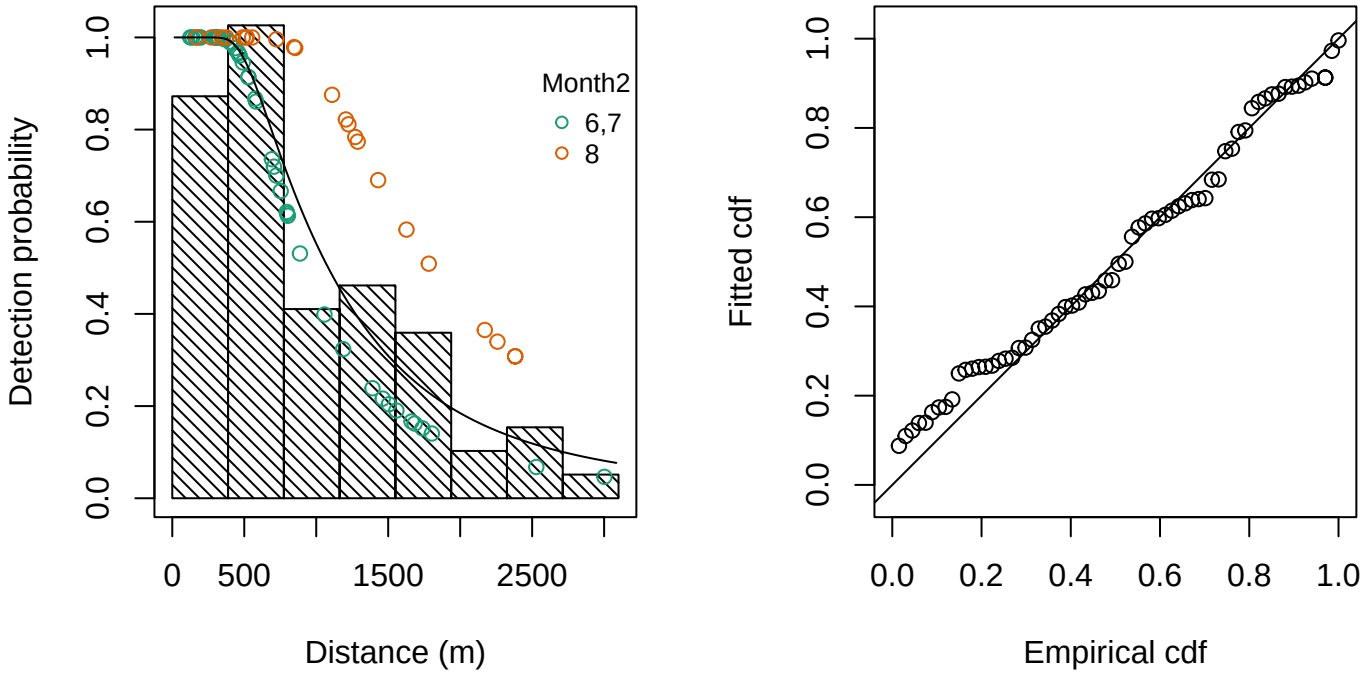


Figure 14: Alternate final fin whale detection function for Norwegian surveys.

Our team decided that although this detection function might be reasonable, it was better to go with the top-ranked function that used the weather code.

5 Bias corrections

5.1 Fin whale

Because this is a shipboard survey, we will assume that availability bias for fin whales is negligible. For perception bias, we'll use the correction factor from Iceland's analysis (TODO: REF) of 0.87. Norway's analysis had a very similar value (0.861), so the bias of using Iceland's for both will be negligible. I believe `dsm()` does now support per-detection-function bias corrections, so it may ultimately be possible to use both, but for now we will just try a single value, as historically required by `dsm()`.

```
g0_fin <- 0.87
```

6 Spatial model

6.1 Fin whale

6.1.1 Filter the segments

TODO: Describe what we did below.

```
segments_retained <- segments |>
  filter((Survey == "Ice_Far" & Beaufort %in% c(0,1,2,3,4,5,6)) |
         (Survey == "Norway" & Beaufort %in% c(0,1,2,3,4) & Weather %in% c(1,2,3,4,7,8)))

obs_retained <- bind_rows(obs_fin_ice_retained, obs_fin_nor_retained)

segs_lost <- nrow(segments) - nrow(segments_retained)
effort_lost <- sum(segments$Effort) - sum(segments_retained$Effort)
obs_lost <- nrow(inner_join(segments, obs_retained, by = "Sample.Label")) -
            nrow(inner_join(segments_retained, obs_retained, by = "Sample.Label"))

cat(sprintf(
  "This filtering resulted in the loss of:
     %d segments (%0.1f %%)
     %.0f km of effort (%0.1f %%)
     %d observations (%0.1f %%)",
  segs_lost, segs_lost / nrow(segments) * 100,
  effort_lost, effort_lost / sum(segments$Effort) * 100,
  obs_lost, obs_lost / nrow(inner_join(segments, obs_retained, by = "Sample.Label")) * 100))
## This filtering resulted in the loss of:
##      32 segments (1.3 %)
##      379 km of effort (1.5 %)
##      0 observations (0.0 %)
```

Due to a bug in mrds, we need to ensure that that the factor covariates to the detection functions do not have any levels without observations. Refit the detection functions now to ensure this.

```
obs_fin_ice_retained_debugged <- obs_fin_ice_retained |>
  mutate(Vessel_Code = factor(as.character(Vessel_Code)))

dfunc_fin_ice_final_debugged <- ds(obs_fin_ice_retained_debugged,
                                   truncation = fin_ice_trunc_dist,
                                   key = "hr",
                                   adjustment = NULL,
                                   formula = ~Vessel_Code)
## Warning: Unknown or uninitialised column: `distbegin`.
## Warning: Unknown or uninitialised column: `distend`.
## Warning: Unknown or uninitialised column: `distbegin`.
## Fitting hazard-rate key function
## AIC= 10265.239
## No survey area information supplied, only estimating detection function.

obs_fin_nor_retained_debugged <- obs_fin_nor_retained |>
  mutate(Weather = factor(as.character(Weather)))

dfunc_fin_nor_final_debugged <- ds(obs_fin_nor_retained_debugged,
                                   truncation = fin_nor_trunc_dist,
                                   key = "hr",
                                   adjustment = NULL,
                                   formula = ~Weather)
## Warning: Unknown or uninitialised column: `distbegin`.
## Warning: Unknown or uninitialised column: `distend`.
```

```
## Warning: Unknown or uninitialised column: `distbegin`.
## Fitting hazard-rate key function
## AIC= 1036.624
## No survey area information supplied, only estimating detection function.

obs_fin_ice_retained_debugged$Weather = factor(
  "2",
  levels = levels(obs_fin_nor_retained_debugged$Weather))

obs_fin_nor_retained_debugged$Vessel_Code = factor(
  "74",
  levels = levels(obs_fin_ice_retained_debugged$Vessel_Code))

obs_retained_debugged <- bind_rows(
  obs_fin_ice_retained_debugged,
  obs_fin_nor_retained_debugged)

# summary(obs_retained_debugged |> select(Weather, Vessel_Code))
```

We also need to classify some of the weather codes for Norway:

```
segments_retained <- segments_retained |>
  mutate(Weather = case_when(
    Survey == "Norway" & Weather == 1 ~ factor("2", levels = levels(Weather)),
    Survey == "Norway" & Weather == 7 ~ factor("8", levels = levels(Weather)),
    TRUE ~ Weather
  ))
```

Effort in the segments was originally given in km. We need it to be in meters for our computations to work out.

```
segments_retained$Effort = segments_retained$Effort * 1000
```

Fix the mrds bug in the segments as well.

```
segments_retained_debugged <- segments_retained |>
  mutate(
    Weather = case_when(
      Survey == "Ice_Far" ~ factor("2", levels = levels(Weather)),
      TRUE ~ Weather),
    Weather = factor(Weather, levels = levels(obs_retained_debugged$Weather)),

    Vessel_Code = case_when(
      Survey == "Norway" ~ factor("74", levels = levels(Vessel_Code)),
      TRUE ~ Vessel_Code),
    Vessel_Code = factor(Vessel_Code, levels = levels(obs_retained_debugged$Vessel_Code)),
  )
```

Because we used two detection functions, we need to create a column called `ddfobj` that tells `dsm()` which detection function to use for each segment and each observation.

```
segments_retained_debugged <- segments_retained_debugged |>
  mutate(ddfobj = if_else(Survey == "Ice_Far", 1, 2))

obs_retained_debugged <- obs_retained_debugged |>
  mutate(ddfobj = if_else(Survey == "Ice_Far", 1, 2))
```

6.1.2 Explore spatial covariates

```
segments_retained_debugged |>
  left_join(obs_retained, by = "Sample.Label") |>
  group_by(Sample.Label) |>
  summarize(
    Depth = mean(Depth),  # They're all the same, so the stat doesn't matter
    WhalesPresent = any(!is.na(distance)),
    .groups = "drop") |>
  ggplot(aes(x = Depth, color = WhalesPresent, fill = WhalesPresent)) +
  geom_density(alpha = 0.1) +
  labs(x = "Depth (m)", y = "Probability Density",
       color = "Whales Present", fill = "Whales Present") +
  theme_bw()
```

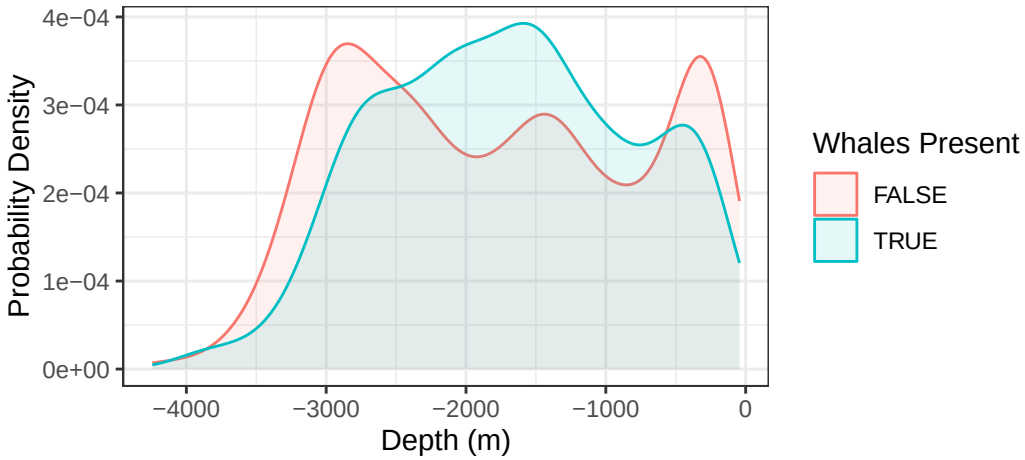


Figure 15: Probability density of the Depth covariate for fin whale.

We are running very low on time, so we'll skip further covariate exploration and just fit a static model with a bivariate x,y smoother. This model will be comparable to the models the other working groups are fitting with point-process methods.

