Journal of Statistical Software

August 2024, Volume 110, Issue 5. *doi: 10.18637/jss.v110.i05*

bayesnec: An R Package for Concentration-Response Modeling and Estimation of Toxicity Metrics

Rebecca Fisher
Australian Institute of Marine Science

Diego R. Barneche
Australian Institute of Marine Science

Gerard F. Ricardo
The University of Queensland

David R. Fox
The University of Melbourne

Abstract

The **bayesnec** package has been developed for R to fit concentration (dose)-response curves (CR) to toxicity data for the purpose of deriving no-effect-concentration (NEC), no-significant-effect-concentration (NSEC), and effect-concentration (of specified percentage "x", ECx) thresholds from non-linear models fitted using Bayesian Hamiltonian Monte Carlo (HMC) via R packages **brms** and **rstan** or **cmdstanr**. In **bayesnec** it is possible to fit a single model, custom model-set, specific model-set or all of the available models. When multiple models are specified, the `bnec()` function returns a model weighted average estimate of predicted posterior values. A range of support functions and methods is also included to work with the returned single, or multi-model objects that allow extraction of raw, or model averaged predicted, NEC, NSEC and ECx values and to interrogate the fitted model or model-set. By combining Bayesian methods with model averaging, **bayesnec** provides a single estimate of toxicity and associated uncertainty that can be directly integrated into risk assessment frameworks.

Keywords: concentration-response, no-effect-concentration, Bayesian, non-linear modeling.

1. Introduction

Concentration-response (CR) modeling (also known as dose-response modeling or dose-response analysis) is a key tool for assessing toxicity and deriving the toxicity thresholds used in the risk assessments that underpin protection of human health and the environment. It is widely used in the disciplines of pharmacology, toxicology and ecotoxicology, where parametric non-linear regression methods are used to model response data of interest, with the resulting fitted models used to estimate critical thresholds of concern. These thresholds may be used directly to assess risk (e.g., see Fisher, Walshe, Bessell-Browne, and Jones 2018), or are subsequently incorporated into a broader population-level risk assessment framework (e.g., Warne *et al.* 2015). Typical thresholds derived from CR modeling include the effect-concentration of defined percentage "x" (ECx) and the no-effect-concentration (NEC), the latter being the generally preferred option (Fox 2008; Warne *et al.* 2015, 2018). In addition, the no-significant-effect-concentration (NSEC) has also been recently defined (Fisher and Fox 2023), and may represent a good alternative estimate of "no-effect" concentrations when the threshold models required to derive true NEC estimates are not appropriate (Fisher and Fox 2023; Fisher, Fox, Negri, Van Dam, Flores, and Koppel 2024b).

In qualitative terms, CR models are typically a decreasing function of concentration, whereby the response may remain relatively stable for the initial portion of the curve, and then decays at some rate to zero (or some other, lower bound at high concentration). An example is death of an organism resulting from ever increasing concentrations of a toxic pollutant. However, often the underlying mechanisms describing CR relationships are not known, and therefore numerous alternative non-linear CR equations have been proposed (e.g., models in **drc**, Ritz, Baty, Streibig, and Gerhard 2015). These can be broadly grouped into two main "model categories" (see Section 2.1 for the mathematical definition of each model): NEC models – threshold models which contain a step function comprising the "break-point" after which the predictor systematically alters the response (Fox 2010); and Smooth transition models that are typically used for estimating effect concentrations of a specified effect (e.g., ECx models). They may or may not encompass the EC_{50} as a parameter (Ritz *et al.* 2015). Each of these two groups can be further split into two categories depending on whether the initial portion of the curve is flat, or increasing – the latter being known as hormesis models (Ritz *et al.* 2015).

The above model categories mostly comprise non-linear equations, thereby increasing the technical complexity of model fitting. CR experimental designs are often also complex, and may require the addition of multi-level, hierarchical effects. Examples might include a random offset to model non-independence of replicates housed together in tanks, or where there are repeated measurements on individual genotypes of a test species. Additionally, CR data are of varied nature, with the measured response data taking a wide range of natural distributions. For instance, response data may be unbounded continuous (e.g., growth when shrinkage is possible), or positive continuous (e.g., growth in the absence of shrinkage), proportional (e.g., fertilization success), or binary (e.g., survival). To the best of our knowledge, there is no open-source statistical software dedicated to CR modeling which allows for appropriate model and error distribution specification depending on the input data. However, there is a wide array of multi-purpose packages for fitting non-linear generalized hierarchical models in R. For example, to list a few, **nlme** (Pinheiro, Bates, DebRoy, Sarkar, and R Core Team 2024), **lme4** (Pinheiro *et al.* 2024), **rjags** (Su and Yajima 2024) and **rstan** (Stan Development Team 2024).

While CR modeling can be carried out using generic non-linear modeling software packages, this can be cumbersome in practice, requiring extensive manual programming to obtain the necessary, often relatively standard outputs. The **drc** package (Ritz and Streibig 2005; Ritz *et al.* 2015) was developed as a user friendly frequentist solution to CR modeling in R, and is currently widely used across a range of disciplines. The **drc** package implements a broad range of non-linear CR models, provides facilities for ranking model fits based on AIC (Burnham and Anderson 2002), joint modeling of multiple response curves, and supports a range of estimation procedures (Ritz *et al.* 2015). Section 4 provides a formal comparison between **drc** and **bayesnec** standard output using default argument values.

Estimates of uncertainty in parameters and derived thresholds are critical for effective integration of threshold estimates into risk assessment and formal decision frameworks (Fisher *et al.* 2018). Bayesian methods allow robust quantification of uncertainty with intuitive and direct probabilistic meaning (Ellison 1996), and are therefore a useful platform for CR modeling in most settings. Furthermore, the posterior draws generated through Bayesian model fitting methods provide a rich resource that can be used to explore synergistic and antagonistic impacts (Fisher, Bessell-Browne, and Jones 2019a; Flores *et al.* 2021), propagate toxicity estimate uncertainty (Charles, Wu, and Ducrot 2020; Gottschalk and Nowack 2013), and test hypotheses of no-effect (Neal 2006).

There is a wide array of packages available for Bayesian model fitting via Markov chain Monte Carlo methods, including **WinBUGS** (Lunn, Thomas, Best, and Spiegelhalter 2000), **JAGS** (Plummer 2003) and Stan (Carpenter *et al.* 2017), that seamlessly interface with R through packages such as **R2WinBUGS**, **R2jags** (Su and Yajima 2024) and **rstan** (Stan Development Team 2024). These packages require coding in R and additional model specification in custom languages – which might involve non-trivial coding, extensive debugging and optimization that is often time consuming and requires specialist expertise, all of which might add a learning/application barrier to many potential users. Several extension packages which aim to reduce that barrier have recently become available, principally **brms** (Bürkner 2017), that allows a broad range of models to be easily fitted using **rstan** (Stan Development Team 2024) or **cmdstanr** (Gabry and Češnovar 2024) through simpler **lme4**-like formula syntax. However, even with packages like **brms**, Bayesian model fitting can be difficult to automate across all potential use cases, particularly with respect to specifying valid initial parameter values and appropriate priors. In addition, as was the motivation for the development of the **drc** package in the frequentist setting, the R code required for fitting non-linear models and extracting the relevant outputs (e.g., NEC, ECx) and their standard errors can be cumbersome in practice, and even more so in the Bayesian setting where model fits contain multiple posterior draws.

The greater complexity associated with Bayesian model fitting has likely hindered the uptake of Bayesian statistics for CR threshold derivation across the broader ecotoxicology and toxicology communities, who may not have access to specialist statistical expertise (Fisher *et al.* 2019b). Package **bayesnec** (Fisher, Barneche, Ricardo, and Fox 2024a) version 2.1.3.0 is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=bayesnec` and builds upon an implementation of the NEC model described in Fox (2010) and Pires, Branco, Picado, and Mendonca (2002). The **bayesnec** package provides an accessible interface specifically for fitting NEC and other CR models using Bayesian methods. A variety of models can be specified based on the known distribution of the "concentration" or "dose" variable (the predictor, x) as well as the "response" (y) variable. The model formula, including priors and initial values required to call **brms** are automatically generated based on information contained in the supplied data. A range of tools is supplied to aid the user in interrogating model fits, plotting and generating predicted values, as well as extracting the standard outputs, such as NEC and ECx, either as a full posterior draw or in summary form.

2. Technical details

In Bayesian inference, model parameters and their inherent uncertainty are estimated as statistical probability distributions. This is achieved by combining an a-priori probability distribution for each parameter (the "priors", $p(\theta)$) with the likelihood of the observed data, D, given the model parameters, $p(D|\theta)$, to yield a so-called posterior probability distribution, $p(\theta|D)$:

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{\int_\theta p(D|\theta)p(\theta)d\theta} \propto p(D|\theta)p(\theta).$$

In many cases, the denominator (also known as "evidence" or "marginal") is analytically intractable, and therefore we resort to numerical approximations such as Markov chain Monte Carlo (MCMC). The **bayesnec** package uses the extensive functionality of the **brms** package to write model code as well as fit models via a `Stan` program (Stan Development Team 2021), which typically defines a Bayesian posterior as a log density function conditioned on the data. `Stan` employs Hamiltonian Monte Carlo (HMC) – a type of MCMC algorithm – for its fitting mechanism, and automatically optimizes the discretization time parameter to match an acceptance-rate target using the no-U-turn sampling (NUTS) algorithm (Hoffman and Gelman 2014). The **bayesnec** package first takes the input model or model-set (via its custom formula class, '`bayesnecformula`') in conjunction with potential user-defined hierarchical effects, to generate an object of class '`brmsformula`'. Then, based on the package-(or user-)specified priors, and response model distribution, it calls **brms** (Bürkner 2017, 2018) to generate the `Stan` program code and fit the model through either **rstan** (Stan Development Team 2024) or **cmdstanr** (Gabry and Češnovar 2024). In doing so, **bayesnec** (as of version 2.1.3.0) allows for response variables to be modeled with a variety of statistical distributions: Gaussian, Poisson, binomial, gamma, negative binomial, beta and beta-binomial. Future implementations of **bayesnec** might include additional distributions which are currently only implemented in **brms**. In addition to greater flexibility in the available response distributions, **bayesnec** includes a larger set of 23 models (see below), including many of the commonly-used models in **drc** (Ritz *et al.* 2015).

2.1. Models in bayesnec

Where possible we have aimed for consistency in the interpretable meaning of the individual parameters across models. Across the currently implemented model sets, models contain from two (basic linear or exponential decay, see `ecxlin` or `ecxexp`) to five possible parameters (`nechorme4`), including:

τ = `top`, usually interpretable as either the response's intercept or the upper plateau representing the mean concentration of the response at zero concentration;

η = `NEC`, the No-Effect-Concentration value (the concentration value where the breakpoint in the regression is);

β = `beta`, generally the exponential decay rate of response, either from 0 concentration or from the estimated η value, with the exception of the `neclinhorme` model where it represents a linear decay from η because slope (α) is required for the linear increase (see below);

δ = `bot`, representing the lower asymptotic response at infinite concentration;

α = `slope`, the linear decay rate in the models `neclin` and `ecxlin`, or the linear increase rate prior to η for all hormesis models;

$\omega =$ `EC`$_{50}$, notionally the 50% effect concentration but may be influenced by scaling and should therefore not be strictly interpreted as such,

$\epsilon =$ `d`, the exponent in the `ecxsigm` and `necisgm` models, and

$\phi =$ `f`, A scaling exponent exclusive to model `ecxll5`.

In addition to the model parameters, all NEC models (prefix `"nec"`) have a step (indicator) function used to define the breakpoint in the regression, which can be defined as:

$$f(x_i, \eta) = \begin{cases} 0, & x_i - \eta < 0, \\ 1, & x_i - \eta \geq 0. \end{cases}$$

Note that strictly positive parameters (for example, β and ϵ) have been estimated on the natural log scale to allow the use of normal priors and stabilize model fitting.

ECx models (prefix `"ecx"`*)*

`ecxlin` is a basic linear decay model, given by the equation: $y_i = \tau - e^{\alpha} x_i$ with the following '`brmsformula`': `y ~ top - exp(slope) * x`. Because the model contains linear, predictors it is not suitable for 0–1 bounded data (i.e., binomial and beta families with an `"identity"` link function). As the model includes a linear decline with concentration, it is also not suitable for 0 bounded data (gamma, Poisson, negative binomial with an `"identity"` link).

`ecxexp` is a basic exponential decay model, given by the equation: $y_i = \tau e^{-e^{\beta} x_i}$ with the following '`brmsformula`': `y ~ top * exp(-exp(beta) * x)`. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function.

`ecxsigm` is a simple sigmoidal decay model, given by the equation: $y_i = \tau e^{-e^{\beta} x_i^{e^{\epsilon}}}$ with the following '`brmsformula`': `y ~ top * exp(-exp(beta) * x^exp(d))`. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function.

`ecx4param` is a 4-parameter sigmoidal decay model, given by the equation: $y_i = \tau + (\delta - \tau)/(1 + e^{e^{\beta}(\omega - x_i)})$ with the following '`brmsformula`': `y ~ top + (bot - top)/(1 + exp((ec50 - x) * exp(beta)))`.

`ecxwb1` is a 4-parameter sigmoidal decay model which is a slight reformulation of the Weibull1 model of Ritz *et al.* (2015), given by the equation: $y_i = \delta + (\tau - \delta) e^{-e^{e^{\beta}(x_i - \omega)}}$ with the following '`brmsformula`': `y ~ bot + (top - bot) * exp(-exp(exp(beta) * (x - ec50)))`.

`ecxwb1p3` is a 3-parameter sigmoidal decay model, equivalent to the `ecxwb1` model where δ (bot) is defined as 0, given by the equation: $y_i = 0 + (\tau - 0) e^{-e^{e^{\beta}(x_i - \omega)}}$ with the following '`brmsformula`': `y ~ 0 + (top - 0) * exp(-exp(exp(beta) * (x - ec50)))`. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function.

`ecxwb2` is a 4-parameter sigmoidal decay model which is a slight reformulation of the Weibull2 model of Ritz *et al.* (2015), given by the equation: $y_i = \delta + (\tau - \delta)(1 - e^{-e^{e^{\beta}(x_i - \omega)}})$ with the following '`brmsformula`': `y ~ bot + (top - bot) * (1 - exp(-exp(-exp(beta) * (x - ec50))))`. While very similar to the `ecxwb1` (according to Ritz *et al.* 2015), fitted `ecxwb1` and `ecxwb2` models can differ slightly.

