Environmetrics AUSTRALIA

Australian Government

Australian Institute of Marine Science

Joint investigation into statistical methodologies underpinning the derivation of toxicant guideline values in Australia and New Zealand

David R Fox, Rebecca Fisher, Joseph L Thorley, Carl Schwarz

Prepared for the Department of Agriculture, Water and the Environment, contracts C05244 (EA) and C05266 (AIMS)

This page intentionally blank

Environmetrics Australia
PO Box 7117
Beaumaris VIC, 3193
+61-3-9018-7121
www.environmetrics.net

Australian Institute of Marine Science

PMB No 3
Townsville MC Qld 4810

PO Box 41775
Casuarina NT 0811

Indian Ocean Marine Research Centre
University of Western Australia, M096
Crawley WA 6009

This report should be cited as:

Fox D.R., Fisher R., Thorley J.L., Schwarz C. (2022) Joint Investigation into statistical methodologies underpinning the derivation of toxicant guideline values in Australia and New Zealand. Report prepared for the Department of Agriculture, Water and the Environment. Environmetrics Australia, Beaumaris, Vic and the Australian Institute of Marine Science, Perth, WA. (167 pp)

DISCLAIMER

While reasonable efforts have been made to ensure that the contents of this document are factually correct, neither AIMS nor Environmetrics Australia make any representation or give any warranty regarding the accuracy, completeness, currency or suitability for any particular purpose of the information or statements contained in this document. To the extent permitted by law AIMS and/or Environmetrics Australia shall not be liable for any loss, damage, cost or expense that may be occasioned directly or indirectly through the use of or reliance on the contents of this document.

Revision History:		*Name*	*Date*	*Comments*
A	Prepared by:	*David Fox and Rebecca Fisher*	*21/01/2022*	
	Reviewed by:	*Ross Jones*	*24/01/2022*	
1	Revised by:	*David Fox and Rebecca Fisher*	*4/3/2021*	
2	Revised by:	*David Fox and Rebecca Fisher*	*16/3/2021*	

Cover photo:
SeaSim, Australian Institute of Marine Science, Cape Cleveland Australia. Image: C. Miller

Contents

EXECUTIVE SUMMARY AND RECOMMENDATIONS 1
1 INTRODUCTION 4
1.1 The species sensitivity distribution (SSD) 4
1.2 SSDs in Australia and New Zealand 6
1.3 Burrs under the saddle 6
1.4 Model averaging 7
1.5 A shiny future for `Burrlioz` 7
1.6 Task overview and summary 9
2 DISTRIBUTION FITTING 10
2.1 Numerical instability of the *Burr III* 10
2.2 Numerical properties of the *inverse Pareto* distribution 12
2.2.1 Unbiased parameter estimation for the *inverse Pareto* distribution 15
2.2.2 HCx estimates from the *inverse Pareto* distribution 15
2.2.3 Bootstrap alternative for HCx confidence intervals from the *inverse Pareto* distribution 17
2.3 `ssdtools` code optimisation, convergence criteria and software enhancements 20
2.4 Benchmark datasets 23
2.5 Evaluating SSD methodologies 24
2.5.1 Benchmark datasets 25
2.5.2 Simulated datasets 28
2.5.3 Simulation Study 1 (*log-normal*, *log-logistic*, *inverse Weibull*) 30
2.5.4 Simulation Study 2 (Johnson family) 37
2.6 Refine mixture-modelling with a view to incorporation 44
2.6.1 Implementation in `ssdtools` (TMB) 44
2.6.2 Simulation Study 1 (*log-normal, log-logistic*, *inverse Weibull*) 44
2.6.1 Simulation Study 2 (Johnson family) 46
2.6.2 Simulation Study 3 (mixtures) 48
3 ALTERNATIVE CI AND HC_X ESTIMATION STRATEGIES 51
3.1 Parameter estimates using L-moments (*Burr III*) 51
3.2 Closed form estimation (*Burr III*) 54
3.3 Parametric versus non-parametric bootstrapping 55
4 DISCUSSION 58
4.1 Comparing SSD methodologies 58
4.2 Mixture distributions 60
4.3 CI and SE estimation methods 61
4.4 Finalise default distributions 61
4.4.1 Sensitivity and numerical stability issues 61
4.4.2 Incorporation of the *inverse Pareto* as a candidate SSD. 65
4.4.3 Parsimony – the minimum "optimal" set 66
5 REFERENCES 71
6 ACKNOWLEDGEMENTS 73
7 APPENDICES 73
Appendix A: Detailed task list 74
Appendix B: *Inverse Pareto* distribution 75
Appendix C: Unbiased estimation for the *inverse Pareto* distribution 101
Appendix D: R-code for fitting mixtures 107
Appendix E: *L*-Moment estimators for the *Burr III* distribution 113
Appendix F: Estimation and inference for the burr iii distribution 126
Appendix G: A note on re-scaling SSDs 142
Appendix H: Reconciliation of HCx estimates for the inverse pareto distribution 144
Appendix I: Additional analyses assessing the recommended distribution set 147
Comparing the recommended candidate distribution set to using all distributions 147
Implications of the recommended set for data following a *Burr III* distribution 150
Mixture distribution mixing proportion and boundary conditions 153
All ssddata fits using the recommended settings 157

TABLE OF FIGURES

Figure 1. Illustration of the competing risks for 'protectors' and 'polluters' of the environment as a function of the level of enforced environmental protection (Fox 1999). As environmental protection increases (x-axis), the risks for those seeking protection of the environment decline, whereas the risk to operators increases. Dotted lines indicate the maximum 'protector' and 'polluter' risk respectively.....5

Figure 2. Plot of kurtosis versus skewness for a variety of theoretical probability distributions with empirical values for selected datasets from `ssddata` (see 2.4 Benchmark datasets).....11

Figure 3. Empirical *cdf* (black line) and fitted American *inverse Pareto* using `Burrlioz` parameter estimates (red line); fitted American *inverse Pareto* with correct MLEs (purple line); and fitted European *inverse Pareto* with MLEs (blue line).....14

Figure 4. The theoretical *cdf* for the *inverse Pareto* distribution with parameters α = 0.5989 and β = 0.2631. 18

Figure 5. Boxplots of HC errors for estimates obtained using `Burrlioz` and new method of this Section for 1000 samples of size *n=100* sampled from the theoretical *inverse Pareto* distribution in Figure 4. HC error is calculated as the difference between the estimated HCx and the theoretical HCx.....19

Figure 6. Boxplots of HC 95% CI widths using `Burrlioz` and new method of this Section for 1000 samples of size *n=100* sampled from the theoretical *inverse Pareto* distribution in Figure 4.....20

Figure 7. HC values estimated using the original `fitdistrplus` version of `ssdtools` with the default distribution set of *log-logistic*, *log-normal* and *gamma*, plotted against `Burrlioz 2.0`. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds. Points with the left half missing have an x value of 0.....25

Figure 8. HC values estimated using the original `fitdistrplus` version of `ssdtools` with the default distribution set of *log-logistic*, *log-normal* and *gamma*, plotted against the new version of `ssdtools` based on TMB using the same distributions. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimates, as well as the lower and upper confidence bounds. 26

Figure 9. HC values estimated using the default distribution set of *log-logistic*, *log-normal* and *gamma*, plotted against the estimate based on all distributions available in the newest version of `ssdtools`. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds. 27

Figure 10. HC values estimated using `ssd_fit_burrlioz` plotted against `Burrlioz 2.0` estimates. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds. Points with the left half missing have an x value of 0.....28

Figure 11. HC values estimated using RBurrlioz plotted against `Burrlioz 2.0`. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds. Values with the left and lower half values missing have x and y values of 0. 30

Figure 12. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the fitted distribution. All fits were done using `ssdtools` (TMB). Note that the inverse.weibull (as implemented in `Burrlioz 2.0`) and the lgumbel (`ssdtools`) are exactly equivalent distributions.....32

Figure 13. Confidence interval width of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the fitted distribution. All fits were done using `ssdtools` (TMB). Note that the inverse.weibull (as implemented in `Burrlioz 2.0`) and the lgumbel (`ssdtools`) are equivalent distributions.....33

Figure 14. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the fitted distribution. All fits were done using `ssdtools` (TMB). Note that the inverse.weibull (as implemented in `Burrlioz 2.0`) and the lgumbel (`ssdtools`) are equivalent distributions. 34

Figure 15. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the applied fitting method and includes RBurrlioz, the `ssd_fit_burrlioz` method implemented in `ssdtools` (TMB) using the non-paremetric bootstrap (in line with RBurrlioz) and two model averaged methods using `ssdtools` (TMB), including one using only the current BC three default distributions (*log-logistic*, *log-normal* and *gamma*, ssdtools_tmb_default) and one using all the available distributions (ssdtools_tmb_all).....35

Figure 16. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the applied fitting method and includes RBurrlioz, the `ssd_fit_burrlioz` method implemented in `ssdtools` (TMB) using the non-parametric bootstrap (inline with RBurrlioz) and two model averaged methods using `ssdtools` (TMB), including one using only the current BC three default distributions (*log-logistic*, *log-normal* and *gamma*, ssdtools_tmb_default) and one using all of the available distributions (ssdtools_tmb_all). 36

Figure 17. Benchmark data (black circles) and generated theoretical curve based on the Johnson family of distributions that was used in simulation studies (solid black line). Horizontal coloured dashed lines show where the HC1, HC5, HC10 and HC20 values cross the theoretical curve.....37

Figure 18. Twenty realisations of the fitted distribution for each of the methods considered, including RBurrlioz (cyan lines) and two model averaged methods using `ssdtools` (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default – red lines; and all - using all the available distributions, gold lines) for four different sample sizes (n). The solid black line shows the theoretical source distribution.....39

Figure 19. Coverage estimates for 95% confidence intervals of the HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the applied fitting method and includes RBurrlioz and two model averaged methods using `ssdtools` (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default; and all - using all the available distributions).....41

Figure 20. Bias (mean estimate-actual) of HC estimates across four different protection levels (plot columns - p = 0.01, 0.05, 0.1 and 0.=2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the applied fitting method and includes RBurrlioz and two model averaged methods using `ssdtools` (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default; and all - using all the available distributions). 42

Figure 21. Mean 95% confidence interval width of HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the applied fitting method and includes `RBurrlioz` and two model averaged methods using `ssdtools` (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default; and all - using all the available distributions). 43

Figure 22. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni). 45

Figure 23. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni). 46

Figure 24. Mean bias (estimated-actual) of HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni). ... 47

Figure 25. Coverage estimates for nominal 95% confidence intervals of the HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni). 48

Figure 26. Density plots showing the simulated data for each of the datasets used in the mixture Simulation Study. Colours indicate the mixing proportion, including 0 (no *log-normal* data, entirely *log-logistic*), 0.5 (an even mixture of *log-normal* and *log-logistic*) and 1 (all *log-normal* data, no *log-logistic*). Values in parentheses show the mean values of the underlying *log-normal* and *log-logistic* distributions respectively. 49

Figure 27. Mean AICc based model weights for each of the fitted distributions, for each dataset in the Simulation Study, as a function of sample size. Colours indicate the mixing proportion, including 0 (no *log-normal* data, entirely *log-logistic*), 0.5 (an even mixture of *log-normal* and *log-logistic*) and 1 (all *log-normal* data, no *log-logistic*). The horizontal line indicates a weight of 0.1, representing the expected value if all distributions were weighted equally. 50

Figure 28. Histogram of metolachlor concentrations in freshwater. 52

Figure 29. Empirical *cdf* of transformed metolachlor toxicity data together with fitted *Burr III* distributions using MLEs and L-moment parameter estimates. 53

Figure 30. Log ratio of the non-parametric versus the parametric bootstrap method for estimating lower and upper bound (lwr and upr) confidence intervals using the `ssd_fit_burrlioz()` function implemented in `ssdtools`. Data are based on the simdat3 simulation dataset (Simulation Study 1). 57

Figure 31. Approximate coverage estimates for a nominal 95% confidence intervals across four different species protection levels (plot columns - p = 0.01, 0.05, 0.1 and 0.2) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the bootstrapping method (parametric versus nonparametric). All fits were done using the `ssd_fit_burrlioz()` function in `ssdtools`. Data are based on the simdat3 simulation dataset (Simulation Study 1). 58

Figure 32. The proportion of simulated datasets/iterations for which the `ssdtools` -fitted distributions were able successfully converge. Note that for the *Burr III* and both mixture distributions, this

proportion includes distributions that may have returned a result, but this was at one of the bounds of the parameter set (i.e. shape1 or shape2 = 20 or 0.05 in the case of *Burr III*, or p=<0.2 in the case of mixtures). 62
Figure 33. AICc based weights for the *Burr III* (burrIII3), *inverse Pareto* (invpareto) and *log-Gumbel* (lgumbel) distributions fitted using `ssdtools` for the Simulation Study 1 (left-hand plot) and 2 (right-hand plot) datasets. Plot rows show the data sample size. For Simulation Study 1, plot columns show the parent distributions. Boxplot colours show the convergence status of the *Burr III* distribution for the model fit, including 0 (parameters were at one of the defined bounds of shape1 or shape2 = 20 or 0.05), 1 (successful convergence, parameters not at bounds) or NA (non-convergence due to issues unrelated to bounds). 65
Figure 34. Cullen-Frey plot of four common distributions used in SSD modelling. The plot shows a distribution's kurtosis (a numerical measure of 'peakedness') as a function of its skewness. 68

LIST OF TABLES

Table 1. Project tasks and priority. 9
Table 2. Comparison of quantiles from *inverse Pareto* distributions. 14
Table 3. Comparison point and interval estimates (lower (LCL) and upper (UCL) 95% confidence limits) for 1, 5, 10 and 20 HC values using bootstrapping (parametric and non-parametric) in `ssdtools` and the method outlined in this Section. All calculations based on n=10,000. Note that in all cases the true HC was captured by the CI. 18
Table 4. Actual coverage estimates for `Burrlioz` and compared to the new method based on 1000 simulated datasets of n=100 from the theoretical *inverse Pareto* distribution (Figure 4). Values are the proportion of simulated outcomes where the true value falls within the estimated 95% confidence interval. 19
Table 5. List of additions and modifications to `ssdtools` 21
Table 6. The proportion of datasets/iterations for which the `ssdtools` -fitted *Burr III* distribution has parameter estimates at one (or more) of the bounds (At bounds), failed to converge entirely (True non-convergence), or was able to successfully converge without reaching one of the bounds (Converged and not at bounds). Results are summarized across both the Study 1 and Study 2 simulated datasets. 63

EXECUTIVE SUMMARY AND RECOMMENDATIONS

Species sensitivity distributions (SSDs) remain a practical tool for the determination of safe threshold concentrations for toxicants in fresh and marine waters. While the fundamental SSD approach employed by jurisdictions around the world has remained similar over the last 20 years, variations do exist in some of the technical details of the methods and associated software tools that have been developed, sometimes leading to marked differences in results which can undermine confidence in SSD approaches. The Australian and New Zealand Guidelines for Fresh and Marine Water Quality (ANZG 2018) currently require the use of a software tool, `Burrlioz 2.0` in the derivation of guideline values. While this has proved a highly valuable tool, a range of issues have also been raised over the years, including reports of 'crashes', numerical instability issues, and failure to converge to a 'solution'. There is also concern that the statistical distributions used by default did not always provide the best representation of the input data and the `Burrlioz` computer code was not open-source and 'locked away'. Australia and New Zealand have recognised that future deployments of `Burrlioz` (or some incarnation of `Burrlioz`) should be open-source and web-based, and there is considerable interest in the `shinyssdtools` `R` Canadian web-based shiny app (https://bcgov-env.shinyapps.io/ssdtools/) front end to the `ssdtools` `R` package as a potential replacement to `Burrlioz`. This report summarises the outcomes of a one-year joint collaborative research project involving Australian and Canadian researchers who undertook extensive investigations into methodologies and tools associated with SSD modelling, focusing on `Burrlioz` and `ssdtools`.

During the project, substantive additions and modifications to `ssdtools` were made, including: replacement of the `fitdistrplus` package by `TMB` to allow more control over model specification and transparency regarding convergence criteria; better assessment of numerical instability issues; the implementation of two mixture distributions (*log-normal-log-normal* and *log-logistic-log-logistic*); implementation of the *inverse Pareto* distribution; development of a function to mimic `Burrlioz`; the option to perform non-parametric bootstrap resampling (as is implemented in `Burrlioz`); and the ability to automatically rescale to aid model fitting.

Research undertaken included exploring and resolving issues with numerical instability and developing alternative solutions for estimating hazard concentrations (HCx) and/or associated 95% confidence intervals of the *Burr III* and *inverse Pareto* distributions; comparisons between `Burrlioz` and `ssdtools` using collated benchmark and synthetic datasets; the development and evaluation of mixture distributions as a potential candidate for accommodating bimodal data; and an assessment of bias and coverage across a range of candidate distribution sets used in model averaging. Based on the findings, the project recommends that the Australian-New Zealand and Canadian jurisdictions adopt the R-package `ssdtools` using an expanded default set of distributions.

RECOMMENDATIONS

1. Software

It is recommended that Australian-New Zealand and Canadian jurisdictions adopt the R-package `ssdtools` (and its on-line implementation `shiny ssdtools`) as the default software tool for fitting SSDs to toxicity data for the purpose of deriving predicted no effect concentrations of chemicals in natural aquatic environments.

2. Default distribution set

A: The default list of candidate distributions in `ssdtools` should be comprised of the following: *log-normal*; *log-logistic*; *gamma*; *inverse Weibull* (*log-Gumbel); Weibull;* mixture of two *log-normal* distributions

and,

B: Australian-New Zealand and Canadian jurisdictions agree on a default set of distributions to use with `ssdtools` whether it be those identified in A above or some other set as may be defined from time to time.

3. Implementation

It is recommended that Australia-New Zealand encourage and facilitate an expeditious transition to `ssdtools` and the model averaging method. A period of overlap may be required whereby the results of either `Burrlioz` or `ssdtools` can be used and reported. In pursuit of this objective, it is further recommended that the responsible government departments in Australia and New Zealand provide support for education and training initiatives associated with the use of model averaging, `ssdtools` and the R computing environment.

4. Periodic review and on-going collaboration

It is recommended that Australian-New Zealand and Canadian jurisdictions agree to establish a framework to continue the R&D collaboration on SSD modelling between the two countries that has been initiated by this project. This framework should also provide oversight of periodic reviews, technical evaluations, and resolution of end-user issues.

SUGGESTIONS FOR FURTHER INVESTIGATION

Within the framework identified in Recommendation 4:

1. Re-visit the candidate distribution set

A: The addition of both versions of the *inverse Pareto* distribution to the candidate distribution list should be further explored to ascertain the desirability and utility of this option in the context of SSD modelling.

B: Additional testing be undertaken to evaluate the alternative procedure more fully for obtaining point and interval estimates of HCx values in the *inverse Pareto* case.

C: Investigate the utility and desirability of including other mixture distributions, such as a *lognormal – log-logistic*.

D: Resolve convergence issues with the *Gompertz* distribution and further evaluate the utility for inclusion in the candidate set.

E: Investigate if the *Burr III* distribution provides additional flexibility that warrants inclusion in the default set. Further work would be required to understand how best to accommodate the *Burr III* into model averaging, given its boundary condition behaviour.

2. Address software anomalies

Undertake more detailed investigations to compare the performance of `Burrlioz` 2.0 and the `ssd_fit_burrlioz` function in `ssdtools`. These investigations should focus on instances where HCx estimates are substantially different to determine the reasons for the discrepancies.

3. Enhance the parameter estimation framework

Explore the use of L-moments: (i) as a possible method of establishing initial parameter estimates for iterative MLE techniques; and/or (ii) to (optionally) provide an alternative parameter estimation framework for SSD model-fitting.

4. Bias correction of parameter estimates

Develop bias corrections for all distributions used in the candidate set and include an option for these corrections to be applied when estimating HCx using `ssdtools`.

5. Convergence issues with the *log-normal-log-normal* mixture distribution
The proportion of bootstrap samples is sometimes low for the current default settings for the mixture distribution. Further investigation is required to understand and resolve this issue in the current development version of `ssdtools`

1 INTRODUCTION

This final report summarises investigations undertaken by Canadian and Australian researchers into statistical and computational aspects of species sensitivity distribution (SSD) modelling. Professor David Fox (*Environmetrics Australia and the University of Melbourne*) and Dr. Rebecca Fisher (*Australian Institute of Marine Science and University of Western Australia Oceans Institute and School of Plant Biology*) were contracted by the Department of Agriculture, Water and the Environment (DAWE) while Dr. Carl Schwarz (*StatMathComp Consulting, Vancouver, BC, Canada*) and Dr. Joseph Thorley (*Poisson Consulting, Nelson, BC, Canada*) were contracted by Environment and Climate Change Canada. The broad objectives of the collaborative research project were: (i) to undertake a review of the statistical methodologies used by Australia and New Zealand to derive toxicant default guideline values (DGVs); and (ii) to make recommendations as to the suitability of the `ssdtools` R package and associated *Shiny App (shinyssdtools)* for the calculation of DGVs in Australian and Canadian jurisdictions. A more detailed list of specific investigations and issues to be addressed is provided in Appendix A of this report.

1.1 The species sensitivity distribution (SSD)

The determination of threshold concentrations for toxicants in fresh and marine waters below which a high proportion of all species will be protected from harmful effects is an exercise in balancing competing risks (Figure 1). Although a conceptual model only, Figure 1 makes clear the fact that the establishment of environmental limits cannot simultaneously minimise the cost to the environment and the cost to the user of that environment. As with most aspects of life, there is a trade-off between the benefits and impacts of human activity and environmental regulation seeks to identify that region identified in Figure 1 where the risks are in some sense 'acceptable'.

Prior to the early 1990s, threshold concentrations of toxicants in receiving waters were set by arbitrarily scaling concentrations derived from acute toxicity studies using a small number of species under controlled laboratory conditions. The scaling factor, also called the 'assessment factor' (AF), was invariably an order of magnitude or more (10, 100, 1000, etc.). Difficulties with this approach are self-evident: the method had no basis in biology or ecology and therefore was difficult to defend; and perhaps more problematically, was that the level of protection afforded by a concentration so determined could not be quantified.

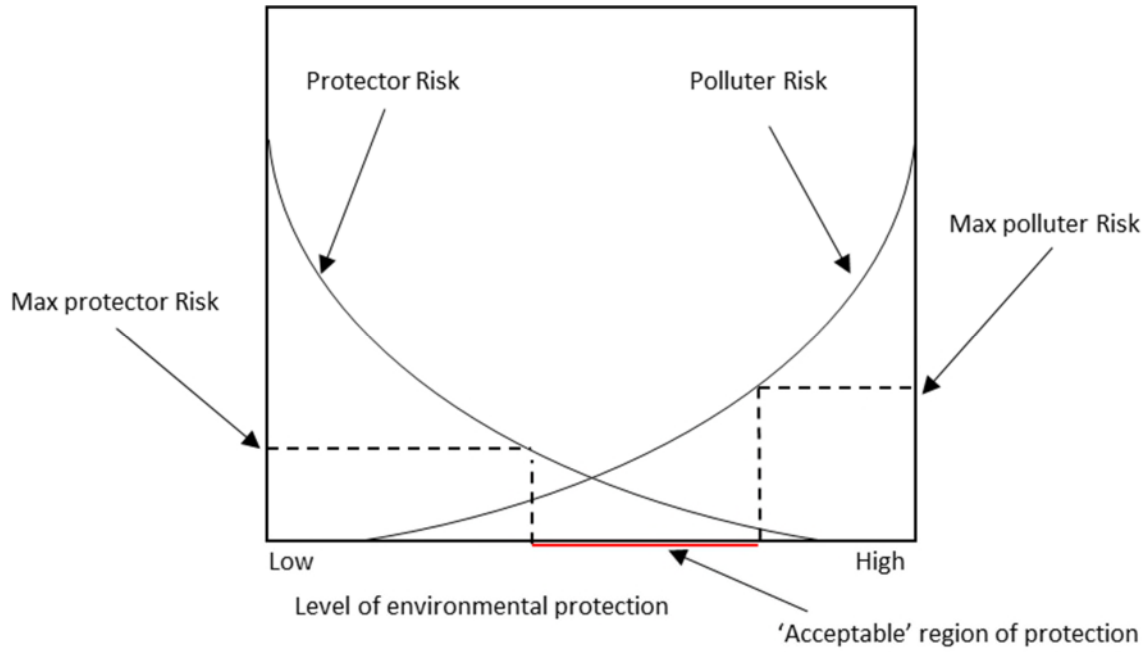


Figure 1. Illustration of the competing risks for 'protectors' and 'polluters' of the environment as a function of the level of enforced environmental protection (Fox 1999). As environmental protection increases (x-axis), the risks for those seeking protection of the environment decline, whereas the risk to operators increases. Dotted lines indicate the maximum 'protector' and 'polluter' risk respectively.

Introduced in the late 1980s, the SSD injected a greater degree of statistical rigour into the process by which default guideline values (DGVs) are established. The SSD describes the cumulative potential for harm to a range of species as the concentration of a contaminant or other stressor increases (Posthuma et al. 2001). SSDs underpin the derivation of protective concentrations (PCx) for x % of all species, or hazardous concentrations (HCx) for 100-x% of all species, that are applied in most current formal water quality guideline value (GV) derivations (Fox et al. 2021). While not without its limitations and shortcomings, the SSD methodology was (to some extent) more statistically defensible and enabled researchers and practitioners to provide estimates of the fraction of species affected/protected together with an assessment of uncertainty in those estimates (Fox et al. 2021).

Uptake of the SSD method over the intervening 40 years has been both slow and patchy and even today, researchers continue to debate the merits of SSD modelling (van den Brink et al. 2001, Forbes & Calow 2002). Despite a significant body of published research and numerous intensive reviews over the past 20 years aimed at improving SSD methods [e.g. (Posthuma et al. 2002, ECETOC 2014, Belanger et al. 2017, Carr & Belanger 2019, Fisher et al. 2019)], the fundamental SSD approach employed by jurisdictions around the world has remained similar. However, variations do exist in some of the technical details of the methods and associated software tools that have been developed and employed. These variations can lead to marked differences in results which has the potential to undermine confidence in SSD approaches. Despite the limitations, SSDs remain a practical tool and until a demonstrably better inferential framework is available, developments and enhancements to conventional SSD practice will and should continue (Fox et al. 2021).

1.2 SSDs in Australia and New Zealand

The use of SSDs in Australia and New Zealand became more widespread in the early 1990s following the publication of 'closed-form' solutions for the determination of DGVs and associated confidence intervals by Aldenberg and Slob (1993) who fitted logistic distributions to No Observed Effect Concentration (NOEC) toxicity data. Motivated by this, David Fox and Graeme Batley worked together to improve the technique by using other probability distributions. Initial investigations into the use of Burr distributions (Burr 1942) were undertaken by Fox and subsequently by CSIRO statistician Quanxi Shao (Shao 2000).

In the mid to late 1990s, Professor Barry Hart of the Water Studies Centre at Monash University chaired a working group established by the Australian Government to oversee the revision of the 1992 ANZECC Water Quality Guidelines. A fundamental shift reflected in the resulting ANZECC/ARMCANZ (2000) Guidelines was the adoption of a risk-based approach to the establishment of DGVs. The kernel of this risk-based approach was the SSD and in particular the use of the *Burr III* distribution. While the *Burr III* distribution provided for a greater variety of distributional shapes than the more commonly used *log-logistic* and *log-normal* distributions, this came at the expense of computational simplicity. Recognising that this increased complexity would severely limit the adoption and utility of the newly advocated approach, the Australian government commissioned CSIRO's Division of Mathematical and Information Sciences (CMIS) to develop user-friendly software to perform the necessary and complex calculations associated with fitting *Burr III* distributions to toxicity data. Version 1.0 of the resulting software tool known as `Burrlioz` was made available in binary (compiled) form to coincide with the release of the ANZECC/ARMCANZ (2000) Guideline documents.