6.1.3 Static models

6.1.3.1 xy only

```
static_fin_xy <- dsm(
  formula = count ~ s(x,y,k=100),
  ddf.obj = list(dfunc_fin_ice_final_debugged, dfunc_fin_nor_final_debugged),
  segment.data = segments_retained_debugged,
```

```
  observation.data = obs_retained_debugged,
  family = tw(),
  availability = g0_fin,
  select = TRUE)

summary(static_fin_xy)
##
## Family: Tweedie(p=1.201)
## Link function: log
##
## Formula:
## count ~ s(x, y, k = 100) + offset(off.set)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -19.1551     0.0983  -194.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##          edf Ref.df     F p-value
## s(x,y) 53.57     99 4.727  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.147   Deviance explained = 34.8%
## -REML = 1597.7  Scale est. = 2.3935    n = 2392
gam.check(static_fin_xy)
##
## Method: REML   Optimizer: outer newton
## full convergence after 11 iterations.
## Gradient range [-9.181225e-05,7.594416e-05]
## (score 1597.723 & scale 2.393513).
## Hessian positive definite, eigenvalue range [0.4681063,1701].
## Model rank =  100 / 100
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'  edf k-index p-value
## s(x,y) 99.0 53.6    0.75  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
gratia::draw(static_fin_xy, rug = FALSE) &
  theme_bw()
```

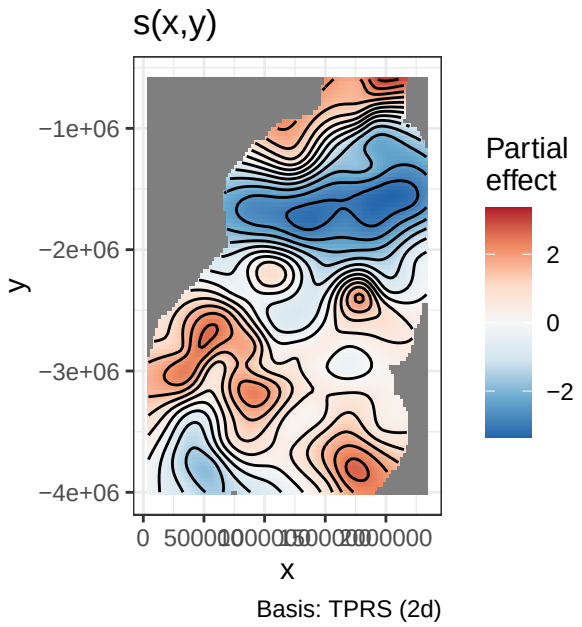


Figure 16: Partial effect plot for the fin whale model static_fin_xy

```
gratia::appraise(static_fin_xy) & theme_bw()
```

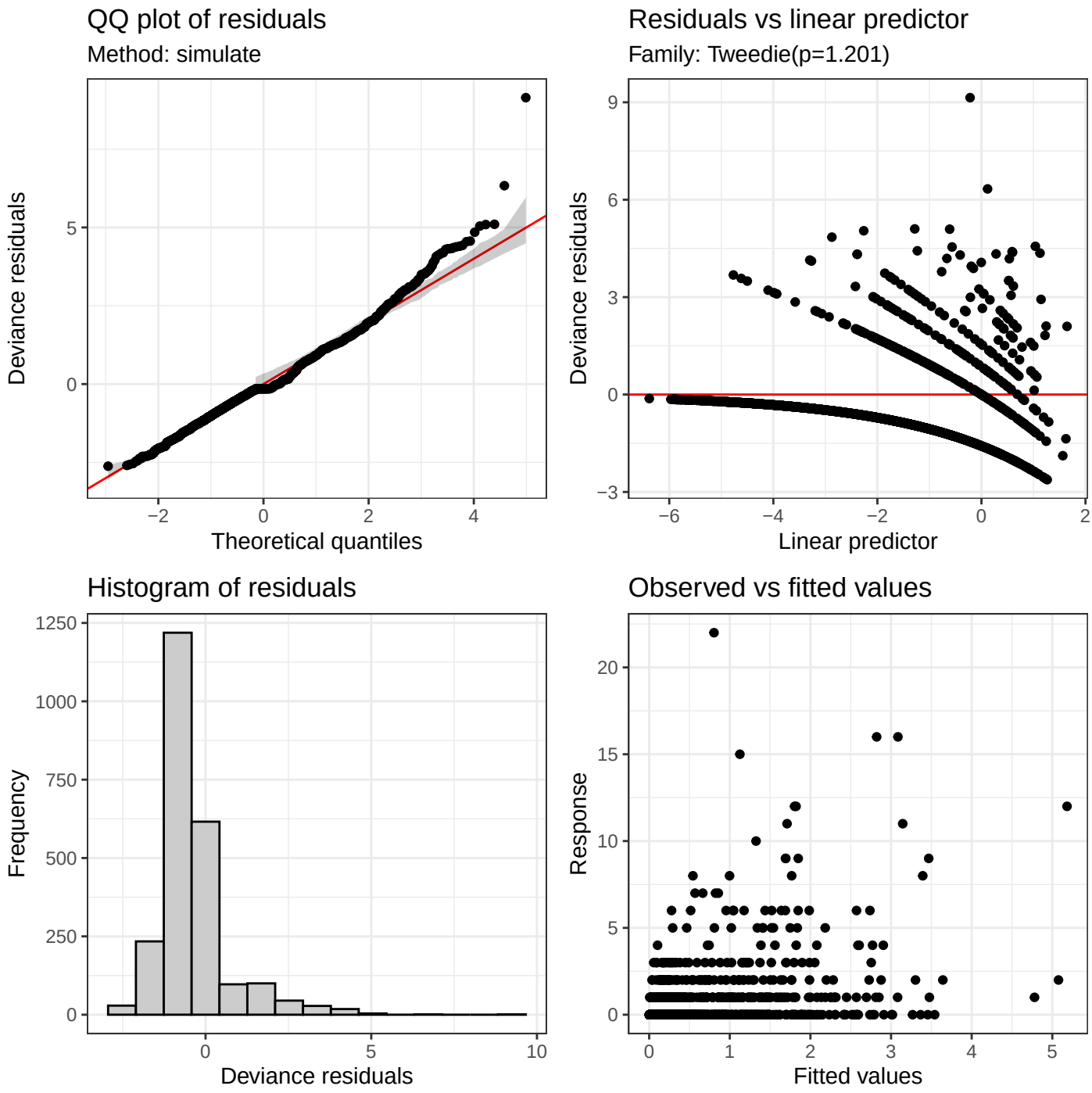


Figure 17: Diagnostic plots for the fin whale model static_fin_xy.

```
rqgam_check(static_fin_xy)
```

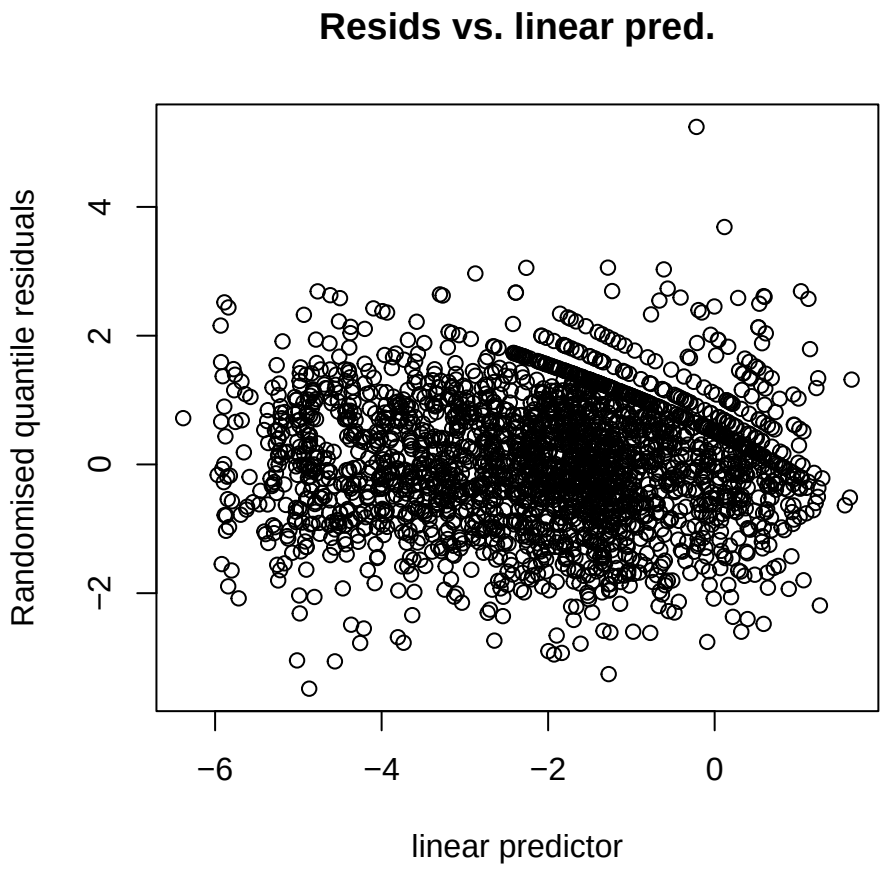


Figure 18: gam.check() diagnostics for the fin whale model static_fin_xy.

Predict the model and make a map.

```
pred_grid$off.set <- (25 * 1000)^2    # Set the offset to the cell size in meters, so we can sum for

pred_static_fin_xy <- predict(static_fin_xy, newdata = pred_grid, type = "response")

pred_points <- pred_grid |>
  select(geom) |>
  mutate(pred = pred_static_fin_xy)

segs_to_plot <- segments_retained_debugged |>
  left_join(segment_lines, by = "Sample.Label") |>
  left_join(obs_retained_debugged, by = "Sample.Label") |>
  group_by(Sample.Label, geom) |>
  summarize(
    WhalesPresent = any(!is.na(distance)),
  .groups = "drop") |>
  st_as_sf(sf_column_name = "geom") |>
  st_set_crs(3413)

tm_shape(pred_points) +
  tm_symbols(
    fill = "pred",
```

```
    shape = 15,
    size = 0.5,
    fill.scale = tm_scale_continuous(values = "matplotlib.turbo"),
    fill.legend = tm_legend(title = "Individuals\nper cell",
                            reverse = TRUE)) +
  tm_shape(segs_to_plot |> filter(!WhalesPresent)) +
  tm_symbols(fill = "white", col = "white", size = 0.1, fill_alpha = 0.3) +
  tm_shape(segs_to_plot |> filter(WhalesPresent)) +
  tm_symbols(fill = "red", col = "red", size = 0.3, fill_alpha = 0.3)
  #tm_basemap("Esri.WorldGrayCanvas") + tm_graticules(lines = FALSE)
```

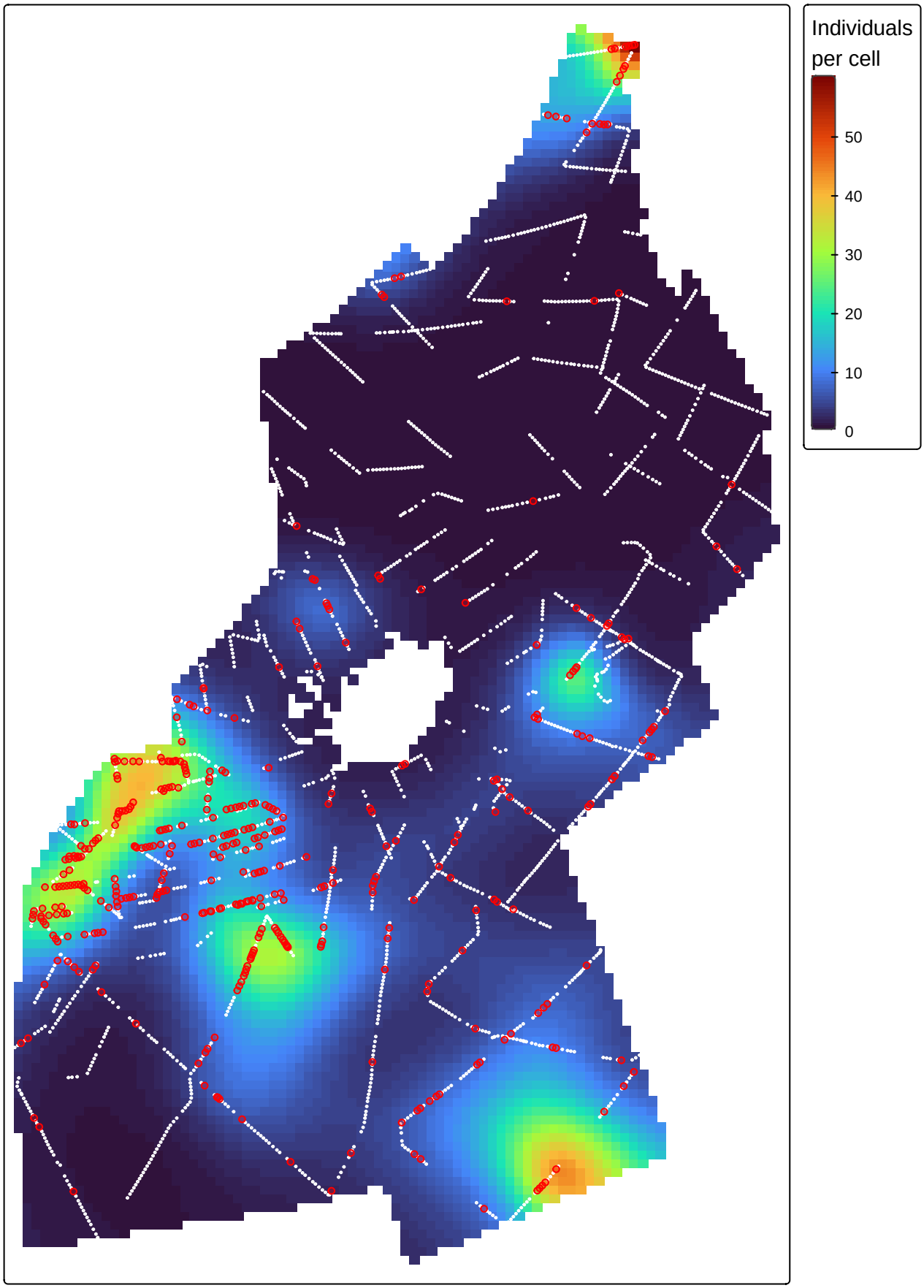


Figure 19: Predictions from the static_fin_xy model.