`ecxwb2p3` is a 3-parameter sigmoidal decay model, equivalent to the `ecxwb2` model where the parameter δ (bot) is defined as 0, given by the equation: $y_i = 0 + (\tau - 0)(1 - e^{-e^{e^{\beta}(x_i - \omega)}})$ with

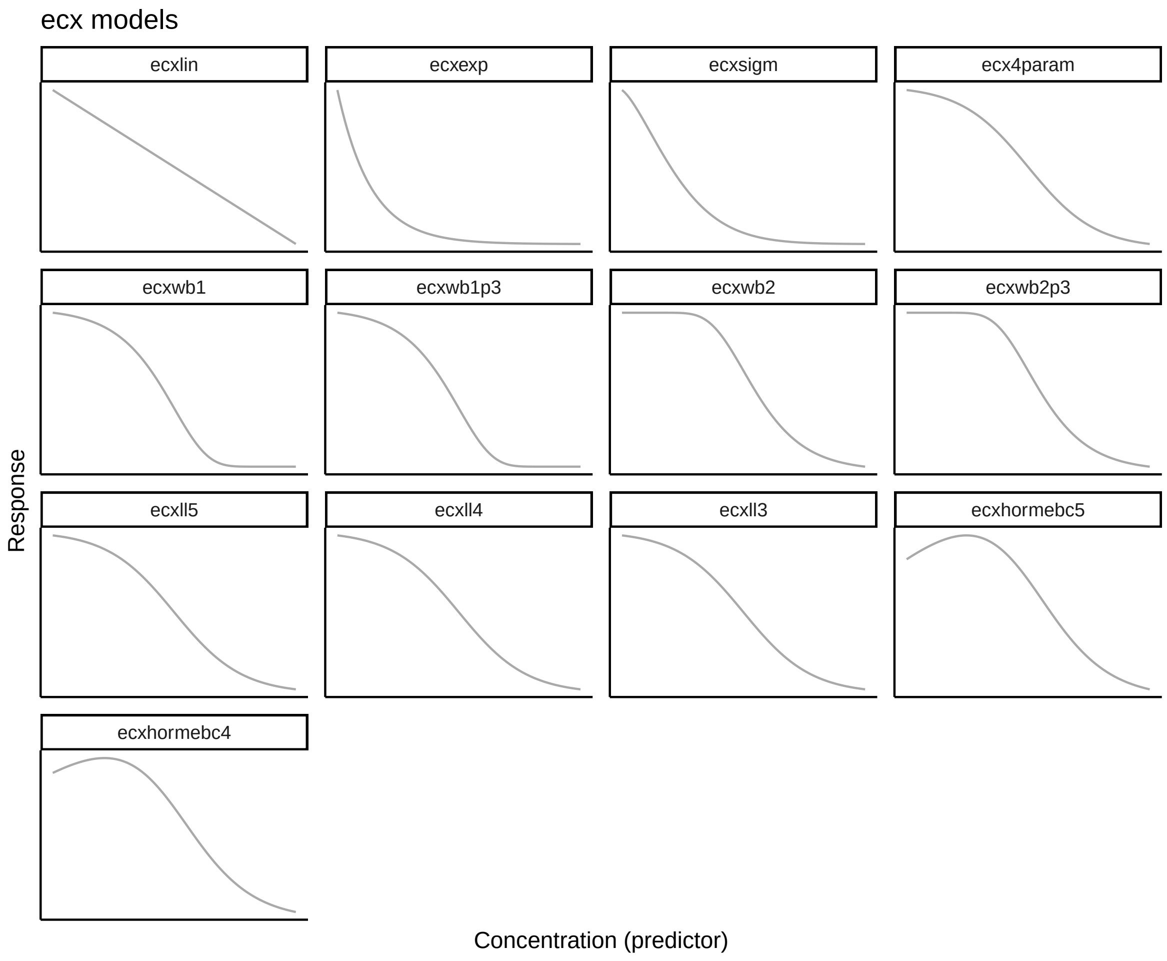

Figure 1: Representative shapes of currently implemented **bayesnec** `ecx` models.

the following 'brmsformula': `y ~ 0 + (top - 0) * (1 - exp(-exp(-exp(beta) * (x - ec50))))`. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a logit or log link function.

`ecxll5` is a 5-parameter sigmoidal log-logistic decay model, which is a slight reformulation of the LL.5 model of Ritz *et al.* (2015), given by the equation: $y_i = \delta + (\tau - \delta)/(1 + e^{-e^{\beta}(x_i - \omega)})^{e^{\phi}}$ with the following 'brmsformula': `y ~ bot + (top - bot)/(1 + exp(exp(beta) * (x - ec50)))^exp(f)`.

`ecxll4` is a 4-parameter sigmoidal log-logistic decay model, equivalent to the `ecxll5` model, but where the parameter ϕ (f) is defined as 0, given by the equation: $y_i = \delta + (\tau - \delta)/(1 + e^{e^{\beta}(x_i - \omega)})$ with the following 'brmsformula': `y ~ bot + (top - bot)/(1 + exp(exp(beta) * (x - ec50)))`.

`ecxll3` is a 3-parameter sigmoidal log-logistic decay model, equivalent to the `ecxll5` model, but where the parameters ϕ (f) and δ (bot) are both defined as 0, given by the equation: $y_i = 0 + (\tau - 0)/(1 + e^{e^{\beta}(x_i - \omega)})$ with the following 'brmsformula': `y ~ 0 + (top - 0)/(1 + exp(exp(beta) * (x - ec50)))`. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function.

`ecxhormebc5` is a 5 parameter log-logistic model modified to accommodate a non-linear hormesis at low concentrations. It has been modified from to the "Brain-Cousens" (BC.5) model of Ritz *et al.* (2015), given by the equation: $y_i = \delta + (\tau - \delta + e^{\alpha} x)/(1 + e^{e^{\beta}(x_i - \omega)})$ with the following 'brmsformula': `y ~ bot + (top - bot + exp(slope) * x) / (1 + exp(exp( beta) * (x - ec50)))`.

`ecxhormebc4` is a 5-parameter log-logistic model similar to the `exchormebc5` model but with a lower bound δ (bot) of 0, given by the equation: $y_i = 0 + (\tau - 0 + e^{\alpha} x)/(1 + e^{e^{\beta}(x_i - \omega)})$ with the following 'brmsformula': `y ~ 0 + (top - 0 + exp(slope) * x)/(1 + exp(exp(beta) * (x - ec50)))`. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function.

NEC models (prefix `"nec"`*)*

`neclin` is a basic linear decay model equivalent to `ecxlin` with the addition of the NEC step function, given by the equation: $y_i = \tau - e^{\alpha} (x_i - \eta) f(x_i, \eta)$ with the following 'brmsformula': `y ~ top - exp(slope) * (x - nec) * step(x - nec)`. Because the model contains linear predictors it is not suitable for 0–1 bounded data (binomial and beta distributions with `"identity"` link). As the model includes a linear decline with concentration, it is also not suitable for 0–∞ bounded data (gamma, Poisson, negative binomial with `"identity"` link).

`nec3param` is a basic exponential decay model equivalent to `ecxexp` with the addition of the NEC step function, given by the equation: $y_i = \tau e^{-e^{\beta}(x_i - \eta) f(x_i, \eta)}$ with the following 'brmsformula': `y ~ top * exp(-exp(beta) * (x - nec) * step(x - nec))`. For binomially distributed response data in the case of `"identity"` link this model is equivalent to that in Fox (2010). The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function.

`nec4param` is a equivalent to the `nec3param` model, but with an additional parameter defining the lower bound (parameter δ (bot)), given by the equation: $y_i = \delta + (\tau - \delta) e^{-e^{\beta}(x_i - \eta) f(x_i, \eta)}$ with the following 'brmsformula': `y ~ bot + (top - bot) * exp(-exp(beta) * (x - nec) * step(x - nec))`.

`nechorme` is a basic exponential decay model with an NEC step function equivalent to `nec3param`, with the addition of a linear increase prior to η, given by the equation $y_i = (\tau + e^{\alpha} x_i) e^{-e^{\beta}(x_i - \eta) f(x_i, \eta)}$ with the following 'brmsformula': `y ~ (top + exp(slope) * x) * exp(-exp(beta) * (x - nec) * step(x - nec))`. The `nechorme` model is a *hormesis* model (Mattson 2008), allowing an initial increase in the response variable at concentrations below η. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function. In this case the linear version (`neclinhorme`) should be used.

`nechormepwr` is a basic exponential decay model with an NEC step function equivalent to `nec3param`, with the addition of a power increase prior to η, given by the equation: $y_i = (\tau + x_i^{1/(1+e^{\alpha})}) e^{-e^{\beta}(x_i - \eta) f(x_i, \eta)}$ with the following 'brmsformula': `y ~ (top + x^(1 / (1 + exp(slope)))) * exp(-exp(beta) * (x - nec) * step(x - nec))`. Being a *hormesis* model (Mattson 2008), `nechormepwr` allows an initial increase in the response variable at concentrations below η. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function. Because the model can generate predictions > 1 it should not be used for binomial and beta distributions with `"identity"` link. In this case the `nechromepwr01` model should be used.

`neclinhorme` is a basic linear decay model with an NEC step function equivalent to `neclin`,

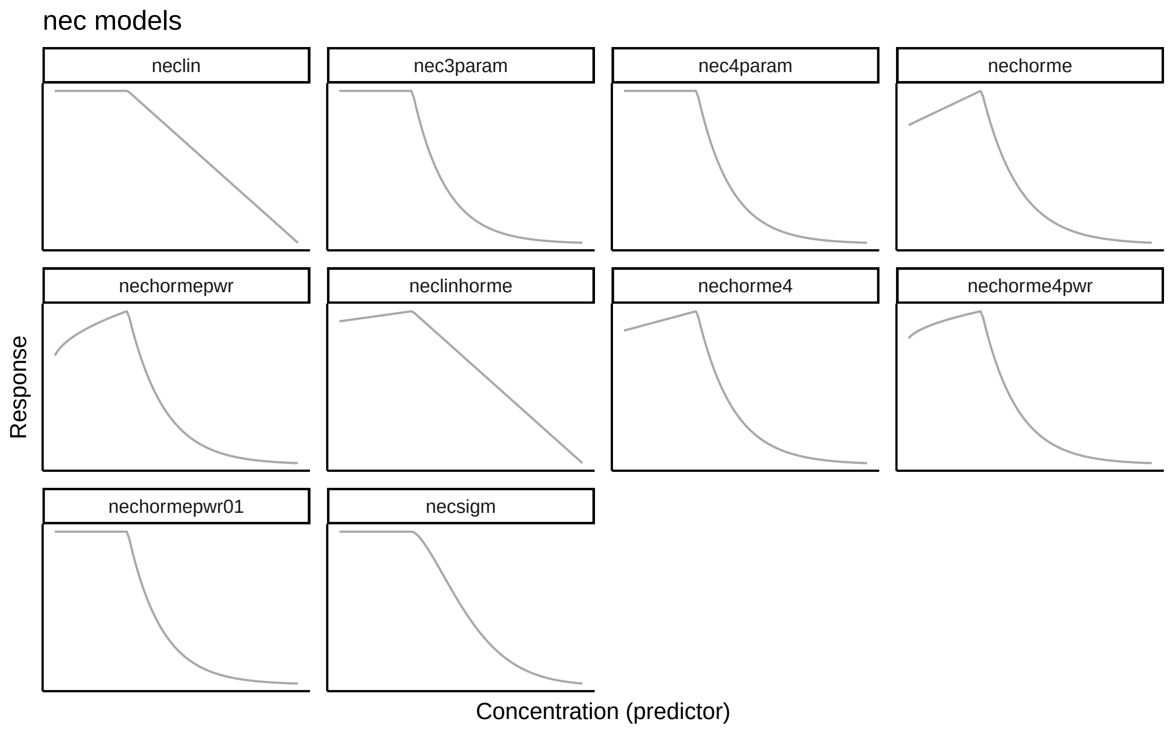

Figure 2: Representative shapes of currently implemented **bayesnec** `nec` models.

with the addition of a linear increase prior to η, given by the equation: $y_i = (\tau + e^{\alpha} x_i) - e^{\beta}(x_i - \eta) f(x_i, \eta)$ and '`brmsformula`': `y ~ (top + exp(slope) * x) - exp(beta) * (x - nec) * step(x - nec)`. The `neclinhorme` model is a *hormesis* model (Mattson 2008), allowing an initial increase in the response variable at concentrations below η. This model contains linear predictors and is not suitable for 0–1 bounded data (binomial and beta distributions with `"identity"` link). As the model includes a linear decline with concentration, it is also not suitable for 0–∞ bounded data (gamma, Poisson, negative binomial with `"identity"` link).

`nechorme4` is five parameter decay model with an NEC step function equivalent to `nec4param` with the addition of a linear increase prior to η, given by the equation: $y_i = \delta + ((\tau + e^{\alpha} x_i) - \delta)e^{-e^{\beta}(x_i - \eta) f(x_i, \eta)}$ with the following '`brmsformula`': `y ~ bot + ((top + exp(slope) * x) - bot) * exp(-exp(beta) * (x - nec) * step(x - nec))`. The `nechorme4` model is a *hormesis* model (Mattson 2008), allowing an initial increase in the response variable at concentrations below η.

`nechorme4pwr` is five parameter decay model similar to `nec4param`: It contains an NEC step function but with a power increase prior to η, given by the equation: $y_i = \delta + ((\tau + x_i^{1/(1+e^{\alpha})}) - \delta)e^{-e^{\beta}(x_i - \eta) f(x_i, \eta)}$ with the following '`brmsformula`': `y ~ bot + ((top + x^(1 / (1 + exp(slope)))) - bot) * exp(-exp(beta) * (x - nec) * step(x - nec))`. Being a *hormesis* model (Mattson 2008), `nechorme4pwr` allows an initial power increase in the response variable at concentrations below η. Because the model can generate predictions > 1 it should not be used for binomial and beta distributions with `"identity"` link. In this case the `nechromepwr01` model should be used.

`nechormepwr01` is a basic exponential decay model with an NEC step function equivalent to `nec3param`, with the addition of a power increase prior to η, given by the equation: $y_i = \left(\frac{1}{(1+((1/\tau)-1)e^{-e^{\alpha}x_i}}\right) e^{-e^{\beta}(x_i-\eta)f(x_i,\eta)}$ with the following '`brmsformula`': `y ~ (1 / (1 + ((1 / top) - 1) * exp(-exp(slope) * x))) * exp(-exp(beta) * (x - nec) * step(x - nec))`. Being a *hormesis* model (Mattson 2008), `nechormepwr01` allows an initial increase in the response variable at concentrations below η. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function. In this case the linear version (`neclinhorme`) should be used.

`necsigm` is a basic exponential decay model equivalent to `ecxlin` with the addition of the NEC step function, given by the equation: $y_i = \tau e^{-e^{\beta}((x_i-\eta)f(x_i,\eta))^{e^{\epsilon}} f(x_i,\eta)}$ with the following '`brmsformula`': `y ~ top * exp(-exp(beta) * (step(x - nec) * (x - nec))^exp(d) * step(x - nec))`. The model is 0-bounded, thus not suitable for Gaussian response data or the use of a `"logit"` or `"log"` link function. Estimation of No-Effect-Concentrations using this model are not currently recommended without further testing, as the behavior is currently unstable, see supplementary material in Fisher *et al.* (2024b).

2.2. Priors on model parameters

To undertake a Bayesian analysis, prior probability densities of the parameters of the model must first be defined. Sometimes there may be substantial prior knowledge, for example when pilot data or data from a previous experiment exist for a given response curve. In this case the prior probability distribution may be quite narrow (highly "informative") and will therefore be influential in the characterization of the posterior, especially when subsequent data are scarce or highly variable. However, in our experience in ecology and related disciplines, such prior knowledge is generally the exception. Where no quantitative prior information exists, it is common in Bayesian statistics to use either "vague" or "weakly" informative priors. The use of "vague", "diffuse", "flat" or otherwise so-called "uninformative" priors is no longer recommended (Banner, Irvine, and Rodhouse 2020). Such priors generally form the default for many Bayesian packages, and are often used in practice without critical thought or evaluation, possibly as a result of fear of being too subjective (Banner *et al.* 2020). However, even vague priors can have a substantial influence on the outcome of an analysis (Depaoli, Winter, and Visser 2020; Gelman, Simpson, and Betancourt 2017). Instead, it is better to use weakly informative, "generative" priors – that is priors that are based on probability distributions that interact sensibly with the likelihood to produce a meaningful data generating mechanism (Gelman *et al.* 2017).