1.3 Burrs under the saddle

Overall, the ANZECC/ARMCANZ (2000) Guidelines and the companion `Burrlioz` tool represented a big step forward along the risk-based path to environmental protection in Australia and New Zealand. The 2000 Guidelines had no legal status, nor were they mandated for use. They were, as the name suggests, *guidelines* that reflected the best available science-based management of aquatic ecosystems at the time.

Feedback from more than 20 years of use by a variety of end-users suggests that `Burrlioz` has been a demonstrable success, although reports of 'crashes', numerical instability issues, and failure to converge to a 'solution' started to emerge almost immediately upon release. Another growing concern was that the `Burrlioz` computer code was 'locked away' and only accessible to the small team of CSIRO statisticians who had developed it. Not only did this preclude users from readily interrogating the software, but it also frustrated the process of continual refinement and improvement – a hallmark of the open-source model. This situation was further compounded by changing structures and priorities within CSIRO which left `Burrlioz` orphaned.

CSIRO was re-engaged by the Commonwealth government in 2014 to revise and update the `Burrlioz` software and, most notably, re-write the core statistical routines using the open-source R statistical computing language. Although R is extraordinarily powerful, it has a very steep learning

curve and, for many people, it is not an intuitive language. Burrlioz 2.0 removed the need for end-users to learn how to use R as it is effectively a compiled GUI that serves as a front-end to the R environment. Elements of the R-code used by Burrlioz 2.0 to fit SSDs can be accessed by users, however the Burrlioz package would not be classed as completely open-source.

Australia and New Zealand have recognised that the ecotoxicology community would be better served if Burrlioz (or some incarnation of Burrlioz) was made open-source and web-based. The advantages of this approach have already been realised with the French web-based tool MOSAIC (https://mosaic.univ-lyon1.fr/ssd, Charles et al. (2018)) and shinyssdtools the Canadian web-based shiny app (https://bcgov-env.shinyapps.io/ssdtools/, Dalgarno (2021)) front end to the ssdtools R package (Thorley & Schwarz 2018).

However, before proceeding down this path it was both prudent and necessary to (i) undertake a review of the status of SSD modelling in Australia and New Zealand; (ii) identify strengths and weaknesses of the Burrlioz 2.0 software; and (iii) decide on a deployment strategy for future software releases. To this end, a technical workshop was held from March 27–29, 2019 at the Australian Institute of Marine Science (AIMS) in Townsville, Queensland, to discuss key issues associated with the derivation of water quality guideline values (GVs). Workshop participants were asked to contemplate the desirability of pursuing established collaborations with Canadian scientists with a view to adopting or replicating their model averaging approach to GV derivations using ssdtools and the associated Shiny app. A comprehensive summary of the workshop deliberations and outcomes can be found in Fisher et al. (2019).

1.4 Model averaging

Many authors have noted that there is no guiding theory in ecotoxicology to justify any particular distributional form for the SSD other than that its domain be restricted to the positive real line (Newman et al. 2000, Zajdlik 2005, Chapman PF & Tarazona J 2007, Fox 2016). Indeed, (Chapman PF & Tarazona J 2007) described the identification of a suitable probability model as one of the most important and difficult choices in the use of SSDs. Compounding this lack of clarity about the functional form of the SSD is the omnipresent, and equally vexatious issue of small sample size, meaning that any plausible candidate model is unlikely to be rejected (Fox et al. 2021). The ssdtools R package uses a model averaging procedure to avoid the need to a-priori select a candidate distribution and instead uses a measure of 'fit' for each model to compute weights to be applied to an initial set of candidate distributions. The method, as applied in the SSD context is described in detail in (Fox et al. 2021), and potentially provides a level of flexibility and parsimony that is difficult to achieve with a single SSD distribution.

1.5 A shiny future for Burrlioz

There was strong support for the use of model averaging, mixture-models and web-based tools such as MOSAIC and ssdtools/shinyssdtools by participants at the Townsville workshop. However, it was acknowledged that the Canadian (shiny)ssdtools would require modification if Australia and New Zealand were to adopt this tool set. Features seen as desirable that are currently not

available in the ssdtools shiny application include additional 'tabs' – for example, an option to use the software in 'research mode' that allows users to delve deeper into (and modify) the inner workings of the R-code, and a separate tab for using 'certified' GV derivation methods, where only government-endorsed methods are available for use.

The Townsville workshop provided additional support and momentum for a core 'SSD research development group' (comprised of Professor David Fox, Dr Graeme Batley, Dr Rick van Dam, and Dr Rebecca Fisher) to progress discussions with their Canadian counterparts working with the British Columbia (BC) Ministry of Environment and Climate Change Strategy. Accordingly, a proposal was developed by this group which sought funds from the Australian Government Department of Agriculture, Water and the Environment (DAWE) to enable Fox to participate in a 3-day workshop in Victoria, BC in late November 2019. Other workshop participants included Angeline Tillmanns (BC Environment) and consultants Joe Thorley and Carl Schwarz. Graeme Batley, Rick van Dam, Rebecca Fisher and Doug Spry and Kathleen McTavish (Environment and Climate Change Canada) participated in a video call each day for a briefing on workshop discussions.

Following the November 2019 workshop, the Canadian software tool developers implemented some of the improvements identified at the workshop while other issues remained to be addressed pending funding approvals and/or the need for further technical consideration. Subsequently, a work program was developed by the research and government collaborators to formally capture the essential and desirable tasks yet to be completed, which was to be used as the basis for further funding submissions. Additional funds were requested from DAWE (for the Australian consultants) and ECCC (for the Canadian consultants) to complete the essential technical work required to reach collective agreement between the Australian and Canadian experts that ssdtools Shiny app is robust and appropriate for the calculation of GVs in both jurisdictions.

This final report provides details and results of the numerous and extensive investigations into many aspects of SSD modelling and the software tools Burrlioz and (shiny)ssdtools. Importantly, the results of comprehensive assessments of the performance of both software applications under a variety of data and modelling scenarios are provided in support of this report's recommendation that the ssdtools R package and Shiny App be adopted for the calculation of GVs in Australia and New Zealand, as part of the Australian and New Zealand Guidelines for Fresh and Marine Water Quality (ANZG 2018). Recommendations arising from the assessment will be considered by the multijurisdictional governance arrangements (i.e., the Project Coordination Group and the Water Quality Policy Sub-Committee).

1.6 Task overview and summary

A listing of Project tasks together with a priority rating is shown in Table 1.

Table 1. Project tasks and priority.

Task	Priority
Distribution-Fitting	
Assess numerical instability issues with Burr III and decide whether to include Burr III	High
Determine convergence criteria	High
Identify benchmark datasets	High
Evaluate SSD methodologies based on datasets	High
Refine mixture modelling with view to incorporating	Low
HCx and CI estimation	
CI for HCx methods – alternatives to bootstrapping	Low
Initialisation of default distributions	
Finalise default distributions	High

The notional effort allocations assigned to each task in Table 1 (Appendix A) severely underestimated the actual amount of effort ultimately expended. This was a consequence of several factors. Most important among those were:

- Unanticipated effort associated with collating actual datasets and developing an R package (`ssddata`).
- Unanticipated effort associated with assisting with the development of a TMB version of `ssdtools`.
- Unanticipated effort associated with accessing and adapting the `Burrlioz` code to run entirely within the R environment.
- Additional effort associated with the exploration of numerous 'side-issues'.
- Underestimation of the time spent on individual tasks such as the exploration of the default distribution set and the generation of synthetic datasets for software evaluation.

2 DISTRIBUTION FITTING

2.1 Numerical instability of the *Burr III*

The *Burr III* family of distributions represents the foundation of the `Burrlioz` software, with the probability distribution function (*pdf*) given by:

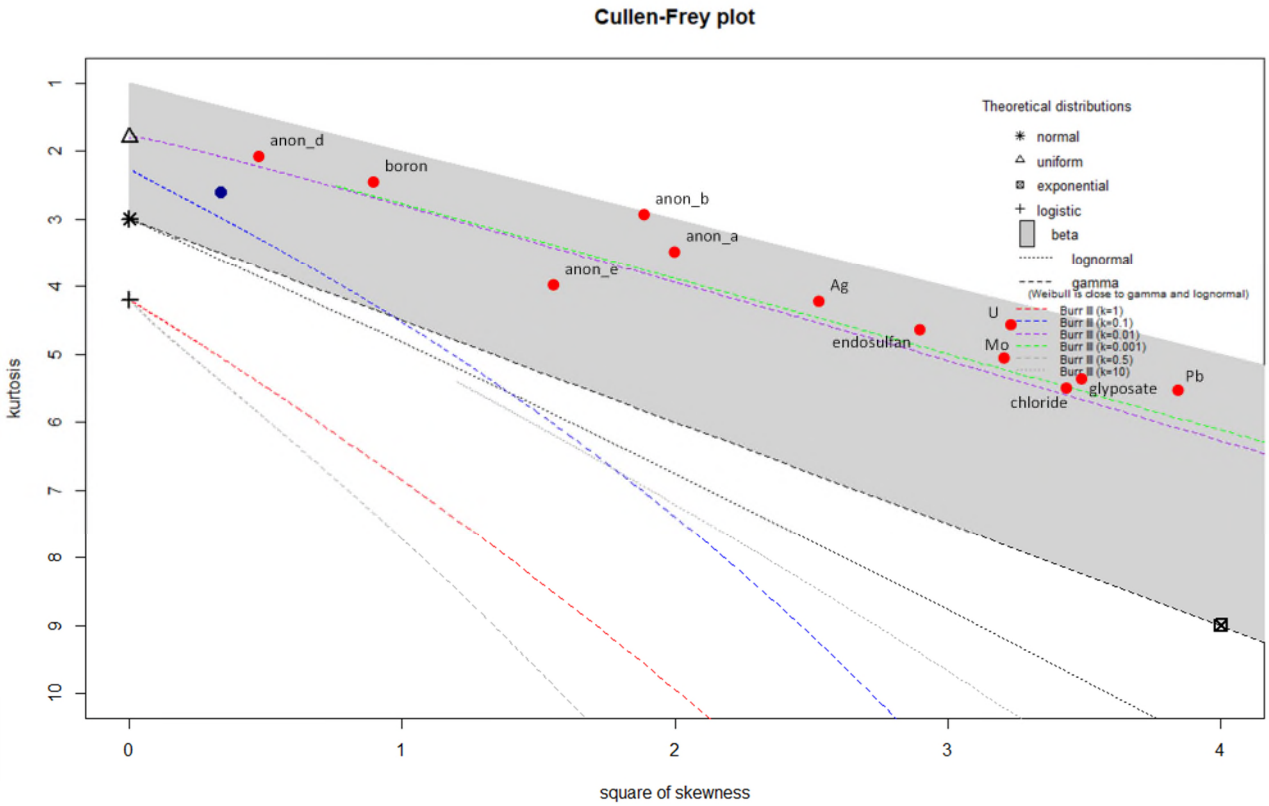
$$f_X\left(x;b,c,k\right)=\frac{bck\left(\frac{b}{x}\right)^{c-1}}{x^2\left[1+\left(\frac{b}{x}\right)^{c}\right]^{k+1}}\quad;\quad x,b,c,k>0 \tag{6}$$

The genesis of its adoption in Australia and New Zealand was detailed in section 1.2. Briefly, the Burr distribution was chosen because: (a) it covered a wide range of distributional shapes; (b) the widely used *log-logistic* distribution is a member of the Burr family; and (c) it has a closed-form expression for quantile determination. Figure 2 shows a Cullen and Frey (1999) plot which is a convenient way of representing the range of shapes associated with theoretical distributions via their skewness and kurtosis parameter values. The wide range of shapes admitted by the Burr family is clear from this plot as is the propensity for empirical benchmark distributions from `ssddata` (Section 2.4) to aggregate along the skewness-kurtosis contour for Burr III distributions having a *k* parameter value of 0.01. This latter point is picked up again in Section 4.4.3.

Over the years, users have periodically encountered numerical stability issues with the `Burrlioz` software. Indeed, our own preliminary investigations showed that `ssdtools` often failed to converge to a unique solution when the *Burr III* was included in the candidate distribution set. Initially this was thought to be a consequence of the very small sample sizes used in ecotoxicology. The `Burrlioz` software has several logical tests to help circumvent or 'trap' situations that would otherwise result in non-convergence.

It is well known (e.g., Tadikamalla (1980)) that the *Burr III* distribution is related to several other theoretical distributions. These include the *Lomax* distribution, the *compound Weibull*, the *Weibull-exponential*, the *logistic*, the *log-logistic*, the *Weibull*, and the *Kappa* family of distributions. Some of these relationships only exist as limiting cases of the *Burr III*, i.e., as one or more of the *Burr III* parameters approaches either zero or infinity. The `Burrlioz` software incorporates logic that aims to identify situations where parameter estimates are trying to converge to either very large or very small values. In such cases, fitting a *Burr III* distribution is abandoned and one of the limiting distributions is fitted instead. In addition to describing these rules, the following advice provided in the `Burrlioz` documentation also reveals the conditions under which the *inverse Weibull* and *inverse Pareto* distributions are fitted:

The 3 parameters of the Burr III distribution, b, c, and k are estimated by maximising the log-likelihood function (which is based on the probability distribution function). This maximisation is performed using the simplex algorithm, an optimisation technique that is not reliant on derivative information. A complication of the Burr III distribution is that, at limits of some of the parameters, the Burr III distribution tends to a limiting distribution. As k tends to infinity the Burr III distribution tends to the reciprocal Weibull distribution. As c tends to infinity the Burr III distribution tends to the reciprocal Pareto distribution. In practical terms, if the Burr III distribution is fitted and k is estimated to be greater than 100,



the estimation procedure is carried out again with a reciprocal Weibull distribution fitted. Similarly for the reciprocal Pareto distribution, if c is greater than 80. This is necessary to ensure numerical stability and does not have significant impacts on the results.

Figure 2. Plot of kurtosis versus skewness for a variety of theoretical probability distributions with empirical values for selected datasets from `ssddata` (see 2.4 **Benchmark datasets**).

Our investigations into stability and non-convergence issues with the *Burr III* distribution suggest that this is a consequence of one or more of the following situations: (i) a flat-likelihood profile; (ii) highly correlated parameter estimates (particularly for the *b* and *k* parameters); and/or (iii) a tendency for successive iterations of estimates of the *c* and *k* parameters to diverge whereby, simultaneously $c \to \infty$ and $k \to 0$. The last of these leads to difficulties with evaluation of the likelihood function which involves computations of the type $\left[1+\left(\frac{b}{x}\right)^{c}\right]^{k}$. Theoretically, the limiting value of this expression as $c \to \infty$ and $k \to 0$ is either 1 for $|b| < |x|$; 2 for $b = x$; and ∞ for $b > x$. So, in the first

two cases even though the theoretical limit is a small number, in practice the computations are done sequentially i.e., resulting in a very large number $\left[1+\left(\frac{b}{x}\right)^c\right]$ being raised to a very small power.

Kleiber and Kotz (2003) suggested that the numerical stability of the maximum likelihood estimation (MLE) algorithm in the case of *Burr III* and related distributions is also sensitive to extreme values and/or outliers in the sample data. This is a common characteristic of ecotoxicological datasets.

The cumulative distribution function (*cdf*) for the *Burr III* distribution is given by Equation 2.

$$F_X\left(x;b,c,k\right)=\frac{1}{\left[1+\left(\frac{b}{x}\right)^c\right]^k}\quad;\quad x,b,c,k>0 \tag{2}$$

As noted by Shao (2000), $\left[1+\left(\frac{b}{x}\right)^c\right]^k \approx \left[1+\left(\frac{b\delta}{x}\right)^c\right]^{\frac{k}{\delta^c}}$ holds for a small, positive constant δ implying $F_X\left(x;b\delta,c,\frac{k}{\delta^c}\right)\approx F_X\left(x;b,c,k\right)$. Also, $\lim_{\delta\to 0}\left\{F_X\left(x;b\delta,c,\frac{k}{\delta^c}\right)\right\}=e^{-k\left(\frac{b}{x}\right)^c}$ which is the *cdf* of the *inverse Weibull* distribution. Further, if we let $k=a^c$, the right-side of this limit becomes $e^{-\left(\frac{ab}{x}\right)^c}$ which is, in effect, a function of only two parameters $\{\lambda,c\}$ where $\lambda=ab$. In other words, under the conditions described, $\{a,b,c\}$ are not all estimable and hence a unique MLE does not exist.

In Section 3.2 we derive mathematical expressions for the standard errors and covariances for the *Burr III* MLEs as well as for an HCx estimated from the fitted distribution and these can be used with empirical data to demonstrate the high degree of correlation between estimates of *b* and *k*.

2.2 Numerical properties of the *inverse Pareto* distribution

As noted in Section 2.1, one of the limiting forms of the *Burr III* distribution is the *inverse Pareto* distribution. The *inverse Pareto* distribution is somewhat unusual in that its theoretical *moments* (mean, variance, skewness, kurtosis, etc.) do not necessarily exist. This gives rise to the paradoxical situation whereby *sample moments* can be computed but they have no connection to the parent distribution. A further complication with the *inverse Pareto* distribution is that there are two distinct versions of its mathematical description – variously referred to as the 'American' and 'European' representations. The difference is important, not least because the *support* for each is different (i.e., the range of 'permissible' values), as are the MLEs. The probability density function (*pdf*) and *cdf* for the two versions of the *inverse Pareto* distribution are given by the equations below.

'American' representation:

$$f_X\left(x;\alpha,\beta\right)=\alpha\beta^\alpha x^{\alpha-1};\quad 0\le x\le\tfrac{1}{\beta};\quad \alpha,\beta>0 \tag{3a}$$

$$F_X\left(x;\alpha,\beta\right)=\left(x\beta\right)^\alpha;\quad 0\le x\le\tfrac{1}{\beta};\quad \alpha,\beta>0 \tag{3b}$$

'European' representation:

$$f_Y(y;a,b)=\frac{a\,b\,y^{a-1}}{(y+b)^{a+1}};\ \ \mathrm{y}>0\ ;\ \ \mathrm{a},b>0 \tag{4a}$$

$$F_Y(y;a,b)=\left(\frac{y}{y+b}\right)^{a};\ \ \mathrm{y}>0;\ \ \mathrm{a,b}>0 \tag{4b}$$

While there is no closed-form solution for the MLEs for *a* and *b* in European version of the *inverse Pareto* distribution (Equation 4a), a closed-form solution for the American representation does exist (Equations 5a and 5b) although `Burrlioz` (somewhat inefficiently) uses numerical optimisation procedures instead to determine the MLEs for α and β .

$$\hat{\alpha}=\left[\ln\left(\frac{g_X}{\hat{\beta}}\right)\right]^{-1} \tag{5a}$$

$$\hat{\beta}=\frac{1}{\max\{X_i\}} \tag{5b}$$

where g_X is the *geometric mean* of the *X*-data.

As it currently stands, `Burrlioz` *only* fits an *inverse Pareto* distribution when the iterative-fitting process for a *Burr III* indicates that $k\to 0$ and $c\to\infty$ in Equation 1 in such a way that the product $kc=\alpha$ is constant. Under these conditions it can be shown that the *inverse Pareto* distribution given by Equation 3b with $\alpha=kc$ and $\beta=\frac{1}{b}$ is the limiting form of the *Burr III* distribution given by Equation 1 and as such no decision is required as to whether to fit the American or European version of the *inverse Pareto* distribution. However, *if* the *inverse Pareto* distribution was to be offered as one of the candidate distributions selected by the user, then further testing and analysis would be required. This is because the differences between the two versions of the *inverse Pareto* distribution have important practical implications. Although both are *inverse Pareto* distributions, the *pdfs* given by Equations 3a and 4a are fundamentally different: one is bounded, the other is not. For a given dataset we will not know which of these is the 'true' underlying distribution and one cannot arbitrarily choose between the two. To see this, consider the following data:

1.21 0.66 1.40 107.49 1.23 0.49 1.51 2.51 2.49 3.10 1.06 3.65 0.54 3.37 1.14 0.67
1.38 1.66 6.65 30.48 0.4716.31 0.46 0.44 1.62 0.42 2.10 5.28 1.25 0.48 14.17 0.63

`Burrlioz` fits an *inverse Pareto* to these data with parameter estimates $\hat{\alpha}=0.3407$ and $\hat{\beta}=0.0328$ on the assumption the data follow the distribution given by Equation (3a) whereas the MLEs for this distribution (Equations 5a and 5b) are $\hat{\alpha}=0.189$ and $\hat{\beta}=0.0093$. Either way, it is evident that the American version of the *inverse Pareto* distribution provides a poor fit to these data while the European version provides a reasonably good fit (Figure 3). The difference in the quantile estimates is, not surprisingly, very pronounced (Table 2).

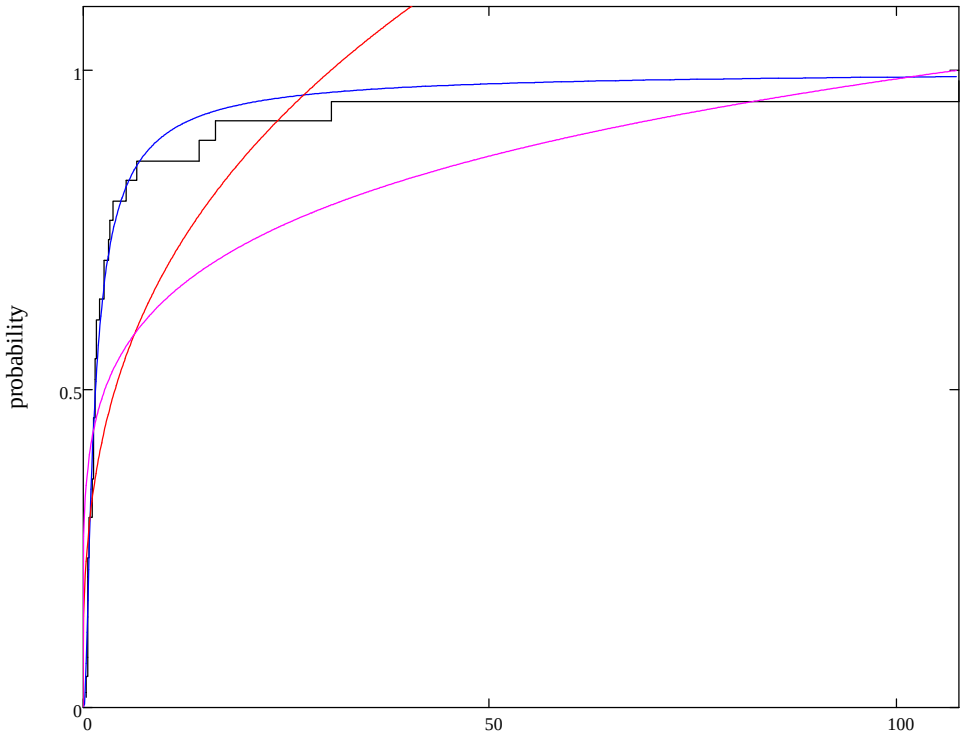


Figure 3. Empirical *cdf* (black line) and fitted American *inverse Pareto* using `Burrlioz` parameter estimates (red line); fitted American *inverse Pareto* with correct MLEs (purple line); and fitted European *inverse Pareto* with MLEs (blue line).

Table 2. Comparison of quantiles from *inverse Pareto* distributions.

Inverse Pareto Distribution	Quantile					
	0.01	**0.05**	**0.1**	**0.2**	**0.95**	**0.99**
US(Burrlioz)	0.000	0.005	0.035	0.271	26.227	29.602
US(MLE)	0.000	0.000	0.001	0.022	81.945	101.924
European(MLE)	0.233	0.358	0.466	0.667	20.936	106.848
Empirical	0.425	0.452	0.467	0.555	22.685	105.103

Again, although no general conclusions can be drawn from this example, it nevertheless provides an example of the failure of `Burrlioz` to correctly estimate the parameters of the (American) *inverse Pareto* distribution as well as demonstrating the care that needs to be exercised in identifying the 'correct' version of the *inverse Pareto* distribution if it is to be used as a candidate distribution for SSD modelling.

Recommendation: corrections to `Burrlioz` 2.0

The current *maximum likelihood estimation* routines for the *inverse Pareto* distribution used by `Burrlioz` do not always yield the correct values and should be updated with correct versions for all future releases of the software (and/or the equivalent R implementation).

Suggestion for further investigation: *inverse Pareto* distributions

> The addition of both versions of the *inverse Pareto* distribution to the candidate distribution list should be further explored to ascertain the desirability and utility of this option in the context of SSD modelling.

The results of supplementary investigations into various computational aspects associated with fitting the *inverse Pareto* distribution are provided in Appendix B.

2.2.1 Unbiased parameter estimation for the *inverse Pareto* distribution

Maximum likelihood estimation (MLE) is the adopted estimation strategy in both `Burrlioz` and `ssdtools` and while the MLEs have several desirable statistical properties, they are *not* guaranteed to be unbiased. The property of unbiasedness is a 'first-order' consideration in statistical estimation – statisticians invariably narrow their search for 'optimal' estimators (i.e., functions of the data) among the class of unbiased functions of the data. Simply put, this means that, in repeated sampling, the estimator neither consistently over nor underestimates the true parameter value.

Additional investigations were undertaken to examine the bias properties of the MLEs for the *inverse Pareto* distribution.

We show in Appendix C that the MLEs obtained using Equations 5a and 5b are biased estimators for the respective parameters although the modified estimators $\left\{\hat{\alpha}^*;\hat{\beta}^*\right\}$ are unbiased:

$$\hat{\alpha}^* = \frac{n-2}{n}\hat{\alpha} \tag{6a}$$

$$\hat{\beta}^* = \left[1-\frac{1}{n\hat{\alpha}}\right]\hat{\beta} \tag{6b}$$

where $\hat{\alpha}$ and $\hat{\beta}$ are given by Equations 5a and 5b respectively.

2.2.2 HCx estimates from the *inverse Pareto* distribution

Computation of the p^{th} percentile for an *inverse Pareto* is particularly easy in view of its *cdf* having a closed-form expression. The computational formulae for both the European and US representations are given by Equations 7a and 7b respectively.

$$\widehat{\xi_p} = \frac{\hat{b}}{\left(\frac{1}{p}\right)^{\frac{1}{\hat{a}}}-1} \tag{7a}$$

$$\widehat{\psi_p} = \frac{1}{\hat{\beta}}p^{\frac{1}{\hat{\alpha}}} \tag{7b}$$

The RBurrlioz program (see Replicating `Burrlioz 2.0` in R for simulation studies) program can be used to fit an *inverse Pareto* distribution to a set of data $\{y_1,\ldots,y_n\}$ with the command:

```
> y.fit <- RBurrlioz::fit(dataframe = y,ldensity = inverse.Pareto.density)
```

And the HCx estimated using:

```
> RBurrlioz::fit(y.fit,p=x)
```

This will yield correct estimates of the HCx provided the *y*-data follow the *inverse Pareto* distribution whose *pdf* is given by Equation 4a. If the *y*-data come from an *inverse Pareto* distribution having *pdf* given by Equation 3a, these HCx estimates will be grossly in error. However, `Burrlioz` can still be used to derive correct HCx estimates after a suitable transformation of the data. The procedure is as follows:

Step 1: Convert the *y*-data $\{y_1,\ldots,y_n\}$ to *z*-data $\{z_1,\ldots,z_n\}$ using:

$$z_i = \frac{1}{b}\left(\frac{y_i}{y_i + b}\right)$$

where *b* is the scale parameter in Equation 3a.