Let's compute total abundance. To do this, we simply need to sum values of the predicted density surface. This works because we used the same linear unit, meters, throughout the analysis:

1. The perpendicular distances of the sightings were given to us in meters and we just used them as-is.
2. The segment lengths were given as kilometers. These appear in the `Effort` column of the segments. But because we had already fitted detection functions in meters, we rescaled the segments to be in meters. You can find where we did this by searching this document for "Effort in the segments was originally given in km".
3. When predicting the dsm in the code chunk immediately above, we set `pred_grid$off.set` to the area of each cell in meters. This means that the prediction would be the number of individuals per grid cell.

So here's total abundance:

```
sum(pred_static_fin_xy)
## [1] 40337.54
```

For our analysis "individuals per grid cell" means "individuals per 25 square km". Density is usually reported in as "individuals per 1 square km". To get a map showing that, you can divide the raster by 25 and then plot it again.

References

Buckland ST, Anderson DR, Burnham KP, Laake JL, Borchers DL, Thomas L (2001) Introduction to distance sampling: Estimating abundance of biological populations. Oxford University Press, Oxford, UK

Miller DL, Burt ML, Rexstad EA, Thomas L (2013) Spatial models for distance sampling data: Recent developments and future directions. Methods Ecol Evol 4:1001–1010. doi: 10.1111/2041-210X.12105

Quick data tour of integrated model of common dolphins using INLA/-bru

Moritz Klaassen

2025-11-10

Contents

Introduction

This notebook gives a quick, runnable tour of the integrated model we use for short-beaked common dolphins (Delphinus delphis) off mainland Portugal (year 2020, monthly resolution). We combine a single month of a designed line-transect distance-sampling survey with multiple months of opportunistic whale-watching (WW) records from four coastal areas. The goal is to illustrate how broad spatial structure from the survey and high-frequency temporal signal from WW can be learned together in one coherent model.

We work in a joint-likelihood log-Gaussian Cox process fitted with INLA/inlabru. The survey contributes a continuous line-transect point-process likelihood with a half-normal detection function. The WW component contributes a presence-only likelihood, integrated only over month-specific availability polygons that capture

where operators were active. Covariates include monthly sea-surface temperature and chlorophyll-a (dynamic) and seabed slope (static). A spatial Matérn field on a triangulated mesh captures broad residual structure.

Preprint: bioRxiv DOI 10.1101/2025.11.07.687170

Methods

The components of our model are: (i) a single joint–likelihood LGCP that integrates a single-stage continuous line–transect distance–sampling likelihood with an availability–restricted presence–only likelihood in a shared latent state; (ii) a temporal formulation with monthly indexing that borrows the signal from WW via dynamic integration domains; (iii) a latent linear predictor comprising nonlinear covariate responses modelled as 1-D stochastic partial differential equation (SPDE) smooths and a residual 2-D Gaussian random field on the spatial domain, represented by a Matérn SPDE on a triangulated mesh; (iv) a barrier–aware Matérn SPDE to prevent correlation across land where appropriate.

Latent spatiotemporal intensity

Let $s \in \Omega \subset \mathbb{R}^2$ denote a two-dimensional spatial location, and let $t \in \mathcal{T} = \{1, \ldots, 12\}$ index months (January to December). The latent point intensity at (s, t) is

$$\lambda(s,t) = \exp\big(\eta(s,t)\big) \tag{1}$$

We write the linear predictor as

$$\eta(s,t) \;=\; \sum_{j=1}^{J} f_j\big(z_j(s,t)\big) \;+\; \omega(s) \tag{2}$$

where J is the number of covariates, z_j is the value of the jth environmental covariate, f_j is its smooth response, and $\omega(s)$ is a Matérn Gaussian random field implemented via the SPDE approach. We use a purely spatial residual field, $\omega(s)$, rather than a spatiotemporal field $\omega(s,t)$. Temporal variation is represented by the time-varying covariates $z_j(s,t)$.

Spatial field

We represent $\omega(s)$ as a continuous-space Matérn Gaussian random field (GRF), formulated through the SPDE representation (Lindgren *et al.* 2011). This formulation enables a sparse Gaussian Markov random field (GMRF) approximation on a triangulated mesh, providing computational efficiency while retaining the properties of the underlying GRF. The Matérn covariance between two locations separated by distance $r = \|s - s'\|$ is

$$Cov(r;\sigma,\rho) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{8\nu}\frac{r}{\rho}\right)^{\nu} K_\nu\left(\sqrt{8\nu}\frac{r}{\rho}\right) \tag{3}$$

where K_ν is the modified Bessel function of the second kind, σ is the marginal standard deviation, ρ is the practical range (Euclidean distance where correlation is ≈ 0.1), and ν is the smoothness parameter. We used penalised complexity (PC) priors on (ρ, σ), specified by tail probabilities $\Pr(\rho < \rho_0) = \alpha_1$ and $\Pr(\sigma > \sigma_0) = \alpha_2$ (Simpson *et al.* 2017; Fuglstad *et al.* 2019). Where appropriate, we employ a barrier variant of the SPDE to prevent correlation across land masses (Bakka *et al.* 2019).

Covariate smooths

We use smooth $f_j(\cdot)$ terms rather than linear effects because species responses to environmental covariates are rarely linear. Each f_j is modelled as a 1-D Matérn Gaussian field over the covariate domain, using an SPDE

representation again with PC priors. Following the SPDE formulation, we discretise $f_j(\cdot)$ on a 1-D mesh over the observed covariate range and use a finite-element basis of degree 2 (piecewise-quadratic functions):

$$f_j(z) = \sum_{m=1}^{M_j} b_{jm}(z)\,\Theta_{jm}, \qquad \Theta_j \sim \mathcal{N}\big(\mathbf{0},\, Q_j(\rho_j, \sigma_j)^{-1}\big) \tag{4}$$

where z is the covariate value, M_j is the number of (mesh) nodes in the 1-D mesh for covariate j, m indexes the mesh nodes; $b_{jm}(z)$ are the quadratic finite-element basis (shape) functions, Θ_{jm} are the Gaussian weights at node m, $\Theta_j = (\Theta_{j1}, \ldots, \Theta_{jM_j})^\top$ is the weight vector; and $Q_j(\rho_j, \sigma_j)$ is the SPDE precision matrix parameterised by the practical range ρ_j and marginal standard deviation σ_j. Each f_j uses the same priors across all analyses: a PC prior on the marginal SD with $\Pr(\sigma > 1) = 0.05$; the range ρ_j is fixed to the covariate span to encourage smoothness and mitigate overfitting.

Observation models

Line-transect survey with distance sampling We formulate our model building on the conventional distance-sampling observation process Buckland *et al.* (2015) so that detections from the survey arise from a thinned point process along transect lines. Let t be the survey month (fixed for this likelihood), C the union of survey strips in that month (half-width W), and $d(s)$ the perpendicular distance from s to the nearest transect line. The observed (thinned) intensity is

$$\lambda_s(s,t) = \exp\big(\beta_s + \eta(s,t)\big)\, g\big(d(s);\sigma\big), \qquad s \in C \tag{5}$$

and the log-likelihood is

$$\ell_s = \sum_{i \in \mathcal{D}_s} \Big(\beta_s + \eta(s_i,t) + \log g(d(s_i);\sigma)\Big) \;-\; \int_C \exp\big(\beta_s + \eta(s,t)\big)\, g\big(d(s);\sigma\big)\,\mathrm{d}s \tag{6}$$

where β_s is the survey-specific intercept, $g(\cdot)$ is the detection function, and $\mathcal{D}_s$ indexes survey detections. We adopt the half-normal key for the detection function as the main specification,

$$g(d;\sigma) = \exp\big(-\tfrac{1}{2}\,(d/\sigma)^2\big) \tag{7}$$

with a weakly informative prior distribution on the scale parameter σ set via an exponential mapping on the natural scale (centred relative to W). As a sensitivity check for the case study, we also fitted a hazard-rate detection function with the shape parameter fixed to $b = 1$:

$$g(d;\sigma) = 1 - \exp\big(-(d/\sigma)^{-1}\big) \tag{8}$$

Whale watching We treat opportunistic WW sightings as a presence-only point process linked to the latent state via an availability-restricted IPP. We assume the key feature of this data source to be concentrated, coordinated effort: multiple observers scan simultaneously (from vessels and shore lookouts), operators communicate by radio, follow each other to sightings, and commercial incentives (e.g., refund policies) favour thorough search within operating footprints. Rather than attempt to reconstruct a fine-scale effort surface for many unknown, overlapping, and mobile platforms, which would rely on unverifiable assumptions, we conditioned on availability. In practice, for time $t \in \{1, \ldots, 12\}$ we define dynamic availability polygons $A(t) \subset \Omega$. Within each $A(t)$ we assume uniform detectability (constant but unknown), which was absorbed into the WW intercept, while month-to-month exposure differences entered via a known scalar. Under this assumption, the WW component informs the shared components of the latent state (covariate responses and spatial field), while absolute scaling is anchored by the survey.

$$\ell_{\mathrm{WW}} \;=\; \sum_{i \in \mathcal{D}_{\mathrm{WW}}} \Big(\log q_{t_i} + \beta_{\mathrm{WW}} + \eta(s_i,t_i)\Big) \;-\; \sum_{t=1}^{12} \int_{A(t)} q_t\, \exp\big(\beta_{\mathrm{WW}} + \eta(s,t)\big)\,\mathrm{d}s, \qquad s \in A(t) \tag{9}$$

where $q_t > 0$ is the month-specific exposure scalar ($q_t = n_{\mathrm{op}}(t)$, the number of operating days in month t), β_{WW} is the whale-watching intercept, $\mathcal{D}_{\mathrm{WW}}$ indexes WW detections, and $A(t)$ are the month-specific availability polygons.