Considerable thought has gone into development of an algorithm to build default "weakly" informative priors for fitting models in **bayesnec**. The default priors are "weakly" informative in that in addition to specifying the relevant statistical `family` that appropriately captures the parameter's theoretical statistical distribution, when external subjective information is unavailable (no priors are supplied by the user) we also use information contained within the observed data to build priors with appropriate scaling. The procedure is described in more detail below, but the algorithm effectively centers the probability density of the prior within a plausible region of the parameter space, ensures priors are appropriately scaled relative to the range of the response and predictor data, and/or priors are constrained to sensible bounds. These weakly informative priors are used to help constrain the underlying routines so that they are less likely to consider what the researcher would deem highly improbable estimates, that may also cause the routines to become unstable resulting in failed model fits. Weakly informative priors can be particularly helpful in complex non-linear modeling to ensure reliable convergence. These types of priors specify the general shape and bounds of the probability distribution for a given parameter, whilst remaining sufficiently broad so as not to influence the parameter's estimated posterior distribution (given a reasonable amount of observed data). In this sense, appropriately weak priors should yield analytical outcomes that share the same level of *objectivity* as equivalent frequentist approaches, whilst yielding robust parameter estimates with probabilistically interpretable uncertainty bounds.

While we sacrifice Bayesian coherence by using features of the data to calibrate our default priors (see Chipman, George, and McCulloch (2010) for another example of such an approach), our primary motivation is to facilitate easy implementation of **bayesnec** in practice, and to ensure model stability and reliable model fits. Note, however that it is critical for users to interrogate these default priors, using for example, sensitivity analysis (Depaoli *et al.* 2020) and ensure they are appropriate given the data (Gelman *et al.* 2017). Priors are automatically saved as part of the '`brmsfit`' and '`bayesnecfit`' models, and there are functions for extracting the priors used, as well as plotting these in comparison to the resulting posterior distribution (see Section 3.5). Care should be taken to ensure that the default priors are sufficiently weak such that they have little influence on posterior estimates.

Priors for response-scaled parameters

Only the parameters $\tau =$ `top` and $\delta =$ `bot` relate directly to the response variable's distribution. For Gaussian-distributed responses (or any response variable for which the link ensures valid values of the response can take from $-\infty$ to ∞, including `"log"` and `"logit"`) priors are Gaussian with a mean set at the 90th and 10th percentiles of the response for parameters $\tau =$ `top` and $\delta =$ `bot`, respectively, and a standard deviation of 2.5 times the standard deviation of the response (on the appropriate link scale). In this way **bayesnec** attempts to construct a prior that scales appropriately with the response data, with greatest density near the most likely region of the response for both $\tau =$ `top` and $\delta =$ `bot`. The priors for top and bot can be set quite narrow and still remain relatively "weak" because **bayesnec** only allows models that represent an overall decrease from the start to the end of the concentration range. Because this directional relationship is pre-defined, it is reasonable to presume that the true value of $\tau =$ `top`, for example, should be relatively near the upper quantile of the observed response data, and a somewhat narrow prior on that assumption can be used without being strongly informative. In the context of a standard deviation across the whole response range, a value of 2.5 can still be considered relatively broad and should have little influence on the parameter's posterior density.

For Poisson-, negative-binomial- and gamma-distributed response variables, the response cannot take negative values and therefore Gaussian priors are unsuitable. Instead, we use gamma priors with a mean scaled to correspond to the 75th and 25th percentiles of the response for $\tau =$ `top` and $\delta =$ `bot`, respectively. The mean (μ) is linked mathematically to the shape (s) and rate parameters (r) by the equation (Becker, Chambers, and Wilks 1988)

$$\mu = s \cdot (1/r)$$

with the shape parameter being set to 2 by default. The value of 2 was selected based on trial and error through initial testing, as this appeared to produce relatively broad priors that were still centered around feasible values for these parameters.

For the binomial, beta, and beta-binomial families, estimates for $\tau =$ `top` and $\delta =$ `bot` must necessarily be constrained between 0 and 1 when modeled on the identity link. Because of this constraint, there is no need to adjust scaling based on the response. In this case **bayesnec** uses `beta(5, 2)` and `beta(2, 5)` priors to provide a broad density centered across the upper and lower 0 to 1 range for the $\tau =$ `top` and $\delta =$ `bot` parameters respectively.

Priors for predictor-scaled parameters

The parameters $\eta =$ `NEC` and $\omega =$ $\texttt{EC}_{50}$ scale according to the predictor variable because both of these are estimated in units of the predictor (usually concentration). To stabilize model fitting, the $\eta =$ `NEC` and $\omega =$ $\texttt{EC}_{50}$ parameters are bounded to the upper and lower observed range in the predictor, under the assumption that the range of concentrations in the experiment were sufficient to cover the full range of the response outcomes. Note that this assumption may not always hold if the data are from an experiment that is poorly designed, and the outcome of any analysis resulting in either η or ω being estimated at the bounds of the predictor data range should be interpreted with caution. The priors used reflect the characteristics of the observed data that are used to predict the appropriate `family`. If the predictor variable is strictly positive, a gamma prior is used, with maximum density (μ, see above) at the median value of the predictor, and a shape parameter of 5. If the predictor variable is truncated at both 0 and 1, a `beta(2, 2)` prior is used. For predictor variables ranging from $-\infty$ to ∞, a Gaussian prior is used, with a mean set at the median of the predictor values and a standard deviation of 10 times the standard deviation of the predictor variable. A much broader prior is required for the $\eta =$ `NEC` and $\omega =$ $\texttt{EC}_{50}$ estimates than for example $\tau =$ `top` and $\delta =$ `bot`, because depending on the curve that is fit estimates for these parameters may fall almost anywhere along the predictor range - especially if the CR experiment was badly designed. We set the maximum density for these parameters as the median of the predictor value (in the hope that the experiment has been well designed and the inflection point is somewhere in the center of the predictor range), but it is important that there is substantial range in the prior, because the true values may be quite low or high across the predictor range, thus a very broad standard deviation of 10 is used.

Priors for other parameters

For the parameters $\beta =$ `beta`, $\alpha =$ `slope` and $\epsilon =$ `d` we first ensured any relevant transformations in the model formula such that theoretical values with the range $-\infty$ to ∞ are allowable, and a `normal(0, 5)` (mean and standard deviation) prior is used. For example in the `nec3param` model, $\beta =$ `beta` is an exponential decay parameter, which must by definition be bounded to 0 and ∞. Calling `exp(beta)` in the model formula ensures the exponent meets these requirements. Note also that a mean of 0 and standard deviation of 5 represents a relatively broad prior on this exponential scaling, so this is usually a weakly informative prior in practice.

User-specified priors

In **bayesnec** we chose to provide default weakly informative priors that are scaled according to the characteristics of the input data (discussed in detail above) in some cases. They were designed to be somewhat informative (relative to each parameter's region) but that would, in data-sufficient cases, return fits without HMC divergent transitions in `Stan`. Default "blanket" priors are not currently provided for non-linear models by the model-building underlying package **brms**, and we note that defining the extent to which a prior is vague or weakly/strongly informative ultimately depends on the likelihood (Gelman *et al.* 2017). Therefore, there may be situations where the default **bayesnec** priors do not produce an appropriate fit, or the user wants to provide customized priors. For example, the default priors may be too informative, yielding unreasonably tight confidence bands (although this is only likely where there are few data or unique values of the predictor variable). Conversely, priors may be too vague, leading to poor model convergence. Alternatively, the default priors may be of the wrong statistical `family` if there was insufficient information in the provided data for **bayesnec** to correctly predict the appropriate ones to use. The priors used in the default model fit can be extracted using `pull_prior`, and a sample or plot of prior values can be obtained from the individual **brms** model fits through the function `sample_priors()` which samples directly from the `"prior"` element in the '`brmsfit`' object (`pull_brmsfit(fit) |> prior_summary() |> sample_priors()`, see Figure 5). We show example usage of these functions under the Section 3.5 below.

2.3. Model averaging

Multi-model inference can be useful where there is a range of plausible models that could be used (Burnham and Anderson 2002) and has been recently adopted in ecotoxicology for Species Sensitivity Distribution (SSD) model inference (Thorley and Schwarz 2023; Fox *et al.* 2021; Dalgarno 2021). The approach may have considerable value in CR modeling because there is often no a priori knowledge of the functional form that the response relationship should take. In this case, model averaging can be a useful way of allowing the data to drive the model selection process, with weights proportional to how well the individual models fit the data. Well-fitting models will have high weights, dominating the model averaged outcome. Conversely, poorly fitting models will have very low model weights and will therefore have little influence on the outcome. Where multiple models fit the data equally well, these can equally influence the outcome, and the resultant posterior predictions reflect that model uncertainty.

The **bayesnec** package adopts the weighting methods implemented via the **loo** (Vehtari *et al.* 2024) package in R. **loo** provides an efficient approximate leave-one-out cross-validation (LOO) algorithm for Bayesian models fit using Markov chain Monte Carlo, as described in Vehtari, Gelman, and Gabry (2017). The approximation uses Pareto smoothed importance sampling (PSIS), a new procedure for regularizing importance weights and follows the implementation described in Vehtari, Simpson, Gelman, Yao, and Gabry (2019). The **loo** package offers two weighting methods, the `"stacking"` method, aimed to minimize prediction error (Yao, Vehtari, Simpson, and Gelman 2018), and the `"pseudobma"` method, with and without Bayesian bootstrap (Vehtari *et al.* 2024, 2017). The stacking method (`method = "stacking"`), combines all models by maximizing the leave-one-out predictive density of the combination distribution, such that it finds the optimal linear combining weights for maximizing the leave-one-out log score (Vehtari *et al.* 2024). The pseudo-BMA method (`method = "pseudobma"`) finds the relative weights proportional to the theoretical expected log pointwise predictive density of each model (Vehtari *et al.* 2024). The Bayesian bootstrap (when using `method = "pseudobma"`) takes into account the uncertainty of finite data points and regularizes the weights away from the extremes of 0 and 1 (Vehtari *et al.* 2024). **bayesnec** currently uses by default the `"pseudobma"` method (`method = "pseudobma"`) with Bayesian bootstrap (`BB = TRUE`), but this can be easily modified via argument `loo_controls`.

3. Usage

3.1. The fitting function `bnec()`

The main working function in **bayesnec** is `bnec()`. We have attempted to make the `bnec()` function as easy to use as possible, targeting the R user that is familiar with the usual model fitting syntax in R, but without specialist expertise in non-linear modeling and Bayesian statistics. We can run `bnec()` by supplying the argument `formula`: A special custom formula which comprises the relationship between response and predictor, and the CR model (or models) chosen to be fitted; and `data`: A '`data.frame`' containing the data for the model fitting.

3.2. The input formula

In its simplest syntax, the basic `bnec()` formula should be of the form:

```
R> response ~ crf(predictor, model = "a_model")
```

where the left-hand side of the formula is implemented exactly as in **brms** (see the "aterms" Section of the `brms::brm()`'s help file). The right-hand side of the formula contains the special internal function `crf()` (which stands for concentration-response function), and takes the predictor (including any simple function transformations such as `"log"`) and the desired CR model or suite of models. As with any formula in R, the name of the terms need to be exactly as they are in the input '`data.frame`'. For binomial or beta-binomial distributed data, the user needs to include the `trials()` term to the left-hand side of the formula, e.g.,

```
R> response | trials(n_trials) ~ crf(log(predictor), model = "a_model")
```

The input formula can either be a character string or an object of class '`bayesnecformula`'. Details about existing possibilities are detailed in the help files of '`bayesnecformula`' and `check_formula`. The argument `model` in the formula function `crf()` is a character string indicating the name(s) of the desired model(s). Alternatively, it may also be one of `"all"`, meaning all of the available models will be fit; `"ecx"` meaning only models excluding the $\eta =$ `NEC` step parameter will be fit; `"nec"` meaning only models with a specific $\eta =$ `NEC` step parameter will be fit; `"bot_free"` meaning only models without a δ (`bot, "bot"`) parameter (without a lower plateau) will be fit; `"zero_bounded"` are models that are bounded to be zero; or `"decline"` excludes all hormesis models, i.e., only allows a strict decline in response across the whole predictor range (see above Section 2.1).

The class '`bayesnecformula`' (generated by the function `bayesnecformula()` and its alias `bnf()`) contains a `model.frame()` method which can be employed to manually inspect the '`data.frame`' that will be used to run checks on the data suitability prior to model fitting, e.g.,

```
R> library("bayesnec")
R> set.seed(17)
R> df <- data.frame(x = rgamma(100, 2, 0.1), y = rnorm(100))
R> form <- bnf(y ~ crf(log(x), model = "nec3param"))
R> head(model.frame(form, df))
```

```
            y   log(x)
1  1.93723559 1.637897
2 -0.50355786 3.249447
3  0.09236529 2.786040
4  1.06937160 1.806777
5 -0.48396058 2.568040
6 -0.41780030 3.366183
```

3.3. Example

Here we use one of the package's built-in data sets, `nec_data`, which is a simulated data set based on the three parameter NEC model `nec3param` described in Section 2.1.

```
R> fit <- bnec(y ~ crf(x, model = "nec3param"), data = nec_data, seed = 17)
```

If a recognized model name is provided, a single model of the specified type is fit, and `bnec()` returns an object of class '`bayesnecfit`'. If a vector of two or more of the available models are supplied, or if one of the model-sets is specified, `bnec()` returns a model object of class '`bayesmanecfit`' containing Bayesian model averaged predictions for the supplied models, providing they were successfully fitted (see Section 2.3 above, and the help file of `bnec()` for further details). By default, `bnec()` sets the number of chains to 4, the number of iterations per chain to 10,000, and the size of the warm-up period to 4/5 of the number of iterations (i.e., 8,000 by default).

`bnec()` will guess the most likely distribution for the response variable. This "guess" is achieved through the internal function `set_distribution()`. This algorithm will assume a binomial distribution for the response if data are integers and `trials()` is passed in the formula call; Poisson if data are integers and there are no `trials()`; gamma if the data are continuous, zero bounded and contain values greater than one; beta if data are continuous and bounded to zero and one; and Gaussian if data are continuous and contain negative values. The `family` can be set manually via the usual R syntax of calling the argument `family` and specifying the desired distribution. **bayesnec** currently supports the use of the above listed families, as well as the negative binomial and beta-binomial families that can be used in the case of over-dispersed binomial and Poisson families respectively.

In the example here, the model was fitted assuming a beta distribution on an identity link because the response is truncated at both 0 and 1 and contains decimal values. Note that the default behavior in **bayesnec** is to use the `"identity"` link because the native link functions for each `family` (e.g., `"logit"` for binomial, `"log"` for Poisson) introduce non-linear transformation on formulas which are already non-linear. Note that estimates of ECx might not be as expected when using link functions other than identity. Additionally, `bnec()` will also generate appropriate priors for the internal `brms::brm()` model call. However, `bnec()` allows the user to pass additional arguments to `brms::brm()` and therefore the user can, for example, manually add specific distributions and link functions via the argument `family`, or custom priors via the argument `prior`. Refer to the rich set of resources available for the **brms** package at `https://github.com/paul-buerkner/brms` for further information.

3.4. Output classes and methods

Models fitted by `bnec()` will invariably inherit a class ‘`bnecfit`’ which carries three exclusive methods: `` `+`() ``, `c()` and `update()`. The first two are used to append one or multiple models to an existing fit, whereas the latter is used to update the fitting characteristics of an existing model (e.g., change the number of iterations or warm-up period, or simply change the HMC fitting parameters).

When `bnec()` fits a single CR model type, the output object also inherits the ‘`bayesnecfit`’ class. This class contains the ‘`brmsfit`’ object in addition to the mean predicted values and summary estimates of each model parameter. Because the original motivation in the development of **bayesnec** was the estimation of no-effect toxicity values, by default `bnec()` also returns a full posterior distribution of the either the NEC (for `nec` models) or the NSEC (for `ecx` models, see Fisher and Fox 2023) estimate. If `bnec()` fits a custom set of models, or a package-predefined model-set (e.g., `model = "decline"` in the input formula), the output object inherits the ‘`bayesmanecfit`’ class. Differently from a ‘`bayesnecfit`’ object, ‘`bayesmanecfit`’ comprises a model weighted estimate of predicted posterior values of N(S)EC. This is a weighted combined average of the NEC or the NSEC values, for all `nec` and `ecx` models respectively, as described in Fisher *et al.* (2024b).