Step 2: Use `RBurrlioz` to fit the *inverse Pareto* to the *z*-data and compute the x^{th} percentile:

```
> z.fit <- RBurrlioz::fit(dataframe = z,ldensity = inverse.Pareto.density)
> hc.z <- RBurrlioz::fit(z.fit,p=x)
```

Step 3: Obtain the x^{th} percentile for the *Y*-data as:

```
> hc.y <- b^2/(1/hc.z-b)
```

2.2.3 Bootstrap alternative for HCx confidence intervals from the *inverse Pareto* distribution

It is shown in Appendix B that $\hat{\alpha}' = \frac{1}{\hat{\alpha}}$ has a $gamma(r, \lambda)$ distribution and the quantity $\left(\hat{\beta}' - \ln \beta\right)$ has a *negative exponential*(λ) distribution where $\lambda = n\alpha$; $r = n - 1$; and $\hat{\beta}' = \ln\left(\hat{\beta}\right)$.

Furthermore, $\hat{\alpha}'$ and $\hat{\beta}'$ are statistically independent. These results can be used as a quick, alternative means of providing interval estimates for an HC*x* for *inverse Pareto* distributed data. In R, this is achieved with the following simple function:

Box 1. R code for rapid computation of HCx confidence intervals from the inverse Pareto distribution

```
hc.ip <- function(N,n,shape.est,scale.est) {
  #  N is the number of simulated {shape,scale} parameters for the CI
estimation;
  #  n is the size of the sample from whch the shape.est and scale.est
parameter estimates were obtained

  p <- list(0.01,0.05,0.1,0.2)
  r <- n-1 ; l <- n * shape.est
  a <- rgamma(N,r,l)
  b <- rexp(N,l) + log(scale.est)
  hc.dat <- lapply(p,function(x) (x^a)/exp(b))
  hc.est <- unlist(lapply(hc.dat,median))
  hc.se <-  unlist(lapply(hc.dat,sd))
  hc.lwr <- unlist(lapply(hc.dat,function(y) quantile(y,0.025)))
  hc.upr <- unlist(lapply(hc.dat,function(y) quantile(y,0.975)))
  hc.out <- data.frame(est=hc.est,se=hc.se,lcl=hc.lwr,ucl=hc.upr)
  rownames(hc.out) <- c("HC1","HC5","HC10","HC20")
  return(round(hc.out,4))
}
```

Example

Consider the following data generated using `ssdtools` with shape = 1.2 and scale = 5.4:

`2.309 2.897 0.564 0.455 4.746 2.390 3.437 1.013 4.027 3.858 4.228 3.215`

The *true* HC1, 5, 10, and 20 values are: $\{0.143,\ 0.487,\ 0.825,\ 1.397\}$. The shape and scale parameter estimates from `Burrlioz` are: $\{\hat{\alpha} = 1.316; \hat{\beta} = 0.2107\}$ and from `ssdtools` are: $\{\hat{\alpha} = 1.2117; \hat{\beta} = 5.0668\}$. *NB: There is an inverse relationship between the scale estimates from* `Burrlioz` *and the original implmenetation in* `ssdtools`*, although this is now aligned in the most recent version.*

A comparison of point and interval estimates obtained using the procedure outlined in this Section and `ssdtools` is provided in Table 3.

Table 3. Comparison point and interval estimates (lower (LCL) and upper (UCL) 95% confidence limits) for 1, 5, 10 and 20 HC values using bootstrapping (parametric and non-parametric) in `ssdtools` and the method outlined in this Section. All calculations based on n=10,000. Note that in all cases the true HC was captured by the CI.

HC	Actual	Estimate		LCL	UCL	Width of ci	Method
1	0.143	0.113	0.247	0.016	0.938	0.92	non-parametric
		0.113	0.193	0.010	0.717	0.71	parametric
		0.196	0.231	0.020	0.878	0.86	this report
5	0.487	0.428	0.400	0.117	1.640	1.52	non-parametric
		0.428	0.350	0.090	1.410	1.32	parametric
		0.584	0.376	0.133	1.561	1.43	this report
10	0.825	0.758	0.473	0.277	2.090	1.81	non-parametric
		0.758	0.436	0.229	1.880	1.65	parametric
		0.933	0.445	0.296	1.999	1.70	this report
20	1.397	1.340	0.522	0.657	2.670	2.01	non-parametric
		1.340	0.507	0.589	2.520	1.93	parametric
		1.497	0.491	0.666	2.555	1.89	this report

A somewhat more comprehensive simulation was undertaken to investigate the efficiency of the proposed methodology. To this end, 1,000 datasets of various sample sizes were generated from an *inverse Pareto* distribution with parameters $\alpha = 0.5989$ and $\beta = 0.2631$. The theoretical *cdf* for this particular distribution is shown in Figure 4.

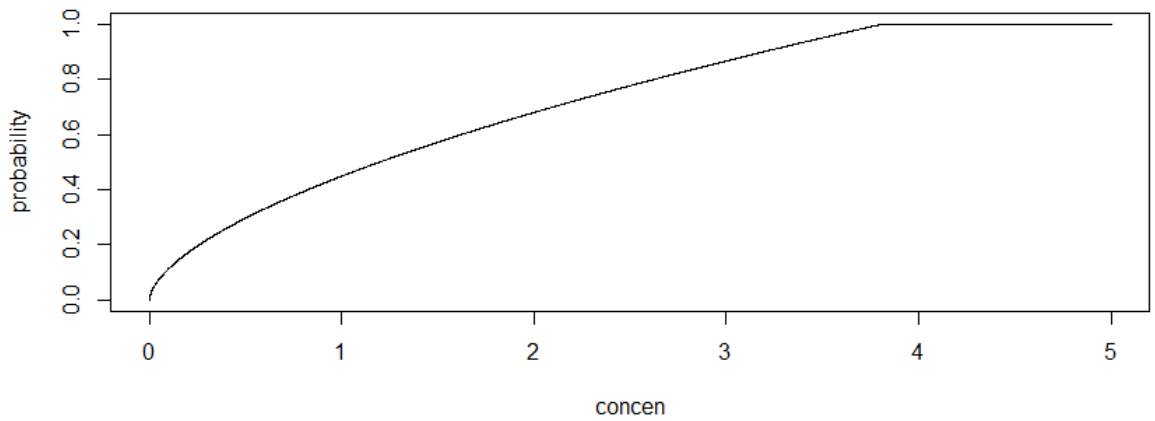


Figure 4. The theoretical *cdf* for the *inverse Pareto* distribution with parameters α = 0.5989 and β = 0.2631.

An *inverse Pareto* distribution was fitted to each of the 1,000 datasets and 1,000 non-parametric bootstrap samples used to obtain estimated 95% confidence limits. A logical variable was created to indicate whether or not the *true* HC value had been captured by the 95% confidence interval. The proportion of cases for which this logical variable was true provided a measure of the *actual* coverage associated with the *nominal* 95% CI. The results from the `Burrlioz` fit were compared to those obtained using the 'new' procedure given in Box 1.

In terms of coverage, both `Burrlioz` and our new procedure gave almost identical results and were all very close to the nominal 95% level (Table 4).

Table 4. Actual coverage estimates for `Burrlioz` and compared to the new method based on 1000 simulated datasets of n=100 from the theoretical *inverse Pareto* distribution (Figure 4). Values are the proportion of simulated outcomes where the true value falls within the estimated 95% confidence interval.

	Method	
	`Burrlioz`	**New methods of this Section**
HC1	0.950	0.949
HC5	0.953	0.953
H10	0.956	0.951
HC20	0.961	0.954

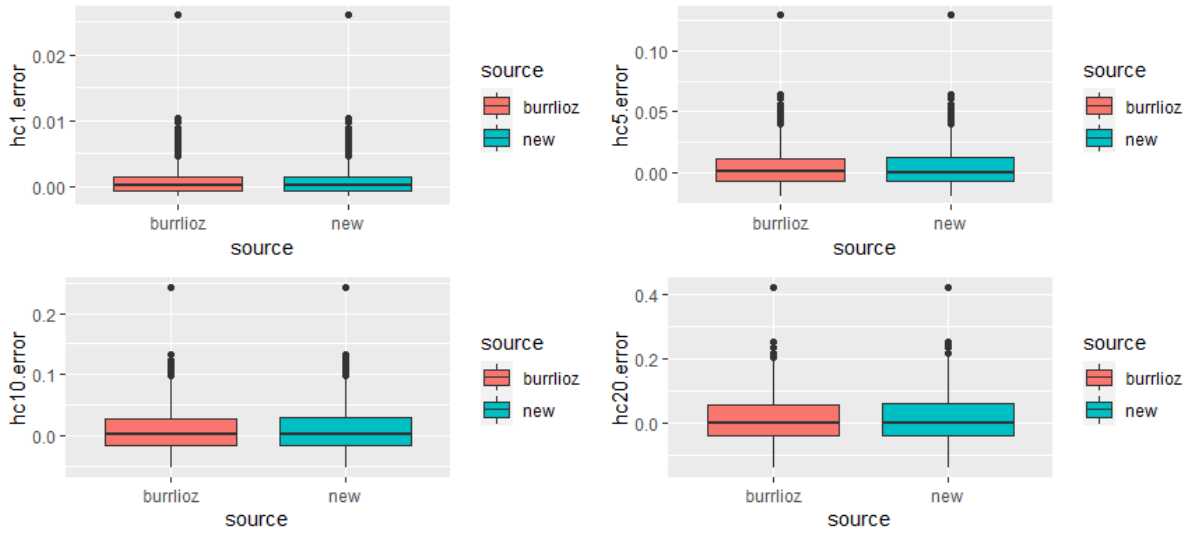


Figure 5. Boxplots of HC errors for estimates obtained using `Burrlioz` and new method of this Section for 1000 samples of size *n=100* sampled from the theoretical *inverse Pareto* distribution in Figure 4. HC error is calculated as the difference between the estimated HCx and the theoretical HCx.

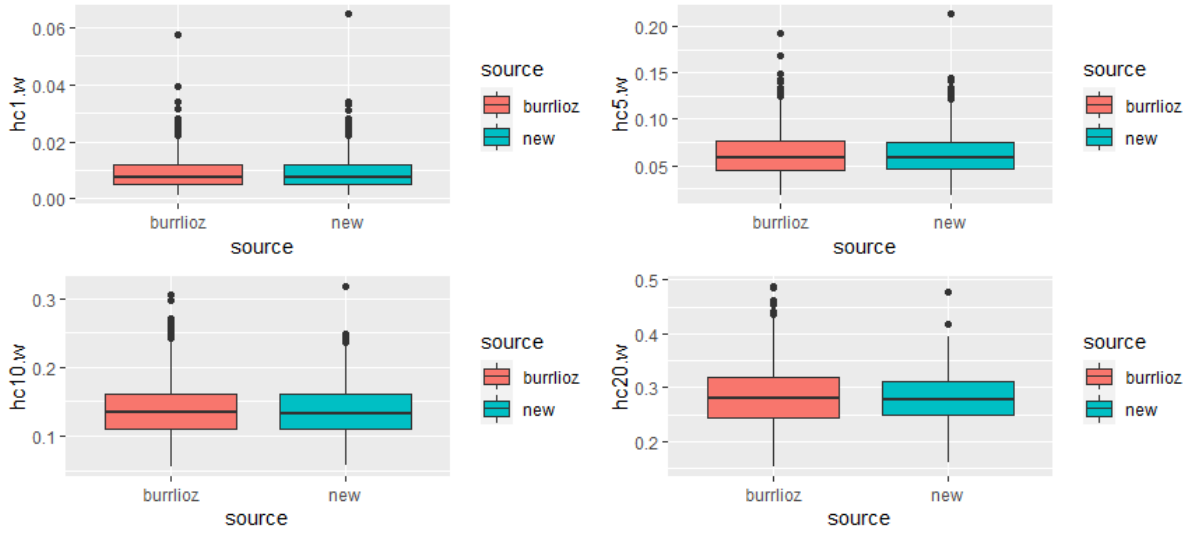


Figure 6. Boxplots of HC 95% CI widths using `Burrlioz` and new method of this Section for 1000 samples of size *n=100* sampled from the theoretical *inverse Pareto* distribution in Figure 4.

Based on this limited analysis, it would appear that the procedure outlined in this Section performs as well as the bootstrapping methods. It yields confidence intervals that achieve the same result but in a fraction of the time taken for bootstrapping (typically 700 to 1,000 times faster) and has interval widths which are between non-parametric and parametric bootstrapped intervals. However, further testing and evaluation would be required before the procedure is considered for inclusion in any software tools.

Suggestion for further investigation: unbiased HCx estimation for the *inverse Pareto*

> Additional testing be undertaken to evaluate the alternative procedure more fully for obtaining point and interval estimates of HCx values in the *inverse Pareto* case.

2.3 `ssdtools` code optimisation, convergence criteria and software enhancements

To facilitate the research program, several modifications and enhancements were made to `ssdtools`. In particular, the `fitdistrplus` package (Delignette-Muller & Dutang 2015) was replaced by TMB (Kristensen et al. 2016) for model fitting to allow more control over model specification and transparency regarding convergence criteria. This in turn allowed us to better assess numerical instability issues with the *Burr III* distribution and to readily implement two mixture distributions (*log-normal-log-normal* and *log-logisitic log-logistic*). It also facilitated implementation of the *inverse Pareto* distribution and development of a function to mimic `Burrlioz`. Confidence interval estimation was facilitated by recording the proportion of bootstrap samples that were successfully fitted to the generative distribution and by adding the option to perform non-parametric bootstrap resampling (as is implemented in `Burrlioz`). Other functionality of note is the ability to automatically rescale data (by dividing by the maximum value) to aid model fitting while returning HCx

and CI values that are automatically adjusted for the rescaling. A comprehensive list of additions and modifications is provided in Table 5 below.

Table 5. List of additions and modifications to `ssdtools`

Item	Outcome
Model fitting	`fitdisrplus` has been replaced with TMB in the refactored code. This has improved handling of censored data and allows greater flexibility for defining constraints on parameters.
Convergence criteria	In addition to requiring the optim function to converge, the following two additional arguments have been added: 1. Bounded parameters are not at boundary (logical) 2. Standard errors for parameter estimates are computable (logical)
Data censoring	Increased flexibility of specifying censored data by: 1. Having a finite value for the left column that is smaller than the finite value in the right column (interval censored) 2. Having a zero or missing value for the left column and a finite value for the right column (left censored) 3. Confidence intervals cannot be estimated for interval censored data.
Akaike weights for model averaging with censored data	Akaike Weights are calculated using AIC for censored data (as the sample size cannot be estimated) but only if all the distributions have the same number of parameters (to ensure the weights are valid)
Distributions	1. Density functions are now defined in C++ as TMB templates (previously, they were exported as R functions to make them accessible to `fitdistrplus`). The distribution, quantile and random generation functions are more generally useful and are still exported but are now prefixed by `ssd_` to prevent clashes with existing functions in other packages. Thus for example `plnorm()`, `qlnorm()` and `rlnorm()` have been renamed `ssd_plnorm()`, `ssd_qlnorm()` and `ssd_rlnorm()` 2. The following distributions were added (or in the case of `*Burr III*` re-added) to the new version - `burrIII3` - *Burr III* three parameter distribution - `invpareto` - *inverse Pareto* (with bias correction in scale order statistic) - `lnorm_lnorm` *log-normal*/*log-normal* mixture distribution - `llogis_llogis` *log-logistic*/*log-logistic* mixture distribution 3. The following arguments were added to `ssd_fit_dists()` - `rescale` (default set to `FALSE`) to specify whether to rescale concentrations values by dividing by the largest (finite) value. This alters the parameter estimates, which can help some distributions converge, but not the estimates of the hazard concentrations/protections. This is akin to converting units between Imperial and metric units and so is mathematically equivalent to the unscaled distribution. - `reweight` (default set to `FALSE`) to specify whether to reweight data points by dividing by the largest weight. - `at_boundary_ok` (default set to `FALSE`) to specifying whether a distribution with one or more parameters at a boundary is considered to have converged. - `min_pmix` (default set to 0.01) to specify the boundary for the minimum proportion for a mixture distribution. - `range_shape1` (default set to `c(0.05, 20)`) to specify the lower and upper boundaries for the shape1 parameter of the *Burr III* distribution. - `range_shape2` (default set to the same as `range_shape2`) to specify the lower and upper boundaries for the shape2 parameter of the *Burr III* distribution. - `control` (default set to an empty list) to pass a list of control parameters to `stats::optim()`. 4. the default value of

Item	Outcome
	- `computable` argument was switched from `FALSE` to `TRUE` to enforce stricter requirements on convergence (see above).
Subsets of distributions	1. Functions were added to handle multiple distributions, including: - `ssd_dists()` to specify subsets of the available distributions (using its `type` argument). - `delta` argument (default set to 7 based on (Burnham & Anderson 2002)) to the `subset()` generic to only keep those distributions within the specified AIC(c) difference of the best supported distribution. This was done to save computation time in calculation of HCx bootstrap confidence values by avoiding the need to bootstrap distributions of very low weight. 2. The function `ssd_fit_burrlioz()` was added to approximate the behaviour of [Burrlioz](https://research.csiro.au/software/burrlioz/).
Hazard concentration/ protection estimation	Hazard concentration estimation is performed by `ssd_hc()` (which is wrapped by `predict()`) and hazard protection estimation by `ssd_hp()`. By default, confidence intervals are estimated by parametric bootstrapping. To reduce the time required for bootstrapping, parallelization was implemented using the [future](https://github.com/HenrikBengtsson/future) package. The following arguments were added to `ssd_hc()` and `ssd_hp()`: - `delta` (by default 7) to only keep those distributions within the specified AIC difference of the best supported distribution. - `min_pboot` (by default 0.99) to specify minimum proportion of bootstrap samples that must successfully fit. - `parametric` (by default `TRUE`) to allow non-parametric bootstrapping. - `control` (by default an empty list) to pass a list of control parameters to `stats::optim()`. The following columns were added to the output data frame: - `wt` to specify the Akaike weight. - `nboot` to indicate how many bootstrap samples were used. - `method` to indicate whether parametric or non-parametric bootstrap was used.
Goodness of fit	The `pvalue` argument (by default `FALSE`) was added to `ssd_gof()` to specify whether to return p-values for the test statistics as opposed to the test statistics themselves.
Plotting	- `ssd_plot_data()` to plot censored and uncensored data by calling `geom_ssdpoint()` for the left and for the right column (alpha parameter values should be adjusted accordingly). - `geom_ssdsegment()` to allow plotting of the range of a censored data points using segments. - `scale_colour_ssd()` (and `scale_color_ssd()`) to provide an 8 color-blind scale. - added `bounds` (by default `c(left = 1, right = 1)`) argument specify how many orders of magnitude to extend the plot beyond the minimum and maximum (non-missing) values. - added `linetype` (by default `NULL`) argument to specify line type. - added `linecolor` (by default `NULL`) argument to specify line color. - changed default value of `ylab` from "Percent of Species Affected" to "Species Affected". Renamed - `GeomSsd` to `GeomSsdpoint`. - `StatSsd` to `StatSsdpoint` Soft-deprecated - `geom_ssd()` for `geom_ssdpoint()`. - `stat_ssd()`. - `ssd_plot_cf()` for `fitdistrplus::descdist()`.
Data	1. The dataset `boron_data` was renamed `ccme_boron` and moved to the [`ssddata`](https://github.com/open-AIMS/ssddata) R package together with the other CCME datasets. The `ssddata` package provides a suite of datasets for testing and comparing SSD fitting software. 2. Some data handling functions have been added: - `ssd_data()` to return original data for a `fitdists` object.

Item	Outcome
	- `ssd_ecd_data()` to get empirical cumulative density for data. - `ssd_sort_data()` to sort data by empirical cumulative density.
Miscellaneous	- `npars()` now orders by distribution name. - All functions and arguments that were soft-deprecated prior to v0.3.0 now warn unconditionally. - Implemented the following generics for `fitdists` objects - `glance()` to get the model likelihoods, information-theoretic criteria etc. - `augment()` to return original dataset. - `logLik()` to return the log-likelihood. - `summary.fitdists()` to summarize.

2.4 Benchmark datasets

A key deliverable of this project was to identify a suite of benchmark datasets that will serve as a reference standard for SSD-fitting and evaluation. This is in keeping with the recommendation by Fox et al. (2021) and is useful for testing packages during development and evaluating the outcome of altering the fitting methodologies. Such data can be used not only in testing and evaluation, but more particularly to identify if any software updates have a measurable impact on the expected outputs. In addition, real case studies of SSDs for which `Burrlioz 2.0` results are available can be used to evaluate differences among fitting methods relative to the current Water Quality Guideline methodology (although rigorous testing of method outcomes must also be done using simulated data for which true values are known because this allows assessment of bias and coverage, see Section 2.5.2 below), which cannot be achieved with real datasets.

We have built `ssddata` (Fisher & Thorley 2021) as a standalone R package to house the set of benchmark datasets, with the intent of becoming a package dependency to `ssdtools` and other related packages. The new package is housed on github at https://github.com/open-aims/ssddata and is published on CRAN at https://CRAN.R-project.org/package=ssddata. The package includes a range of datasets sourced from the Canadian Council of Ministers of the Environment (CCME), the Australian Institute of Marine Science (AIMS), the Commonwealth Scientific and Industrial Research Organisation (CSIRO), and the Australian and New Zealand water quality guidelines website (ANZG), as well as anonymous datasets supplied by DAWE and other parties. The source of each dataset is indicated using a lower-case pre-fix in the data name (e.g., ccme, aims, etc), with the chemical name following (e.g., ccme_boron).

During development, substantial R code was written to provide a workflow that allowed efficient documentation of the individual datasets within the R package environment, thus making it possible to add additional datasets with minimal effort. The package documentation is hosted on the GitHub site at (https://open-aims.github.io/ssddata/), which shows instructions for package installation on the home page, and documentation for all the constituent datasets on the Reference page. The package is now published on CRAN, so can be included as a dependency to `ssdtools`. Please see the relevant Reference pages for the CCME, AIMS, CSIRO, ANZG and anonymous datasets for more details on each of the individual datasets available.

Included in this data package is a dataset containing various software fits to these data that can be used for comparison purposes (see https://open-aims.github.io/ssddata/reference/ssd_fits.html), that includes HC estimates for data contained within the package. These fits were obtained either as an entire dataset for a given chemical from a data source (e.g., ccme_boron) using a geometric mean where a dataset contains multiple species, or based on a subset using some kind of filter. We created a helper function `get_ssddata` that can be used to generate the geometric mean, given the dataset_name and a filter_val. Most commonly this filter was applied to subset data into either a tropical or temperate Domain. As an example, the `aims_aluminium_marine` dataset contains replicate measurements for some species, as there were data for both Temperate and Tropical subsets, thus for an overall SSD their geometric mean must be computed and a single value included in the final SSD to replicate the results (van Dam et al. 2018). Likewise, the source of this same dataset also provides SSD results from `Burrlioz 2.0` that were computed using just the tropical and temperate data, and both are included in the collated `ssd_fits` dataset.

At present, the `ssddata` package contains estimates for 249 dataset/filter/PC combinations across 20 unique datasets, fitted using either `Burrlioz 2.0`, or the `fitdistrplus`-based version of `ssdtools` (version 0.3.4) using the default distribution set of *log-logistic*, *log-normal* and *gamma*. Many of these estimates include both the `Burrlioz 2.0` or `ssdtools` estimates, as well as the lower and upper 95% confidence bounds based on bootstrapping.

2.5 Evaluating SSD methodologies

In this Section we fully explore, evaluate, and compare the Australian/New Zealand and the Canadian-BC SSD methodologies. This includes comparing the estimated outcomes of different methodologies based on benchmark datasets, as well as the accuracy and precision of HCx estimation using simulated data. Methods to compare include:

1. `BurrliOz` (the currently adopted method in Australia/New Zealand)
2. Model averaging using `ssdtools` (TMB-based) with the three current default distributions (*log-logistic*, *log-normal* and *gamma*; the currently adopted Canadian BC approach)
3. Model averaging using `ssdtools` (TMB-based) using different suites of possible distributions, including the three already adopted (*log-logistic*, *log-normal* and *gamma*), those currently used in `BurrliOz` (*Burr III*, *inverse Weibull* [named lgumbel in `ssdtools`], and *inverse pareto*), as well as the other two available in `ssdtools` (i.e.,*gompertz* and *Weibull*).
4. Model averaging using `ssdtools` (TMB-based) using all the above distributions as well as two mixture distributions (*llogis_llogis*, and *lnorm_lnorm*).
5. Model averaging using the current CRAN release version of `ssdtools` (based on `fitdistrplus`) with the three current default distributions (*log-logistic*, *log-normal* and *gamma*; the currently adopted Canadian BC approach).
6. An `ssdtools` (TMB-based) implementation of the `Burrlioz 2.0` fitting framework, using the `ssd_fit_burrlioz()` function.

These comparisons were performed across the common levels of species protection (i.e., 80, 90, 95 and 99% protection) and the influence of sample size on the performance and outcome of different methods was also investigated in the case of simulated datasets (see below).

2.5.1 Benchmark datasets

We used the `ssd_fit` dataset containing actual `Burrlioz` estimates of HC, to compare HC estimates between `Burrlioz 2.0` and the current CRAN version of `ssdtools`, using a model average of the current default distribution set from the BC Environment [shiny app] (https://bcgov-env.shinyapps.io/ssdtools/ *log-logistic*, *gamma* and *log-norm*). This version of `ssdtools` uses the R package `fitdistrplus` R package for fitting the individual distributions to the data, with model weights based on AICc.

2.5.1.1 `Burrlioz 2.0` versus `ssdtools` (fitdistrplus-based - default distributions)

Overall, there was very good agreement in the estimates among the two methods for the PC80/HC20 and PC90/HC10, with variation increasing for the smaller species protection values (95% and 99%, Figure 7). Estimates of the upper 95% confidence bands were relatively similar, but estimates for the lower bound varied widely, particularly for the two smaller protection values (Figure 7). Overall, there was little deviation from the 1:1 line for the two methods, indicating that the estimates from the two methods were on average very similar (Figure 7).

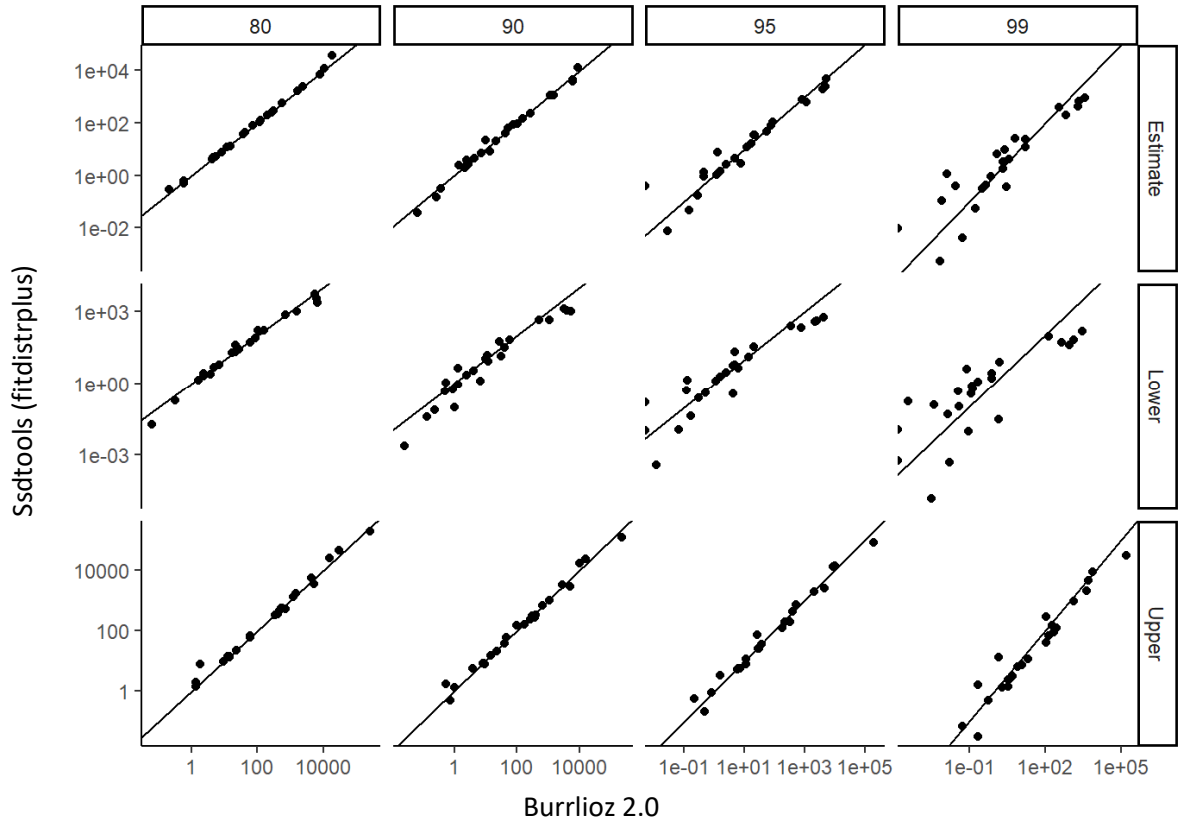


Figure 7. HC values estimated using the original `fitdistrplus` version of `ssdtools` with the default distribution set of *log-logistic*, *log-normal* and *gamma*, plotted against `Burrlioz 2.0`. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds. Points with the left half missing have an x value of 0.