Joint log-likelihood and posterior

Let $y = (y_s, y_{\mathrm{WW}})$ be the observed data, $x = (\Theta_\omega, \{\Theta_j\}_{j=1}^J, \beta_s, \beta_{\mathrm{WW}})$ the latent Gaussian variables (Θ_ω are SPDE weights for $\omega(s)$; Θ_j are 1-D SPDE weights for $f_j(\cdot)$), and $\boldsymbol{\theta} = (\rho_\omega, \sigma_\omega, \{\sigma_j\}_{j=1}^J, \sigma_{\mathrm{det}})$ the hyperparameters, where $(\rho_\omega, \sigma_\omega)$ are the Matérn range and marginal SD of $\omega(s)$, σ_j are the marginal SDs of the 1-D smooths (with ρ_j fixed), and σ_{det} is the detection-scale parameter in g. Assuming conditional independence given $(x, \boldsymbol{\theta})$, the joint log-likelihood factorises as

$$\ell_{\mathrm{joint}}(y \mid x, \boldsymbol{\theta}) = \ell_s(y_s \mid x, \boldsymbol{\theta}) + \ell_{\mathrm{WW}}(y_{\mathrm{WW}} \mid x, \boldsymbol{\theta}) \tag{10}$$

The log-posterior probability density of $(x, \boldsymbol{\theta})$ given y is

$$\log \pi(x, \boldsymbol{\theta} \mid y) \;\propto\; \ell_{\mathrm{joint}}(y \mid x, \boldsymbol{\theta}) \;+\; \log \pi(x \mid \boldsymbol{\theta}) \;+\; \log \pi(\boldsymbol{\theta}) \tag{11}$$

Code

1) Load data

First, lets load data: species points, sampler geometries (survey lines and WW polygons), the SPDE mesh and boundaries, and the environmental raster stack.

```
##############################################
# Mega-Int
# 4. Model
# 4.2 Model
##############################################

# Directories ------------------------------------------------------------------
base_dir    <- dirname(getwd())
outputs_dir <- file.path(base_dir, "outputs")

# Libraries (project helper; loads pkgs if present) ----------------------------
libfile <- file.path(base_dir, "Libraries.R")
if (file.exists(libfile)) source(libfile)

# CRSs -------------------------------------------------------------------------
crs_utm <- st_crs(32629)
crs_km  <- "+proj=utm +zone=29 +datum=WGS84 +units=km +no_defs"
crs_ll  <- st_crs(4326)

# Load species points ----------------------------------------------------------
spea_path <- file.path(outputs_dir, "points", "spea_pts.shp")
lis_path  <- file.path(outputs_dir, "points", "lisbon_pts.shp")
albu_path <- file.path(outputs_dir, "points", "albu_pts.shp")
sagres_path <- file.path(outputs_dir, "points", "sagres_pts.shp")
faro_path   <- file.path(outputs_dir, "points", "faro_pts.shp")

spea_pts  <- st_read(spea_path,  quiet = TRUE)
lis_pts   <- st_read(lis_path,   quiet = TRUE)
albu_pts  <- st_read(albu_path,  quiet = TRUE)
sagres_pts<- st_read(sagres_path,quiet = TRUE)
faro_pts  <- st_read(faro_path,  quiet = TRUE)

# Load sampler geometries ------------------------------------------------------
lines_raw_spea <- st_read(file.path(outputs_dir, "samplers", "spea_samplers.shp"), quiet = TRUE)
```

```r
polygons_raw_lis  <- st_read(file.path(outputs_dir, "samplers", "lisbon_samplers.shp"), quiet = TRUE)
polygons_raw_albu <- st_read(file.path(outputs_dir, "samplers", "albu_samplers.shp"), quiet = TRUE)
polygons_raw_sagres <- st_read(file.path(outputs_dir, "samplers", "sagres_samplers_hulls.shp"), quiet =
polygons_raw_faro   <- st_read(file.path(outputs_dir, "samplers", "faro_samplers.shp"), quiet = TRUE)

# Load mesh & boundary ------------------------------------------------------
mesh <- readRDS(file.path(outputs_dir, "mesh", "mesh.RDS"))
boundary_inner_path <- file.path(outputs_dir, "mesh", "boundary_inner.shp")
boundary_inner   <- st_read(boundary_inner_path, quiet = TRUE)
boundary_inner_sf<- st_transform(boundary_inner, st_crs(crs_km))

# Load covariates stack -----------------------------------------------------
env_stack_full <- terra::rast(file.path(outputs_dir, "env_stack.tif"))
```

2) Prep species points and sampler polygons/lines

Next we standardise the raw inputs into sf layers with consistent columns and CRS (km). For species points we keep geometry, programme, month, and day (plus distance for the survey). For WW we also compute month-level exposure (days_out) per programme and join it onto monthly availability polygons; this becomes the weight used as the exposure scalar.

```r
# prep species data---------------------------------------------------------

pts_spea_sf <- spea_pts |>
  filter(month == 2) |>
  transmute(geometry,
            programme = "SPEA",
            month,
            day,
            distance = dst_md_k) |>
  st_transform(crs_km)

pts_lis_sf <- lis_pts |>
  mutate(day = day(as_date(ts))) |>
  transmute(geometry, programme = "LISBON", month, day) |>
  st_transform(crs_km)

pts_albu_sf <- albu_pts |>
  transmute(geometry, programme = "ALBU", month, day) |>
  st_transform(crs_km)

pts_sagres_sf <- sagres_pts |>
  mutate(day = day(as_date(ts))) |>
  transmute(geometry, programme = "SAGRES", month, day) |>
  st_transform(crs_km)

pts_faro_sf <- faro_pts |>
  mutate(day = day(as_date(ts))) |>
  transmute(geometry, programme = "FARO", month, day) |>
  st_transform(crs_km)

all_sf   <- bind_rows(pts_spea_sf, pts_lis_sf, pts_albu_sf, pts_sagres_sf, pts_faro_sf)
```

```r
head(all_sf)
```

```
## Simple feature collection with 6 features and 4 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 421.7714 ymin: 4084.101 xmax: 573.9475 ymax: 4464.385
## Projected CRS: +proj=utm +zone=29 +datum=WGS84 +units=km +no_defs
##   programme month day distance                  geometry
## 1      SPEA     2  11    0.075  POINT (573.9378 4085.21)
## 2      SPEA     2  24    0.075 POINT (497.4516 4464.385)
## 3      SPEA     2  24    0.075 POINT (473.6158 4450.002)
## 4      SPEA     2  19    0.075   POINT (421.7714 4287.2)
## 5      SPEA     2  11    0.075 POINT (573.9475 4084.101)
## 6      SPEA     2  24    0.025 POINT (473.6158 4450.002)
```

```r
# add weight to samplers-----------------------------------------------------

ww_days <- bind_rows(
  pts_lis_sf    |> st_drop_geometry() |> select(programme, month, day),
  pts_albu_sf   |> st_drop_geometry() |> select(programme, month, day),
  pts_sagres_sf |> st_drop_geometry() |> select(programme, month, day),
  pts_faro_sf   |> st_drop_geometry() |> select(programme, month, day)
) |>
  group_by(programme, month) |>
  summarise(days_out = n_distinct(day), .groups = "drop")


samplers_ww <- bind_rows(
  polygons_raw_lis    |> st_transform(crs_km) |> mutate(programme = "LISBON"),
  polygons_raw_albu   |> st_transform(crs_km) |> mutate(programme = "ALBU"),
  polygons_raw_sagres |> st_transform(crs_km) |> mutate(programme = "SAGRES"),
  polygons_raw_faro   |> st_transform(crs_km) |> mutate(programme = "FARO")
) |>
  group_by(programme, month) |>
  summarise(
    geometry = st_union(geometry),
    n_polys  = dplyr::n(),
    .groups  = "drop"
  ) |>
  mutate(
    area_km2 = units::set_units(st_area(geometry), "km^2") |> units::drop_units()
  ) |>
  left_join(ww_days, by = c("programme","month")) |>
  mutate(
    days_out = tidyr::replace_na(days_out, 0L),
    weight   = days_out) |>
  select(programme, month, weight, geometry)


# split back per programme for individual likelihoods---------------------------

samplers_lis    <- samplers_ww |> filter(programme == "LISBON")
samplers_albu   <- samplers_ww |> filter(programme == "ALBU")
samplers_sagres <- samplers_ww |> filter(programme == "SAGRES")
```

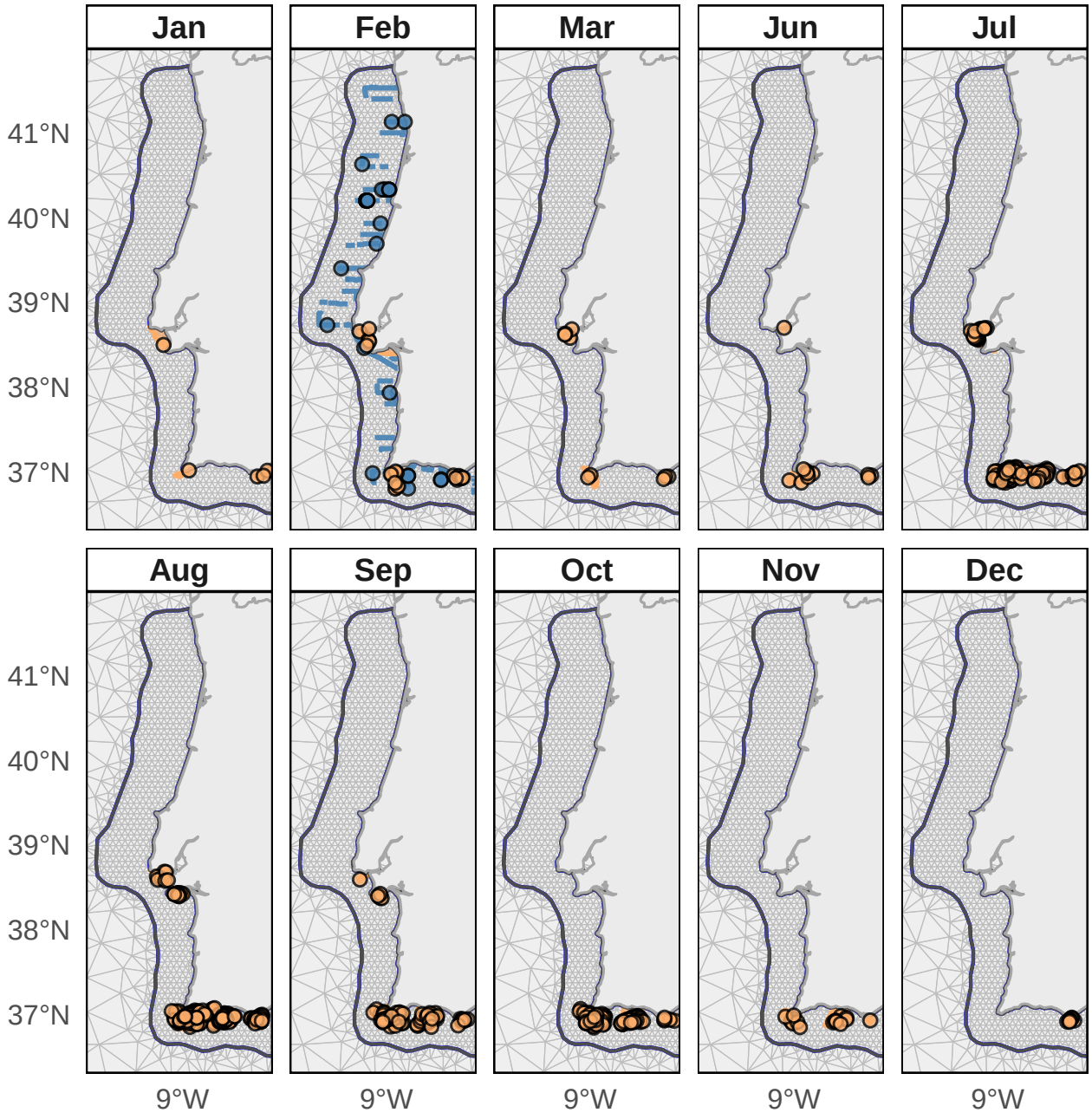
```r
samplers_faro   <- samplers_ww |> filter(programme == "FARO")

# samplers for survey-------------------------------------------------------------

samplers_spea <- lines_raw_spea |>
  st_transform(crs_km) |>
  mutate(programme = "SPEA") |>
  group_by(programme, month) |>
  summarise(
    geometry = st_union(geometry),
    .groups  = "drop")
```