Regardless of whether `bnec()` generates a ‘`bayesnecfit`’ or ‘`bayesmanecfit`’ class, the underlying ‘`brmsfit`’ object can be extracted using the function `pull_brmsfit()`. The ‘`brmsfit`’ contains all of the information usually returned from a call to `brm()`, including the posterior samples of all parameters in the model, from which predictions can be made and custom probabilities calculated.

Both ‘`bayesnecfit`’ and ‘`bayesmanecfit`’ classes contain methods for `summary()`, `print()`, `predict()`, `model.frame()`, `fitted()`, `posterior_predict()`, `posterior_edpred()` and plotting (`plot()` and `autoplot()`). Wherever possible, these methods have been implemented such they are consistent with other model fitting packages in R, and in particular **brms**. We have also implemented a range of custom functions for extracting effect concentrations and related threshold values (`nec()`, `ecx()` and `nsec()`) that, in the case of a ‘`bayesmanecfit`’, return model weighted estimates.

The `summary()` method provides the usual summary of model parameters and any relevant model fit statistics as returned in the underlying `brm()` model fit(s). In the specific case of a ‘`bayesmanecfit`’ object, the summary includes a list of fitted models, their respective model weights, and a model-averaged no-effect toxicity estimate. Where the fitted model(s) are `nec` models (threshold models, containing a step function) the no-effect estimate is a true no-effect-concentration (NEC, see Fox 2010). Where the fitted model(s) are smooth `ecx` models with no step function, the no-effect estimate is a no-significant-effect-concentration (NSEC, see Fisher and Fox 2023). In the case of a ‘`bayesmanecfit`’ that contains a mixture of both `nec` and `ecx` models, the no-effect estimate is a model averaged combination of the NEC and NSEC estimates, and is reported as the N(S)EC (see Fisher *et al.* 2024b).

```
R> summary(fit)
```

```
Object of class bayesnecfit containing the nec3param model

 Family: beta
  Links: mu = identity; phi = identity
```

```
Formula: y ~ top * exp(-exp(beta) * (x - nec) * step(x - nec))
         top ~ 1
         beta ~ 1
         nec ~ 1
   Data: data (Number of observations: 100)
  Draws: 4 chains, each with iter = 10000; warmup = 8000; thin = 1;
         total post-warmup draws = 8000

Population-Level Effects:
     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
top      0.89      0.01     0.88     0.90 1.00     6929     5865
beta     0.54      0.06     0.43     0.65 1.00     5867     5229
nec      1.54      0.02     1.50     1.57 1.00     5770     4875

Family Specific Parameters:
    Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
phi    51.99      7.32    38.83    67.80 1.00     6461     5368

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).

    Estimate Q2.5 Q97.5
NEC     1.54 1.50  1.57

Bayesian R2 estimates:
   Estimate Est.Error Q2.5 Q97.5
R2     0.96      0.00 0.96  0.97
```

As mentioned above, the visualization of a particular model fit can be done via either **base** R (`plot()`) and **ggplot2** (Wickham 2016) (`autoplot()`).

```
R> round_digits <- function(x) sprintf("%.2f", x)
R> autoplot(fit, xform = exp) +
+   scale_x_continuous(trans = "log", labels = round_digits)
```

By default the plot shows the fitted posterior curve with 95% credible intervals, along with an estimate of the $\eta =$ `NEC` value. For more examples using **bayesnec** models for inference see the on-line the vignettes at `https://open-aims.github.io/bayesnec/articles/`, as well as Section 6.1.

3.5. Model diagnostics

The **bayesnec** package will return warning messages as part of the summary method where parts of the model have not converged (rhat, $\widehat{R} > 1.05$; see Vehtari, Gelman, Simpson, Carpenter, and Burkner 2021) and indicate the number of any divergent transitions (if any). These messages include guidance on running more iterations, adjusting priors and or adjusting

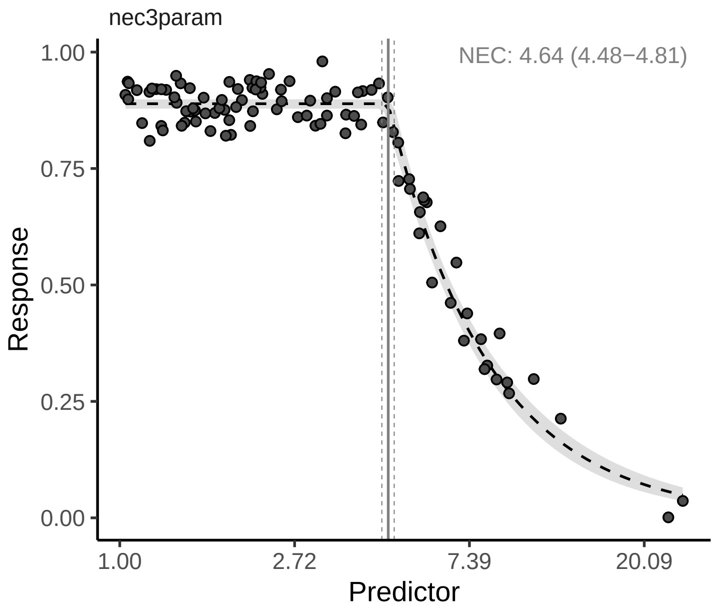

Figure 3: **ggplot2** `autoplot()` of the example fit. The solid black line is the fitted median of the posterior prediction, dashed black lines are the 95% credible intervals, and the vertical lines show the estimated NEC value.

other fitting criteria, such as `adapt_delta`. The summary method for a '`bayesnecfit`' object also indicates the effective sample size for estimates of each of the parameters, and for a '`bayesmanecfit`' a warning is returned if any of the models have parameters with an effective sample size of < 100.

In addition to the diagnostic information reported by the summary method, a range of tools is available to assess model fit, including an estimate of overdispersion (for relevant families), an extension of the **brms** `rhat()` function that can be applied to both '`bayesnecfit`' and '`bayesmanecfit`' model objects, and a function `check_chains()` that can be used to visually assess chain mixing and stability.

All diagnostic functions available in **brms** and **rstan** can be used on the underlying `brm` model fit by extracting the fitted **brms** model from the '`bayesnecfit`' or '`bayesmanecfit`' model object using the function `pull_brmsfit()`. For example, we can use the default **brms** plotting method to obtain a diagnostic plot of the individual fit of the `nec4param` model using:

```
R> brms_fit <- pull_brmsfit(fit)
R> plot(brms_fit)
```

which yields a plot of the posterior densities as well as trace plots of chains for each parameter in the specified model (Figure 4).

Several helper functions have been included that allow the user to add or drop models from a '`bayesmanecfit`' object, or change the model weighting method (`amend()`); extract a single or subset of models from the '`bayesmanecfit`' object (`pull_out()`); and examine the priors used for model fitting (`pull_prior()`, `sample_priors()` and `check_priors()`).

The priors used in the default model fit can be extracted using `pull_prior()`, and a sample or plot of prior values can be obtained from the individual **brms** model fits through the function

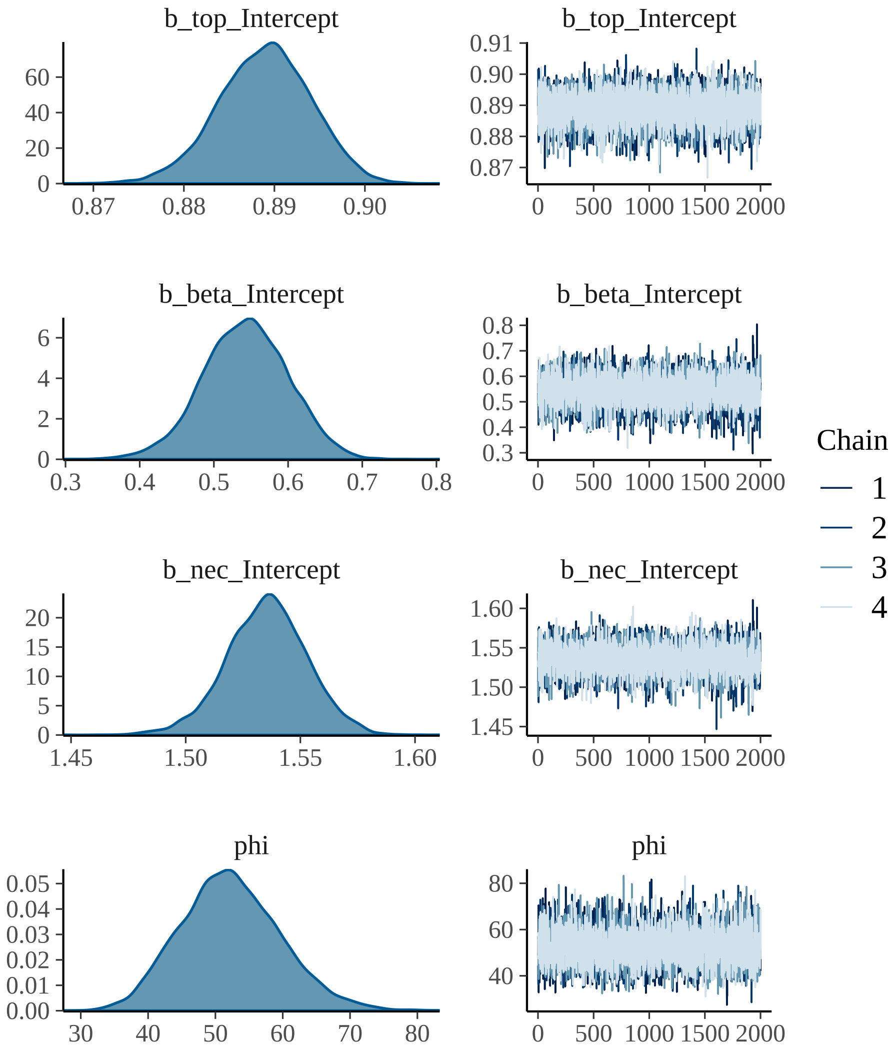

Figure 4: Default **brms** plot of the `nec3param` model showing the posterior densities and chain mixing for each of the included parameters.

`sample_priors()` which samples directly from the `"prior"` element in the '`brmsfit`' object (`pull_brmsfit(fit) |> prior_summary() |> sample_priors()`, see Figure 5).

We can also use the function `check_priors()` (based on the `hypothesis()` function of **brms**) to assess how the posterior probability density for each parameter differs from that of the prior. Here we show the prior and posterior probability densities for the parameters in the `nec3param` model fit (`check_priors(fit)`, see Figure 6). There is also a '`bayesmanecfit`'-specific method that can be used to sequentially view all plots in a `bnec()` call with multiple models, or write to a portable document format (PDF) file as in `check_chains()`.

3.6. Model comparison

With **bayesnec** we have included a function (`compare_posterior()`) that allows bootstrapped comparisons of posterior predictions. This function allows the user to fit several different `bnec()` model fits and compare differences in their posterior predictions. Comparisons can

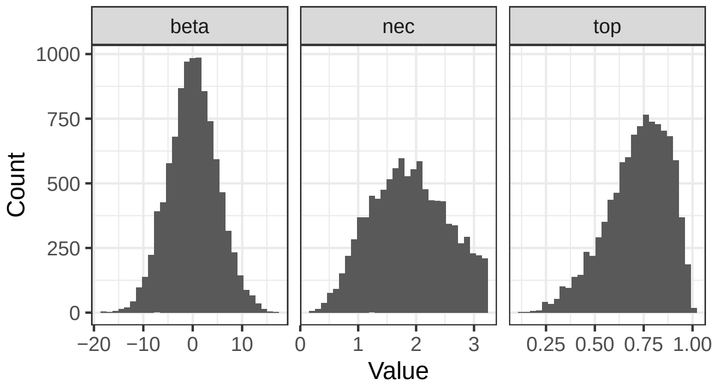

Figure 5: Frequency histograms of samples of the default priors used by **bayesnec** for fitting the `nec3param` model to the example data.

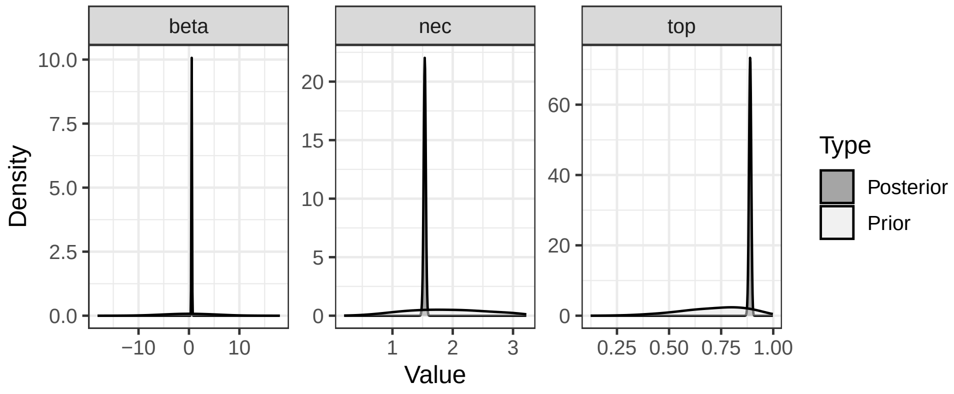

Figure 6: A comparison of the prior and posterior parameter probability densities for the `nec3param` model fit to the example data.

be made across the model fits for individual threshold estimates (e.g., NEC, N(S)EC, NSEC or ECx) or across a range of predictor values. Usage is demonstrated in the relevant vignette at `https://open-aims.github.io/bayesnec/articles/example4.html` by comparing different types of models and model-sets using a single data set. However, the intent of this function is to allow comparison across different data sets that might represent, for example, different levels of a fixed factor covariate. For example, this function has been used to compare toxicity of herbicides across three different climate scenarios, to examine the cumulative impacts of pesticides and global warming on corals (Flores *et al.* 2021).

At this time `bnec()` does not allow for an inclusion of an interaction with a fixed factor. Including an interaction term within each of the non-linear models implemented in **bayesnec** is relatively straightforward, and may be introduced in future releases. However, in many cases the functional form of the response may change with different levels of a given factor. The substantial complexity of defining all possible non-linear model combinations at each factor level means it unlikely this could be feasibly implemented in **bayesnec** in the short term. In the meantime the greatest flexibility in the functional form of individual model fits can be readily obtained using models fitted independently to data within each factor level.

3.7. Hierarchical effects

Most ecotoxicological and toxicology experiments include a range of grouping elements, such as tanks, vials or batches of samples that contain multiple measurements that cannot be considered strictly independent (also known as they are pseudo-replicates). To avoid criticism around potential issues with pseudo-replication, it is often the practice for ecotoxicologists to pool such observations and carry out modeling using, for example, the group mean. Where the number of within group observations varies substantially across groups, this will have the undesirable effect of equally weighting the group means even though some may be based on far fewer observations than others. In addition, there are often instances of ecotoxicology data from multiple experiments or other grouping factors within an experiment (such as genotype) that cover the full range of the predictor variable that cannot be averaged prior to modeling, resulting in the ecotoxicologist either ignoring the potential non-independence, or fitting many independent data sets and subsequently needing to aggregate the threshold estimates. Carrying out multiple fits on separate data sets is undesirable because each fit is based on fewer data and will have greater uncertainty.

The current version of **bayesnec** harnesses the powerful modeling flexibility of **brms** for accommodating hierarchical designs and other forms of non-independence. This is achieved by allowing a list of grouping terms to be added to a '`bayesnecformula`', which are then used to generate the underlying '`brmsformula`' for the **brms** internal call. Hierarchical effects can be in the form of an offset, which effectively allows different mean response levels across groups, and is achieved by specifying the `ogl()` (offset group-level) formula term. Hierarchical effects can also be added to any or all of the (non-)linear parameters in the model by specifying the `pgl()` (parameter group-level) formula term. Note that implementing hierarchical effects in a non-linear modeling setting is non-trivial and considerable thought and testing should go into selecting an appropriate hierarchical structure, and potentially suitable priors. Examples of how to implement hierarchical effects in **bayesnec** can be found on the help file of function `bnf()`.