2.5.1.2 `ssdtools` (fitdisrplus) versus `ssdtools` (TMB)

We also used the datasets in the `ssddata` R package to compare the original version of `ssdtools` based on `fitdistrplus` and the development version of `ssdtools` based on TMB. For these datasets, there is complete agreement among the two packages in terms of the estimate, as well as the confidence bands (Figure 8), and no further comparisons were done between these two platforms.

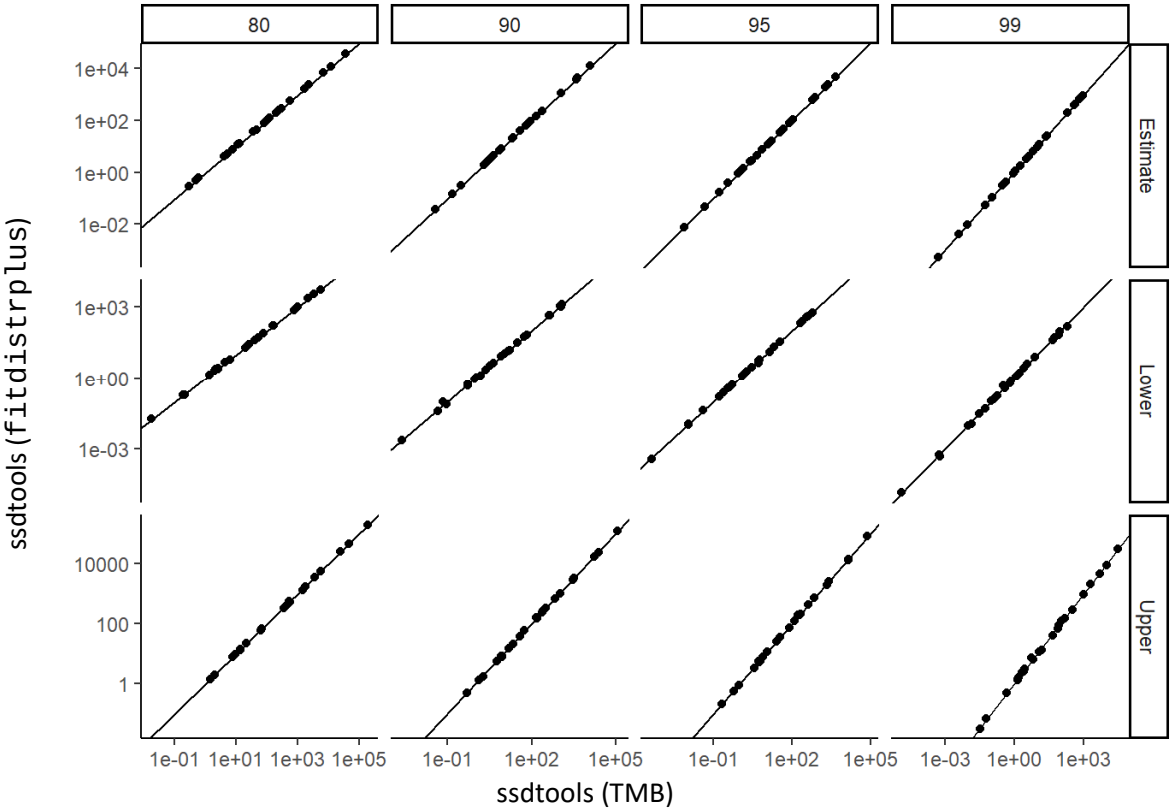


Figure 8. HC values estimated using the original `fitdistrplus` version of `ssdtools` with the default distribution set of *log-logistic*, *log-normal* and *gamma*, plotted against the new version of `ssdtools` based on TMB using the same distributions. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimates, as well as the lower and upper confidence bounds.

2.5.1.3 Comparing the default distributions to all distributions in `ssdtools`

The development version of `ssdtools` contains a large range of distributions, including the *Burr III* (burrIII3*)*, *gamma*, *Gompertz (*gompertz*), inverse Pareto (*invpareto*), inverse Weibull* (or *log-Gumbel*, lgumbel*)*, *log-logistic* (llogis*)*, *Weibull* (weibull), and *log-logistic* (llogis_llogis*)* and *log-normal* (lnorm_lnorm) mixtures. We compared estimates based on the current default set (*log-logistic*, *log-normal* and *gamma*), to estimates based on all distributions and observed overall good agreement, particularly for the 80th and 90th PC values (Figure 9). There was a tendency for estimates based on the 'all' set to show slightly higher PC values than the default distribution set for some datasets (Figure 9). This suggests that the choice of distributions in the set will have an impact on HC estimates and should be carefully considered using simulations studies (see Section 2.5.2).

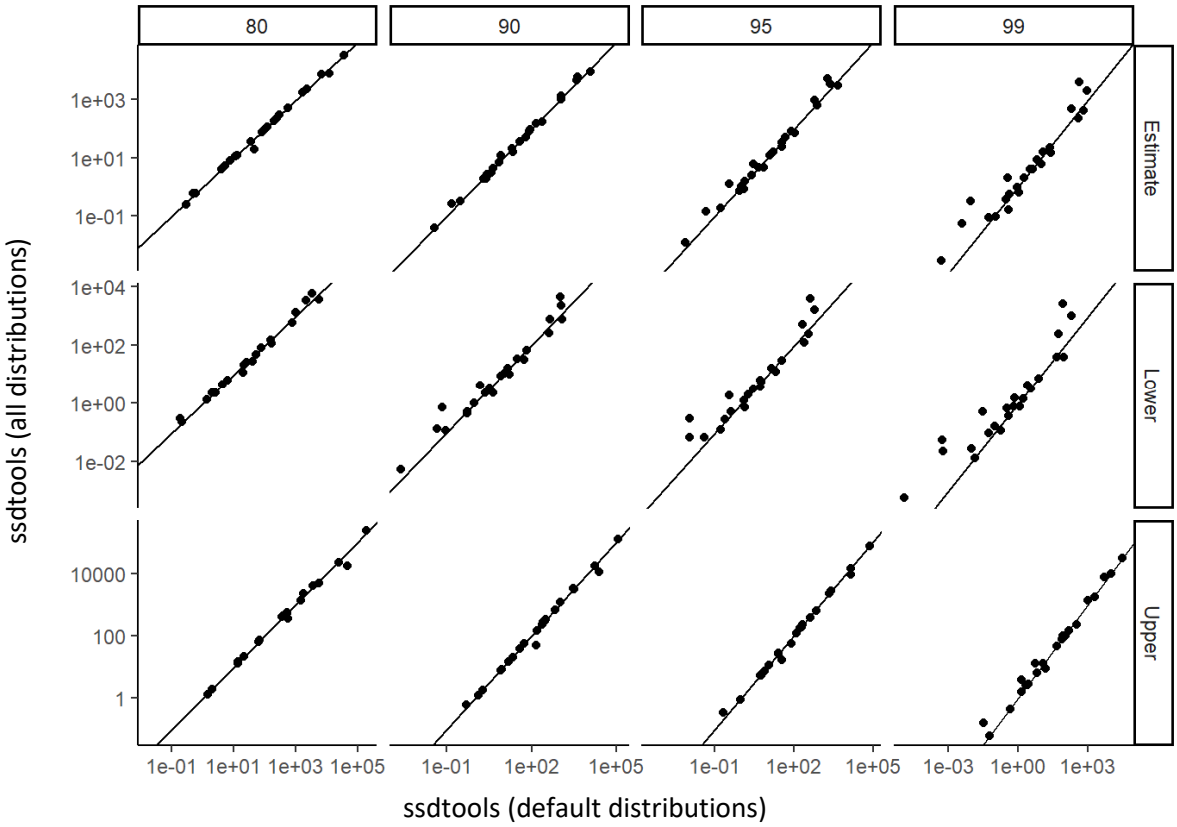


Figure 9. HC values estimated using the default distribution set of *log-logistic*, *log-normal* and *gamma*, plotted against the estimate based on all distributions available in the newest version of `ssdtools`. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds.

2.5.1.4 The `ssd_fit_burrlioz` function in `ssdtools`

The development version of `ssdtools` (TMB-based) includes an `ssd_fit_burrlioz()` function that is designed to mimic the analytical framework of `Burrlioz 2.0`, but using the `ssdtools` package (and therefore TMB for distribution estimation). We compared how well this tool performs in replicating the existing `Burrlioz 2.0` estimates using the benchmark datasets. Overall, we found very good agreement between the two methods; however, there are a few large outliers, suggesting the `ssd_fit_burrlioz()` function cannot be used to generate PC estimates as a direct replacement to `Burrlioz 2.0` (Figure 10). Without further, more detailed testing, we cannot say what is responsible for this situation, although we suspect that it may be a consequence of a very flat likelihood and that the two algorithms have converged to different, but equally good, solutions.

Suggestion for further investigation: `Burrlioz 2.0 comparisons`

> Undertake more detailed investigations to compare the performance of `Burrlioz 2.0` and the `ssd_fit_burrlioz` function in `ssdtools`. Focussing on instances where HCx estimates are substantially different, determine reasons for this discrepancy.

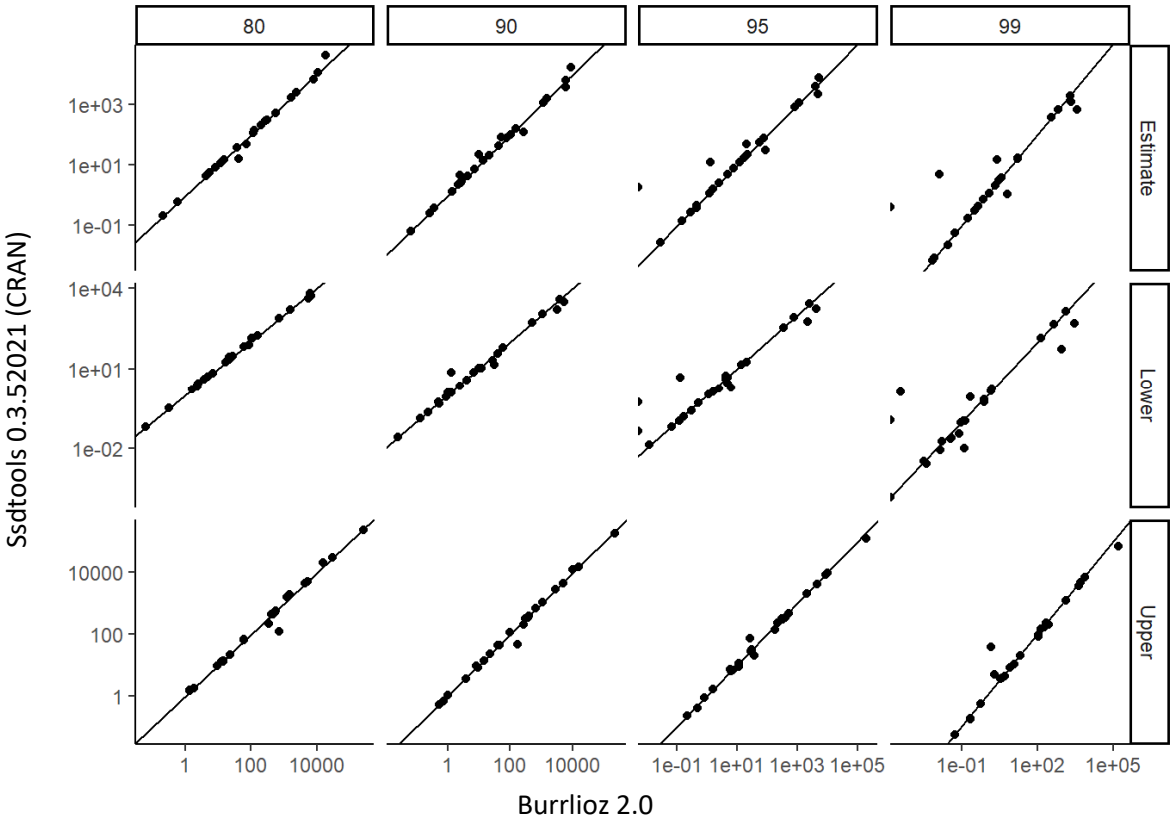


Figure 10. HC values estimated using `ssd_fit_burrlioz` plotted against `Burrlioz 2.0` estimates. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds. Points with the left half missing have an x value of 0.

2.5.2 Simulated datasets

Simulated datasets are an essential tool for evaluating alternative methods for estimating statistical distributions because they allow a comparison of the estimated HC values against known true values, which cannot be done using observed SSD datasets, such as those in the `ssddata` package (see Section 2.5.1). While the benchmark datasets can be used to compare the outcome of two fitting methods (as was done in Section 2.5.1), when the fits differ, there is no way to know which outcome is 'correct'. By simulating data with known HC characteristics, it is possible to estimate values of bias (estimated HC – true HC), as well as the observed coverage of the confidence interval estimate – i.e., does the true HC fall within the 95% confidence interval, 95% of the time? The problem with implementing simulation studies to examine the performance of `Burrlioz 2.0` against other methods are twofold: (i) the manual interface of `Burrlioz 2.0` precludes it's use in simulations, and (ii) the simulated data need to be generated by some underlying distribution. To address the first issue, we developed and validated an R only version of `Burrlioz 2.0`. To address the second, we ran simulations using two sets of distributions – the first included distributions that were contained within the methods being compared [Simulation Study 1 (*log-normal*, *log-logistic*, *inverse Weibull*)], and the second based on a family of distributions not included in any of the methods under consideration [Simulation Study 2 (Johnson family)]. These are described in more detail below.

Replicating `Burrlioz 2.0` in R for simulation studies

To allow the use of a `Burrlioz`-*like* method in our simulation studies, it was necessary to implement a version of `Burrlioz` that was contained entirely within the R environment and that could be called upon thousands of times without manual intervention. Although a standalone package, `Burrlioz` is in effect a front-end to the R statistical computing environment, i.e., it interacts with the user through a GUI to specify data sources, set preferences, and select various fitting options. This is then passed to the R statistical computing package which performs all the computations and returns the results to `Burrlioz`. Accordingly, the relevant mathematical functions and methods are contained within an `.RData` file that is saved in the package directory upon installation. This `.RData` file, along with the `Burrlioz 2.0` user manual, was used to develop an R package that could be installed and run on any machine.

Initial attempts to replicate the `Burrlioz 2.0` software failed to produce sufficiently similar results as we did not have access to the default input values, as well as critical control parameters used by the optimisation algorithm. After requesting and obtaining this information from the package developers, we found estimates for the HC values to be very similar, with HC estimates from the two software packages yielding a very tight 1:1 line (Figure 11). The 95% confidence bands on the HC estimates also agreed closely, with a tight relationship for the Upper estimates, and only minor deviations on the Lower estimate for 1000 bootstrap iterations (Figure 11). Note that the fits using the `RBurrlioz` implementation are more like the actual `Burrlioz 2.0` output than the `ssdtools sdd_fit_burrlioz` output (see above, Figure 10).

Overall, it appears that the `RBurrlioz` implementation closely replicates the behaviour of the `Burrlioz 2.0` software, and is suitable for use in simulation studies evaluating the performance of alternative methods as an approximation for the expected outcome were it possible to use `Burrlioz 2.0` directly in these simulations.

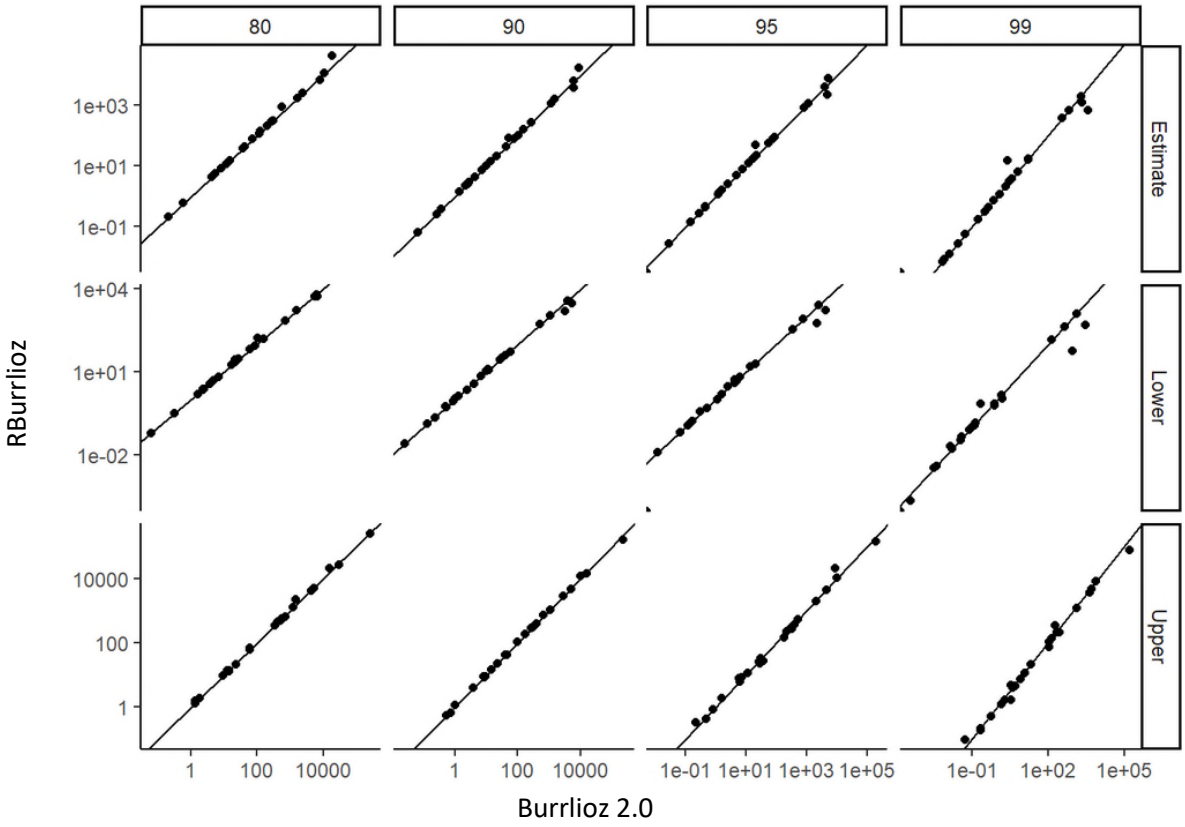


Figure 11. HC values estimated using `RBurrlioz` plotted against `Burrlioz 2.0`. Plot columns show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot rows are the actual estimate, as well as the lower and upper confidence bounds. Values with the left and lower half values missing have x and y values of 0.

2.5.3 Simulation Study 1 (*log-normal*, *log-logistic*, *inverse Weibull*)

We simulated data from an expanded grid of parameters across three distributions – the *log-normal*, *log-logistic* and *inverse Weibull* (also equivalent to the *log-Gumbel* [lgumbel] in `ssdtools`). This very large set of synthetic data was initially screened to avoid unrealistic distributions (e.g., 'bathtub' shapes) and a subset was selected (1,814 in total) to cover a broad range of skewness and kurtosis representations (composition: 610 *log-logistic*; 610 *log-normal*; and 594 *inverse Weibull*) such that the final dataset would have relatively equal representation of the parent distributions. The data were further screened to exclude those with a standard deviation less than 1 or greater than 5000. This ensured the data were consistent with the range in standard deviations across the observed benchmark datasets (all had standard deviations > 1). Although some observed datasets in `ssddata` have much larger standard deviations, data of that nature would generally be analysed in `Burrlioz` on a log scale, which we did not implement in our simulation workflow.

The associated meta-data, including the initial parent distribution, the original generating parameters and the true HC1, HC5, HC10 and HC20 estimates were collated and used in analysis of the outcome of the simulated data. As the `fitdistrplus` version of `ssdtools` and the TMB version `ssdtools` were yielding identical results for the benchmark data, we focused our efforts instead on fitting the simulated data using:

1. The RBurrlioz implementation of Burrlioz 2.0;
2. Model averaging using the new version of ssdtools (TMB) including only the three currently adopted distributions (*log-normal*, *log-logistic* and *gamma*);
3. Model averaging using ssdtools (TMB) using all currently available unimodal distributions in ssdtools: *Burr III* (burrIII3), *gamma*, *gompertz*, *inverse Pareto* (invpareto), *inverse Weibull* (*log-Gumbel,* lgumbel), *log-logistic* (llogis), *log-normal* (lnorm), and *Weibull*, as well as two mixture distributions (llogis_llogis, and lnorm_lnorm);
4. An ssdtools implementation of the Burrlioz 2.0 fitting framework using the function ssd_fit_burrlioz().

Fitted distributions were used to extract HC estimates that were compared to the true HC values to assess bias, as well as to examine the empirical coverage associated with the computed 95% confidence bounds returned by each method. Estimates were obtained across a range of samples sizes by subsampling from the initially generated 1,000 values for each synthetic dataset. Given the large number of datasets and the relatively long computation time required to estimate HC confidence limits using bootstrap methods, it was not possible to run multiple iterations of subsampling across the range of sample sizes. While we endeavoured to set a seed for this subsampling that was the same across all methods (so the same subsampled data was analysed for every method), this subsampling was not repeated iteratively, thus coverage estimates based on this first simulation study must be considered approximate. True coverage estimates were explored in more detail in the second simulation study described in more detail below.

2.5.3.1 Checking simulation expectations based on the parent distributions

Initially we examined bias and coverage as a function of sample size, using estimates based only on the individual distributions representing the parent distributions. We found that in all cases, and for all three parent distributions, bias is low when data are fitted using the same distribution as the parent distribution and shrinks to near zero as sample sizes increase (Figure 12). Where the fitted distribution is different to the parent distribution there is generally some bias in the estimates, and in some cases this bias can be quite extreme, particularly for the smallest HC values (Figure 12). For example, estimates obtained from a *log-logistic* or *log-normal* distributions fitted to data generated from the *inverse Weibull* distribution have a strong negative bias, although this disappears for the HC20 estimate (Figure 12). Both the *log-normal* and *inverse Weibull* (*log-Gumbel*) distributions tend to overestimate HC1 for data generated using a *log-logistic* distribution (Figure 12).

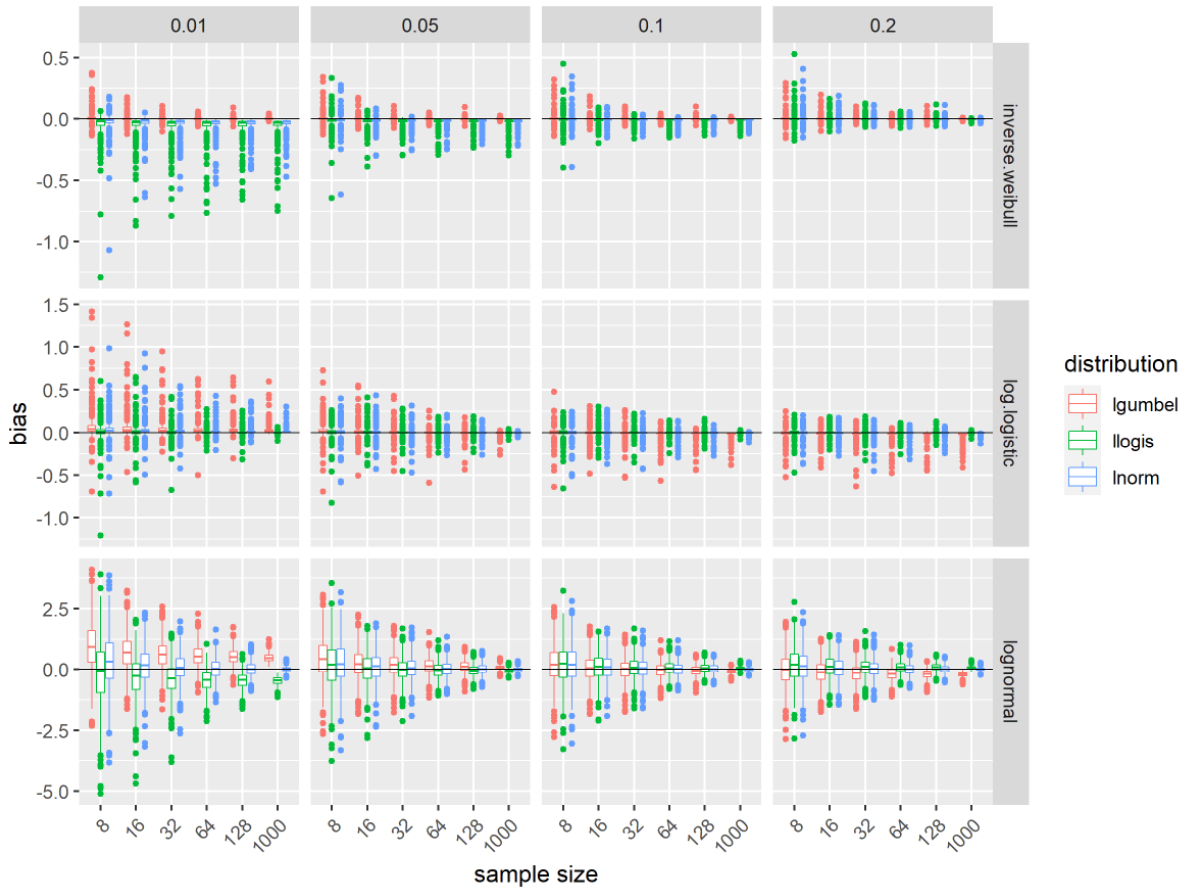


Figure 12. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the fitted distribution. All fits were done using `ssdtools` (TMB). Note that the inverse.weibull (as implemented in `Burrlioz 2.0`) and the lgumbel (`ssdtools`) are exactly equivalent distributions.

With respect to confidence interval width, the widest intervals were obtained when the fitted SSD is the *log-logistic* distribution, closely followed by the *log-normal*, with the *log-Gumbel* (*inverse Weibull*) having the narrowest intervals (Figure 13). This observation held true regardless of the parent distribution used to generate the data.