3) Plot data

Lets have a quick look at our data over the mesh. We see that we have 10 months of data for whale watching and one month for the SPEA survey.

```r
# Quick plot before we start modelling

mesh_ll <- if (!identical(crs_ll$wkt, mesh$crs$wkt)) fm_transform(mesh, crs_ll) else mesh
boundary_ll <- st_transform(boundary_inner, crs_ll)
coast_ll <- ne_countries(scale = 10, returnclass = "sf") |> st_transform(crs_ll) |> st_make_valid()

samplers_all_ll <- bind_rows(
  st_transform(samplers_spea, crs_ll) |> mutate(programme = "SPEA"),
  list(samplers_lis, samplers_albu, samplers_sagres, samplers_faro) |>
    lapply(st_transform, crs_ll) |> bind_rows() |> mutate(programme = "WW")
) |> st_make_valid()

pts_all_ll <- bind_rows(
  st_transform(pts_spea_sf, crs_ll) |> mutate(programme = "SPEA"),
  list(pts_lis_sf, pts_albu_sf, pts_sagres_sf, pts_faro_sf) |>
    bind_rows() |> mutate(programme = "WW") |> st_transform(crs_ll)
)

month_labs <- format(as.Date(paste0("2020-", 1:12, "-01")), "%b")
samplers_all_ll$month <- as.integer(samplers_all_ll$month)
pts_all_ll$month <- as.integer(pts_all_ll$month)
samplers_all_ll$month_lab <- factor(month_labs[samplers_all_ll$month], levels = month_labs)
pts_all_ll$month_lab <- factor(month_labs[pts_all_ll$month], levels = month_labs)

bb0 <- st_bbox(boundary_ll)
pad_x <- 0.15; pad_y <- 0.20
bb <- bb0
bb["xmin"] <- bb0["xmin"] - pad_x
bb["xmax"] <- bb0["xmax"] + pad_x - 0.4
bb["ymin"] <- bb0["ymin"] - pad_y
bb["ymax"] <- bb0["ymax"] + pad_y

coast_win    <- suppressWarnings(st_crop(coast_ll, bb))
boundary_win <- suppressWarnings(st_crop(boundary_ll, bb))
sam_win      <- suppressWarnings(st_crop(samplers_all_ll, bb))
pts_win      <- suppressWarnings(st_crop(pts_all_ll, bb))
```

```r
deg_lab <- \(x) ifelse(x < 0, paste0(abs(x), "°W"), paste0(x, "°N"))
lon_brks <- seq(floor(bb["xmin"]), ceiling(bb["xmax"]), 2)
lat_brks <- seq(floor(bb["ymin"]), ceiling(bb["ymax"]), 1)
col_prog <- c(SPEA = "steelblue", WW = "#fdae6b")

p_facet <- ggplot() +
  geom_fm(data = mesh_ll, colour = "grey75", linewidth = .14, alpha = .6) +
  geom_sf(data = sam_win |> filter(programme == "WW"),
          fill = col_prog["WW"], colour = col_prog["WW"], linewidth = .6, alpha = .7) +
  geom_sf(data = sam_win |> filter(programme == "SPEA"),
          colour = col_prog["SPEA"], linewidth = .7, alpha = .9) +
  geom_sf(data = boundary_win, fill = NA, colour = "grey30", linewidth = .45) +
  geom_sf(data = coast_win, fill = "grey92", colour = "grey65", linewidth = .35) +
  geom_sf(data = pts_win, aes(fill = programme),
          shape = 21, colour = "black", size = 1.5, alpha = .8) +
  scale_fill_manual(values = col_prog, guide = "none") +
  coord_sf(crs = crs_ll,
           xlim = c(bb["xmin"], bb["xmax"]),
           ylim = c(bb["ymin"], bb["ymax"]),
           expand = FALSE) +
  scale_x_continuous(breaks = lon_brks, labels = \(x) gsub("N$", "", deg_lab(x))) +
  scale_y_continuous(breaks = lat_brks, labels = \(x) gsub("W$", "", deg_lab(x))) +
  facet_wrap(~month_lab, ncol = 5) +
  theme_minimal(base_size = 10) +
  theme(
    panel.grid = element_blank(),
    axis.title = element_blank(),
    strip.background = element_rect(fill = "white", colour = "black", linewidth = .4),
    strip.text = element_text(face = "bold", size = 9, margin = margin(2, 4, 2, 4)),
    panel.border = element_rect(colour = "black", fill = NA, linewidth = .4),
    plot.margin = margin(4, 4, 4, 4)
  )

p_facet
```

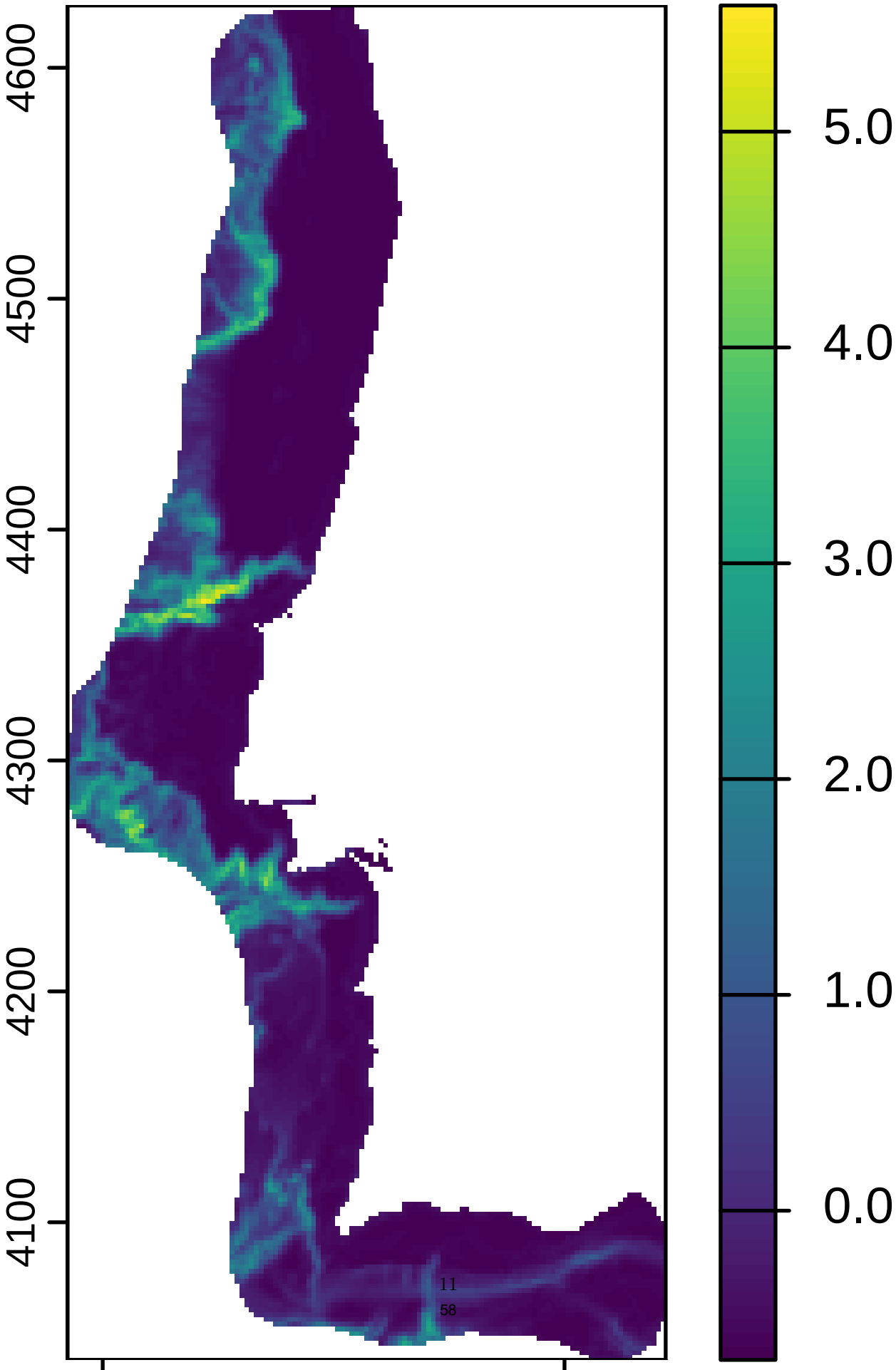


4) Prepare covariates

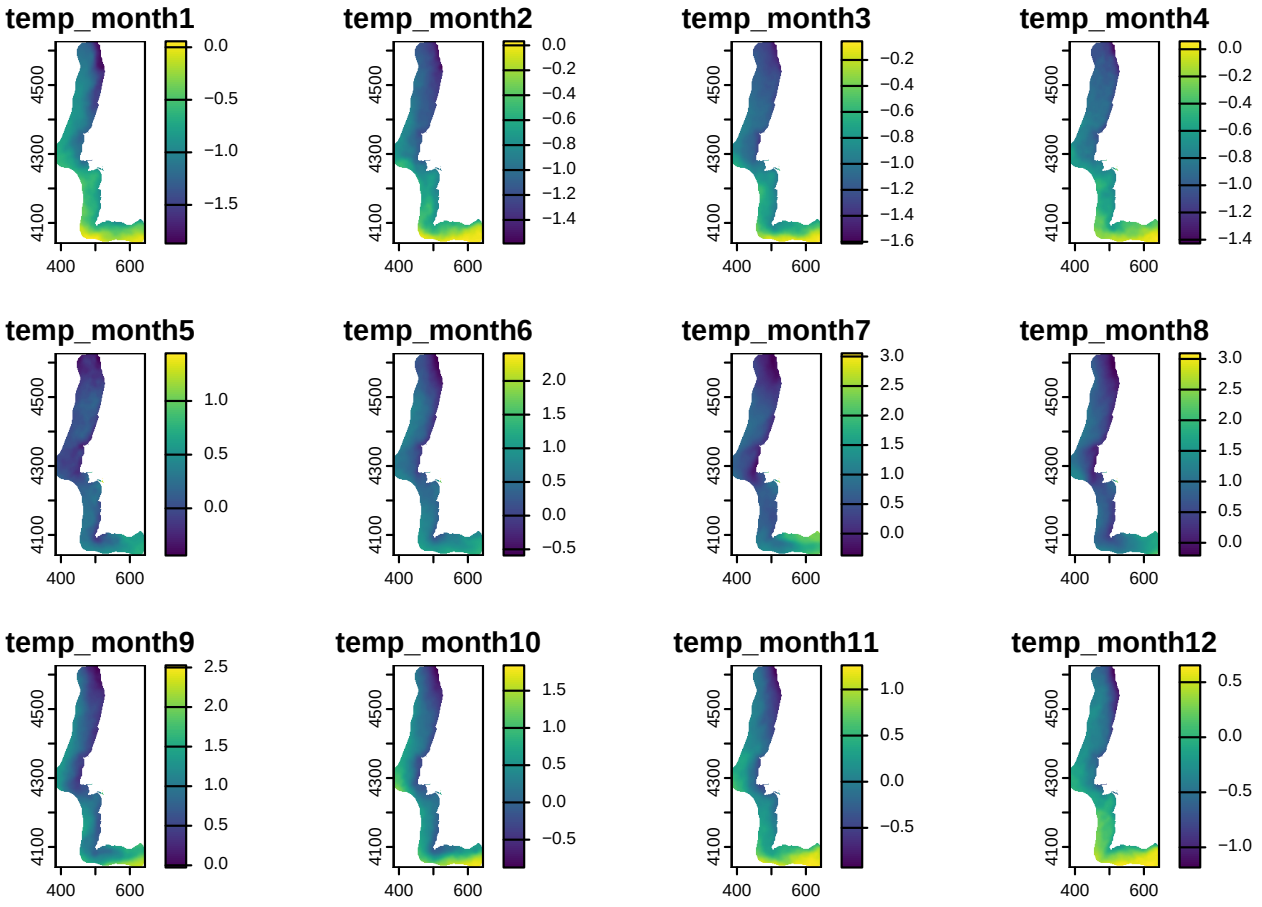
We use one static covariate (slope) and two dynamic stacks (monthly temperature and chlorophyll-a). First we project the full stack to the km CRS, then extract and standardize slope. For the dynamic stacks we standardize to mean 0 and SD 1 using the global mean and SD across all months, rename layers to temp_month1..12 and chl_month1..12, and clip to the inner boundary.

```
idx_static     <- c(26)  # slope only
env_stack_full <- project(env_stack_full, crs_km)

static_stack <- env_stack_full[[idx_static]] |>
  crop(boundary_inner_sf) |>
```

```
  mask(boundary_inner_sf) |>
  scale()

names(static_stack) <- c("slope")
plot(static_stack)
```