4. Existing alternatives

Package **bayesnec** is built upon the precursor R package **jagsNEC** (Fisher, Ricardo, and Fox 2020), which writes and fits CR models in **JAGS** (Plummer 2003). The **bayesnec** package was then expanded to include several additional CR models and further generalized to allow a large range of response variables to be modeled using their appropriate statistical distribution. In addition, **bayesnec** allows the addition of hierarchical effects (see above). The simpler syntax of **brms** allows **bayesnec** to be more easily expanded to include additional response distributions as well as CR model formula. In addition, **brms** is well developed and comes with a large range of supporting functions not available to the **JAGS** equivalents.

While there are some commercial propriety software packages to support the analysis of toxicity data, such as GraphPad **Prism** (GraphPad Software 2024) and **ToxCalc** (Tidepool Scientific, LLC 2022), these provide limited flexibility and most importantly do not support fully reproducible analysis in an open-source environment. Ensuring that the raw data from the experiment are available, and that the statistical code and documentation to reproduce the analysis are also available are two major components to a reproducible study (Peng 2015).

The open-source flexible computing environment R provides an ideal platform for reproducible analysis of toxicity data sets. The main existing tool in R that is widely used in ecotoxicology

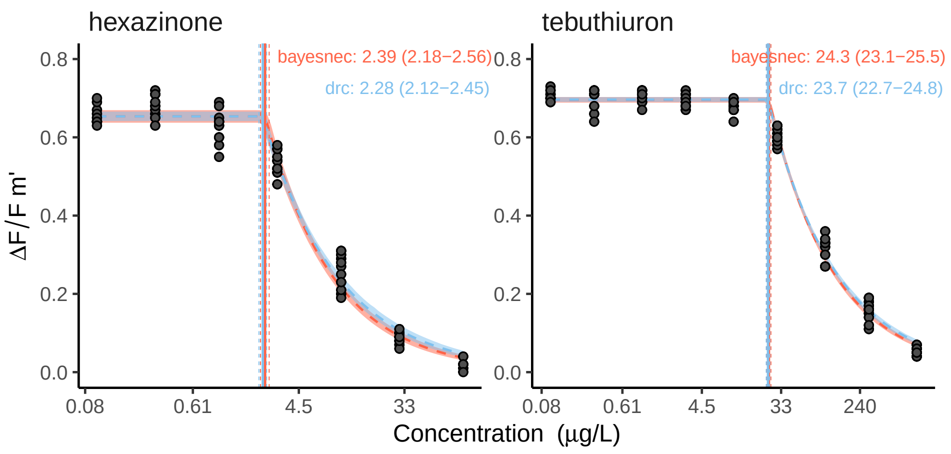

Figure 7: A comparison of the **bayesnec** and **drc** model fits and estimated NEC values for the `nec3param` model, fit to data on maximum effective quantum yield ($\Delta F/Fm'$) of symbiotic dinoflagellates (in hospite) in *Seriatopora hystrix* exposed to elevated hexazinone and tebuthiuron (range 0.3 to 1000 µg/L) for 10 h.

and toxicology is the frequentist-based package **drc** (Ritz *et al.* 2015). **drc** provides a suite of flexible and versatile model fitting and after-fitting functions for the analysis of dose-response data. The package includes a large range of built-in dose-reponse models that are parameterized using a unified structure with a coefficient `b` denoting the steepness of the dose-response curve (β = `beta` in **bayesnec**); `c` and `d`, the lower and upper asymptotic limits of the response (τ = `top` and δ = `bot` in **bayesnec**); and, for some models, `e`, the effective dose $\texttt{ED}_{50}$ (ω = $\texttt{EC}_{50}$ in **bayesnec**) (Ritz *et al.* 2015). Estimation in **drc** is based on the maximum likelihood principle, which under the assumption of normally distributed response values simplifies to non-linear least squares. The **bayesnec** package provides a Bayesian implementation of many of the non-linear models offered by **drc**.

We compared the **drc** and **bayesnec** fits for the three parameter no-effect-concentration model implemented in WinBugs by Fox (2010) (the `nec3param` model in **bayesnec**, see Section 2.1.2) for two selected herbicides from the data from Jones and Kerswell (2003). The data comprise assays of herbicide phytotoxicity on chlorophyll fluorescence measurements (Fv/Fm) of symbiotic dinoflagellates still in the host tissue of the coral. Full detail on this example data set is provided in Section 5. In **bayesnec** this model is fit with the call `bnec(fvfm ~ crf(log_x, model = "nec3param"), data = .x)`, and in **drc** using `drm(fvfm ~ log_x, fct = NEC.3(), data = .x)`. The herbicides hexazinone and tebuthiuron were selected specifically for this comparison as visual inspection indicated they should provide a reasonable fit to the Fox (2010) model as there was some evidence of a threshold effect. For both herbicides, the predicted **drc** and **bayesnec** values were nearly identical using the default behavior of each package (Figure 7).

While **drc** is an excellent tool for fitting CR models using frequentist methods and is widely used (Ritz *et al.* 2015 is cited nearly 2,000 times), **bayesnec** provides a Bayesian alternative using similarly simple syntax. The advantages of Bayesian methods in this setting are numerous, and include direct characterization of parameter uncertainty and posterior predicted samples that provide a valuable resource for model inference (such as comparisons of relative toxicity, see Section 5 and Figure 11). In addition, we have observed that even the use of only weakly informative priors tends to improve the reliability of model fits compared to **drc**, and this may be true of MLE estimation more generally (Krull 2020).

5. Illustrative example

So far we have demonstrated the basic usage of **bayesnec** and compared the results for a single model fit to **drc**. Here we work through an illustrative example demonstrating the use of the package for a data set on herbicide toxicity. The aim here is to indicate the usual workflow and highlight the advantages of model averaging combined with Bayesian methods as a rigorous means of estimating and comparing relative toxicity, and the associated uncertainty. The data we analyse in this example are from Jones and Kerswell (2003).

5.1. Example case study

In our case study, the no-effect-toxicity values of a range of herbicides are first estimated and then their relative toxicity is compared. The response data are the maximum effective quantum yield ($\Delta F/Fm'$) of symbiotic dinoflagellates still in the host tissue of the coral *Seriatopora hystrix* (in hospite or in vivo). $\Delta F/Fm'$ was calculated from Chlorophyll fluorescence parameters measured using a DIVING-PAM chlorophyll fluorometer (Walz) as described in more detail in Jones and Kerswell (2003) and Jones, Muller, Haynes, and Schreiber (2003). The corals were exposed to elevated levels of eight different herbicides (Irgarol 1051, ametryn, diuron, hexazinone, atrazine, simazine, tebuthiuron, ioxynil) at concentrations ranging from 0.3 to 1000 µg/L) for 10 h. Data for ioxynil were excluded from analysis here as this herbicide did not show sufficient response at the maximum concentration.

5.2. Single herbicide analysis

We start by describing the analysis workflow for a single herbicide, ametryn. We first filter ametryn from the larger herbicide data set. The concentration data are log transformed prior to analysis to improve model stability and because this is the natural scaling of the concentration series for these data. As there was little evidence of hormesis (an initial increase in the response at low concentration) in these data (or in the other herbicides, see Section 5.3 below), we used only the `decline` model set as candidate models for the model averaging. Setting `model = "decline"` results in **bayesnec** attempting to fit a set of 14 models, and returning an object of class '`bayesmancfit`'.

```
R> ametryn <- herbicide |>
+   dplyr::mutate(concentration = log(concentration)) |>
+   dplyr::filter(herbicide == "ametryn")
R> manecfit_ametryn <- bayesnec::bnec(
+    fvfm ~ crf(concentration, model = "decline"), data = ametryn, seed = 17)
```

Other than selecting a model set to use, here we leave all other `bnec()` arguments as their default. In this case **bayesnec** correctly chooses a beta distribution to use as the `family`, defaults to the identity link, and drops the models "neclin" and "ecxlin" from the complete list of "decline" models (as these are not appropriate for a zero bounded distribution, such as the beta distribution, Section 2.1). Note that in this example (or in the one detailed in

Section 5.3) we do not show all of the console output and messages generated by both the `bnec()` and underlying `brm` functions, because across 14 models this results in substantial output.

Following model fitting, the quality of the fits should be examined using `check_chains`, as well as `check_priors` to ensure there is good chain mixing and that the default priors were suitable. The results from these checks are omitted here for brevity, but can be easily saved to PDF output for visual inspection and inclusion into any analysis supplementary material by setting the argument `filename` to any non empty string, as in the code below:

```
R> check_chains(manecfit_ametryn, filename = "ametryn_check_chains")
R> check_priors(manecfit_ametryn, filename = "ametryn_check_priors")
```

We can also check the $\widehat{R}$ values of the fitted models using the `rhat()` function, based on the method of Vehtari *et al.* (2021):

```
R> rhat(manecfit_ametryn)
```

Once we are satisfied with the model fits, we can examine the model statistics using `summary`:

```
R> summary(manecfit_ametryn)

Object of class bayesmanecfit

 Family: beta
  Links: mu = identity; phi = identity

Number of posterior draws per model:  8000

Model weights (Method: pseudobma_bb_weights):
             waic   wi
nec3param -450.89 0.02
nec4param -456.61 0.03
ecxexp    -320.62 0.00
ecx4param -465.07 0.30
ecxwb1    -443.51 0.01
ecxwb2    -461.38 0.08
ecxwb1p3  -323.35 0.00
ecxwb2p3  -446.89 0.02
ecxll5    -464.40 0.23
ecxll4    -465.19 0.31
ecxll3    -431.80 0.00

Summary of weighted N(S)EC posterior estimates:
NB: Model set contains a combination of ECx and NEC
    models, and is therefore a model averaged
    combination of NEC and NSEC estimates.
       Estimate  Q2.5 Q97.5
```

```
N(S)EC    -1.60 -2.09 -0.47

Bayesian R2 estimates:
          Estimate Est.Error Q2.5 Q97.5
nec3param     0.99      0.00 0.99  0.99
nec4param     0.99      0.00 0.99  0.99
ecxexp        0.89      0.01 0.87  0.91
ecx4param     0.99      0.00 0.99  0.99
ecxwb1        0.99      0.00 0.98  0.99
ecxwb2        0.99      0.00 0.99  0.99
ecxwb1p3      0.94      0.01 0.90  0.96
ecxwb2p3      0.99      0.00 0.99  0.99
ecxll5        0.99      0.00 0.99  0.99
ecxll4        0.99      0.00 0.99  0.99
ecxll3        0.99      0.00 0.98  0.99
```

For a 'bayesmanecfit' object with multiple model fits, summary first displays the class, the family and links that have been used for the model fits, the number of posterior draws contained within each model fit, and a table of the model weights for each model, showing the Widely Applicable Information Criterion (waic, Watanabe and Opper 2010) from **loo** and weights (wi) which are based by default on the "pseudobma" method (method = "pseudobma") with Bayesian bootstrap (BB = TRUE) (see above), but this can be easily modified via argument loo_controls using amend. For the ametryn data set, weights are highest for the ecx4param model, followed closely by the ecxll4 model, with some lesser support for the ecxwb2 model, and a very small amount of support for the two nec models (nec3param and nec4parm).

Because **bayesnec** was primarily developed for estimating no-effect-concentrations, an estimate of the model averaged no-effect toxicity estimate is also provided. In this case the no-effect toxicity estimate is reported as an N(S)EC value as the model set contains a combination of "nec" and "ecx" models. In units of log concentration, the N(S)EC value for ametryn is -1.6, which is equivalent to 0.202 µg/L. The 95% credible intervals are also provided, based on the 0.025 and 0.975 quantiles of the weighted pooled posterior sample.

Finally, Bayesian R^2 estimates are also provided (Gelman, Goodrich, Gabry, and Vehtari 2019), as an indicator of overall model fit. This is useful because model weights are always relative to the models actually included in the model set for the given data. The R^2 provides an indicator of model fit that can be compared objectively across data sets as an indication of the quality of the fit of any of the supplied models to the data.

We can plot all the models contained within the 'bayesmanecfit' using the autoplot() function, with argument all_models = TRUE (Figure 8):

```
R> ametryn_plot_all <- autoplot(manecfit_ametryn, all_models = TRUE)
```

We can also plot the model averaged fit that is used to derive the model averaged no-effect-concentration for ametryn, as displayed in the summary (Figure 9):

```
R> ametryn_plot <- autoplot(manecfit_ametryn)
```

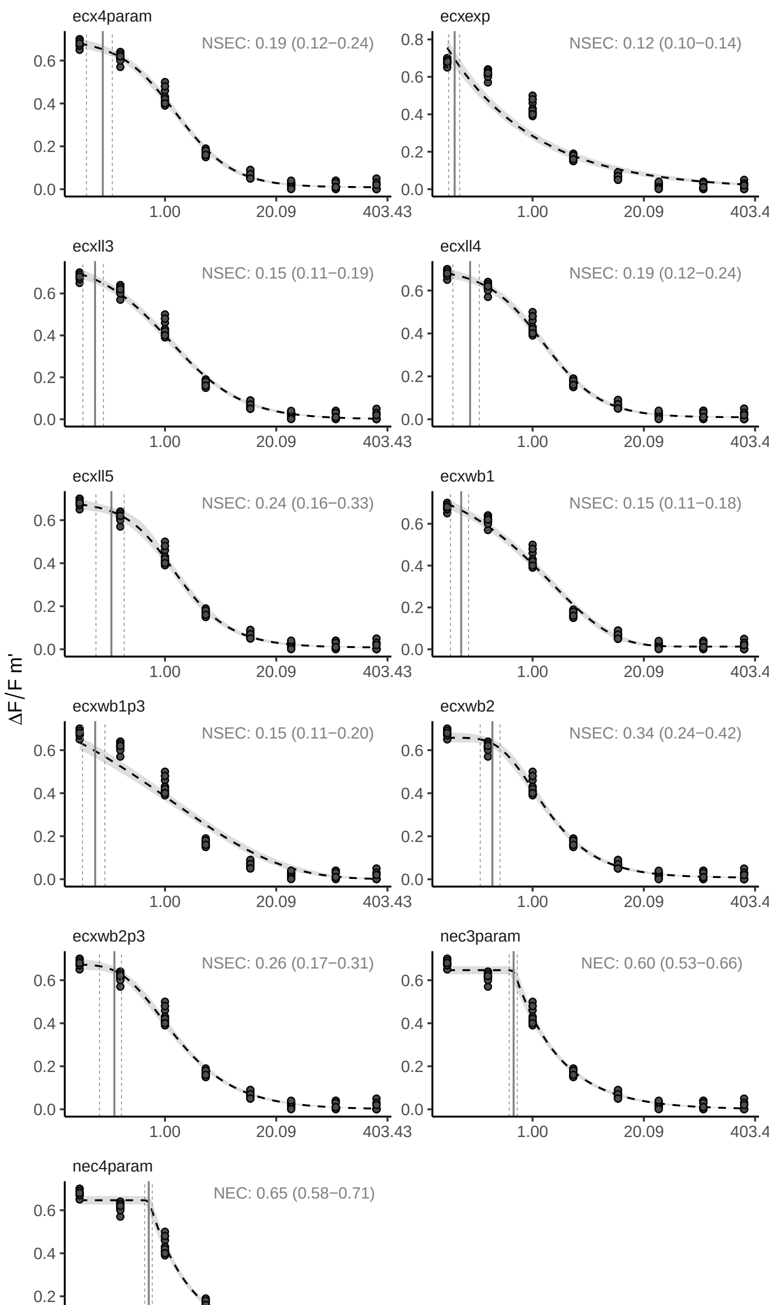

Figure 8: Individual model fits to the ametryn data set, showing the estimated no effect concentration for each. Data are the maximum effective quantum yield ($\Delta F/Fm'$) of symbiotic dinoflagellates (in hospite) in *Seriatopora hystrix* exposed to elevated ametryn (range 0.3 to 1000 µg/L) for 10 h. No-effect toxicity values presented are the median and 95% credible intervals of the posterior estimates of the NEC parameter obtained for all `nec` models, and the posterior predicted NSEC values estimated from all smooth `ecx` models.