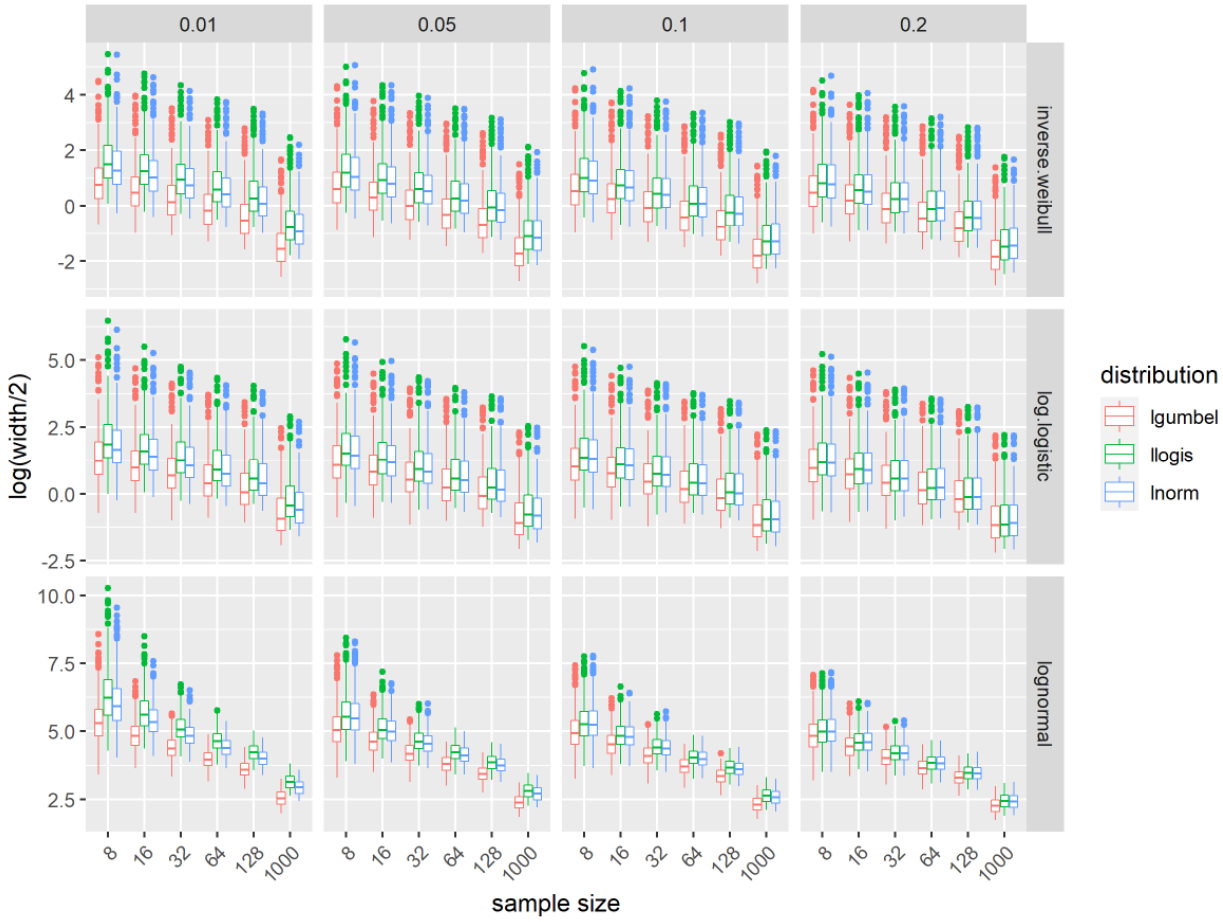


Figure 13. Confidence interval width of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the fitted distribution. All fits were done using `ssdtools` (TMB). Note that the inverse.weibull (as implemented in `Burrlioz 2.0`) and the lgumbel (`ssdtools`) are equivalent distributions.

As would be expected, estimates of coverage are close to nominal levels when the fitted SSD is a member of the same family as the underlying parent distribution (Figure 14). Actual coverage is less than the nominal 95% for small sample sizes, with rapid convergence to the 95% level for larger-sized samples where the fitted and parent distribution are of the same functional form (Figure 14). Inconsistent coverage results were obtained when the parent and fitted distributions were of different functional forms. Thus, we see, for example, that coverage for HC20 estimates for data generated from an *inverse Weibull* distribution were very close to the nominal 95% regardless of either the fitted distribution or the sample size, while for these same data, coverage for the HC1 and HC5 estimates showed a monotonic decline with increasing sample size when the fitted SSD was either a *log-logistic* or *log-normal* distribution. To explain this somewhat counter-intuitive result, one needs to go back to the plots of bias and confidence interval width for the HC1 and HC5 estimates for data generated from the *inverse Weibull* distribution (columns 1 and 2 of the first row of Figure 12 and Figure 13 respectively). With respect to bias, it is evident that the fitted *log-logistic* and *log-normal* SSDs yield severe underestimates of the true HC values and that the direction of this bias is unaffected by sample size.

On the other hand, as the sample size increases, the width of the confidence intervals consistently shrinks. Putting this together, we conclude that, as the sample size increases, the centres of the confidence intervals remain consistently to the left of the true HCx while the width of those confidence intervals shrinks. The net effect is that it becomes increasingly unlikely that the true HCx will be 'captured' by the confidence interval.

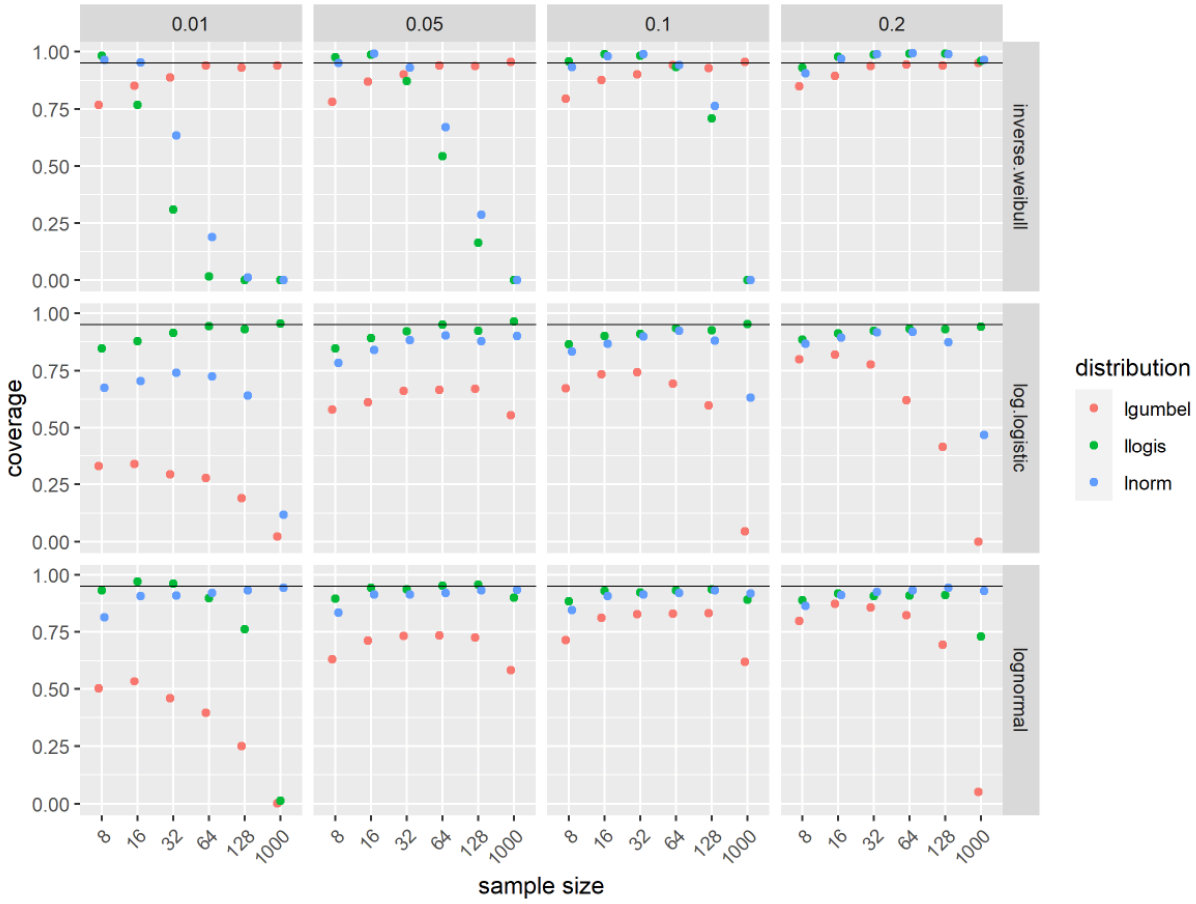


Figure 14. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the fitted distribution. All fits were done using `ssdtools` (TMB). Note that the inverse.weibull (as implemented in `Burrlioz 2.0`) and the lgumbel (`ssdtools`) are equivalent distributions.

2.5.3.2 Comparing methods across parent distributions

Overall, bias decreases as sample size increases as would be expected for all the methods considered (Figure 15). As discussed above, the clearest exception to this pattern occurs when the parent distribution used to generate the data and the fitted SSD are of different distributional forms. The most obvious case of this is for the model average approach using the 'default' set of only three distributions (*log-logistic*, *log-normal* and *gamma*, Figure 15). While the expected behaviour of decreasing bias with increasing sample size occurs for data generated by the *log-normal* or *log-logistic* distributions, bias remains relatively high across samples sizes for data generated using the *inverse Weibull* (Figure 15), which is not represented by this default set.

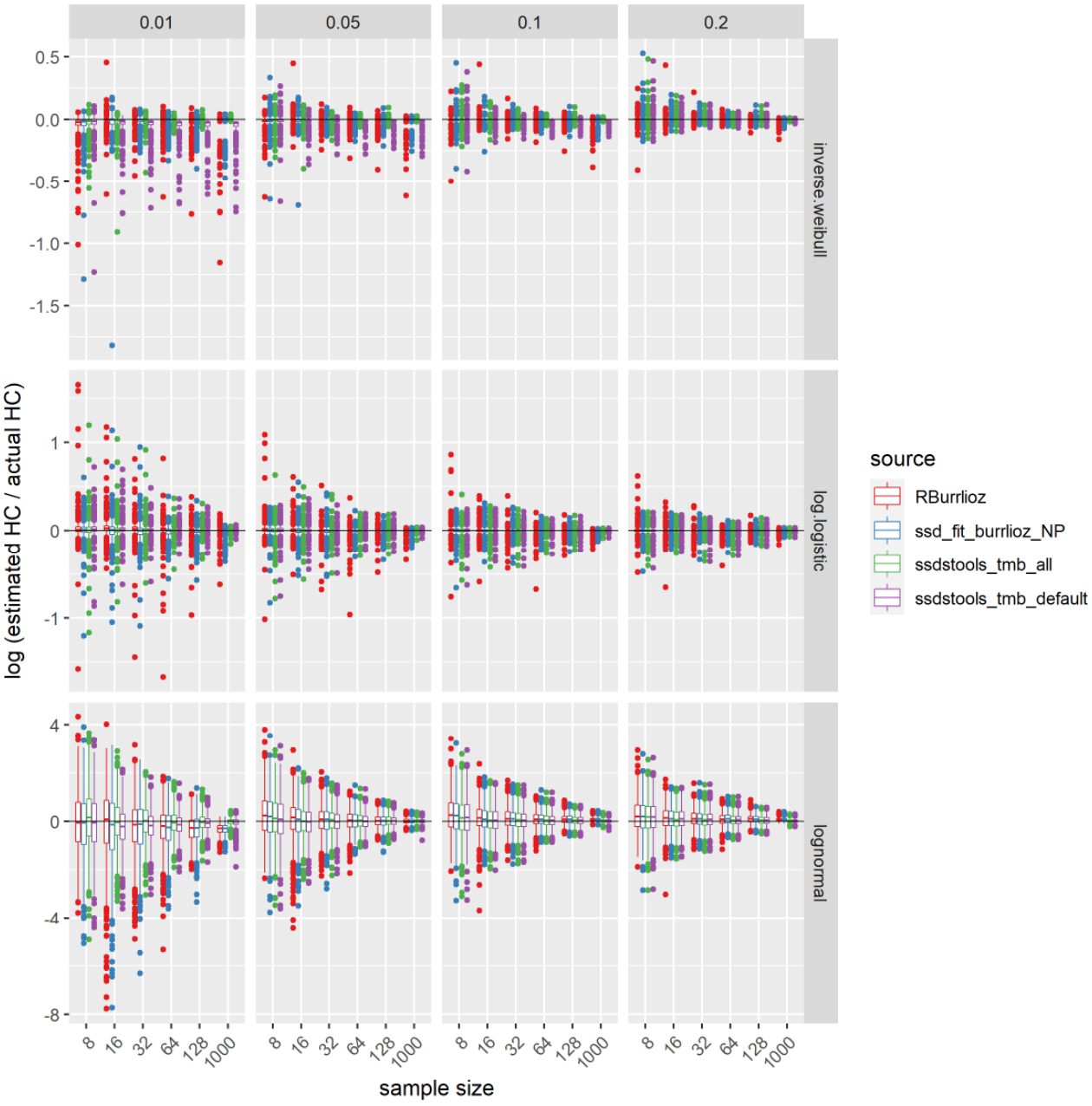


Figure 15. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the applied fitting method and includes `RBurrlioz`, the `ssd_fit_burrlioz` method implemented in `ssdtools` (TMB) using the non-paremetric bootstrap (in line with `RBurrlioz`) and two model averaged methods using `ssdtools` (TMB), including one using only the current BC three default distributions (*log-logistic*, *log-normal* and *gamma*, ssdtools_tmb_default) and one using all the available distributions (ssdtools_tmb_all).

The ‘all’ model average method appears to perform as well (if not better) than the `RBurrlioz` method at low sample sizes and has a clear advantage over `RBurrlioz` at large sample sizes, with bias converging uniformly to zero as sample size increases for all distributions (Figure 15).

Results for approximate coverage across these simulated data largely reflect the levels of bias observed across the methods (Figure 16). Coverage of the ‘all’ model average method is the best across all scenarios, whereas coverage of the ‘default’ model average method is extremely poor when the distribution used to generate the data is not one of the included default distributions (i.e.,the *inverse Weibull*/*log-Gumbel*, Figure 16). Coverage for the `RBurrlioz` and `sdd_fit_burrlioz()` methods behave a bit more unpredictably across sample sizes (Figure 16), possibly as a consequence of complex inter-relationships with the actual fitted distribution and sample size.

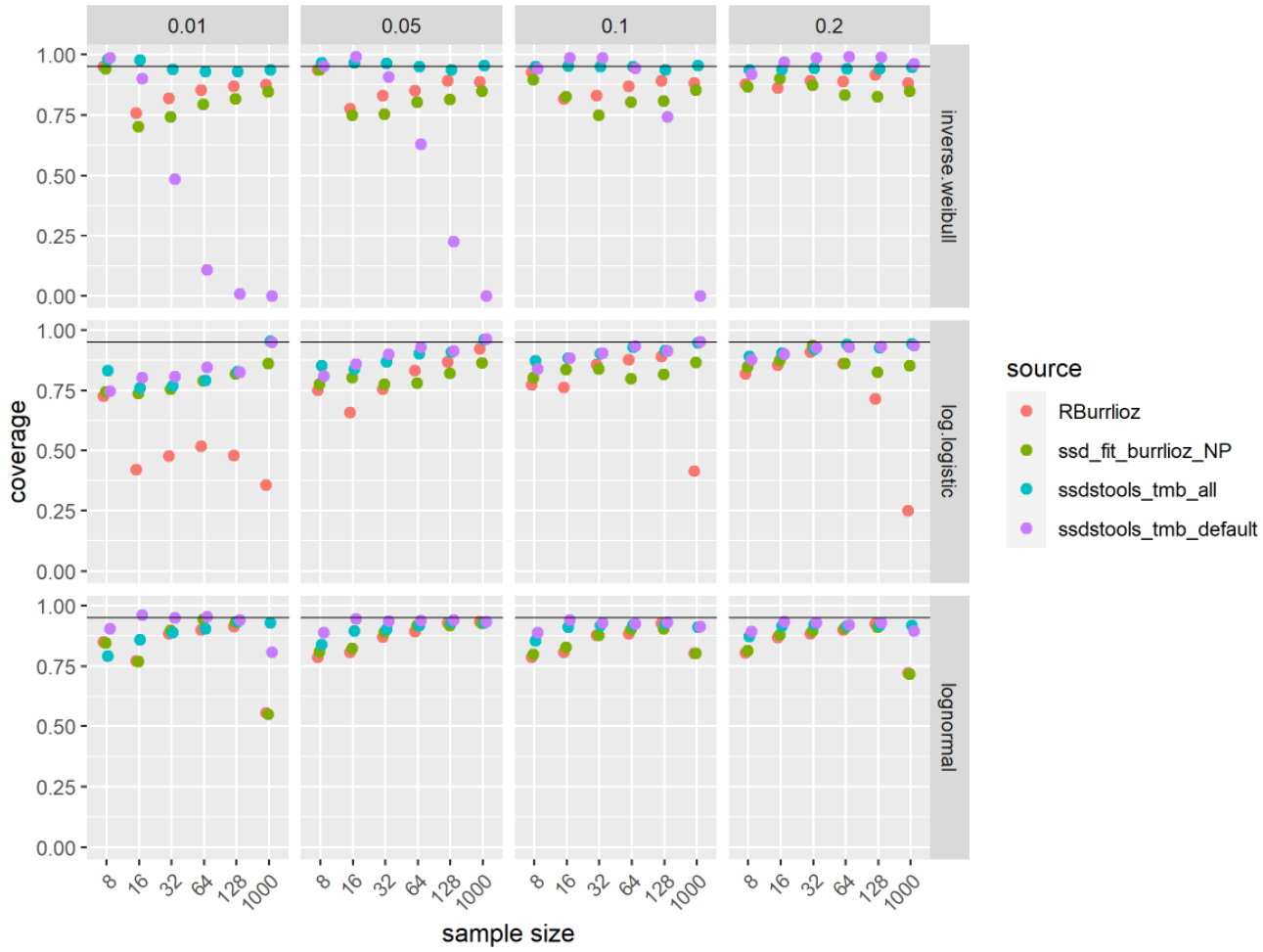


Figure 16. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the applied fitting method and includes `RBurrlioz`, the `ssd_fit_burrlioz` method implemented in `ssdtools` (TMB) using the non-parametric bootstrap (inline with `RBurrlioz`) and two model averaged methods using `ssdtools` (TMB), including one using only the current BC three default distributions (*log-logistic*, *log-normal* and *gamma*, ssdtools_tmb_default) and one using all of the available distributions (ssdtools_tmb_all).

2.5.4 Simulation Study 2 (Johnson family)

In Simulation Study 1 of Section 2.5.3, synthetic data was generated from a range of distributions, all of which were members of the 'all' candidate model set of the model average method, and some of which were also available in `RBurrlioz` and the 'default' set currently used in `ssdtools`. Not surprisingly, the performance of the different SSD-fitting methods among the datasets was in part a consequence of the commonality of parent and fitted distributions. Due to excessive computational demands, it was not possible to explore the true coverage of each SSD-fitting method for each simulated dataset. As a work-around, we chose parameter values for members from the Johnson family of distributions (Johnson 1949) to mimic the distributional properties of six of the original benchmark datasets in `ssddata` identified as "anon_a", "anon_b", "ccme_boron", "ccme_glyphosate", "ccme_silver", and "ccme_uranium" and generated 'parent populations' of size *n=1,000* for each. Using the *actual* datasets themselves was not an option because: (i) the *true* HC values were unknown; and (ii) the sample sizes were too small to permit the construction of random sub-samples of a variety of sizes. A comparison of the theoretical and empirical *cdfs* for the six datasets is shown in Figure 17.

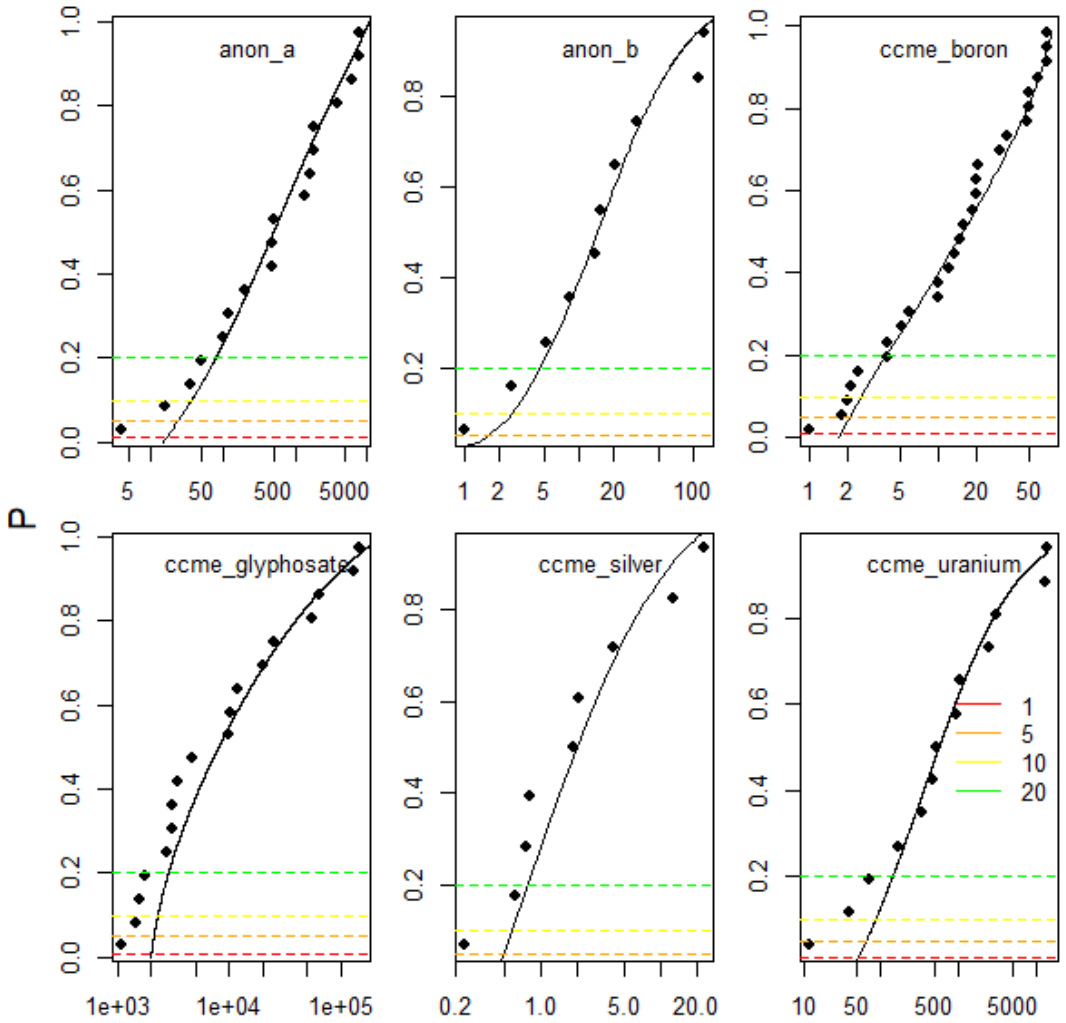


Figure 17. Benchmark data (black circles) and generated theoretical curve based on the Johnson family of distributions that was used in simulation studies (solid black line). Horizontal coloured dashed lines show where the HC1, HC5, HC10 and HC20 values cross the theoretical curve.

Although the first four moments of the fitted and empirical distributions match, it is evident from Figure 17 that the Johnson distributions do not necessarily do a particularly good job at replicating the left-tail behaviour exhibited by the data. Indeed, the fitted Johnson distributions have, to varying degrees, no long-tail behaviour and instead of approaching the horizontal axis smoothly, they intercept it abruptly. It could be argued that this is unrealistic of (eco-) toxicity data and therefore our simulation studies do not provide a useful basis on which to make comparisons or recommendations. Our counterargument is that regardless of the degree of ecotoxicological realism, the distributions in Figure 17 provide a 'stress-test' in the sense that these represent challenging cases for the SSD modelling tools – particularly in the context of their ability to cope with 'unknown' distributions. Furthermore, the 'handicapping' introduced by using 'unknown' distributions is the same for all SSD modelling tools and to that extent no systematic bias has been introduced.

A graphic illustration of the ability (or inability) of the various modelling tools to replicate the left-tail behaviour of the Johnson distributions of Figure 17 is shown by the sequence of plots in Figure 18, which shows 20 realisations (i.e., 'fits') from each of the SSD modelling tools for each of the replica datasets as a function of sample size.

In the analyses that follow, we investigate the performance characteristics of the SSD modelling tools in terms of bias, confidence interval coverage and confidence interval width. Some of these characteristics are readily explained by Figure 18. For example, the reason that the bias in HC20 estimates for the `ccme_uranium` replica data are all positive is explained by the fact that the majority of points where the horizontal line at *p=0.2* intercepts the fitted curves are to the right of the point where *p=0.2* intercepts the black curve (i.e.,the true HC20). Furthermore, because the fitted SSDs are more compactly distributed about the true (black) curve as the sample size increases, this means that the *magnitude* of the bias diminishes.

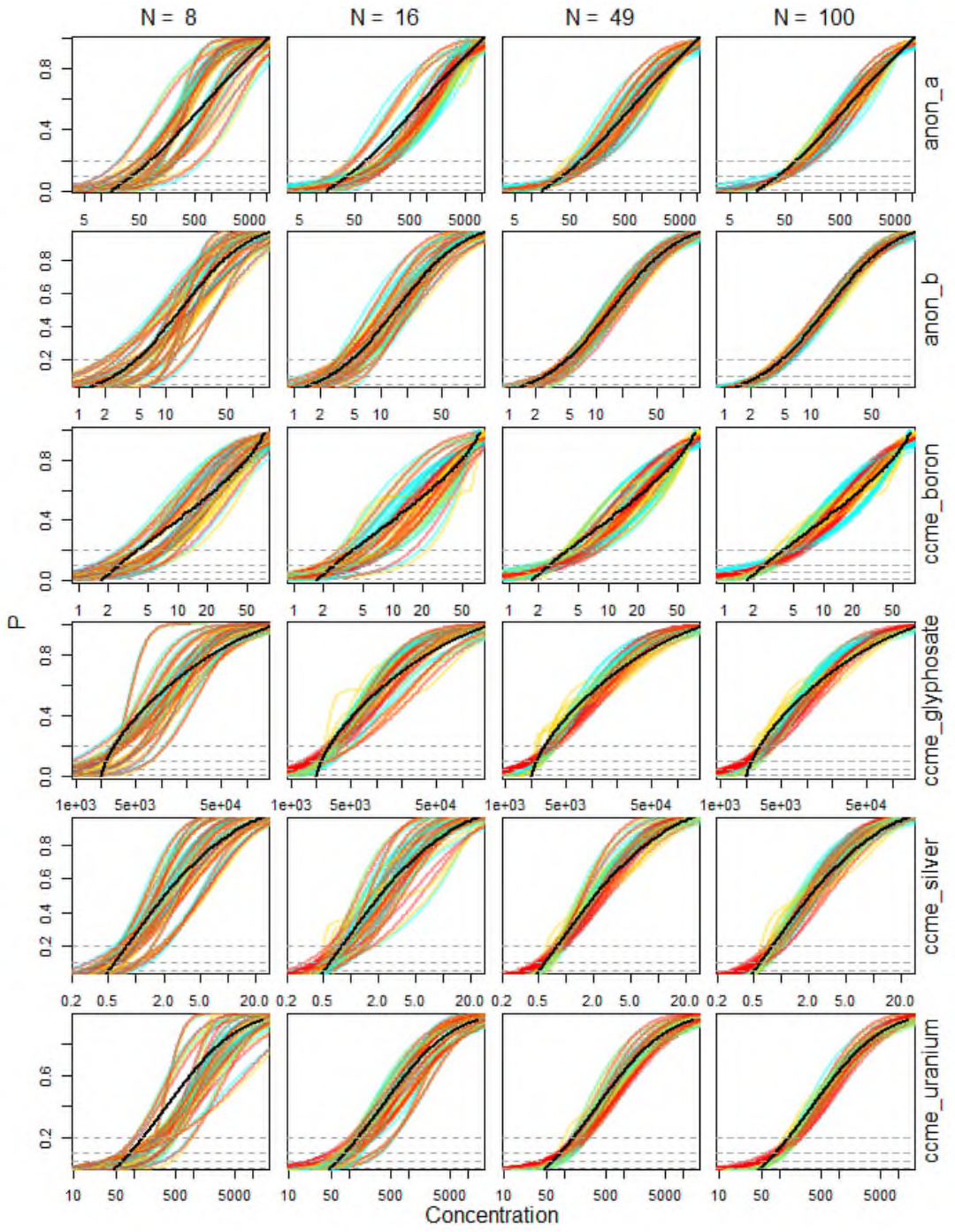


Figure 18. Twenty realisations of the fitted distribution for each of the methods considered, including RBurrlioz (cyan lines) and two model averaged methods using ssdtools (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default – red lines; and all - using all the available distributions, gold lines) for four different sample sizes (n). The solid black line shows the theoretical source distribution.