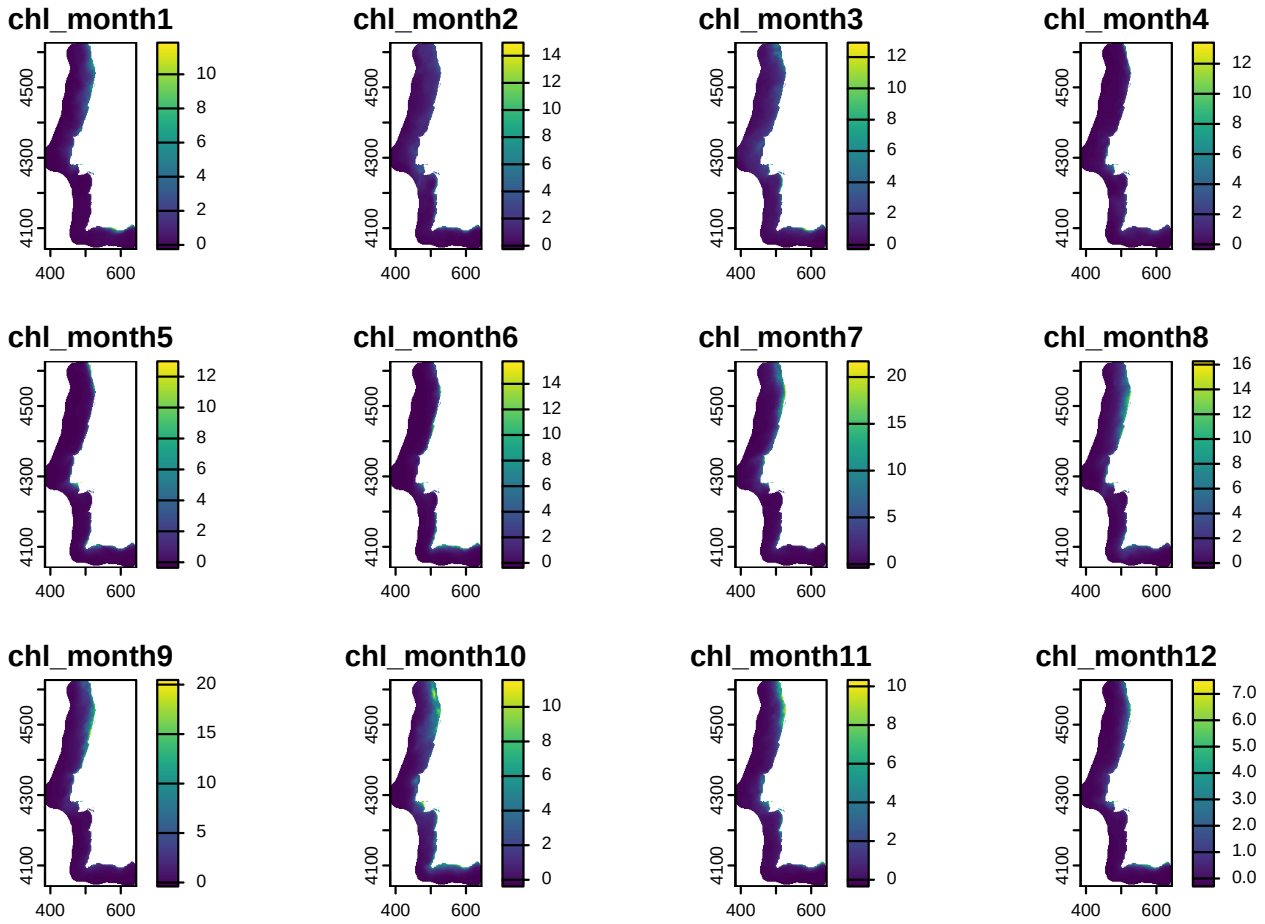

4600
4500
4400
4300
4200
4100
5.0
4.0
3.0
2.0
1.0
0.0
11
58

```r
# dynamic covariates temperature and chlorophyll--------------------------------
temp_stack <- env_stack_full[[ grep("temp_month", names(env_stack_full)) ]]

tvals      <- values(temp_stack)
temp_mean  <- mean(tvals, na.rm = TRUE)
temp_sd    <- sd(tvals, na.rm = TRUE)
temp_stack <- (temp_stack - temp_mean) / temp_sd
names(temp_stack) <- paste0("temp_month", 1:12)

temp_stack <- temp_stack |>
  crop(boundary_inner_sf) |>
  mask(boundary_inner_sf)

plot(temp_stack)
```




```r
# chlorophyll
chl_stack <- env_stack_full[[ grep("chl_month", names(env_stack_full)) ]]

chlvals      <- values(chl_stack)
chl_mean  <- mean(chlvals, na.rm = TRUE)
chl_sd    <- sd(chlvals, na.rm = TRUE)
chl_stack <- (chl_stack - chl_mean) / chl_sd
names(chl_stack) <- paste0("chl_month", 1:12)

chl_stack <- chl_stack |>
```

```
  crop(boundary_inner_sf) |>
  mask(boundary_inner_sf)

plot(chl_stack)
```



5) Prepare covariate smooths (1-D SPDEs)

Next, we prepare nonlinear responses to each covariate with a 1-D SPDE smoother on the covariate domain. For each covariate we:

We define a Matérn SPDE with PC priors: $\Pr(\sigma > 1)=0.05$; the prior range is set to the covariate span to encourage smoothness and reduce overfitting. We found this to work well in our case study, but urge that prior selection is different for different data / model and there is no "one-fits-all" solution

```
# Prep covariate smooths via stochastic partial differential equations (SPDEs)

val_temp  <- values(temp_stack)
val_chl   <- values(chl_stack)
val_slope <- values(static_stack[["slope"]])

# 1. Temperature
temp_range  <- range(val_temp, na.rm = TRUE)

temp_mesh   <- fm_mesh_1d(
```

```
  seq(temp_range[1], temp_range[2], length.out = 20),
  degree   = 2,
  boundary = "free"
)

temp_spde1d <- inla.spde2.pcmatern(
  mesh        = temp_mesh,
  alpha       = 2,
  prior.range = c(diff(temp_range), NA),
  prior.sigma = c(1, 0.05),
  constr      = TRUE
)

# 2. chlorophyll

chl_range  <- range(val_chl, na.rm = TRUE)

chl_mesh   <- fm_mesh_1d(
  seq(chl_range[1], chl_range[2], length.out = 20),
  degree   = 2,
  boundary = "free"
)

chl_spde1d <- inla.spde2.pcmatern(
  mesh        = chl_mesh,
  alpha       = 2,
  prior.range = c(diff(chl_range), NA),
  prior.sigma = c(1, 0.05),
  constr      = TRUE
)

# 3. slope

slope_range <- range(val_slope, na.rm = TRUE)
slope_mesh  <- fm_mesh_1d(
  seq(slope_range[1], slope_range[2], length.out = 20),
  degree   = 2,
  boundary = "free"
)

slope_spde1d <- inla.spde2.pcmatern(
  mesh        = slope_mesh,
  alpha       = 2,
  prior.range = c(diff(slope_range), NA),
  prior.sigma = c(1, 0.05),
  constr      = TRUE
)
```

6) Prep random field

We capture broad, residual spatial structure with a Matérn Gaussian random field $\omega(s)$ represented via the SPDE approach on the triangulated mesh mesh (units in km). We place penalised-complexity (PC) priors on the Matérn hyperparameters:

Pr(ρ < 150 km)=0.05, which favors ranges larger than 150 km while allowing shorter ranges if strongly supported. Pr(σ > 0.5)=0.05, shrinking the marginal SD toward small values unless the data require more spatial variation.

```
# Gaussian random field via SPDE-------------------------------------------------

xy           <- mesh$loc[,1:2]
range0_km    <- max(dist(xy)) /4

spde_space   <- INLA::inla.spde2.pcmatern(
  mesh,
  prior.range = c(150, 0.05),
  prior.sigma = c(0.5, 0.05),
  constr = TRUE
)
```

7) Prep distance mesh

For the line-transect likelihood we integrate the detection component over perpendicular distance d∈[0,W]. Our coordinates are in kilometers, so we set the truncation at W=0.25 km. We then build a 1-D finite-element mesh on [0,W] for numerical quadrature of the detection function.

```
# Distance mesh------------------------------------------------------------------

W_km <- 0.25
dist_mesh <- fm_mesh_1d(seq(0, W_km, length.out = 20))
```

8) Components and observation likelihoods

We defined a shared linear predictor and dataset-specific likelihoods. The predictor has:

one intercept for the survey and one intercept for each WW area. These absorb platform-specific exposure and constant detectability within footprint, so all sources share the same latent ecology while allowing scale differences.

a spatial Matérn field field on the 2-D mesh.

three covariate smooths: static slope and monthly temperature and chlorophyll, all as 1-D SPDE smooths.

For the survey, we use a continuous distance-sampling point-process formulation. The half-normal detection is included as log(hn(distance, σ)), plus log(2) to account for left and right sides of the transect. Presence-only WW likelihoods integrate only over their month-specific availability polygons. Quadrature uses a subdivided integration mesh mesh_cp (this is computationally friendly because the 2D SPDE still uses the coarser mesh). The detection-scale parameter σ has a weakly informative prior specified via an exponential mapping relative to the truncation width W_km.

```
# components---------------------------------------------------------------------
hn      <- function(distance, sigma) exp(-0.5 * (distance / sigma)^2)

cmp_dist <- ~
  Intercept_spea(1) +
  Intercept_lisbon(1) +
  Intercept_albu(1) +
  Intercept_sagres(1) +
  Intercept_faro(1) +
  field(
```

```
    main          = geometry,
    model         = spde_space
  ) +
  slope(main      = static_stack[["slope"]],      model = slope_spde1d) +
  temperature(
    eval_spatial(temp_stack, .data., layer = names(temp_stack)[month]),
    model = temp_spde1d
  ) +
  chlorophyll(
    eval_spatial(chl_stack, .data., layer = names(chl_stack)[month]),
    model = chl_spde1d
  ) +
  sigma(1,
        prec.linear = 1,
        marginal   = bru_mapper_marginal(qexp, pexp, dexp,
                                         rate = 1 / (W_km * .5)))

# likelihoods-------------------------------------------------------------------
mesh_cp <- fm_subdivide(mesh,4)


lik_spea <- bru_obs(
  "cp",
  data       = pts_spea_sf,
  samplers   = samplers_spea,
  domain     = list(geometry = mesh_cp, distance = dist_mesh, month = 2),
  formula    = geometry + distance + month ~
    Intercept_spea + field + temperature + slope + chlorophyll +
    log(hn(distance, sigma)) + log(2)
)

lik_lis <- bru_obs(
  "cp",
  data       = pts_lis_sf,
  samplers   = samplers_lis,
  domain     = list(geometry = mesh_cp, month = sort(unique(samplers_lis$month))),
  formula    = geometry + month ~ Intercept_lisbon + field + slope + temperature + chlorophyll
)

lik_albu <- bru_obs(
  "cp",
  data       = pts_albu_sf,
  samplers   = samplers_albu,
  domain     = list(geometry = mesh_cp, month = sort(unique(samplers_albu$month))),
  formula    = geometry + month ~ Intercept_albu + field + slope + temperature + chlorophyll
)

lik_sagres <- bru_obs(
  "cp",
  data       = pts_sagres_sf,
  samplers   = samplers_sagres,
  domain     = list(geometry = mesh_cp, month = sort(unique(samplers_sagres$month))),
  formula    = geometry + month ~ Intercept_sagres + field + slope + temperature + chlorophyll
```

```
)

lik_faro <- bru_obs(
  "cp",
  data       = pts_faro_sf,
  samplers   = samplers_faro,
  domain     = list(geometry = mesh_cp, month = sort(unique(samplers_faro$month))),
  formula    = geometry + month ~ Intercept_faro + field + slope + temperature + chlorophyll
)
```