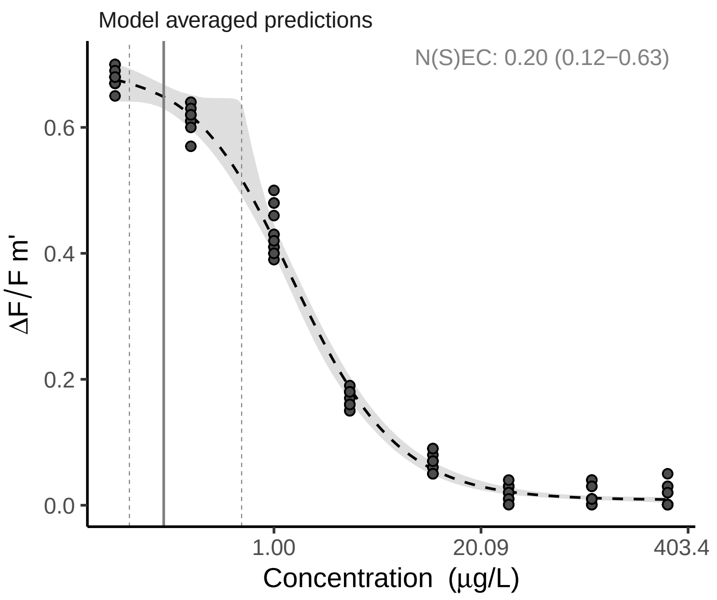

Figure 9: Full model averaged '`bayesmanecfit`' to the ametryn data set, showing the estimated model-averaged no effect concentration. Data are the maximum effective quantum yield ($\Delta F/Fm'$) of symbiotic dinoflagellates (in hospite) in *Seriatopora hystrix* exposed to elevated ametryn (range 0.3 to 1000 µg/L) for 10 h. No-effect toxicity values presented are model averaged posterior densities of the NEC parameter obtained from all fitted `nec` models, and the NSEC values estimated from all smooth `ecx` models, summarized as a median and 95% credible intervals. Only the `decline` model set was used (i.e., hormesis models were excluded).

5.3. Comparing toxicity across herbicides

Above we describe the workflow for a single herbicide. We now show how to use the same workflow across all herbicides to generate full '`bayesmanecfit`' model averaged fits and no-effect-toxicity estimates, and use this to compare their relative toxicity.

We start by modeling the concentration-response curves for all seven photo toxicity data sets using the **bayesnec** package via the following code:

```
R> manecfit <- herbicide |>
+   dplyr::mutate(concentration = log(concentration)) |>
+   split(f = ~ herbicide) |>
+   purrr::map(function(x) {
+     bayesnec::bnec(fvfm ~ crf(concentration, model = "decline"),
+       data = x, seed = 17)
+   })
R> save(manecfit, file = "manecfits.RData")
```

Because we want to run the analysis for all seven herbicides separately we first split the data, then call the `bnec()` function for each herbicide using **purrr** (Henry and Wickham 2023). We use the same settings and default arguments as for our single herbicide example above (ametryn, see Section 5.2). Note that fitting a large set of models using Bayesian methods can take some time (see Section 7), and we recommend running the analysis at a convenient time, and saving the resulting output to a `.RData` file to work with later.

Once we have our list of fitted '`bayesmanecfit`' objects for each herbicide, we can use `rhat` to check that all models fitted successfully for each, as well as check the chains and priors for each fitted model, although we have skipped these steps here. It is also possible to simply remove any models that fail the `rhat` criteria of <1.05 using the function `amend()`. Note for the "decline" model set, there are no fits with poor `rhat` values for this example. If there are models that fail to converge (have high `rhat` values for some parameters, divergent transitions or issues identified with chain mixing) it may be possible to improve those fits by re-running the model with a greater number of iterations, modified priors, or adjusting other fitting options within **brms** such as `adapt_delta`. Unfortunately, at this time `bnec()` will not support model averaging across models fitted using varying numbers of iterations, so to improve the fit of a single model, all models in the set will need to be re-fit with the same (higher) number of iterations. We recommend exploring the required changes using a single '`bayesnecfit`' of any problematic models before re-running the complete '`bayesmanecfit`' set.

```
R> cleaned_fits <- purrr::map(manecfit, function(x) {
+   bad_fits <- rhat(x)$failed
+   out_fit <- x
+   if (length(bad_fits) > 0) {
+     out_fit <- bayesnec::amend(x, drop = bad_fits)
+   }
+   out_fit
+ })
```

To facilitate comparison across the herbicides, we create a collated table of model weights by extracting the `"mod_stats"` element from each herbicide's '`bayesmanecfit`', again using **purrr**:

```
R> library("tidyr")
R> library("stringr")
R> modtab <- purrr::map_dfr(cleaned_fits, function(x) {
+   x$mod_stats |>
+     dplyr::select(model, wi) |>
+     dplyr::mutate(wi = round(wi, 3))
+ }, .id = "herbicide") |>
+   tidyr::pivot_wider(names_from = herbicide, values_from = wi) |>
+   data.frame()
R> colnames(modtab) <- stringr::str_to_title(colnames(modtab))
```

This collated table of model weights shows that the best fitting models vary substantially across the CR curves for the seven herbicides (Table 1). Few herbicides showed any weight for the `nec` threshold models, with the exception of ametryn which had some, albeit limited, support. The weights for the various `ecx` models varied substantially, with at least some support for more than one model in all cases. This shows clearly the value of the model averaging approach adopted in **bayesnec**, which effectively accommodates this model uncertainty by seamlessly providing weighted model averaged inferences.

Model	Ametryn	Atrazine	Diuron	Hexazinone	Irgarol	Simazine	Tebuthiuron
`nec3param`	0.021	0.000	0.000	0.000	0.001	0.000	0.018
`nec4param`	0.028	0.001	0.000	0.000	0.001	0.000	0.003
`ecxexp`	0.000	0.000	0.000	0.000	0.000	0.000	0.000
`ecx4param`	0.296	0.142	0.251	0.113	0.255	0.175	0.001
`ecxwb1`	0.010	0.439	0.124	0.000	0.258	0.107	0.000
`ecxwb2`	0.082	0.001	0.001	0.016	0.028	0.018	0.165
`ecxwb1p3`	0.000	0.000	0.000	0.019	0.000	0.015	0.000
`ecxwb2p3`	0.017	0.001	0.001	0.057	0.001	0.053	0.502
`ecxll5`	0.232	0.289	0.311	0.022	0.189	0.156	0.310
`ecxll4`	0.314	0.127	0.224	0.115	0.268	0.175	0.001
`ecxll3`	0.000	0.000	0.088	0.658	0.000	0.301	0.000

Table 1: Fitted valid models and their relative pseudo-BMA weights for CR curves for the effects of seven herbicides on maximum effective quantum yield ($\Delta F/Fm'$) of symbiotic dinoflagellates of the coral *Seriatopora hystrix*.

We use the **bayesnec** `autoplot`, together with **ggpubr** (Kassambara 2023) to make a panel plot of the weighted model averaged predicted curves for all seven herbicides (Figure 10).

```
R> library("ggpubr")
R> all_plots <- lapply(cleaned_fits, function(x) {
+   autoplot(x, xform = exp) +
+     scale_x_continuous(trans = "log", labels = round_digits) +
+     theme(axis.title.x = element_blank(),
+           axis.title.y = element_blank(),
+           strip.background = element_blank(),
+           strip.text.x = element_blank()) +
+     ggtitle("")
+ })
R> figure <- ggpubr::ggarrange(plotlist = all_plots, nrow = 4,
+   ncol = 2, labels = names(all_plots), align = "hv",
+   font.label = list(color = "black", size = 12, face = "plain"))
```

Across the seven herbicides, the '`bayesmanecfit`' model averaged fits model the input data very well, with predictions generally very confident (Figure 10). The slight uncertainty in the appropriate model form for the ametryn data set is evident in the weighted average predicted values as a broader confidence band at the estimated position of the NEC threshold point (Figure 10). The N(S)EC values are model averaged posterior densities of the NEC parameter obtained from all fitted `nec` models, and the NSEC values estimated from all smooth `ecx` models. These values are the **bayesnec** estimates for the no-(significant)-effect concentration required for the integration of this toxicity data into the relevant regulatory framework in Australia, the Australian and New Zealand Water Quality Guidelines (ANZG 2018). While the recommendation that NEC is the preferred toxicity estimate in this framework is well established (Warne *et al.* 2015, 2018), use of the NSEC is recent (Fisher and Fox 2023) and while yet to gain formal approval for use in the Australian setting presents a potential alternative no-effect estimate for smooth curves.

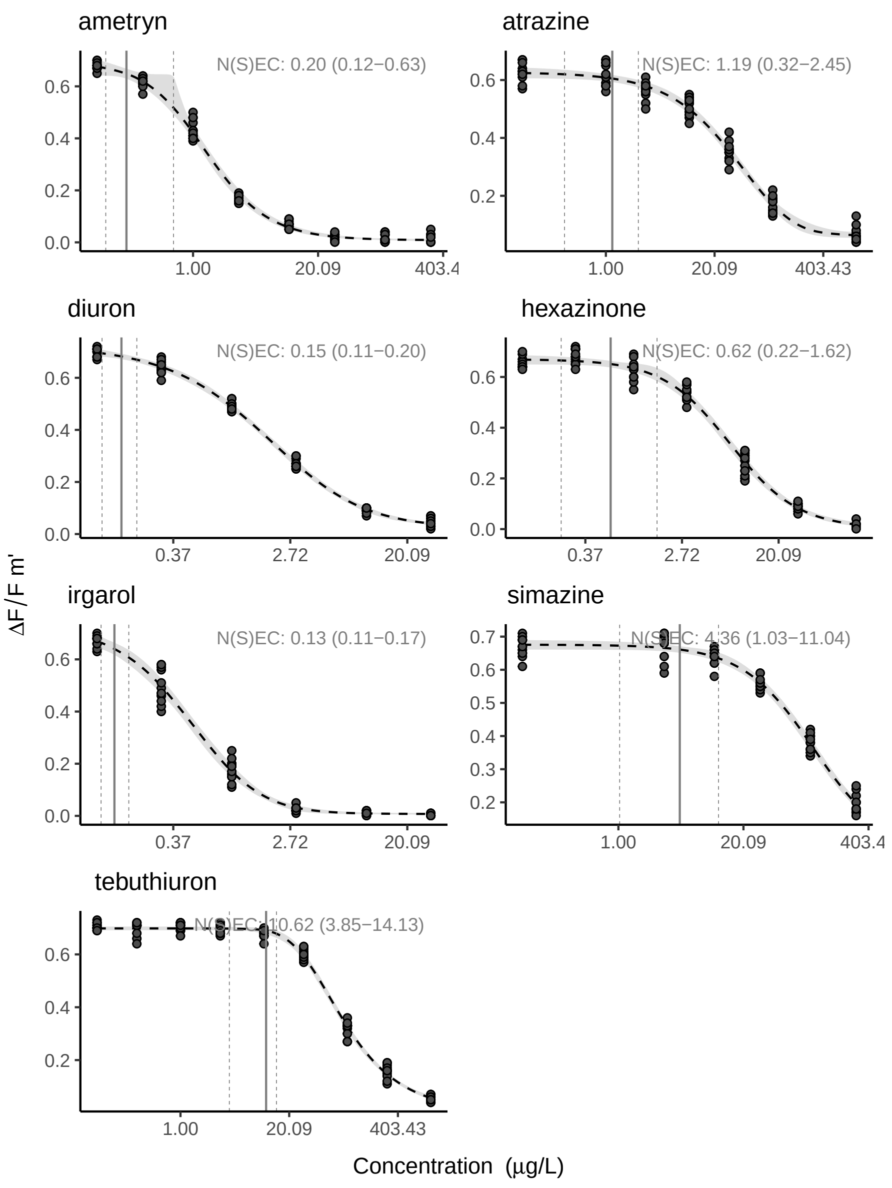

Figure 10: Full model averaged '**`bayesmanecfit`**'s to seven phototoxicity data sets, showing estimated no effect concentrations. Data are the maximum effective quantum yield ($\Delta F/Fm'$) of symbiotic dinoflagellates (in hospite) in *Seriatopora hystrix* exposed to elevated Irgarol 1051, ametryn, diuron, hexazinone, atrazine, simazine, or tebuthiuron (range 0.3 to 1000 µg/L) for 10 h. N(S)EC values presented are model averaged posterior densities of the NEC parameter obtained from all fitted `nec` models, and the NSEC values estimated from all smooth `ecx` models, summarized as a median and 95% credible intervals. Only the `decline` model set was used (i.e., hormesis models were excluded).

Finally, we also use the `compare_posterior()` function to extract and plot the weighted averaged posterior samples for the N(S)EC toxicity values for all herbicides (Figure 11). This shows clearly that irgarol, diuron and ametryn are the most toxic, and exhibit relatively similar toxicity, with their posterior densities substantially overlapping (Figure 11). The

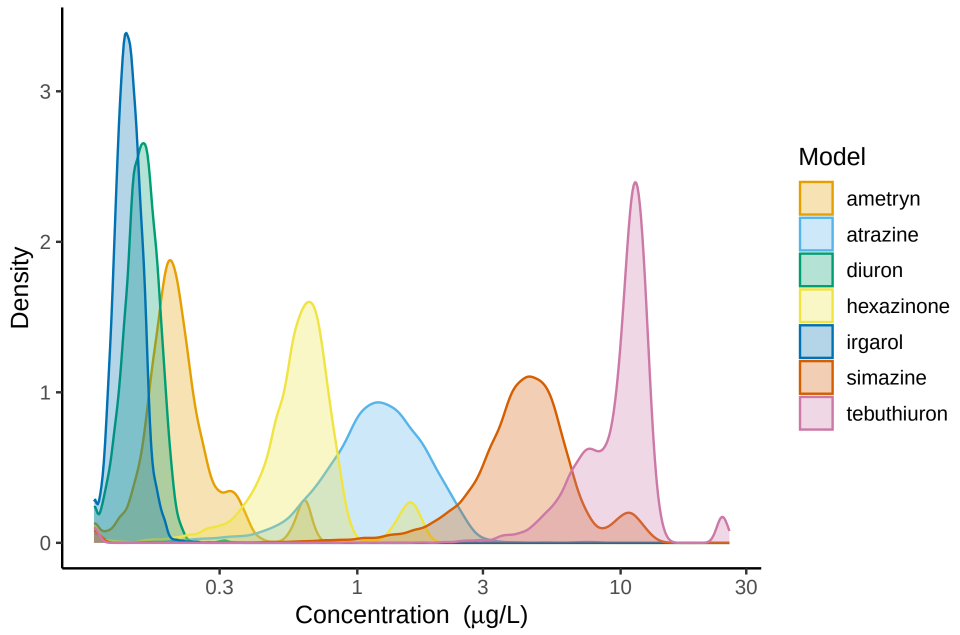

Figure 11: Posterior distributions for N(S)EC toxicity estimates for the effect of seven herbicides on maximum effective quantum yield ($\Delta F/Fm'$) of symbiotic dinoflagellates of the coral *Seriatopora hystrix*. Shown are medians with 80% uncertainty intervals.