Coverage estimates for these six simulated datasets generally declines with increasing sample size, and this was true across all methods, although not for all HC values for all datasets (Figure 19). Except for the replica anon_b dataset, the coverage for the HC1 declines monotonically for sample sizes greater than ~ 12 and is uniformly worse when the default distribution set in ssdtools is used and best when all distributions are used in ssdtools. The performance of Burrlioz lies between these two extremes. The default set in ssdtools performs particularly poorly for the replica ccme_uranium dataset. Taken as a whole, the set of results is inconsistent. The following additional observations are pertinent:

Replica anon_a synthetic dataset

- coverages for HC5 and HC10 are consistently close to nominal 95%; less so for HC10
- coverage for HC20 declines steadily when using the default candidate list from ssdtools and the sample size is greater than 50.

Replica anon_b synthetic dataset

- coverage estimates are uniformly good across all sample sizes and all fitting strategies

Replica ccme_boron synthetic dataset

- coverages for HC5 are very close to the nominal 95% for al sample sizes
- poor coverage is observed for Burrlioz when the sample size is less than ~ 36 and ssdtools 'all' when the sample size is greater than ~ 80.
- Coverage for the HC20 is poor for all fitting methods with the ssdtools 'default' option performing worse, the ssdtools 'all' option performing best and Burrlioz in between.

Replica ccme_glyphosate synthetic dataset

- coverages for HC5 using ssdtools 'all' option are consistently close to the nominal 95% irrespective of sample size while the 'default' option fares particularly poorly and Burrlioz somewhat better.
- coverages for HC10 are consistently close to 95% with both ssdtools option performing better than Burrlioz
- coverages for HC20 are poor for the ssdtools 'all' option and Burrlioz although good for the ssdtools 'all' option with sample sizes less than ~64.

Replica ccme_silver synthetic dataset

- coverages for HC1 are all poor and decrease rapidly with increasing sample size with the ssdtools 'default' set performing worst.
- coverages for HC5 are very good for both Burrlioz and ssdtools 'all' for all sample sizes. The ssdtools 'default' option has very poor coverage which declines linearly with increasing sample size.
- coverages for HC10 are generally good for both ssdtools options but less so for Burrlioz.
- coverages for HC20 are reasonable for both ssdtools options with Burrlioz performing poorly.

Replica ccme_uranium synthetic dataset

- coverages for HC1 are all poor and decrease rapidly with increasing sample size with the ssdtools 'default' set performing worst.

- coverages for HC5 are very good for `Burrlioz` and `ssdtools` 'all' and `ssdtools` 'default' for sample sizes <~80.
- coverages for HC10 are very good for `ssdtools` 'default' but not any other method.
- coverages for HC20 are all similarly poor.

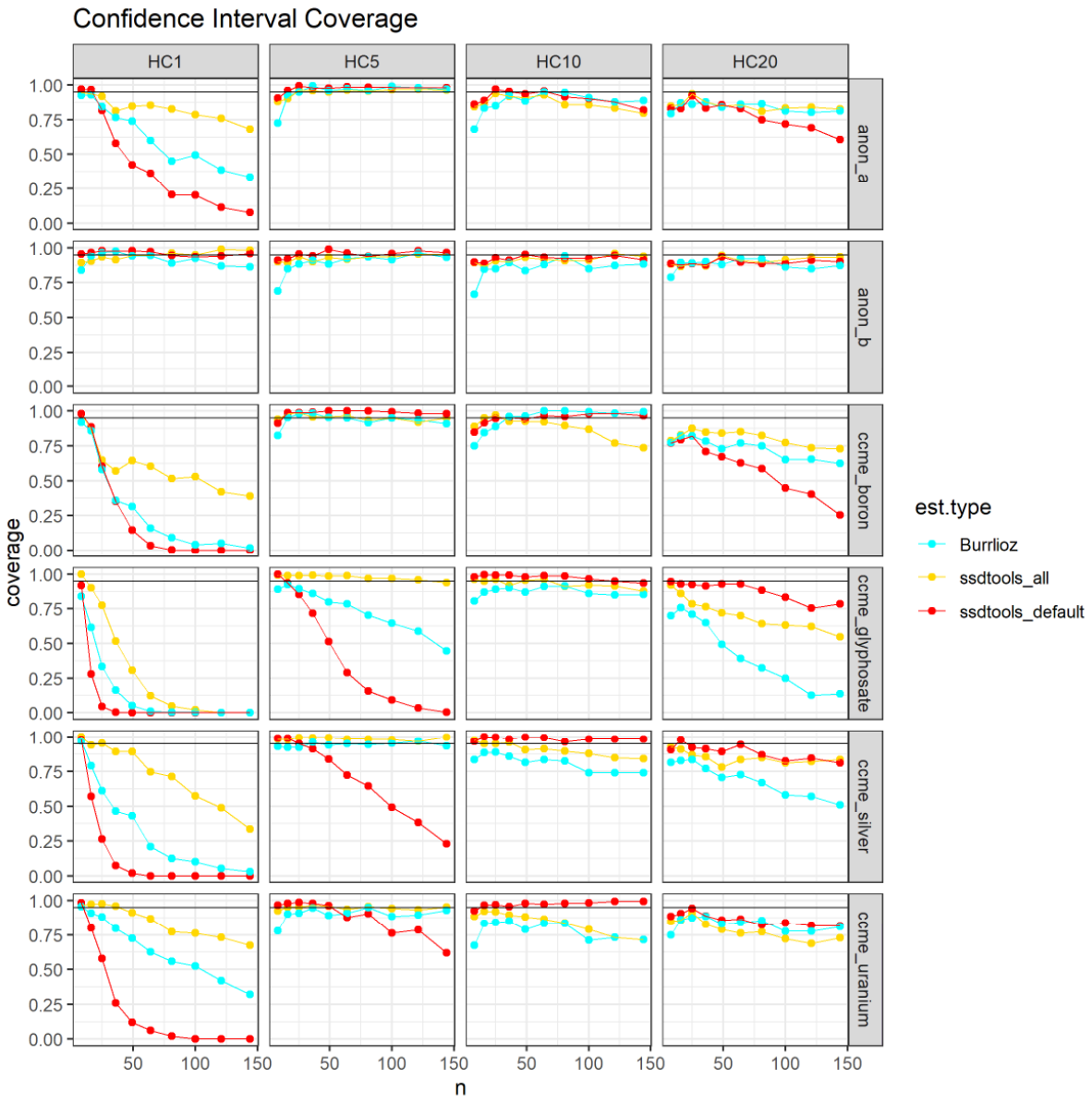


Figure 19. Coverage estimates for 95% confidence intervals of the HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the applied fitting method and includes R`Burrlioz` and two model averaged methods using `ssdtools` (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default; and all - using all the available distributions).

In terms of bias, the relationship with sample size was very consistent and, in most instances, displayed the expected asymptotic decline to zero with increasing sample size (Figure 20).

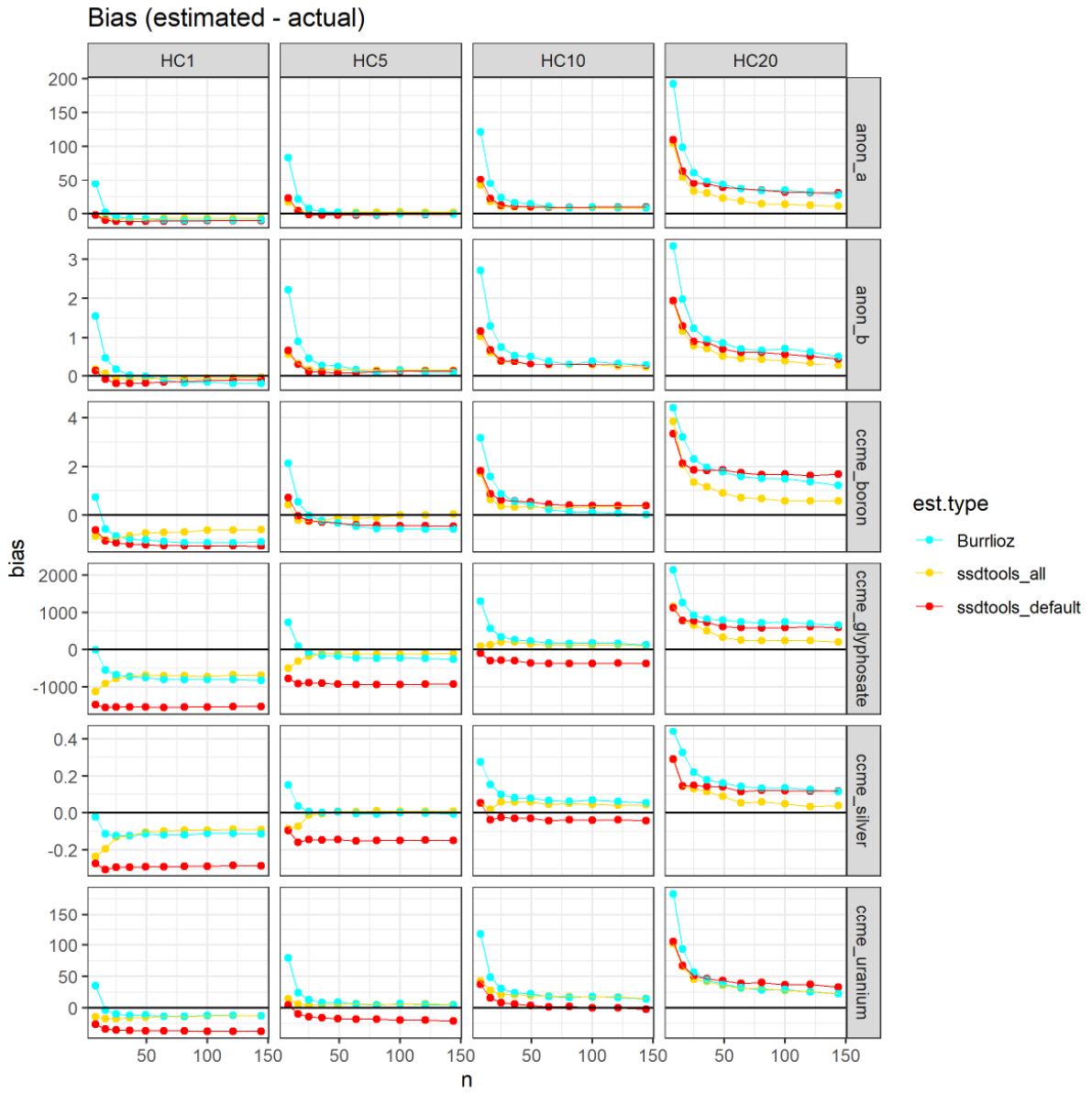


Figure 20. Bias (mean estimate-actual) of HC estimates across four different protection levels (plot columns - p = 0.01, 0.05, 0.1 and 0.=2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the applied fitting method and includes RBurrlioz and two model averaged methods using ssdtools (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default; and all - using all the available distributions).

There were however a few notable exceptions:

- over-estimation of HC10 and HC20 for all methods and all sample sizes for the datasets based on `anon_a` , `anon_b`, `ccme_boron`
- over-estimation of HC20 for all methods and all sample sizes for replica datasets: `ccme_boron; ccme_glyphosate, ccme_silver, and ccme_uranium`

- under-estimation of HC1 for all methods and all sample sizes (with one or two exceptions for very small sample sizes) for all replica datasets. The magnitude of bias in the `Burrlioz` estimates was generally greater than either of the estimates from `ssdtools`. The `ssdtools` 'all' option tended to have the lowest absolute bias over all replica datasets and sample sizes. In terms of confidence interval width, there was a consistent inverse relationship with sample size with all estimation methods performing almost identically in all instances (Figure 21). The small number of instances where differences between the methods was observed was invariably a result of the `ssdtools` 'default' option having wider intervals.

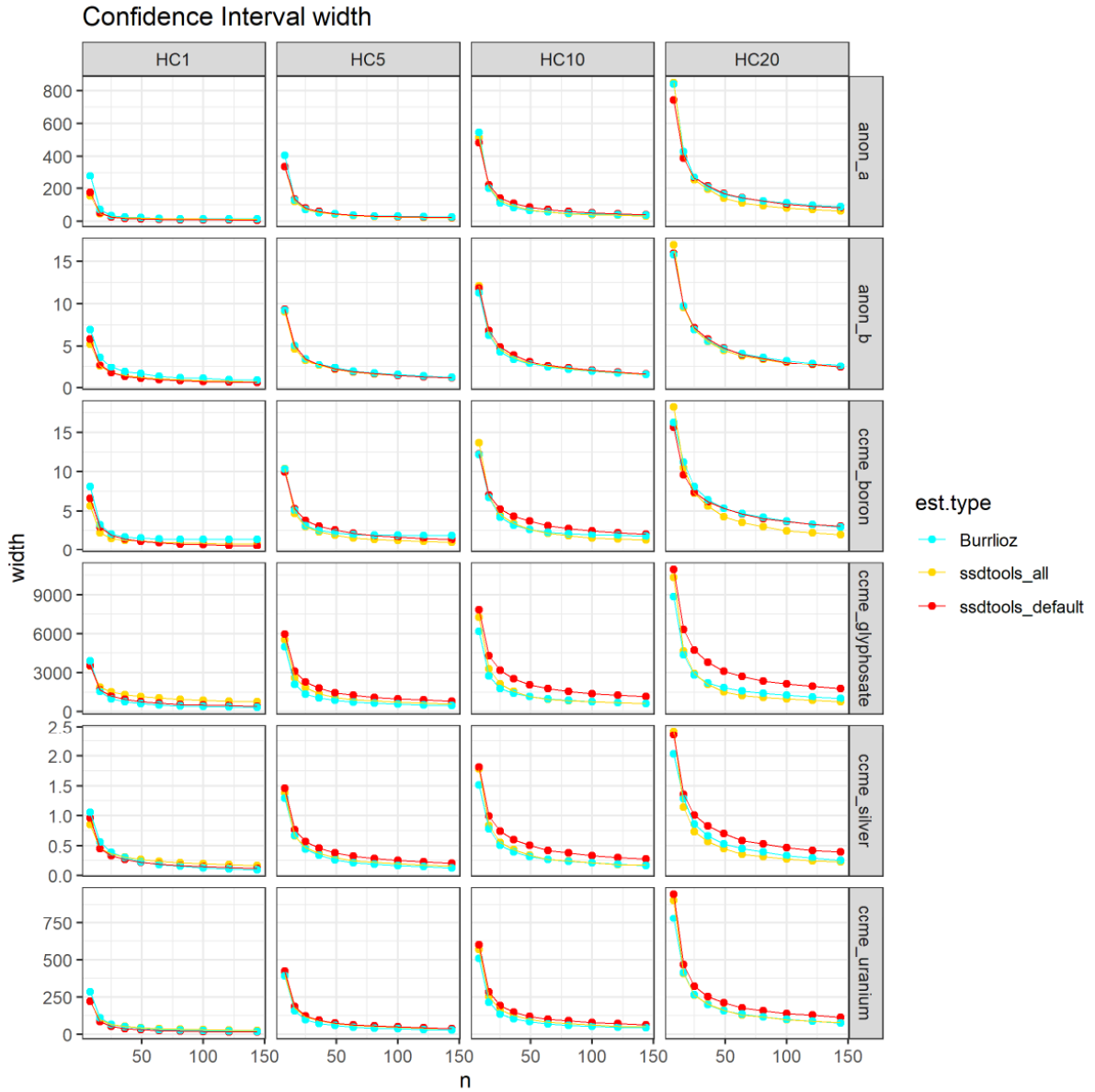


Figure 21. Mean 95% confidence interval width of HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the applied fitting method and includes RBurrlioz and two model averaged methods using ssdtools (default - using only the current BC default distributions: *log-logistic*, *log-normal* and *gamma*, ssdtools_ default; and all - using all the available distributions).

2.6 Refine mixture-modelling with a view to incorporation

The use of statistical mixture-models was promoted by Fox as a convenient and more realistic way of modelling bimodal toxicity data (Fisher et al. 2019). Although parameter heavy, statistical mixture-models provide a better conceptual match to the inherent underlying data generating process since they directly model bimodality as a mixture of 2 underlying univariate distributions that represent, for example, different modes of action (Fox et al. 2021). It has been postulated that a mixture-model would only be selected in a model-averaging context when the fit afforded by the mixture is demonstrably better than the fit afforded by any single distribution. This is a consequence of the high penalty in AICc associated with the increased number of parameters (*p* in Equation 7 of Fox et al. (2021)) and will be most pronounced for relatively small sample sizes.

We first describe how mixtures have been incorporated into the `ssdtools` package and then use simulation studies to examine their impact in terms of confidence interval coverage and bias relative to single distributions, and the extent to which sample size influences their relative weight in a model averaging context.

2.6.1 Implementation in `ssdtools` (TMB)

The TMB version of `ssdtools` now includes the option of fitting two mixture distributions, individually or as part of a model average set. These are the *`lnorm-lnorm`* and the *`llogis-llogis`*. The underlying code for these mixtures has three components: the likelihood function required for TMB; exported R functions to allow the usual methods for a distribution to be called (p, q and r); and a set of supporting R functions (see Appendix D for more details). Both mixtures have five parameters - two parameters for each of the component distributions and a mixing parameter (`pmix`) that defines the weighting of the two distributions in the 'mixture.'

2.6.2 Simulation Study 1 (*log-normal, log-logistic*, *inverse Weibull*)

The first simulation study was based on three different univariate parent distributions, the *log-normal*, *log-logistic* and *inverse Weibull* - so this simulation study indicates the outcome of including mixtures in simulated data that are known to have at most a single mode, and for which the parent distribution is in the distribution set being considered. We found that in terms of bias, both the 'all' (including mixtures) and 'uni' (all unimodal distribution) sets yielded very similar results, with little to no systematic bias, and overall bias converging to the expected 0 with increasing sample size (Figure 22). Similarly, the coverage of the model set using 'all' distributions was very high, and in fact slightly higher than the set including only the unimodal distributions (Figure 23).

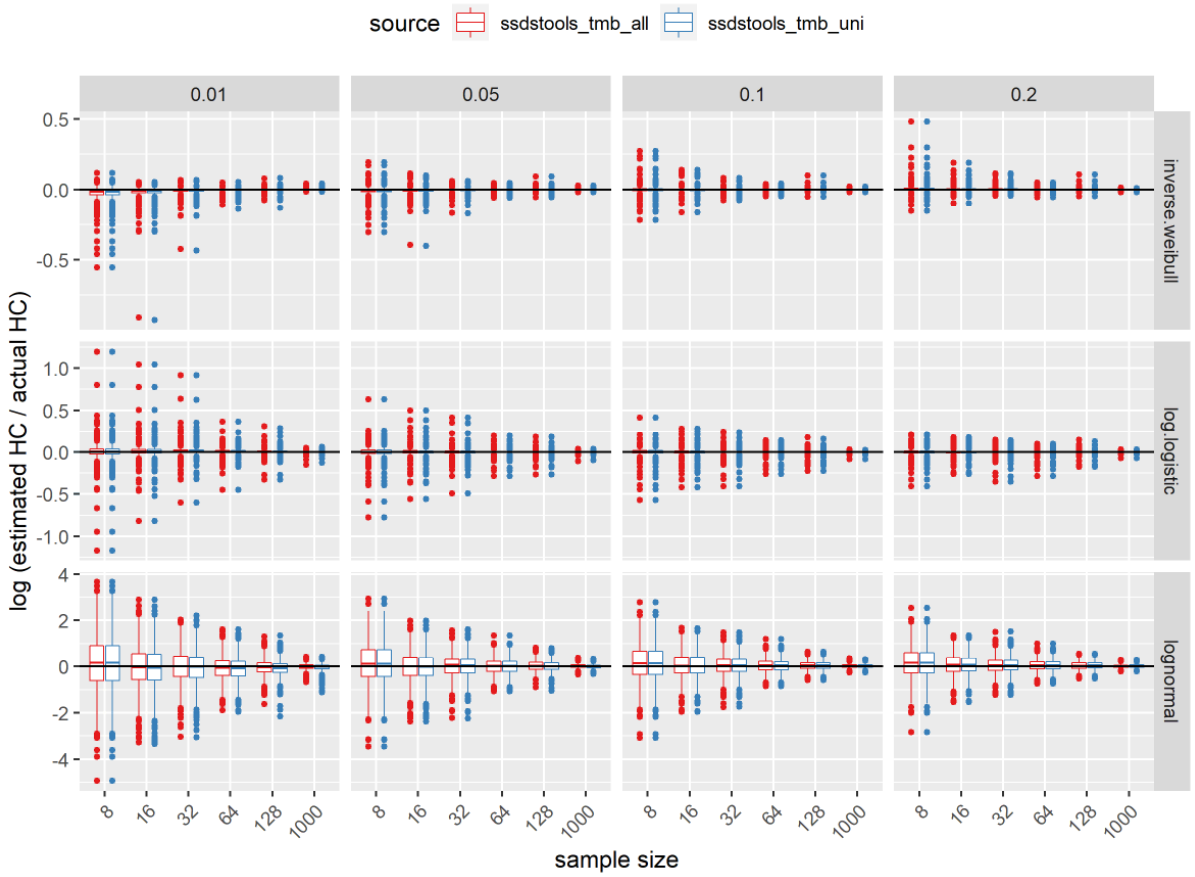


Figure 22. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni).

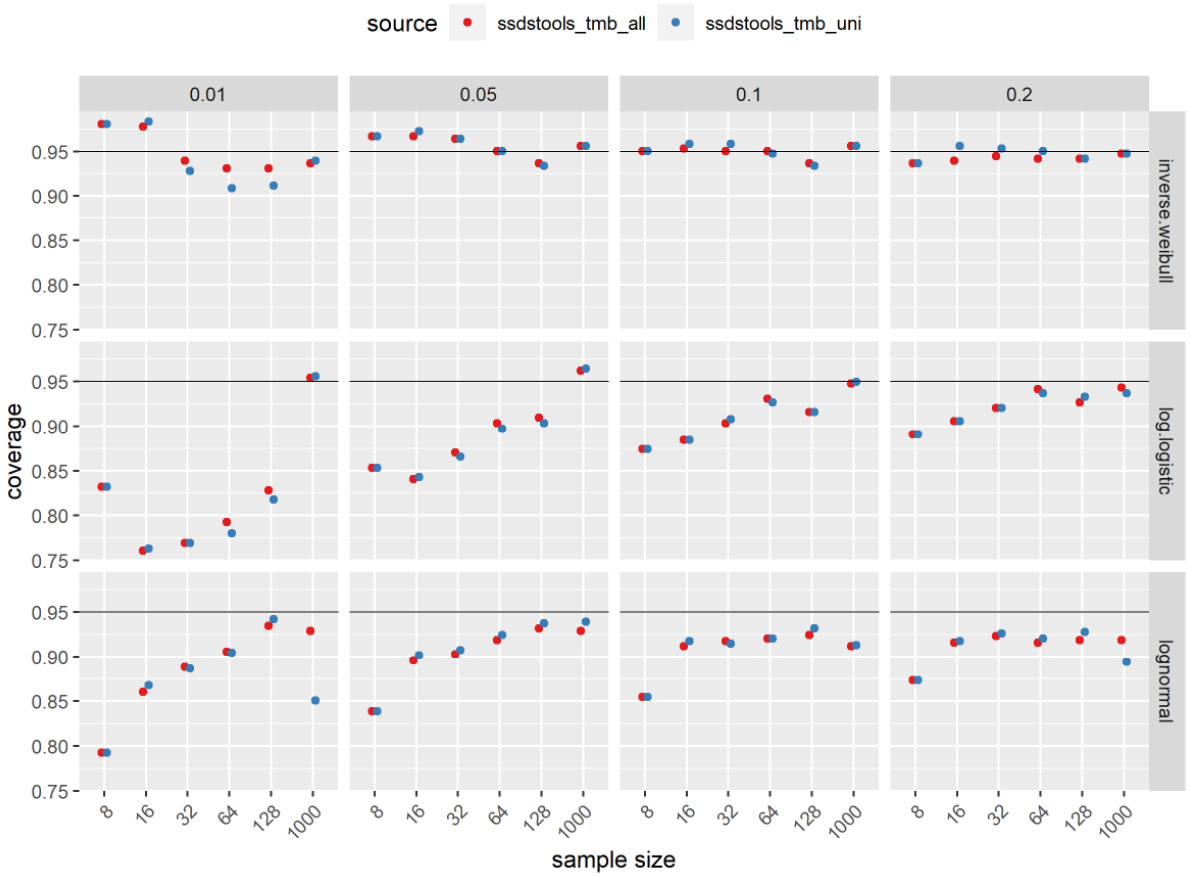


Figure 23. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni).

2.6.1 Simulation Study 2 (Johnson family)

The second study used data simulated from the Johnson family of distributions (Johnson 1949) that replicated the characteristics of six of the `ssddata` benchmark datasets[1]. In this example the parent distributions were also unimodal but were not included in the model set considered by `ssdtools`. We found that, in this scenario, including the two mixture distributions either had no impact on the HC estimate, or alternatively (in quite a few cases) substantially improved the bias in the HC estimates (i.e., estimates were much closer to the theoretical value) (Figure 24). Similarly, including the mixture-models also either had no impact on coverage of the estimated HC confidence intervals, or substantially improved it (Figure 25). In some cases, this improvement was sufficient to reach the expected 95% level, even when coverage of the unimodal set was extremely low (for example, the HC5 for `ccme_boron`, Figure 25).

[1] The use of a theoretical distribution meant that the *true* HCx was known and also allowed for the generation of many samples of varying sample sizes.