9) Fit the model

Now we can fit the joint model. The INLA integration strategy is empirical Bayes (int.strategy = "eb"). We request DIC and the marginal posteriors of the latent predictor for later plotting. After fitting, it is useful to sanity check: overall fit (DIC), fixed effects (dataset intercepts), key hyperparameters (spatial field SD and range, 1-D smooth SDs, and the detection scale sigma), and the implied detection curve.

```
fit_int <- bru(
  cmp_dist,
  lik_spea,
  lik_lis,
  lik_albu,
  lik_sagres,
  lik_faro,
  options = list(
    control.compute = list(dic = TRUE, return.marginals.predictor = TRUE),
    control.inla    = list(int.strategy = "eb")
  )
)

summary(fit_int)
```

```
## inlabru version: 2.13.0
## INLA version: 25.06.07
## Components:
## Intercept_spea: main = linear(1), group = exchangeable(1L), replicate = iid(1L), NULL
## field: main = spde(geometry), group = exchangeable(1L), replicate = iid(1L), NULL
## slope: main = spde(static_stack[["slope"]]), group = exchangeable(1L), replicate = iid(1L), NULL
## temperature: main = spde(eval_spatial(temp_stack, .data., layer = names(temp_stack)[month])), group =
## chlorophyll: main = spde(eval_spatial(chl_stack, .data., layer = names(chl_stack)[month])), group = c
## sigma: main = linear(1), group = exchangeable(1L), replicate = iid(1L), NULL
## Intercept_lisbon: main = linear(1), group = exchangeable(1L), replicate = iid(1L), NULL
## Intercept_albu: main = linear(1), group = exchangeable(1L), replicate = iid(1L), NULL
## Intercept_sagres: main = linear(1), group = exchangeable(1L), replicate = iid(1L), NULL
## Intercept_faro: main = linear(1), group = exchangeable(1L), replicate = iid(1L), NULL
## Observation models:
##   Family: 'cp'
##     Tag: <No tag>
##     Data class: 'sf', 'tbl_df', 'tbl', 'data.frame'
##     Response class: 'numeric'
##     Predictor:
##         geometry + distance + month ~ Intercept_spea + field + temperature +
##             slope + chlorophyll + log(hn(distance, sigma)) + log(2)
```

```
##     Additive/Linear: FALSE/FALSE
##     Used components: effects[Intercept_spea, field, slope, temperature, chlorophyll, sigma], latent[]
##   Family: 'cp'
##     Tag: <No tag>
##     Data class: 'sf', 'tbl_df', 'tbl', 'data.frame'
##     Response class: 'numeric'
##     Predictor:
##         geometry + month ~ Intercept_lisbon + field + slope + temperature +
##             chlorophyll
##     Additive/Linear: TRUE/TRUE
##     Used components: effects[Intercept_lisbon, field, slope, temperature, chlorophyll], latent[]
##   Family: 'cp'
##     Tag: <No tag>
##     Data class: 'sf', 'tbl_df', 'tbl', 'data.frame'
##     Response class: 'numeric'
##     Predictor:
##         geometry + month ~ Intercept_albu + field + slope + temperature +
##             chlorophyll
##     Additive/Linear: TRUE/TRUE
##     Used components: effects[Intercept_albu, field, slope, temperature, chlorophyll], latent[]
##   Family: 'cp'
##     Tag: <No tag>
##     Data class: 'sf', 'tbl_df', 'tbl', 'data.frame'
##     Response class: 'numeric'
##     Predictor:
##         geometry + month ~ Intercept_sagres + field + slope + temperature +
##             chlorophyll
##     Additive/Linear: TRUE/TRUE
##     Used components: effects[Intercept_sagres, field, slope, temperature, chlorophyll], latent[]
##   Family: 'cp'
##     Tag: <No tag>
##     Data class: 'sf', 'tbl_df', 'tbl', 'data.frame'
##     Response class: 'numeric'
##     Predictor:
##         geometry + month ~ Intercept_faro + field + slope + temperature +
##             chlorophyll
##     Additive/Linear: TRUE/TRUE
##     Used components: effects[Intercept_faro, field, slope, temperature, chlorophyll], latent[]
## Time used:
##     Pre = 3.25, Running = 6.89, Post = 47.7, Total = 57.8
## Fixed effects:
##                    mean    sd 0.025quant 0.5quant 0.975quant   mode kld
## Intercept_spea   -3.677 0.616     -4.884   -3.677     -2.470 -3.677   0
## sigma             0.331 0.199     -0.060    0.331      0.721  0.331   0
## Intercept_lisbon -4.983 0.902     -6.751   -4.983     -3.214 -4.983   0
## Intercept_albu   -7.944 0.760     -9.433   -7.944     -6.454 -7.944   0
## Intercept_sagres -6.947 0.701     -8.321   -6.947     -5.573 -6.947   0
## Intercept_faro   -6.688 0.782     -8.221   -6.688     -5.154 -6.688   0
##
## Random effects:
##   Name     Model
##     field SPDE2 model
##    slope SPDE2 model
##    temperature SPDE2 model
```

```
##    chlorophyll SPDE2 model
##
## Model hyperparameters:
##                          mean    sd 0.025quant 0.5quant 0.975quant   mode
## Range for field       41.949 8.372     28.128   41.056      60.95 39.212
## Stdev for field        1.572 0.240      1.157    1.553       2.10  1.513
## Stdev for slope        0.600 0.352      0.165    0.520       1.50  0.382
## Stdev for temperature  0.486 0.251      0.158    0.434       1.12  0.342
## Stdev for chlorophyll  0.669 0.374      0.205    0.585       1.63  0.445
##
## Deviance Information Criterion (DIC) ...............: -9137.78
## Deviance Information Criterion (DIC, saturated) ....: -9158.29
## Effective number of parameters .....................: -10346.26
##
## Marginal log-Likelihood:  -5926.13
##  is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

10) Evaluate covariate effects

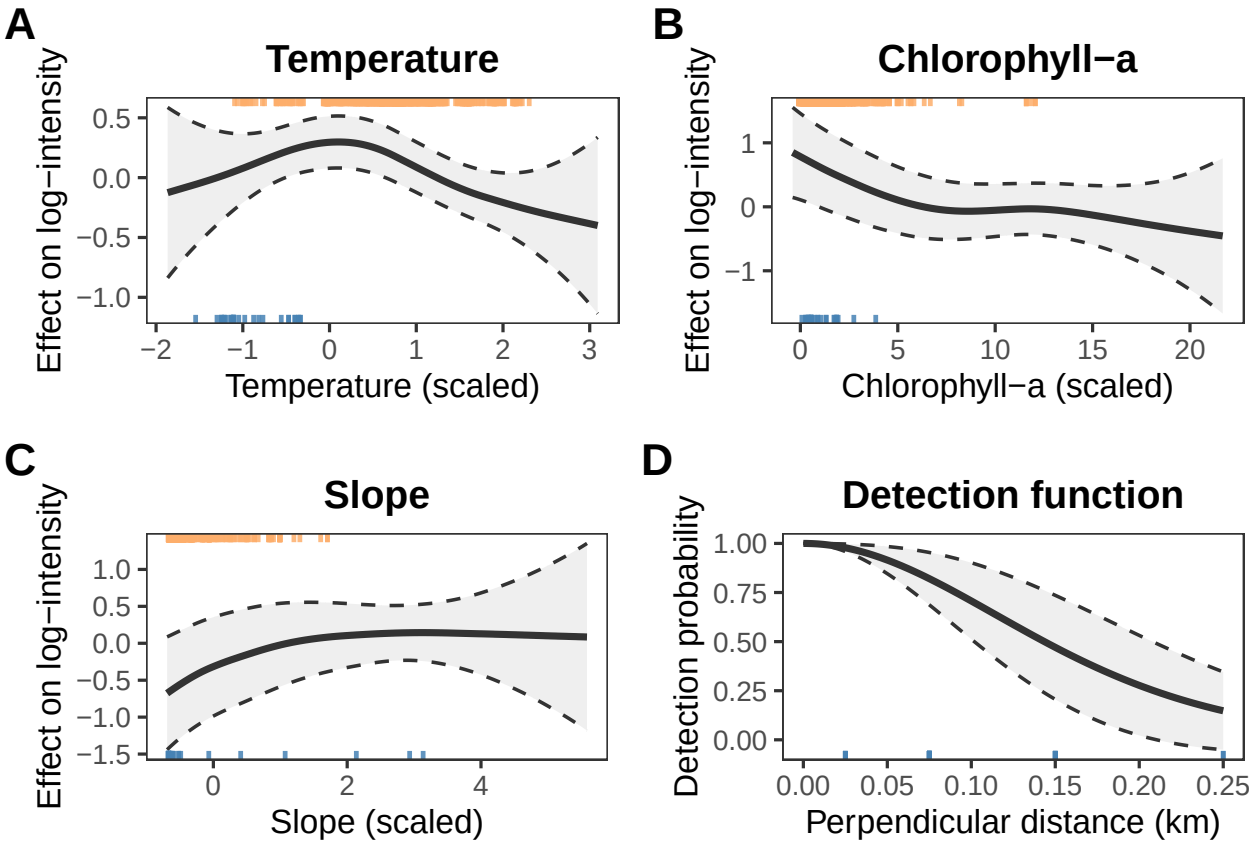
Now, we can evaluate our model. For this example, we first look at the covariate smooths and the fitted detection function.

```r
# Helper
plot_smooth <- function(fit, comp, grid_vals, rug_w, rug_s, x_lab) {

  df   <- setNames(data.frame(grid_vals), comp)
  form <- as.formula(sprintf("~ %s_eval(%s)", comp, comp))

  pred <- predict(
    fit, df,
    formula   = form,
    n.samples = 2000,
    include   = character(0)
  )

  ggplot(pred, aes(x = .data[[comp]], y = mean)) +
    geom_ribbon(aes(ymin = mean - 1.96*sd,
                    ymax = mean + 1.96*sd),
                fill = "#C0C0C0", alpha = 0.25) +
    geom_line(aes(y = mean + 1.96*sd),
              linetype = "dashed", colour = "#333333", linewidth = 0.6) +
    geom_line(aes(y = mean - 1.96*sd),
              linetype = "dashed", colour = "#333333", linewidth = 0.6) +
    geom_line(colour = "#333333", linewidth = 1.2) +
    # survey rugs (bottom)
    geom_rug(
      data = setNames(data.frame(rug_s), comp),
      mapping = aes(x = .data[[comp]]),
      sides   = "b",
      colour  = "steelblue", alpha = 0.85,
      length  = grid::unit(0.04, "npc"), linewidth = 0.8,
      inherit.aes = FALSE
```

```
    ) +
    # pooled WW rugs (top)
    geom_rug(
      data = setNames(data.frame(rug_w), comp),
      mapping = aes(x = .data[[comp]]),
      sides   = "t",
      colour  = "#fdae6b", alpha = 0.85,
      length  = grid::unit(0.04, "npc"), linewidth = 0.8,
      inherit.aes = FALSE
    ) +
    labs(
      x     = sprintf("%s (scaled)", x_lab),
      y     = "Effect on log-intensity",
      title = x_lab
    ) +
    theme_bw(base_size = 12) +
    theme(
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      plot.title = element_text(hjust = 0.5, face = "bold")
    )
}

# Pool WW points for rugs
ww_pts_sf <- dplyr::bind_rows(pts_lis_sf, pts_albu_sf, pts_sagres_sf, pts_faro_sf)
s_pts_sf  <- pts_spea_sf

# Temperature ------------------------------------------------------------------
temp_rng  <- range(terra::values(temp_stack), na.rm = TRUE)
temp_grid <- seq(temp_rng[1], temp_rng[2], length.out = 400)

get_pt_temp <- function(sf_pts) {
  eval_spatial(
    temp_stack, sf_pts,
    layer = names(temp_stack)[sf_pts$month]
  )
}
temp_ww <- get_pt_temp(ww_pts_sf)
temp_s  <- get_pt_temp(s_pts_sf)

p_temp <- plot_smooth(
  fit_int, "temperature",
  temp_grid, temp_ww, temp_s,
  "Temperature"
)

# Chlorophyll-a ----------------------------------------------------------------
chl_rng  <- range(terra::values(chl_stack), na.rm = TRUE)
chl_grid <- seq(chl_rng[1], chl_rng[2], length.out = 400)

get_pt_chl <- function(sf_pts) {
  eval_spatial(
    chl_stack, sf_pts,
```

```
    layer = names(chl_stack)[sf_pts$month]
  )
}
chl_ww <- get_pt_chl(ww_pts_sf)
chl_s  <- get_pt_chl(s_pts_sf)

p_chl <- plot_smooth(
  fit_int, "chlorophyll",
  chl_grid, chl_ww, chl_s,
  "Chlorophyll-a"
)

# Slope (static) ---------------------------------------------------------
slope_rng  <- range(terra::values(static_stack[["slope"]]), na.rm = TRUE)
slope_grid <- seq(slope_rng[1], slope_rng[2], length.out = 400)

slope_ww <- eval_spatial(static_stack[["slope"]], ww_pts_sf)
slope_s  <- eval_spatial(static_stack[["slope"]], s_pts_sf)

p_slope <- plot_smooth(
  fit_int, "slope",
  slope_grid, slope_ww, slope_s,
  "Slope"
)