Herbicide	Atrazine	Diuron	Hexazinone	Irgarol	Simazine	Tebuthiuron
Ametryn	0.019	0.866	0.054	0.940	0.012	0.011
Atrazine	–	0.989	0.857	0.989	0.038	0.012
Diuron	–	–	0.015	0.717	0.012	0.011
Hexazinone	–	–	–	0.986	0.0178	0.011
Irgarol	–	–	–	–	0.011	0.011
Simazine	–	–	–	–	–	0.080

Table 2: Probability of no-difference in no-effect toxicity for seven herbicides. Values are based on the proportional overlap in predicted posterior probability density of the N(S)EC estimates.

herbicide tebuthiuron is the least toxic of these seven, followed by simazine, atrazine and finally hexazinone, which exhibits intermediate toxicity (Figure 11). `compare_posterior` also calculates the probability of difference in toxicity across the herbicides, which confirm the visual results and can be used to infer significant differences in toxicity response (Table 2).

```
R> post_comp <- compare_posterior(cleaned_fits, comparison = "n(s)ec")
R> prob_diff <- post_comp$prob_diff |>
+   tidyr::separate(col = comparison, into = c("herbicide", "columns")) |>
+   tidyr::pivot_wider(names_from = columns, values_from = prob) |>
+   dplyr::mutate(across(where(is.numeric), ~round(.x, 3)))
R> colnames(prob_diff) <- stringr::str_to_sentence(colnames(prob_diff))
R> prob_diff$Herbicide <- stringr::str_to_sentence(prob_diff$Herbicide)
```

6. Discussion

In order to be accessible to a broad community of varying statistical capabilities, we have simplified fitting a **bayesnec** model as much as possible, whilst retaining the ability to modify a wide range of arguments as necessary. Where possible we have tried to set default values to align with those in **brms**. Wherever we deviate, this is generally towards being more conservative and/or we have clearly explained our reasoning. Specific examples include: (1) `iter`, which we increased from the **brms** default of 2,000 to 10,000 as we found that a higher number of iterations are generally required for these non-linear models; and (2) the use of `pointwise = TRUE` (where possible) and `sample_prior = "yes"` to avoid excessive crashes in the R programming environment when used in the Windows operating system and allow the use of the `hypothesis()` function respectively. We welcome constructive criticism of our selections and users must expect that default settings may change accordingly in later releases. We encourage users to modify these default values themselves whenever appropriate.

We have made considerable effort to ensure that **bayesnec** makes a sensible prediction for the appropriate `family`, constructs appropriate weakly informative priors, and generates sensible initial values. However, this is a difficult task across such a broad range of non-linear models, and across the potential range of ecotoxicological data that may be used. The user must interrogate their model fits using the wide array of helper functions, and use their own judgment regarding the appropriateness of model inferences for their own application. Of particular importance are examination of model fit statistics through the `summary()` and `rhat()` methods, visual inspection of all model fits in '`bayesmanecfit`' objects (via `plot(..., all_models = TRUE)` and `check_chains(..., all_models = TRUE)`) and an assessment of the posterior versus prior probability densities to ensure default priors are appropriate (using `check_priors()`).

The model averaging approach implemented in **bayesnec** is widely used in a range of settings (in ecology for example, see Dormann *et al.* 2018, for a thorough review). However, model averaging is relatively new to ecotoxicology (but see, for example, Shao and Gift 2014; Thorley and Schwarz 2023; Fox *et al.* 2021; Wheeler and Bailer 2009). In **bayesnec** we have implemented a broad range of potential models, and the default behavior is to fit them all (if appropriate for the natural range of the response). More research is required to understand how model-set selection influences model inference. While some studies suggest using a broad range of models may be optimal (Wheeler and Bailer 2009), others indicate that including multiple models of similar shape may overweight the representation of that shape in model averaged predictions (Fox *et al.* 2021). In addition, it is important to understand that when models are added or removed from the model-set, this can sometimes have a substantial influence on model predictions (potentially changing estimated ECx values, for example). As the model-set in **bayesnec** may change with further package development it is important to keep a record of the models that were actually fitted in a given analysis, in the event it is necessary to reproduce a set of results.

6.1. Model suitability for NEC and EC$_x$ estimation

In principle all models provide an estimate for a "no-effect" toxicity concentration. As seen above, for model strings with `nec` as a prefix, the NEC is directly estimated as parameter η = `NEC` in the model, as per Fox (2010). On the other hand, model strings with `ecx` as a prefix are continuous curve models with no threshold, typically used for extracting ECx values from concentration-response data. In this instance, the no-effect toxicity value reported is actually the No-Significant-Effect-Concentration (NSEC, see details in Fisher and Fox 2023), defined as the concentration at which there is a user supplied certainty (based on the Bayesian posterior estimate) that the response falls below the estimated value of the upper asymptote ($\tau =$ `top`) of the response (i.e., the response value is significantly lower than that expected in the case of no exposure). The default value for this NSEC proportion is 0.01, which corresponds to an alpha value (Type-I error rate) of 0.01 for a one-sided test of significance. The NSEC concept has been recently explored using simulation studies and case study examples, and when combined with the NEC estimates of threshold models within a model- averaging approach, can yield robust estimates of N(S)EC and of their uncertainty within a single analysis framework (Fisher *et al.* 2024b). Both NEC and NSEC can be calculated from fitted models using the functions `nec()` and `nsec()`. The model averaged N(S)EC is automatically returned as part of the fitted model for any '`bayesmanecfit`' that contains a combination of both `"nec"` and `"ecx"` models. The significance level used can be adjusted from the default value of 0.01 using `amend()`.

ECx estimates can be equally obtained from both `"nec"` and `"ecx"` models. ECx estimates will usually be lower (more conservative) for `"ecx"` models fitted to the same data as `"nec"` models. There is ambiguity in the definition of ECx estimates from hormesis models – these allow an initial increase in the response (see Mattson 2008) and include models with the string `horme` in their name – as well as those that have no natural lower bound on the scale of the response (models with the string `lin` in their name, in the case of Gaussian response data). For this reason, the `ecx()` function has arguments `hormesis_def` and `type`, both character vectors indicating the desired behavior. For `hormesis_def = "max"`, ECx values are calculated as a decline from the maximum estimates (i.e., the peak at $\eta =$ `NEC`); and `hormesis_def = "control"` (the default) indicates that ECx values should be calculated relative to the control, which is assumed to be the lowest observed concentration. For `type = "relative"` ECx is calculated as the percentage decrease from the maximum predicted value of the response ($\tau =$ `top`) to the minimum predicted value of the response (i.e., `relative` to the observed data). For `type = "absolute"` (the default) ECx is calculated as the percentage decrease from the maximum value of the response ($\tau =$ `top`) to 0. For `type = "direct"`, a direct interpolation of the response on the predictor is obtained.

6.2. Model suitability for response types

Models that have an exponential decay (most models with parameter $\beta =$ `beta`) and lacking the $\delta =$ `bot` parameter are 0-bounded and are not suitable for the Gaussian `family`, or any `family` modeled using a `"logit"` or `"log"` link because they cannot generate predictions of negative response values. Conversely, models with a linear decay (containing the string `lin` in their name) are not suitable for modeling families that are 0-bounded (gamma, Poisson, negative binomial, beta, binomial, beta-binomial) using an `"identity"` link. These restrictions do not need to be controlled by the user, as a call to `bnec()` with `models = "all"` in the formula will simply exclude inappropriate models, albeit with a message.

Strictly speaking, models with a linear hormesis increase are not suitable for modeling responses that are 0, 1-bounded (binomial-, beta- and beta-binomial-distributed), however they are currently allowed in **bayesnec**, with a reasonable fit achieved through a combination of the appropriate distribution being applied to the response, and **bayesnec**'s `make_inits()` function which ensures initial values passed to **brms** yield response values within the range of the user-supplied response data.

7. Computational considerations

7.1. Analytical reproducibility

Considerations of analytical reproducibility are particularly relevant to CR modeling, where the model outcomes can often have far reaching management implications. It is challenging to fit complex non-linear models in practice, particularly for non-Gaussian response variables. As noted above, the Bayesian approach adopted in **bayesnec** using weakly informative priors to develop appropriate initial values works reasonably well to produce relatively stable model fits across a range of data sets. However, some models can fail and this can result in changes in the model set, possibly leading to variation in the resulting multi-model inference.

To help with reproducibility **bayesnec** now allows a seed to be passed to **brm** and Stan. If a seed is used in the `bnec()` call, it will also be used internally to generate initial values. Although in R seeds are consistent across versions and operational systems, and therefore the initial values will be the same across different users for a given seed, the underlying Stan model fitting mechanism may yield slightly different parameter estimates for known reasons relating to floating point operations (see Chapter 20 in Stan Development Team 2021). A potentially better strategy for ensuring reproducibility is to build a docker (`https://docs.docker.com/get-docker/`) container, an approach representing one strategy towards overcoming the reproducibility crisis (Baker 2016). Also note that while setting a seed can be useful to obtain consistent outputs it might be worth examining how robust the inference is across different seeds.

Due to the fact that the underlying '`brmsfit`' model fitted using **cmdstanr** does not retain initial values as part of the returned model object, reproducibility may be reduced when using **cmdstanr**.

7.2. Computational details

All computations in this paper were performed using **rmarkdown** (Allaire *et al.* 2024) with R version 4.3.3 (2024-02-29), aarch64, darwin20, and the base packages (R Core Team 2024) **stats**, **graphics**, **grDevices**, **utils**, **datasets**, **methods**, **base** along with and **bayesnec** (Fisher *et al.* 2024a), **brms** (Bürkner 2017), **ggplot2** (Wickham 2016), and **Rcpp** (Eddelbuettel *et al.* 2024).

7.3. Computation times

Bayesian analysis can take considerable time to run, and can also generate relatively large data files that can require substantial computer power to work with the resulting output. Here we provide a benchmark of the time taken to run the examples in this article. Run times based using an 11th Gen Intel Core i9-11950H @ 2.60GHz with 33.5 Gb RAM were:

- 0.9 minutes for a simple single model fit to the example `nec_data` data set (Section 3.3);
- 11.8 minutes for the full `decline` model set fit to the herbicide ametryn in the example herbicide data set (Section 5.2); and
- 84.2 minutes to fit the full `decline` model set to all seven herbicides in the example herbicide data set (Section 5.3).

When the same analysis was performed on an Apple M1 Max with 68.7 Gb RAM, run times were:

- 0.2 minutes for a simple single model fit to the example `nec_data` data set (Section 3.3);
- 2.1 minutes for the full `decline` model set fit to the herbicide ametryn in the example herbicide data set (Section 5.2); and
- 16 minutes to fit the full `decline` model set to all seven herbicides in the example herbicide data set (Section 5.3).

These computing times can be substantially reduced by running the four chains in parallel, by passing the argument `cores = 4` via the `bnec(...)` call.

7.4. Data requirements

Due to the relatively long compute times of **bayesnec** fits, especially when multiple models are fit at once, we recommend that when running **bayesnec** the resulting model fit is saved as an `.RData` file to be read in and used in later workflows to examine model diagnostics, plotting, parameter estimates and inference.

The data requirements for saving model fits can be relatively large. The single model fit (Section 3.3) generates an object of 62.4 Mb; the full `decline` model set fit to the herbicide ametryn (see Section 5.2) an object of 74.7 Mb; and the full `decline` model set to all seven herbicides in the example data set (Section 5.3) an object of 522.9 Mb.

7.5. Dependencies

bayesnec has been built using **brms** (Bürkner 2017) as the main dependency which provides an interface to fit Bayesian generalized (non-)linear multivariate multilevel models using Stan program (Stan Development Team 2021), a C++ package for performing full Bayesian inference (`https://mc-stan.org/`).

brms can use two alternative interfaces to Stan, including **rstan** (Stan Development Team 2024) and **cmdstanr** (Gabry and Češnovar 2024) both of which require Rtools and the g++ compiler to be properly configured in R. Making sure **brms** is properly working on your machine is essential before any attempt to use the **bayesnec** package for analyses, as if this dependency is not working, **bayesnec** will not work. Instructions for installing these two packages can be found for **cmdstanr** (`https://mc-stan.org/cmdstanr/articles/cmdstanr.html`) and **rstan** (`https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started`).

8. Future directions

The **bayesnec** package is a work in progress, and we welcome suggestions and feedback that will improve the package performance and function. Our goal is to make **bayesnec** as user friendly as possible, and capable of dealing with most real world CR modeling applications in the hope that Bayesian statistics will become more widely used in applied risk assessment. Submit bugs and feature requests through the package issues (`https://github.com/open-AIMS/bayesnec/issues`) on GitHub. Some suggested future enhancements include:

- The addition of other families to accommodate a broader range of response variables. This could include a range of distributions already available in **brms**, such as the zero-inflated Poisson or zero-truncated Gaussian. In addition, there are other useful distributions currently unavailable in **brms**, such as the Tweedie distribution and ordered beta model. Currently **bayesnec** implements adjustments away from 0 (gamma, beta) or 1 (beta) as a strategy for allowing modeling with these types of data using the closest most convenient statistical distribution. There are no readily available distributions able to model data that includes 0 and 1 on the continuous scale in **brms** and **bayesnec** currently does 0 and 1 adjustments followed by modeling using a beta distribution. The ordered beta model has been suggested as a better method for modeling continuous data with lower an upper bounds (see Kubinec *et al.* 2021) that could be readily implemented in the **brms** customs families framework. For data that are 0 to ∞ on the continuous scale the Tweedie distribution may prove a much better option than the current zero-bounded gamma, and has been used extensively in fisheries research for biomass data (Shono 2008). As this `family` is not currently available in **brms** this would also need to be implemented as a custom `family`, which for the Tweedie is not trivial.
- A hypothesis method for testing against toxicity thresholds. The **brms** package includes a `hypothesis()` function that allows for testing parameter estimates against specified criteria. This is used in **bayesnec** in the `check_prior()` function, which is a wrapper that examines the deviation of each parameter in the given model relative to 0 as a means of generating posterior and prior probability density plots for comparison. However, an additional wrapper function could be developed that allows toxicity to be assessed, as measured through NEC, or ECx for example, against a required pre-defined threshold. Such a feature may be useful where toxicity testing is used as a trigger in risk management (for example, using whole-effluent-toxicity (WET) testing, Karman and Smit 2019).

Acknowledgments

The development of **bayesnec** was supported by an AIMS internal grant. Usage, testing and functionality of both the **jagsNEC** and **bayesnec** packages were substantially aided through input from Joost van Dam, Andrew Negri, Florita Flores, Heidi Luter, Marie Thomas and Mikaela Nordborg. Florita Flores and Murray Logan provided valuable comments on the manuscript text. Ron Jones provided technical computing support.

References

Allaire JJ, Xie Y, Dervieux C, McPherson J, Luraschi J, Ushey K, Atkins A, Wickham H, Cheng J, Chang W, Iannone R (2024). ***rmarkdown**: Dynamic Documents for R*. `doi:10.32614/CRAN.package.rmarkdown`. R package version 2.27.

ANZG (2018). "Australian and New Zealand Guidelines for Fresh and Marine Water Quality." Australian and New Zealand Governments and Australian State and Territory Governments, URL `https://www.waterquality.gov.au/guidelines/anz-fresh-marine`.

Baker M (2016). "1,500 Scientists Lift the Lid on Reproducibility." *Nature*, **533**, 452–454. doi:10.1038/533452a.

Banner KM, Irvine KM, Rodhouse TJ (2020). "The Use of Bayesian Priors in Ecology: The Good, the Bad and the Not Great." *Methods in Ecology and Evolution*, **11**, 882–889. doi:10.1111/2041-210x.13407.

Becker RA, Chambers JM, Wilks AR (1988). *The New S Language.* Wadsworth and Brooks/Cole.