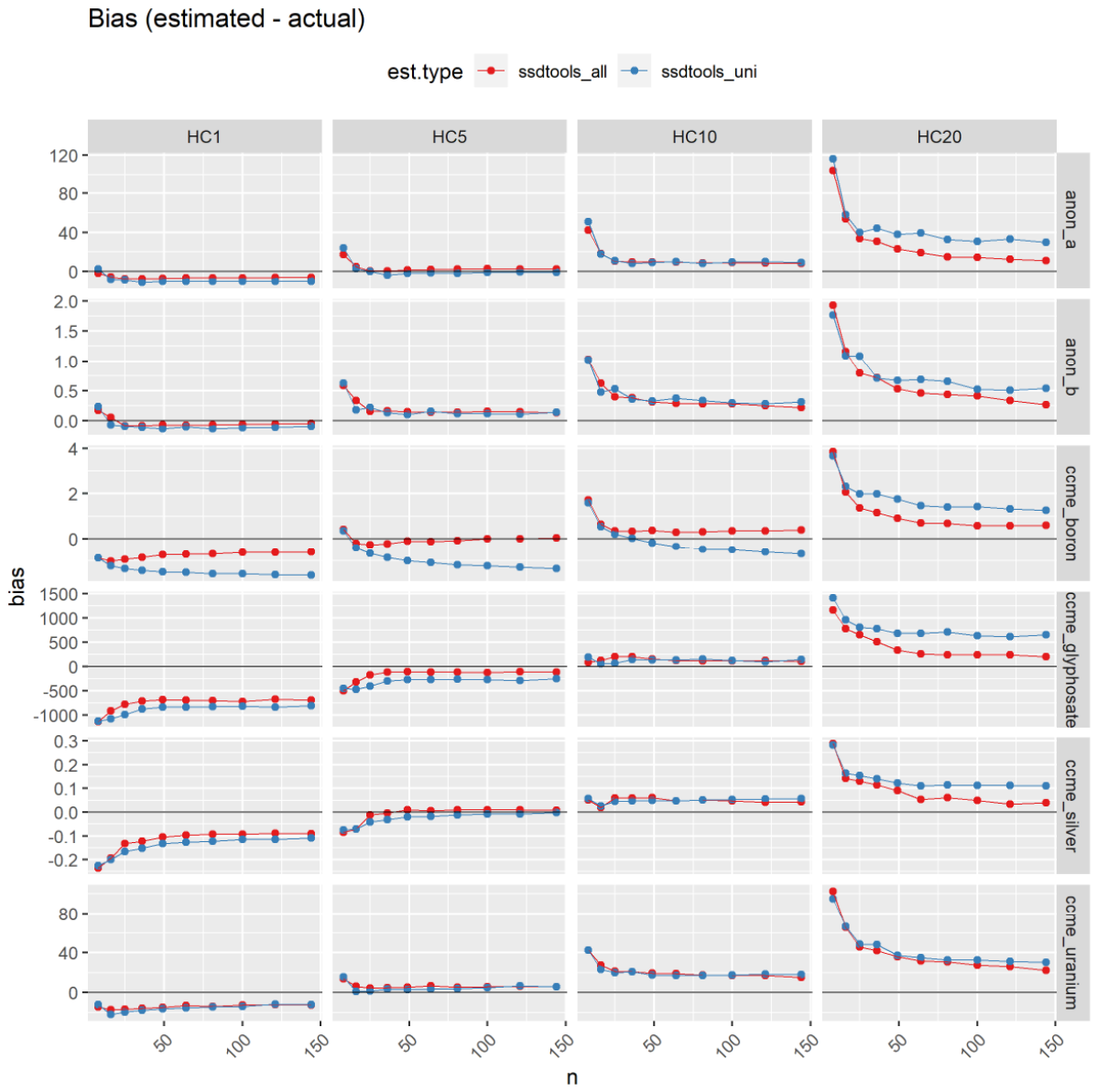


Figure 24. Mean bias (estimated-actual) of HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni).

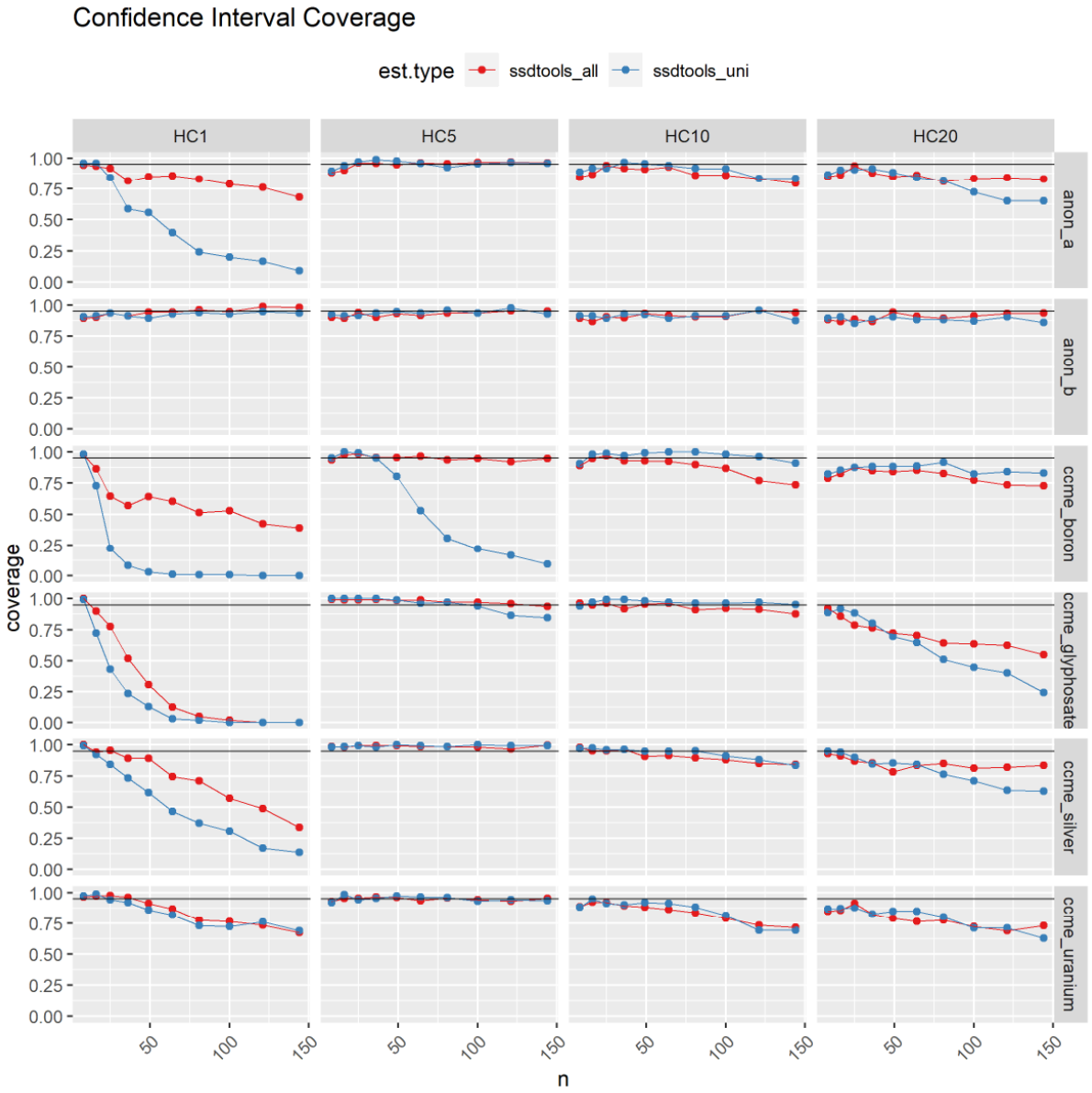


Figure 25. Coverage estimates for nominal 95% confidence intervals of the HC estimates across four different protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from six different source distributions (plot rows). Plots are coloured according to the distribution set used in model averaging, including all available distributions (all) and only the unimodal distribution (uni).

2.6.2 Simulation Study 3 (mixtures)

To examine how the AICc weighting of models in the 'all' distribution set changes with sample size, we undertook a Simulation Study based on a combination of a *log-normal* and a *log-logistic* distribution. The data for each simulation were based on 1000 random draws from the two distributions, with the number of draws from each defined by a mixing proportion (the proportion of the data comprising the *log-normal*), including 0 (no *log-normal* data, entirely *log-logistic*), 0.5 (an

even mixture of *log-normal* and *log-logistic*) and 1 (all *log-normal* data, no *log-logistic*). The parameters of the underlying *log-normal* and *log-logistic* distributions were selected to ensure mixture data showed at least some visual evidence of bimodality (Figure 26). Two values for the parameter defining the mean of each distribution (`meanlog`; `locationlog`) were selected, with the parameters controlling dispersion set as 0.5 and 0.2 respectively (`sdlog` and `scalelog`). Simulation data were generated from an expanded grid of both parameter and mixing values, resulting in a total of 12 simulated datasets, four of which showed clear bi-modality (Figure 26).

When we fit these data using all the distributions available in the TMB version of `ssdtools`, we found that at the lowest sample size examined (n=8), weights were relatively evenly distributed across the seven two-parameter distributions (Figure 27). As would be expected, the mean weight for the underlying data generating distribution increased systematically with sample size for the datasets comprised solely of *log-logistic* (mixing = 0) or *log-normal* (mixing = 1) data (Figure 27). Similarly, the weight for the *Burr III* distribution also increased with sample size for these unimodal datasets and tended to be higher for data generated from the *log-logistic* distribution (Figure 27).

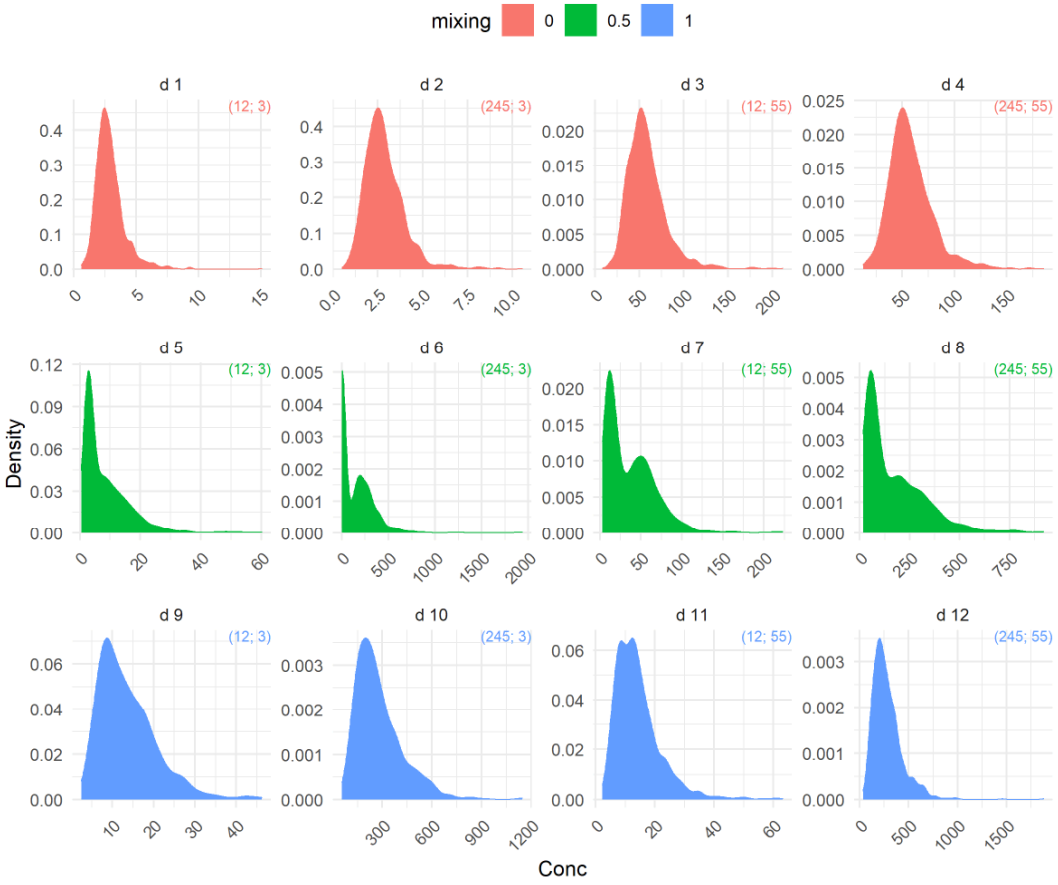


Figure 26. Density plots showing the simulated data for each of the datasets used in the mixture Simulation Study. Colours indicate the mixing proportion, including 0 (no *log-normal* data, entirely *log-logistic*), 0.5 (an even mixture of *log-normal* and *log-logistic*) and 1 (all *log-normal* data, no *log-logistic*). Values in parentheses show the mean values of the underlying *log-normal* and *log-logistic* distributions respectively.

While at a sample size of 8 neither of the two mixture distributions received any support (Figure 27), at a sample size as low as 16 the mixture-models gained substantial support (summed mean AICc weight is near 1) for dataset d6 (Figure 27). This is the mixture dataset exhibiting the most extreme bimodality (Figure 26). For the remaining simulated bimodal data, the mixture distributions do not receive substantial support (>0.25) until sample sizes of 32 are reached, with higher support requiring even larger sample sizes (Figure 27).

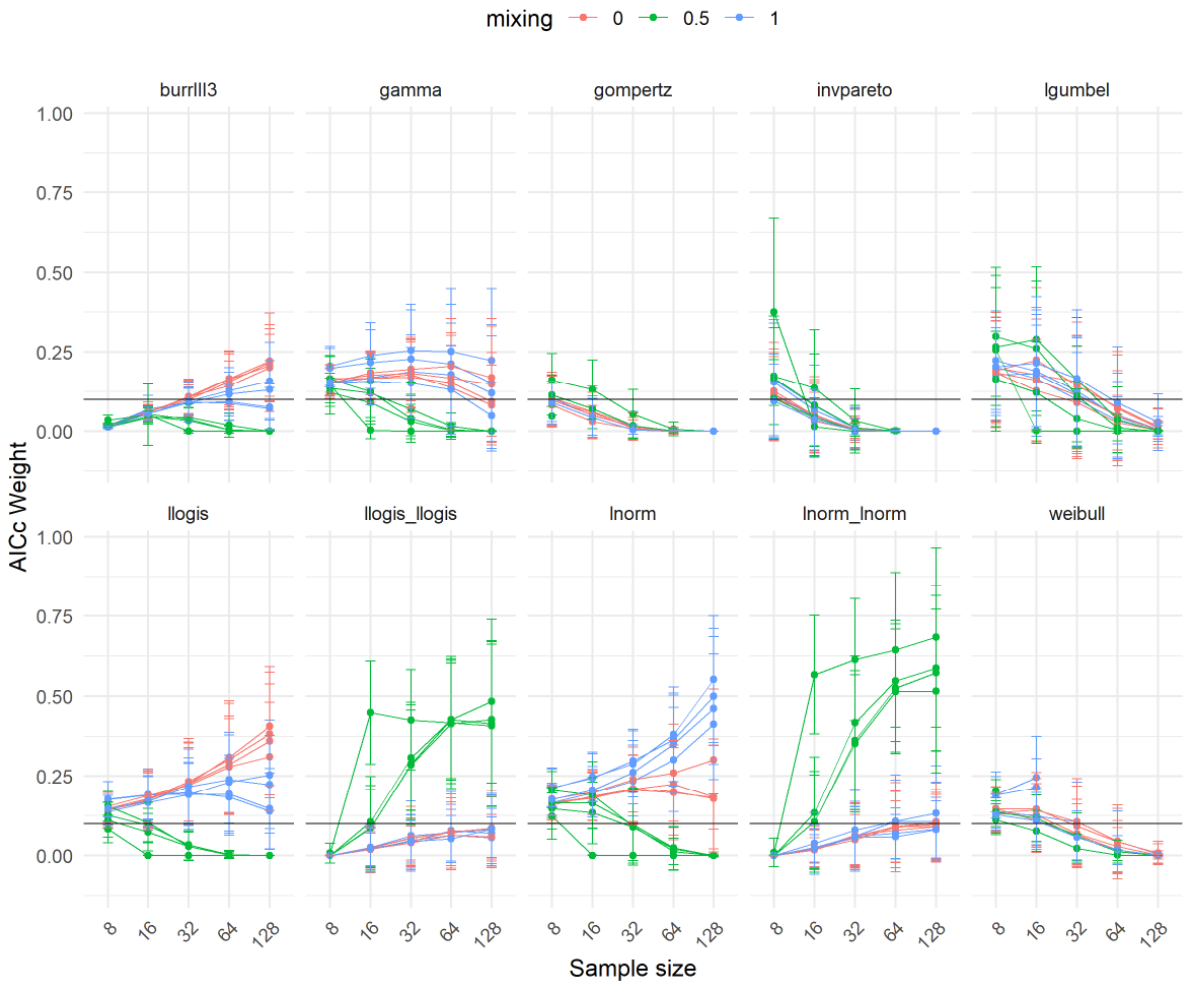


Figure 27. Mean AICc based model weights for each of the fitted distributions, for each dataset in the Simulation Study, as a function of sample size. Colours indicate the mixing proportion, including 0 (no *log-normal* data, entirely *log-logistic*), 0.5 (an even mixture of *log-normal* and *log-logistic*) and 1 (all *log-normal* data, no *log-logistic*). The horizontal line indicates a weight of 0.1, representing the expected value if all distributions were weighted equally.

3 ALTERNATIVE CI AND HC_X ESTIMATION STRATEGIES

3.1 Parameter estimates using L-moments (*Burr III*)

Maximum likelihood estimation (MLE) is ubiquitous and generally the default estimation strategy used in computational statistics. There are sound theoretical reasons underpinning this choice although it needs to be recognised that MLE is not the only estimation strategy. A recurring difficulty with maximum likelihood estimation of *Burr III* parameters when applied to small ecotoxicological datasets is the difficulty the numerical algorithms have in converging to the MLEs. This is because for these datasets the likelihood surface is sometimes very flat in the vicinity of the MLE and thus has no well-defined peak or maximum. This means that the value of the likelihood function is almost identical for wildly different parameter estimates and thus there is no basis for selecting one solution in preference to another.

We have explored the possibility of using the EM algorithm and L-moment estimation (Hosking 1990, Hosking 2006). The EM algorithm is generally used for estimating the parameters of latent-variable models (i.e., models that contain variables for which direct observation is not possible) or in situations where data are missing. Very little work appears to have been done on using the EM algorithm in the context of Burr distributions although the conference paper by Ismail and Khalid (2014) used the EM algorithm with *Burr III* distributions applied to censored data. There may be some merit in exploring the use of the EM algorithm when dealing with censored toxicity data, however these situations are already adequately handled by existing software tools (`MOSAIC`, `ssdtoolbox` and `ssdtools`) and as such this is unlikely to be a profitable avenue of further investigation.

In the remainder of this section we provide theoretical results and detailed procedures for the estimation of *Burr III* parameters using L-moments. The use of L-moments shows considerable promise in alleviating (if not avoiding) many of the difficulties encountered with maximum likelihood estimation for Burr distributions. A draft technical report is provided in Appendix E.

The concept of L-moments was introduced by Hosking (1990) and they have found to be particularly useful for describing probability distributions. Unlike conventional moments (which are also widely used to describe and fit probability distributions), L-moments have several unique properties. Among these is the fact that any distribution with finite mean is uniquely determined by its L-moments. Although the use of conventional moments for estimating the parameters of a probability distribution (the so-called method-of-moments or MoM estimation) has a long history, their use in this context has largely given way to likelihood-based approaches (MLEs).

Maximum likelihood estimators are generally more accurate than conventional MoM estimators and they enjoy several desirable statistical properties not shared by MoM estimators such as asymptotically normality. Furthermore, for some monotonic function $g(\cdot)$, the estimate $g\left(\hat{\theta}\right)$ is the MLE of $g\left(\theta\right)$ where $\hat{\theta}$ is the MLE of θ.

Advantages of L-moment estimators are:

- They are more robust than conventional moments to outliers in the data;
- L-moment estimators can usually be used when MLEs are either unavailable or difficult to compute
- They enable more robust inference to be made from small samples about an underlying probability distribution;
- They can yield more efficient parameter estimates than maximum likelihood estimates (in the sense that they make better use of the available data);
- They characterise a wider range of distributions than is possible with conventional moments;
- Small sample bias of L-moment estimators is generally less than conventional moment-based estimators;
- L-moments can be used to specify a distribution even when some of its conventional moments do not exist.

Since their introduction 30 years ago, L-moments have been widely used by hydrologists for flood-frequency analyses (Kjeldsen et al. 2002, Kroll & Vogel 2002, Lim & Lye 2003).

Mathematical and computational details for estimating the parameters of the *Burr III* distribution using L-moments are given in Appendix E. Although we have not undertaken any comprehensive testing, the following example suggests this may be a productive avenue for further investigation with a view to using in future releases of software for fitting SSDs.

Example: `ssddata` metolachlor dataset.

This dataset consists of 21 measurements of metolachlor (an herbicide) concentrations in freshwater (Figure 28). These were comprised of chronic toxicity values µg/L for fish, macrophytes, and microalgae. Metolachlor is moderately persistent to persistent in the environment and can impact birds, small mammals, and endangered fish. Further details may be found in (ANZG 2020).

Using `Burrlioz` we obtain the following MLEs for the *Burr III* distribution:
$\hat{b} = 776.232;\ \hat{c} = 0.970;$ and $\hat{k} = 0.416$. With these parameter estimates, the HC estimates are:
HC1 = 0.0085; HC5 = 0.46; HC10 = 2.58; and HC20 = 14.65.

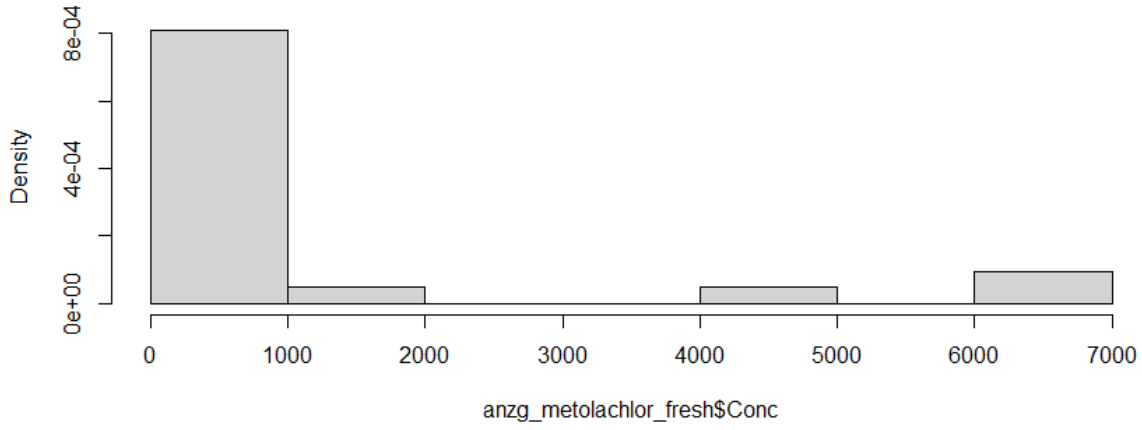


Figure 28. Histogram of metolachlor concentrations in freshwater.

Given the wide range of metolachlor toxicity values (0.53 to 6,528), it is more convenient to work with the (natural) log-transformed data. The log-transformed data range from -0.635 to 8.784. The *Burr III* distribution cannot be fitted directly to these transformed data due to the negative values. To overcome this, the log-transformed values were subtracted from 10. Thus, in terms of the original data (*x*), the transformed values (*y*) were obtained as $y_i = 10 - \ln(x_i)$. The first four sample L-moments for the y values are 5.296, 1.611, 0.175, and 0.147. A comparison of the empirical and fitted distributions is shown in Figure 29. The HC estimates from the *L*-moment fit are: HC1=0.021; HC5=0.73; HC10=2.90; and HC20=12.01.

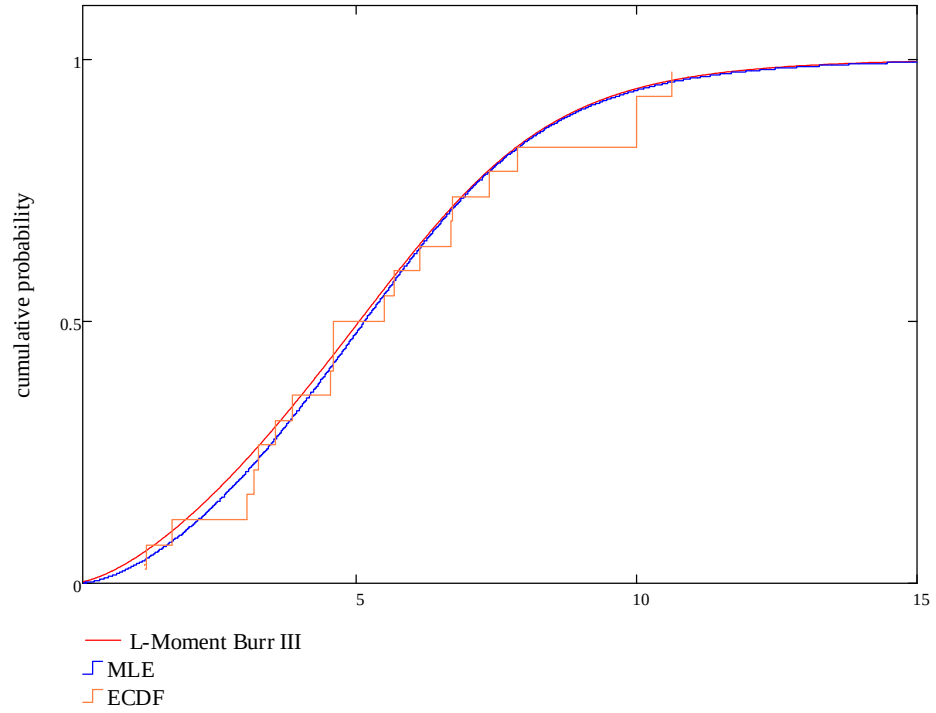


Figure 29. Empirical *cdf* of transformed metolachlor toxicity data together with fitted *Burr III* distributions using MLEs and L-moment parameter estimates.

The fits and HC estimates are similar but not identical – this is to be expected. By way of comparison, the `ssdtools` package gave the following: HC1=0.093; HC5=0.74; HC10=2.48; and HC20=11.8 which are similar to those obtained using the *L*-moment estimates.

Although no general conclusions can be drawn from this limited analysis, we believe that *L*-moments may have a role to play in SSD modelling – if only to provide quickly and conveniently a 'good' set of initial parameter estimates to supply to software tools that estimate parameters iteratively.

Suggestion for further investigation: utility of L-moments

> The use of *L*-moments be explored: (i) as a possible method of establishing initial parameter estimates for iterative MLE techniques; and/or (ii) to (optionally) provide an alternative parameter estimation framework for SSD model-fitting.

3.2 Closed form estimation (*Burr III*)

As mentioned in Section 2.1, one of the appealing properties of the *Burr* family of distributions is the fact that the *cdf* (see Equation 2) is 'invertible' – meaning there is a closed-form expression for quantile estimation. The p^{th} quantile for the BurrIII is given by Equation 8.

$$q(p;b,c,k)=\frac{b}{\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{\frac{1}{c}}} \qquad 0<p<1;\ b,c,k>0 \tag{8}$$

Thus, while the point estimate for the HCx is a straightforward computation using Equation 8 with $p=x/100$, the determination of a confidence interval is not straightforward with `Burrlioz` resorting to computationally-intensive bootstrapping methods to achieve this.

As part of our evaluation of the *Burr III* distribution, we have derived a closed-form expression for the variance-covariance matrix of the parameter estimates $\left\{\hat{b},\hat{c},\hat{k}\right\}$ which can be coupled with the delta-method to provide a good approximation to the standard error of the HCx obtained from Equation 8. Note that only main results are presented here, with details of the mathematical derivation provided in Appendix F.