# Detection function -----------------------------------------------------
hn <- function(distance, sigma) exp(-0.5 * (distance / sigma)^2)

dist_grid <- data.frame(distance = seq(0, W_km, length.out = 150))
det_pred  <- predict(
  fit_int, dist_grid,
  formula   = ~ hn(distance, sigma),
  n.samples = 2000,
  include   = "sigma"
)
dist_obs  <- s_pts_sf$distance

p_det <- ggplot(det_pred, aes(distance, mean)) +
  geom_ribbon(aes(ymin = mean - 1.96*sd, ymax = mean + 1.96*sd),
              fill = "#C0C0C0", alpha = 0.25) +
  geom_line(aes(y = mean + 1.96*sd),
            linetype = "dashed", colour = "#333333", linewidth = 0.6) +
  geom_line(aes(y = mean - 1.96*sd),
            linetype = "dashed", colour = "#333333", linewidth = 0.6) +
  geom_line(colour = "#333333", linewidth = 1.2) +
  geom_rug(data = data.frame(distance = dist_obs), aes(distance),
           sides   = "b",
           colour  = "steelblue", alpha = 0.85,
           length  = grid::unit(0.04, "npc"), linewidth = 0.8,
           inherit.aes = FALSE) +
  labs(x = "Perpendicular distance (km)",
       y = "Detection probability",
       title = "Detection function") +
```

```
  theme_bw(base_size = 12) +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    plot.title = element_text(hjust = 0.5, face = "bold")
  )

(p_temp | p_chl) /
(p_slope | p_det) +
  plot_annotation(tag_levels = "A") &
  theme(plot.tag = element_text(size = 16, face = "bold"))
```



11) Evaluate spatially

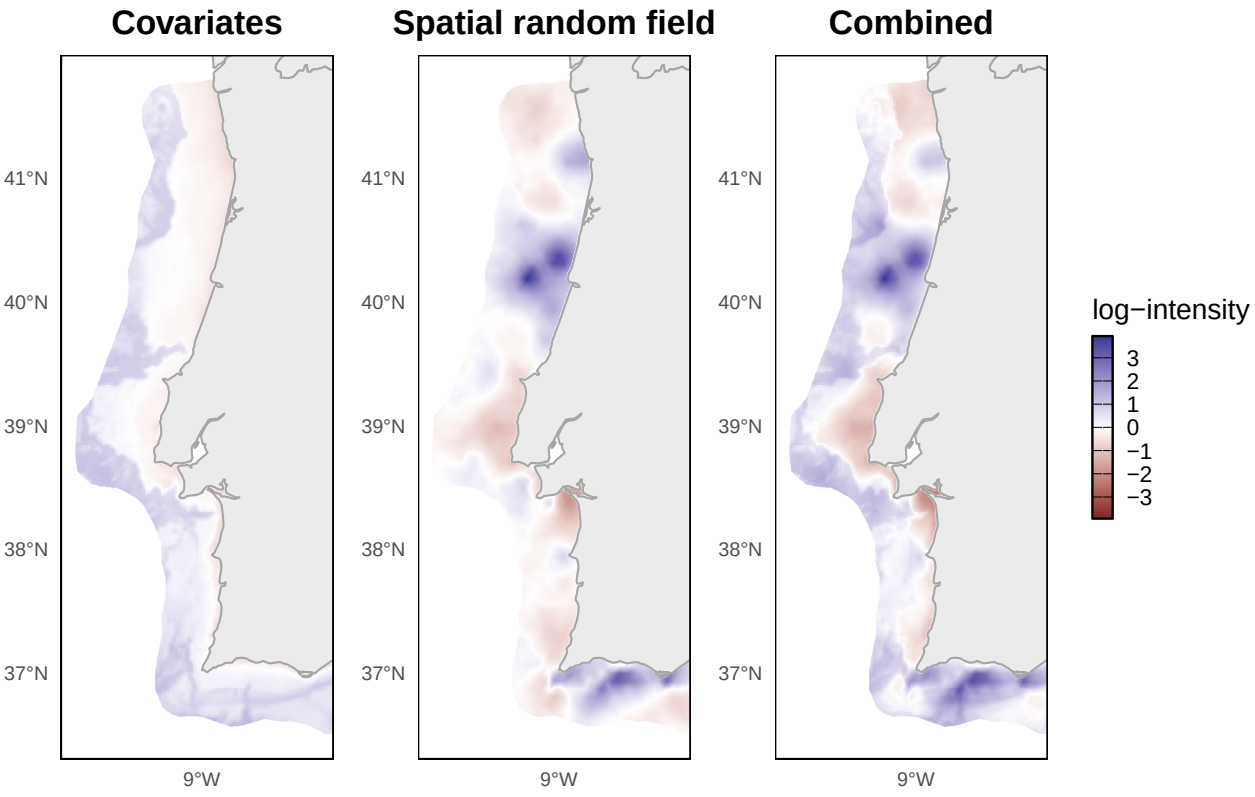
We map three components of the linear predictor on the **log-intensity** scale for a chosen month `t_month`: 1) **Covariate-only**: slope + temperature$_t$ + chlorophyll$_t$
2) **Spatial GRF**: $\omega(s)$
3) **Combined**: covariates + GRF (intercepts excluded)

```
# Component surfaces

cell_km <- 0.5
bb      <- st_bbox(boundary_inner_sf)

nx <- ceiling((bb["xmax"] - bb["xmin"]) / cell_km)
ny <- ceiling((bb["ymax"] - bb["ymin"]) / cell_km)
```

```r
pxl_base <- fm_pixels(
  mesh, dims = c(nx, ny),
  mask = boundary_inner_sf
)

r_template <- rast(
  xmin = bb["xmin"], xmax = bb["xmax"],
  ymin = bb["ymin"], ymax = bb["ymax"],
  ncol = nx, nrow = ny,
  crs  = st_crs(pxl_base)$proj4string
)

predict_feb <- function(rhs, pxl_template, fit_int, r_template) {
  pxl_feb <- pxl_template
  pxl_feb$month <- 2  # february
  pr <- predict(fit_int, pxl_feb, formula = as.formula(paste("~", rhs)), seed = 123)
  v  <- terra::vect(structure(pr, class = c("sf","data.frame")))
  r  <- rasterize(v, r_template, field = "mean")
  names(r) <- gsub("[ ~]", "", rhs)
  r
}

r_cov   <- predict_feb("slope + chlorophyll + temperature", pxl_base, fit_int, r_template)
r_field <- predict_feb("field",                             pxl_base, fit_int, r_template)
r_full  <- predict_feb("field + slope + chlorophyll + temperature", pxl_base, fit_int, r_template)

# Project to lon/lat
p_cov_ll   <- project(r_cov,   "EPSG:4326", method = "bilinear")
p_field_ll <- project(r_field, "EPSG:4326", method = "bilinear")
p_full_ll  <- project(r_full,  "EPSG:4326", method = "bilinear")

mm_all <- range(c(terra::minmax(p_cov_ll), terra::minmax(p_field_ll), terra::minmax(p_full_ll)), na.rm =
ticks  <- scales::pretty_breaks(7)(mm_all)

bb0 <- sf::st_bbox(sf::st_union(boundary_ll))
pad_x <- 0.15
pad_y <- 0.20
bb_tight <- sf::st_bbox(c(
  xmin = as.numeric(bb0["xmin"]) - pad_x,
  xmax = as.numeric(bb0["xmax"]) + pad_x,
  ymin = as.numeric(bb0["ymin"]) - pad_y,
  ymax = as.numeric(bb0["ymax"]) + pad_y
), crs = crs_ll)

trim_east <- 0.4
bb_custom <- bb_tight
bb_custom["xmax"] <- as.numeric(bb_tight["xmax"]) - trim_east

ext_win <- terra::ext(
  as.numeric(bb_custom["xmin"]),
  as.numeric(bb_custom["xmax"]),
  as.numeric(bb_custom["ymin"]),
  as.numeric(bb_custom["ymax"])
```

```
)
p_cov_ll   <- terra::crop(p_cov_ll,   ext_win, snap = "out")
p_field_ll <- terra::crop(p_field_ll, ext_win, snap = "out")
p_full_ll  <- terra::crop(p_full_ll,  ext_win, snap = "out")

coast_win    <- suppressWarnings(sf::st_crop(coast_ll,    bb_custom))
boundary_win <- suppressWarnings(sf::st_crop(boundary_ll, bb_custom))

ext_vals <- c(
  terra::minmax(p_cov_ll),
  terra::minmax(p_field_ll),
  terra::minmax(p_full_ll)
)
v_lim <- max(abs(range(ext_vals, na.rm = TRUE)))
lims  <- c(-v_lim, v_lim)
ticks <- scales::pretty_breaks(7)(lims)

deg_lab  <- function(x) ifelse(x < 0, paste0(abs(x), "°W"), paste0(x, "°N"))
lon_brks <- seq(floor(as.numeric(bb_custom["xmin"])),
                ceiling(as.numeric(bb_custom["xmax"])), by = 2)
lat_brks <- seq(floor(as.numeric(bb_custom["ymin"])),
                ceiling(as.numeric(bb_custom["ymax"])), by = 1)

make_component_map <- function(r_ll, title) {
  df <- as.data.frame(r_ll, xy = TRUE, na.rm = FALSE)
  names(df)[3] <- "value"

  ggplot(df) +
    geom_raster(aes(x, y, fill = value)) +
    geom_sf(data = coast_win, fill = "grey92", colour = "grey65", linewidth = 0.35) +
    coord_sf(
      crs   = crs_ll,
      xlim  = c(as.numeric(bb_custom["xmin"]), as.numeric(bb_custom["xmax"])),
      ylim  = c(as.numeric(bb_custom["ymin"]), as.numeric(bb_custom["ymax"])),
      expand = FALSE
    ) +
    scale_x_continuous(breaks = lon_brks,
                       labels = function(x) gsub("N$", "", deg_lab(x)),
                       expand = c(0, 0)) +
    scale_y_continuous(breaks = lat_brks,
                       labels = function(x) gsub("W$", "", deg_lab(x)),
                       expand = c(0, 0)) +
    scale_fill_gradient2(
      name     = "log-intensity",
      limits   = lims,
      breaks   = ticks,
      midpoint = 0,
      na.value = "white",
      guide = guide_colourbar(
        barwidth  = .5, barheight = 4.5,
        frame.col = "black", frame.linewidth = .4, ticks.col = "black"
      )
    ) +
```

```
    labs(title = title) +
    theme_minimal(base_size = 10) +
    theme(
      panel.background = element_rect(fill = "white", colour = NA),
      panel.border     = element_rect(colour = "black", fill = NA, linewidth = 0.4),
      panel.grid       = element_blank(),
      axis.title       = element_blank(),
      axis.text        = element_text(size = 7),
      strip.background = element_rect(fill = "white", colour = "black", linewidth = 0.4),
      plot.title       = element_text(face = "bold", hjust = 0.5),
      legend.position  = "right"
    )
}

p_cov_map   <- make_component_map(p_cov_ll,   "Covariates")
p_field_map <- make_component_map(p_field_ll, "Spatial random field")
p_full_map  <- make_component_map(p_full_ll,  "Combined")

p_components <- (p_cov_map | p_field_map | p_full_map) + patchwork::plot_layout(ncol = 3, guides = "col
p_components
```



Interpretation. The spatial GRF clearly dominates the linear predictor in this case. The dynamic covariates contribute smaller adjustments, so month-to-month differences are weak. To avoid a long sequence of near-identical panels, we present a single representative month (`t_month`) here and do not map all months.

Bakka, H., Vanhatalo, J., Illian, J.B., Simpson, D. & Rue, H. (2019). Non-stationary Gaussian models with physical barriers. *Spatial Statistics*, **29**, 268–288.

Buckland, S.T., Rexstad, E.A., Marques, T.A. & Oedekoven, C.S. (2015). *Distance sampling: Methods and*

applications. Springer International Publishing, Cham. Retrieved from https://link.springer.com/book/10.1007/978-3-319-19219-2

Fuglstad, G.-A., Simpson, D., Lindgren, F. & Rue, H. (2019). Constructing priors that penalize the complexity of Gaussian random fields. *Journal of the American Statistical Association*, **114**, 445–452.

Lindgren, F., Rue, H. & Lindström, J. (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: The stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **73**, 423–498.

Simpson, D., Rue, H., Riebler, A., Martins, T.G. & Sørbye, S.H. (2017). Penalising model component complexity: A principled, practical approach to constructing priors. *Statistical Science*, **32**, 1–28.