Bürkner PC (2017). "**brms**: An R Package for Bayesian Multilevel Models Using **Stan**." *Journal of Statistical Software*, **80**(1), 1–28. doi:10.18637/jss.v080.i01.

Bürkner PC (2018). "Advanced Bayesian Multilevel Modeling with the R Package **brms**." *The R Journal*, **10**(1), 395–411. doi:10.32614/rj-2018-017.

Burnham KP, Anderson DR (2002). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach.* 2nd edition. Springer-Verlag, New York. doi:10.1007/b97636.

Carpenter B, Gelman A, Hoffman MD, Lee D, Goodrich B, Betancourt M, Brubaker M, Guo J, Li P, Riddell A (2017). "Stan: A Probabilistic Programming Language." *Journal of Statistical Software*, **76**(1), 1–32. doi:10.18637/jss.v076.i01.

Charles S, Wu D, Ducrot V (2020). "How to Account for the Uncertainty from Standard Toxicity Tests in Species Sensitivity Distributions: An Example in Non-Target Plants." *bioRxiv*. doi:10.1101/2020.07.02.183863.

Chipman HA, George EI, McCulloch RE (2010). "BART: Bayesian Additive Regression Trees." *The Annals of Applied Statistics*, **4**(1), 266–298. doi:10.1214/09-aoas285.

Dalgarno S (2021). "**shinyssdtools**: A Web Application for Fitting Species Sensitivity Distributions (SSDs)." *Journal of Open Source Software*, **6**(57), 2848. doi:10.21105/joss.02848.

Depaoli S, Winter SD, Visser M (2020). "The Importance of Prior Sensitivity Analysis in Bayesian Statistics: Demonstrations Using an Interactive **shiny** App." *Frontiers in Psychology*, **11**, 608045. doi:10.3389/fpsyg.2020.608045.

Dormann CF, Calabrese JM, Guillera-Arroita G, Matechou E, Bahn V, Bartoń K, Beale CM, Ciuti S, Elith J, Gerstner K, Guelat J, Keil P, Lahoz-Monfort JJ, Pollock LJ, Reineking B, Roberts DR, Schröder B, Thuiller W, Warton DI, Wintle BA, Wood SN, Wüest RO, Hartig F (2018). "Model Averaging in Ecology: a Review of Bayesian, Information-Theoretic, and Tactical Approaches for Predictive Inference." *Ecological Monographs*, **88**(4), 485–504. doi:10.1002/ecm.1309.

Eddelbuettel D, Francois R, Allaire JJ, Ushey K, Kou Q, Russell N, Ucar I, Bates D, Chambers J (2024). ***Rcpp**: Seamless R and C++ Integration.* doi:10.32614/CRAN.package.Rcpp. R package version 1.0.13.

Ellison AM (1996). "An Introduction to Bayesian Inference for Ecological Research and Environmental Decision-Making." *Ecological Applications*, **6**(4), 1036–1046. doi:10.2307/2269588.

Fisher R, Barneche D, Ricardo G, Fox D (2024a). ***bayesnec**: A Bayesian No-Effect-Concentration (NEC) Algorithm.* doi:10.32614/CRAN.package.bayesnec. R package version 2.1.3.0.

Fisher R, Bessell-Browne P, Jones R (2019a). "Synergistic and Antagonistic Impacts of Suspended Sediments and Thermal Stress on Corals." *Nature Communications*, **10**, 2346. doi:10.1038/s41467-019-10288-9.

Fisher R, Fox DR (2023). "Introducing the No Significant Effect Concentration (NSEC)." *Environmental Toxicology and Chemistry*, **42**(9), 2019–2028. doi:10.1002/etc.5610.

Fisher R, Fox DR, Negri AP, Van Dam J, Flores F, Koppel D (2024b). "Methods for Estimating No-Effect Toxicity Concentrations in Ecotoxicology." *Integrated Environmental Assessment and Management*, **20**(1), 279–293. doi:10.1002/ieam.4809.

Fisher R, Ricardo G, Fox D (2020). "Bayesian Concentration-Response Modelling Using **jagsNEC**." doi:10.5281/zenodo.3966864.

Fisher R, Van Dam RA, Batley GE, Fox DR, Harford AJ, Humphrey CL, King CK, Menendez P, Negri AP, Proctor A, Shao Q, Stauber JL, Van Dam JW, Warne MSJ (2019b). "Key Issues in the Derivation of Water Quality Guideline Values: A Workshop Report." doi:10.13140/rg.2.2.29774.82241. Australian Institute of Marine Science Report, Crawley, WA, Australia. 57 pp.

Fisher R, Walshe T, Bessell-Browne P, Jones R (2018). "Accounting for Environmental Uncertainty in the Management of Dredging Impacts Using Probabilistic Dose-Response Relationships and Thresholds." *Journal of Applied Ecology*, **55**(1), 415–425. doi:10.1111/1365-2664.12936.

Flores F, Marques JA, Uthicke S, Fisher R, Patel F, Kaserzon S, Negri AP (2021). "Combined Effects of Climate Change and the Herbicide Diuron on the Coral *Acropora Millepora*." *Marine Pollution Bulletin*, **169**, 112582. doi:10.1016/j.marpolbul.2021.112582.

Fox DR (2008). "NECS, NOECS and the ECx." *Australasian Journal of Ecotoxicology*, **14**, 7–9. URL https://www.leusch.info/ecotox/aje/archives/vol14p7.pdf.

Fox DR (2010). "A Bayesian Approach for Determining the No Effect Concentration and Hazardous Concentration in Ecotoxicology." *Ecotoxicology and Environmental Safety*, **73**(2), 123–131. doi:10.1016/j.ecoenv.2009.09.012.

Fox DR, Van Dam RA, Fisher R, Batley GE, Tillmanns AR, Thorley J, Schwarz CJ, Spry DJ, McTavish K (2021). "Recent Developments in Species Sensitivity Distribution Modeling." *Environmental Toxicology and Chemistry*, **40**(2), 293–308. doi:10.1002/etc.4925.

Gabry J, Češnovar R (2024). ***cmdstanr**: R Interface to **CmdStan***. URL https://mc-stan.org/cmdstanr/.

Gelman A, Goodrich B, Gabry J, Vehtari A (2019). "R-Squared for Bayesian Regression Models." *The American Statistician*, **73**(3), 307–309. doi:10.1080/00031305.2018.1549100.

Gelman A, Simpson D, Betancourt M (2017). "The Prior Can Often Only Be Understood in the Context of the Likelihood." *Entropy*, **19**(10). doi:10.3390/e19100555.

Gottschalk F, Nowack B (2013). "A Probabilistic Method for Species Sensitivity Distributions Taking Into Account the Inherent Uncertainty and Variability of Effects to Estimate Environmental Risk." *Integrated Environmental Assessment and Management*, **9**(1), 79–86. doi:10.1002/ieam.1334.

GraphPad Software (2024). "GraphPad **Prism**." URL https://www.graphpad.com/.

Henry L, Wickham H (2023). ***purrr**: Functional Programming Tools*. doi:10.32614/CRAN.package.purrr. R package version 1.0.2.

Hoffman MD, Gelman A (2014). "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research*, **15**(47), 1593–1623.

Jones RJ, Kerswell AP (2003). "Phytotoxicity of Photosystem II (PSII) Herbicides to Coral." *Marine Ecology Progress Series*, **261**, 149–159. doi:10.3354/meps261149.

Jones RJ, Muller J, Haynes D, Schreiber U (2003). "Effects of Herbicides Diuron and Atrazine on Corals of the Great Barrier Reef, Australia." *Marine Ecology Progress Series*, **251**, 153–167. doi:10.3354/meps251153.

Karman CC, Smit MGD (2019). "Whole Effluent Toxicity Data and Discharge Volumes to Assess the Likelihood That Environmental Risks of Offshore Produced Water Discharges Are Adequately Controlled." *Integrated Environmental Assessment and Management*, **15**(4), 584–595. doi:10.1002/ieam.4139.

Kassambara A (2023). ***ggpubr**: **ggplot2** Based Publication Ready Plots*. doi:10.32614/CRAN.package.ggpubr. R package version 0.6.0.

Krull M (2020). "Comparing Statistical Analyses to Estimate Thresholds in Ecotoxicology." *PLOS One*, **15**(4), e0231149. doi:10.1371/journal.pone.0231149.

Kubinec R, Barcel'o J, Goldszmidt R, Grujic V, Model T, Schenk C, Cheng C, Hale T, Hartnett AS, Messerschmidt L, Petherick A, Thorvaldsdottir S (2021). "Statistically Validated Indices for COVID-19 Public Health Policies." *SocArXiv*, SocArXiv E-Print Archive. doi:10.31235/osf.io/rn9xk.

Lunn DJ, Thomas A, Best N, Spiegelhalter D (2000). "**WinBUGS** – A Bayesian Modelling Framework: Concepts, Structure, and Extensibility." *Statistics and Computing*, **10**(4), 325–337. doi:10.1023/a:1008929526011.

Mattson MP (2008). "Hormesis Defined." *Ageing Research Reviews*, **7**(1), 1–7. doi:10.1016/j.arr.2007.08.007.

Neal T (2006). "Hypothesis Testing and Bayesian Estimation Using a Sigmoid Emax Model Applied to Sparse Dose-Response Designs." *Journal of Biopharmaceutical Statistics*, **16**(5), 657–677. doi:10.1080/10543400600860469.

Peng R (2015). "The Reproducibility Crisis in Science: A Statistical Counterattack." *Significance*, **12**(3), 30–32. doi:10.1111/j.1740-9713.2015.00827.x.

Pinheiro J, Bates D, DebRoy S, Sarkar D, R Core Team (2024). ***nlme**: Linear and Nonlinear Mixed Effects Models*. doi:10.32614/CRAN.package.nlme. R package version 3.1-165.

Pires AM, Branco JA, Picado A, Mendonca E (2002). "Models for the Estimation of a 'No Effect Concentration'." *Environmetrics*, **13**, 15–27. doi:10.1002/env.501.

Plummer M (2003). "**JAGS**: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling." In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)*. Vienna. URL https://www.R-project.org/conferences/DSC-2003/Proceedings/.

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Ritz C, Baty F, Streibig JC, Gerhard D (2015). "Dose-Response Analysis Using R." *PLOS One*, **10**(12), e0146021. doi:10.1371/journal.pone.0146021.

Ritz C, Streibig JC (2005). "Bioassay Analysis using R." *Journal of Statistical Software*, **12**(5), 1–22. doi:10.18637/jss.v012.i05.

Shao K, Gift JS (2014). "Model Uncertainty and Bayesian Model Averaged Benchmark Dose Estimation for Continuous Data." *Risk Analysis*, **34**, 101–20. doi:10.1111/risa.12078.

Shono H (2008). "Application of the Tweedie Distribution to Zero-Catch Data in CPUE Analysis." *Fisheries Research*, **93**(1), 154–162. doi:10.1016/j.fishres.2008.03.006.

Stan Development Team (2021). "Stan Modeling Language Users Guide and Reference Manual." Version 2.28, URL https://mc-stan.org/.

Stan Development Team (2024). "**rstan**: The R Interface to Stan." R package version 2.32.6, URL http://mc-stan.org/rstan/.

Su YS, Yajima M (2024). ***R2jags**: Using R to Run **JAGS***. doi:10.32614/CRAN.package.R2jags. R package version 0.8-5.

Thorley J, Schwarz C (2023). ***ssdtools**: Species Sensitivity Distributions*. doi:10.32614/CRAN.package.ssdtools. R package version 1.0.6.

Tidepool Scientific, LLC (2022). "**ToxCalc**." URL https://tidepool-scientific.com/.

Vehtari A, Gabry J, Magnusson M, Yao Y, Bürkner PC, Paananen T, Gelman A (2024). ***loo**: Efficient Leave-One-Out Cross-Validation and WAIC for Bayesian Models*. doi:10.32614/CRAN.package.loo. R package version 2.8.0.

Vehtari A, Gelman A, Gabry J (2017). "Practical Bayesian Model Evaluation Using Leave-One-Out Cross-Validation and WAIC." *Statistics and Computing*, **27**, 1413–1432. doi:10.1007/s11222-016-9696-4.

Vehtari A, Gelman A, Simpson D, Carpenter B, Burkner PC (2021). "Rank-Normalization, Folding, and Localization: An Improved $\widehat{\mathrm{R}}$ for Assessing Convergence of MCMC." *Bayesian Analysis*, **16**(2), 667–718. doi:10.1214/20-ba1221.

Vehtari A, Simpson D, Gelman A, Yao Y, Gabry J (2019). "Pareto Smoothed Importance Sampling." *arXiv 1507.02646*, arXiv.org E-Print Archive. doi:10.48550/arXiv.1507.02646.

Warne MSJ, Batley GE, Van Dam RA, Chapman JC, Fox DR, Hickey CW, Stauber JL (2015). "Revised Method for Deriving Australian and New Zealand Water Quality Guideline Values for Toxicants." Prepared for the Council of Australian Government's Standing Council on Environment and Water (SCEW). Department of Science, Information Technology and Innovation, Brisbane, Queensland. 43 pp., URL `https://www.waterquality.gov.au/sites/default/files/documents/warne-wqg-derivation2015.pdf`.

Warne MSJ, Batley GE, Van Dam RA, Chapman JC, Fox DR, Hickey CW, Stauber JL (2018). "Revised Method for Deriving Australian and New Zealand Water Quality Guideline Values for Toxicants Update of 2015 Version." Prepared for the Revision of the Australian and New Zealand Guidelines for Fresh and Marine Water Quality. Australian and New Zealand Governments and Australian state and territory governments, Canberra, 48 pp., URL `https://www.waterquality.gov.au/sites/default/files/documents/warne-wqg-derivation2018.pdf`.

Watanabe S, Opper M (2010). "Asymptotic Equivalence of Bayes Cross Validation and Widely Applicable Information Criterion in Singular Learning Theory." *Journal of Machine Learning Research*, **11**(12).

Wheeler MW, Bailer AJ (2009). "Comparing Model Averaging With Other Model Selection Strategies for Benchmark Dose Estimation." *Environmental and Ecological Statistics*, **16**, 37–51. `doi:10.1007/s10651-007-0071-7`.

Wickham H (2016). ***ggplot2**: Elegant Graphics for Data Analysis.* Springer-Verlag. `doi:10.1007/978-0-387-98141-3`.

Yao Y, Vehtari A, Simpson D, Gelman A (2018). "Using Stacking to Average Bayesian Predictive Distributions." *Bayesian Analysis*, **13**(3), 917–1007. `doi:10.1214/17-ba1091`.

Affiliation:

Rebecca Fisher, Diego R. Barneche
Australian Institute of Marine Science
Crawley, WA 6009, Australia
and
Oceans Institute
University of Western Australia, Crawley, Western Australia, Australia
Crawley, WA 6009, Australia
E-mail: `r.fisher@aims.gov.au`, `d.barneche@aims.gov.au`

Gerard F. Ricardo
Marine Spatial Ecology Lab, School of the Environment
The University of Queensland
St Lucia, QLD 4072, Australia
and
Australian Institute of Marine Science
Cape Cleveland, Qld 4810, Australia
E-mail: `g.ricardo@uq.edu.au`

David R. Fox
Department of Infrastructure Engineering
The University of Melbourne
Parkville, Vic 3010, Australia
and
Environmetrics Australia Pty Ltd
Beaumaris, Vic 3193, Australia
E-mail: `david.fox@environmetrics.net.au`

Journal of Statistical Software `https://www.jstatsoft.org/`
published by the Foundation for Open Access Statistics `https://www.foastat.org/`
August 2024, Volume 110, Issue 5 *Submitted:* 2022-12-15
`doi:10.18637/jss.v110.i05` *Accepted:* 2023-10-26