The variance-covariance matrix for $\hat{\Theta}^T=\left[\hat{b},\hat{c},\hat{k}\right]$ is given by Equation 9.

$$Cov\left[\hat{\Theta}\right]=\mathcal{I}\left(\hat{\theta}\right)^{-1}=\Omega_{\hat{\theta}} \tag{9}$$

where $\mathcal{I}\left(\hat{\theta}\right)$ is the *information matrix*. The elements of $\mathcal{I}$ are complex and are given in Appendix F. Using the *delta method*, an estimate of the variance of the estimated HCx is given by Equation 10.

$$Var\left[Q\left(\underline{\hat{\Theta}}\right)\right]\simeq\nabla\left(\underline{\hat{\Theta}}\right)^T\Omega_{\underline{\hat{\Theta}}}\nabla\left(\underline{\hat{\Theta}}\right) \tag{10}$$

where $Q\left(\underline{\Theta}\right)=q\left(p;\underline{\Theta}\right)$ for some p and $\underline{\Theta}=\left\{b,c,k\right\}$ and $\nabla_{\underline{\Theta}}(\bullet)$ is the gradient vector given by Equation 11.

$$\begin{aligned}\nabla_{\underline{\Theta}}\left(\Theta\right)^T&=\nabla_{b,c,k}\left(b,c,k\right)^T\\&=\left\{\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{-\frac{1}{c}},\frac{b}{c^2}\ln\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{-\frac{1}{c}},\frac{b}{ck^2}\ln\left(\frac{1}{p}\right)\left(\frac{1}{p}\right)^{\frac{1}{k}}\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{-\frac{1}{c}-1}\right\}\end{aligned} \tag{11}$$

Example: `ssddata` *cadmium dataset*

`Burrlioz` parameter estimates for this sample of *n=36* toxicity values are:

$\left\{\hat{b} = 0.00119; \hat{c} = 0.468; \hat{k} = 30.007\right\}$.

Using Equations provided in Appendix F, we obtain:

$$\mathcal{I}\left(\hat{\theta}\right)^{-1} = \begin{pmatrix} 5.272\text{x}10^{-4} & 1.814\text{x}10^{-3} & -5.706 \\ 1.814\text{x}10^{-3} & 9.557\text{x}10^{-3} & -19.018 \\ -5.706 & -19.018 & 6.189\text{x}10^{4} \end{pmatrix}.$$

As an aside, we can compute the correlation matrix, $\mathcal{R}$ of the parameter estimates from $\mathcal{I}\left(\hat{\theta}\right)^{-1}$:

$$\mathcal{R} = \begin{pmatrix} 1.0 & 0.808 & -0.999 \\ 0.808 & 1.0 & -0.782 \\ -0.999 & -0.782 & 1.0 \end{pmatrix}$$

from which we observe the near perfect (negative) correlation between the estimates of *b* and *k* – an issue that was noted in Section 2.1. From Equation 11 we have:

$\nabla_{\underline{\Theta}}\left(\hat{\underline{\Theta}}\right)^{T} = \begin{bmatrix} 124.049 & -1.516 & 0.011 \end{bmatrix}$ and with p = 0.05 in Equation 9 we obtain

$Var\left[Q\left(\hat{\underline{\Theta}}\right)\right] \simeq 3.955 \text{ x } 10^{-3}$ and hence $SE\left[Q\left(\hat{\underline{\Theta}}\right)\right] \simeq 0.063$. The *estimated* HC5 *fro*m Equation 8 is $\widehat{HC_5} = 0.147$ and thus an approximate $(1-\alpha)100\%$ confidence interval is:

$$\widehat{HC_5} \pm t_{n-3,\left(1-\frac{\alpha}{2}\right)} \cdot SE\left[\widehat{HC_5}\right] \tag{12}$$

With $\alpha = 0.05$ in Equation 11 we have an approximate 95% CI for the true HC5 $= \{0.019; 0.275\}$.

By way of comparison, `Burrlioz` gives $\widehat{HC_5} = 0.147$; $SE\left[\widehat{HC_5}\right] = 0.0975$;and 95% CI $= \{0.0150; 0.192\}$.

This example demonstrates that, to a first-order approximation, the formulae presented in this section provide a reliable means of estimating the standard error of an HCx thereby eliminating the need to use the more computationally intensive bootstrapping approximation.

3.3 Parametric versus non-parametric bootstrapping

`Burrlioz 2.0` uses a non-parametric bootstrap method to obtain confidence intervals on the HC estimate. Non-parametric bootstrapping is carried out by repeatedly resampling the raw data with replacement, and refitting the distribution many times. The 95% confidence limits are then obtained by calculating the lower 0.025th and upper 0.975th quantiles of the resulting HC estimates across all

the bootstrap samples (typically >1000). This type of bootstrap takes into account uncertainty in the distribution fit based on uncertainty in the data.

The `ssdtools` package by default uses a parametric bootstrap. Instead of resampling the data, parametric bootstrapping draws a random a set of new data (of the same sample size as the original) from the fitted distribution to repeatedly refit the distribution. Upper and lower 95% bounds are again calculated as the lower 0.025^{th} and upper 0.975^{th} quantiles of the resulting HC estimates across all the bootstrap samples (again, typically >1000). This will capture the possible uncertainty that may occur for a sample size from a given distribution, but it assumes no uncertainty in that original fit, so it is not accounting for uncertainty in the input data.

The new TMB version of `ssdtools` has the capacity to do bootstrapping either using the `Burrlioz` non-parametric method, or the original parametric method of `ssdtools` (based on `fitdistrplus`), and we used this functionality to examine bias and compare the resulting coverage of the two bootstrapping methods.

We found that both the non-parametric and parametric bootstrapping methods result in very similar estimates of the confidence intervals across all the simulation data, with the difference between the two methods converging to near zero at large samples sizes, particularly for the high PC values (i.e.,20% protection, Figure 30). There is a tendency for the lower bound estimate to be higher for the non-parametric than the parametric bootstrap method when sample sizes are low, and this was the case across all three simulated distributions (Figure 30).

Approximate coverage was actually very low in many cases for the `ssd_fit_burrlioz()` estimates regardless of the bootstrapping method used (Figure 31). Coverage was always higher for the parametric bootstrap method compared to the non-parametric bootstrap method – although this difference is generally marginal (Figure 31). Therefore, we recommend that the parametric bootstrapping currently employed by `ssdtools` should be the preferred bootstrapping method for estimating confidence intervals.

Coverage is often better at sample size 8 – presumably because at that sample size the algorithm uses the *log-logistic* distribution that does a reasonable job of representing the data generated by all three distributions (Figure 31). Coverage patterns are very complex, probably because of the inter relationships between sample size and the resulting fitted distribution (Figure 31), and these issues are discussed in more detail in Sections 2.5.3 and 2.5.4.

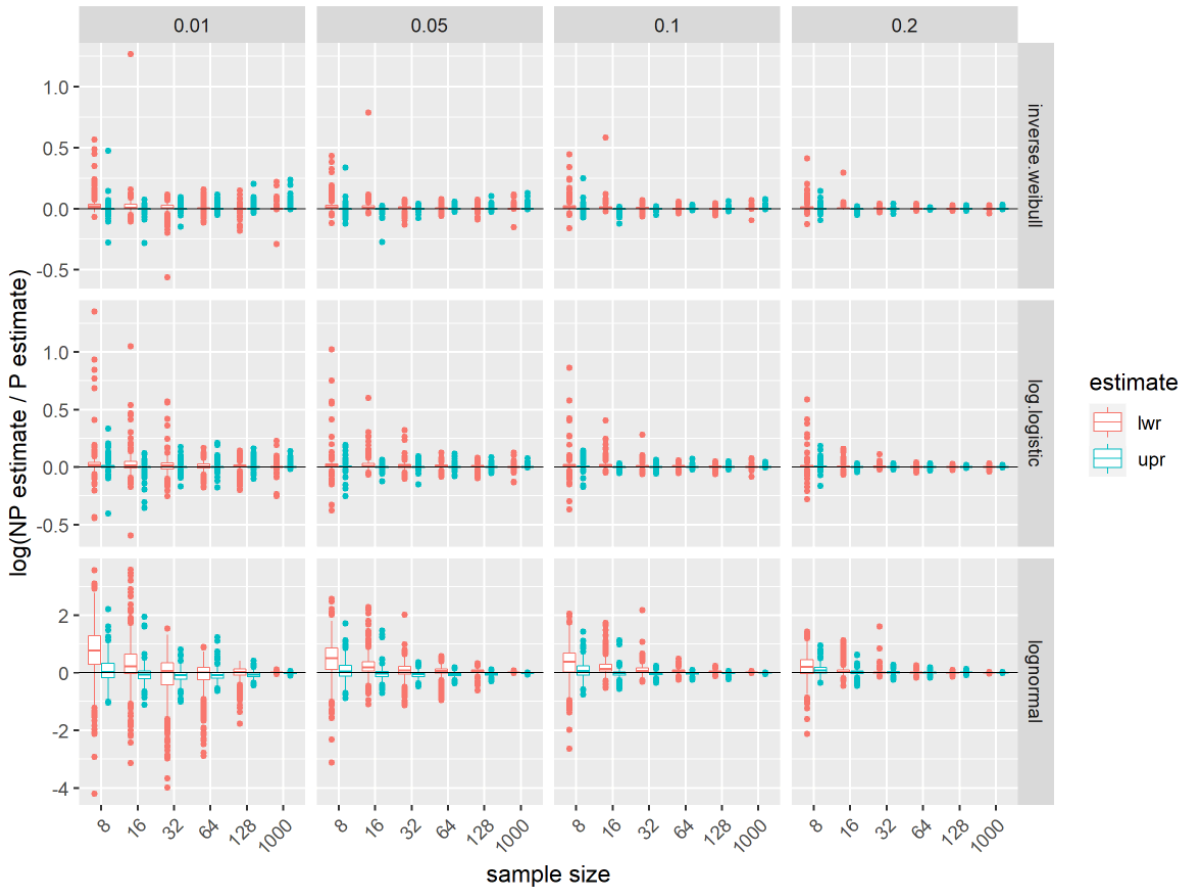


Figure 30. Log ratio of the non-parametric versus the parametric bootstrap method for estimating lower and upper bound (lwr and upr) confidence intervals using the `ssd_fit_burrlioz()` function implemented in `ssdtools`. Data are based on the simdat3 simulation dataset (Simulation Study 1).

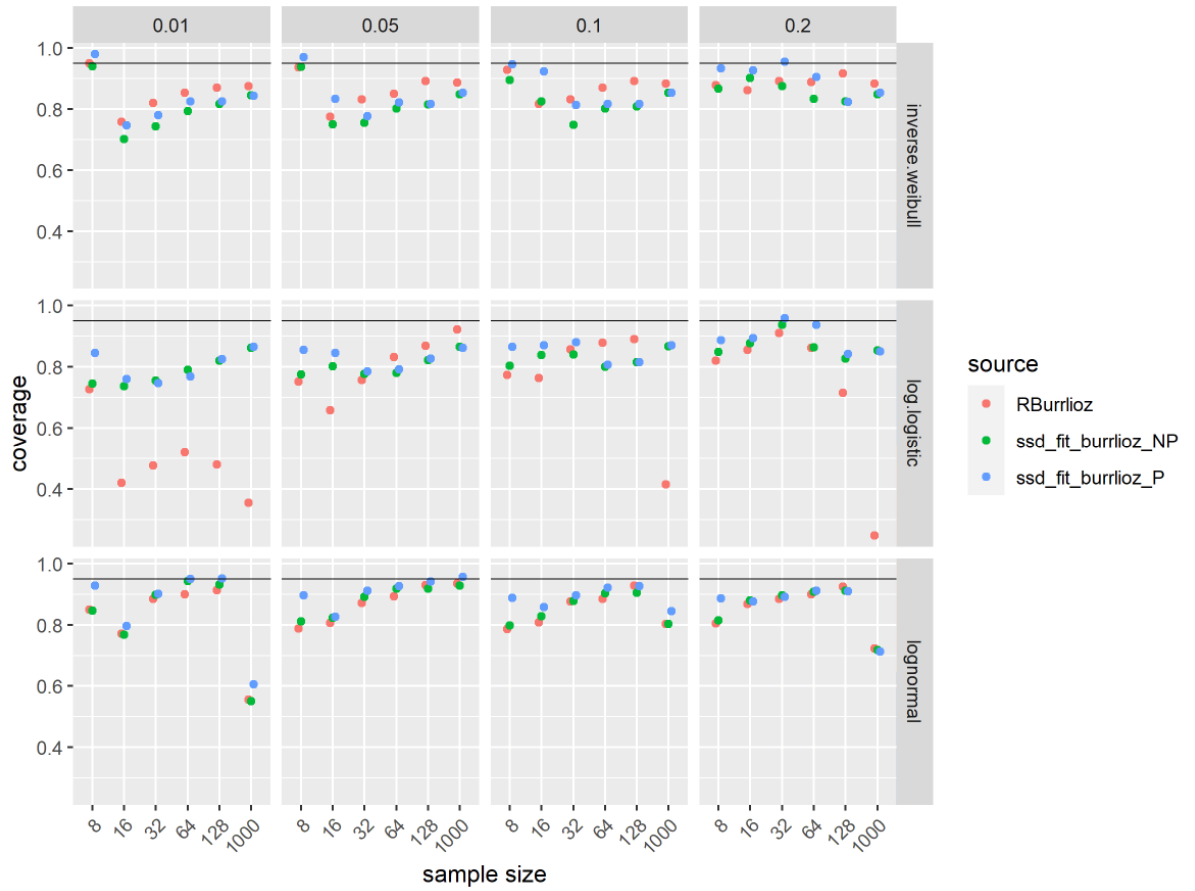


Figure 31. Approximate coverage estimates for a nominal 95% confidence intervals across four different species protection levels (plot columns - p = 0.01, 0.05, 0.1 and 0.2) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the bootstrapping method (parametric versus nonparametric). All fits were done using the `ssd_fit_burrlioz()` function in `ssdtools`. Data are based on the simdat3 simulation dataset (Simulation Study 1).

4 DISCUSSION

4.1 Comparing SSD methodologies

Comparisons across the various estimation methods suggested that for the most part HC estimates are generally quite similar, regardless of the method used – with strong relationships observed between methods for the benchmark datasets, as well as our simulated data. There was, however, evidence of bias associated with some methods, with the result being low realised coverage of the HC 95% confidence interval and a high probability that derived HC values are not robust.

From our Simulation Study 1 we found that bias is low when data are fitted using the same distribution as the parent distribution and shrinks to near zero as sample size increases. The sample size required to ensure that the nominal 95% confidence interval has actual coverage close to the 95% nominal level differs depending on both the simulating parent distribution, as well as the fitting distribution. Some distributions, such as the *log-logistic*, tend to produce very wide confidence bands, particularly at low

sample sizes. Such distributions can yield good coverage at sample sizes as low as 8, even when applied to data simulated using alternative distributions, albeit at the cost of very high estimation uncertainty. Where the fitting distribution is the same as the simulating distribution, samples sizes in the order of >60 are required to yield the notional 95% coverage, and such large sample sizes are rarely available in practice.

Where the fitting distribution is not the same as the generating distribution, bias can be extremely high, and this was explored in both Simulation Studies 1 and 2. When bias is high, increasing sample size reduces the confidence interval widths. This interplay results in the sometimes counter-intuitive outcome that coverage declines with increasing sample size. The complex interplay between bias and confidence interval width and the relationship with sample size, means that the safest option is to use the model averaging approach that includes all (or at least a large range, see more below) of the available distributions. For simulations for which the parent distributions are included in the 'all' available set, this approach is always optimal, because it yields low bias and good coverage in all cases. The more distributions that are included in this set, the more likely one of them will closely reflect the underlying generating distribution of the data, yielding lower bias and better coverage of resulting estimates. Of course, there is also the issue that included distributions need to be sufficiently different to avoid the potential issue associated with over-weighting a particular shape (Fox et al. 2021). However, the sometimes very poor performance of the model averaging approach when the true generating distribution is not included in the set considered suggests that the most robust approach will be to include all available distributions that can capture different shapes, providing that their estimation is numerically stable and relatively unbiased (see more below). Consequently, it seems important to include distributions that may, for the most part, produce very close estimates on average. A good example would be the *log-logistic* and *log-normal*, which can appear very similar in practice. Because the *log-logistic* distribution can have a longer left tail, a *log-normal* fit to such data can be quite biased and yield very low coverage when estimating very low species protection values.

The TMB version of `ssdtools` includes a large range of potential distributions, and if we assume that the true underlying distribution is represented by at least one of these, a model averaging approach including all of them is quite robust. When we generate data using a distribution not represented by any within this set (as we did in Simulation Study 2), the model averaging approach can yield biased results and low coverage of 95% confidence intervals even at very high sample sizes. Across the six simulated datasets that were used to investigate this issue, using the 'all' model average set resulted in HC1 and HC5 coverage that was often similar or greater than the 'default' model averaging method using the restricted range of distributions. For the HC10 and HC20 estimates, there are cases where the 'default' set provides better coverage than the 'all' set (see silver, uranium and glyphosate, Figure 19). Although not shown here, this trend was not driven by the inclusion of the two mixture-model distributions in the 'all' model average candidate set, and the observed pattern simply reflects the slightly better fit (i.e., less bias) of this 'default' set through the region of the curve representing these two larger HC estimates (Figure 18). Given that this portion of the curve is generally of the least interest, particularly when considering that the Australian/NZ guidelines use either the HC5 or HC1 for deriving guideline values, it seems clear that the 'all' model set should be preferred, even when the underlying synthetic data were generated by a family of distributions that are not included in this model set.

Notably, RBurrlioz performed worse than either of the 'default' or 'all' model averaging methods in all cases, both in terms of coverage (Figure 19) and bias (Figure 20), reflecting its reduced flexibility and capacity to mimic the underlying data (Figure 18). So regardless of whether the true distribution is included in the model average set or not, model averaging is clearly a superior method to that currently being used via Burrlioz 2.0.

4.2 Mixture distributions

Including mixture distributions in simulations based on unimodal parent distributions generated outcomes at least as good as a model averaging approach based only on unimodal distributions. Importantly, including mixtures did not lead to any systematic bias (i.e., HC values that were consistently either over- or under- estimated). This suggests there is generally no 'harm' in including mixtures in the candidate set, even when data are known to be unimodal.

Including mixture distributions had little to no effect when the unimodal parent distribution was included in the candidate model set. Although surprisingly, including mixture distributions in the candidate set substantially improved coverage in some cases when the unimodal parent distribution was not included as part of the model set. This suggests that even where there is no strong evidence of bimodality, including mixtures can potentially yield more robust HC estimates when the underlying data do not come from any single unimodal distribution from the candidate list. This is not surprising and simply reflects their enhanced degree of flexibility in distributional shape over the sample domain.

For very small sample sizes, neither of the mixture distributions received any substantial support, reflecting the large penalty on the number of estimated parameters associated with AICc. Similarly, the *Burr III* distribution also received little support at these small sample sizes, presumably also a consequence of the larger parameter count for this distribution. For datasets having two, well-separated modes, a 2-component mixture-model was preferentially chosen over any single distribution for sample sizes as low as 16. For simulated mixture datasets showing slightly less extreme bimodality, much larger samples sizes were required for the mixture distributions to attain a high weight in the candidate set.

With respect to confidence interval coverage, the inclusion of mixture-models can lead to better outcomes – namely *actual* coverage closer to the *nominal* confidence with intervals that are narrower than those afforded by univariate models. General conclusions about the utility of mixture-models are difficult to make since this depends on the interplay between sample size, distributional fit and separation of the modes for any given dataset. What is clear, however, is that statistical mixture-models provide a more comprehensive 'arsenal' of distributional shapes and are a natural modelling framework for toxicity data generated by two or more distinct processes.

Suggestion for further investigation: Additional mixture distributions

> Investigate the utility and desirability of including other mixture distributions, such as a *lognormal – log-logistic*.

4.3 CI and SE estimation methods

The default method for obtaining HC confidence intervals in `ssdtools` is the parametric bootstrap. This differs to `Burrlioz 2.0`, which uses the non-parametric bootstrap. Results from our Simulation Study show that the results of the two methods are in fact very similar, although coverage was always slightly higher for the parametric bootstrap (Figure 31), suggesting that this should be the preferred approach to adopt for any future tool.

We explored the use of alternative methods for obtaining the confidence interval of HC estimates. This included using the closed-form expression for the variance-covariance matrix of the parameters of the *Burr III* distribution, coupled with the delta-method, as well as an alternative bootstrap method for the *inverse Pareto* distribution based on statistical properties of the parameters. In both cases, it appeared that these methods can give results similar to other traditional bootstrapping approaches in much less time, and are therefore potentially worth further investigation. However, implementation of such methods across all the distributions now available in `ssdtools` would be a substantial undertaking.

An alarming outcome of the coverage comparisons for the "`Burrlioz`" -like methods (R`Burrlioz` and `ssd_fit_burrlioz()` with parametric and non-parametric bootstrapping) is that all of them show very low coverage at low to moderate sample sizes - which are the sizes typical of SSD datasets. Ironically, for these methods, higher coverage was obtained at the lowest examined sample size of 8 for our simulated data (Figure 31). This outcome is likely due to the use of the *log-logistic* distribution for sample sizes <8 in the `Burrlioz` framework, which apparently yields a relatively good fit to these simulated data. Of course, simulating data using an alternative distribution would likely yield a different outcome. Our coverage estimates from the simulation results suggests that overall coverage of the 95% confidence intervals of the HC values can be substantially improved using a model averaging approach that includes all the available distributions.

4.4 Finalise default distributions

Deciding on a final default set of distributions to adopt using the model averaging approach is not trivial, and we acknowledge that there is probably no cut and dry 'solution' to this issue. However, the default set should be underpinned by a guiding principle of *parsimony*, i.e., the set should be as large as is necessary to cover a wide variety of distributional shapes and contingencies but no bigger. Further, the default set should result in model averaged estimates of HCx values that: 1) minimise bias; 2) have *actual* coverages of confidence intervals that are close to the nominal level of confidence; 3) estimated HCx and confidence intervals of HCx are robust to small changes in the data; and 4) represent a positively continuous distribution that has both right and left tails.

4.4.1 Sensitivity and numerical stability issues

While it seems reasonable to aim for a distribution set that results in model average estimates of HCx that are insensitive to small changes in the data, this may be difficult to achieve in practice. Estimation of low-order quantiles using small sample sizes is fraught, and some would say, statistically reckless. While the extreme left tail of most distributions will always show some sensitivity, there are some

distributions that appear to be inherently unstable. Practical experience has revealed that the *Gompertz* and *Burr III* distributions are particularly sensitive to small changes in the data as well as the initial starting values for parameter estimates. We examined how frequently each of the distributions currently included in the development version of `ssdtools` were able to successfully converge across the datasets used in both simulation studies (Figure 32). The *gamma*, *inverse Pareto*, *log-Gumbel* (*inverse Weibull*), *log-logistic and log-normal* are almost always reliably estimated (Figure 32). Of the remaining distributions, the *Gompertz* is the only distribution that fails to yield consistent convergence for any of the tested datasets, at any sample size (Figure 32).

sample_size	source/parent	gamma	invpareto	lgumbel	llogis	lnorm	burrIII3	gompertz	llogis_llogis	lnorm_lnorm	weibull
	A) Simulation study 1 (log-logistic, inverse Weibull, log-normal)										
8	inverse.weibull	0.98	0.98	1.00	1.00	1.00	0.02	0.61	0.64	0.52	0.65
8	log.logistic	0.99	0.99	1.00	1.00	1.00	0.07	0.63	0.66	0.60	0.66
8	lognormal	1.00	1.00	1.00	1.00	1.00	0.38	0.25	0.81	0.79	0.47
16	inverse.weibull	0.98	0.98	1.00	1.00	1.00	0.04	0.62	0.56	0.47	0.61
16	log.logistic	0.99	0.99	1.00	1.00	1.00	0.13	0.64	0.53	0.47	0.67
16	lognormal	1.00	1.00	1.00	1.00	1.00	0.72	0.14	0.70	0.64	0.34
32	inverse.weibull	0.97	0.99	1.00	1.00	1.00	0.04	0.63	0.56	0.46	0.63
32	log.logistic	0.97	0.99	1.00	1.00	1.00	0.18	0.65	0.49	0.48	0.66
32	lognormal	1.00	1.00	1.00	1.00	1.00	0.90	0.07	0.59	0.53	0.22
64	inverse.weibull	0.99	0.98	1.00	1.00	1.00	0.04	0.62	0.61	0.47	0.65
64	log.logistic	0.99	0.99	1.00	1.00	1.00	0.20	0.65	0.41	0.41	0.68
64	lognormal	1.00	1.00	1.00	1.00	1.00	0.98	0.02	0.53	0.42	0.16
128	inverse.weibull	0.99	0.97	1.00	0.99	1.00	0.05	0.62	0.72	0.47	0.65
128	log.logistic	1.00	0.95	1.00	1.00	1.00	0.21	0.64	0.38	0.40	0.69
128	lognormal	1.00	1.00	1.00	1.00	1.00	1.00	0.01	0.50	0.36	0.11
1000	inverse.weibull	0.96	0.84	1.00	0.99	0.89	0.02	0.60	0.95	0.68	0.67
1000	log.logistic	0.96	0.81	1.00	1.00	0.90	0.21	0.63	0.35	0.53	0.68
1000	lognormal	1.00	1.00	1.00	1.00	0.86	1.00	0.00	0.78	0.33	0.05
	B) Simulation study 2 (Johnson family)										
100	anon_a	1.00	1.00	1.00	1.00	1.00	0.90	0.00	0.80	0.82	0.00
100	anon_b	1.00	1.00	1.00	1.00	1.00	1.00	0.04	0.80	0.60	0.04
100	ccme_boron	1.00	1.00	1.00	1.00	1.00	0.14	0.88	1.00	0.98	1.00
100	ccme_glyphosate	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.88	0.88	0.00
100	ccme_silver	1.00	1.00	1.00	1.00	1.00	0.00	0.04	0.90	0.92	1.00
100	ccme_uranium	1.00	1.00	1.00	1.00	1.00	0.16	0.00	0.64	0.56	0.00
16	anon_a	1.00	1.00	1.00	1.00	1.00	0.54	0.00	0.90	0.82	0.00
16	anon_b	1.00	1.00	1.00	1.00	1.00	0.62	0.16	0.64	0.58	0.38
16	ccme_boron	1.00	1.00	1.00	1.00	1.00	0.02	0.68	0.90	0.92	1.00
16	ccme_glyphosate	1.00	1.00	1.00	1.00	1.00	0.04	0.00	0.84	0.88	0.00
16	ccme_silver	1.00	1.00	1.00	1.00	1.00	0.18	0.26	0.80	0.82	1.00
16	ccme_uranium	1.00	1.00	1.00	1.00	1.00	0.42	0.00	0.74	0.68	0.00
49	anon_a	1.00	1.00	1.00	1.00	1.00	0.76	0.00	0.82	0.82	0.00
49	anon_b	1.00	1.00	1.00	1.00	1.00	0.94	0.06	0.64	0.56	0.22
49	ccme_boron	1.00	1.00	1.00	1.00	1.00	0.26	0.84	0.92	0.92	1.00
49	ccme_glyphosate	1.00	1.00	1.00	1.00	1.00	0.02	0.00	0.88	0.92	0.00
49	ccme_silver	1.00	1.00	1.00	1.00	1.00	0.04	0.04	0.86	0.90	1.00
49	ccme_uranium	1.00	1.00	1.00	1.00	1.00	0.30	0.00	0.58	0.60	0.00
8	anon_a	1.00	1.00	1.00	1.00	1.00	0.24	0.00	0.84	0.82	0.00
8	anon_b	1.00	1.00	1.00	1.00	1.00	0.50	0.46	0.84	0.86	0.66
8	ccme_boron	1.00	1.00	1.00	1.00	1.00	0.14	0.64	0.96	0.96	1.00
8	ccme_glyphosate	1.00	1.00	1.00	1.00	1.00	0.10	0.00	0.88	0.92	0.00
8	ccme_silver	1.00	1.00	1.00	1.00	1.00	0.16	0.38	0.82	0.82	1.00
8	ccme_uranium	1.00	1.00	1.00	1.00	1.00	0.24	0.00	0.80	0.80	0.00

Figure 32. The proportion of simulated datasets/iterations for which the `ssdtools`-fitted distributions were able successfully converge. Note that for the *Burr III* and both mixture distributions, this proportion includes distributions that may have returned a result, but this was at one of the bounds of the parameter set (i.e. shape1 or shape2 = 20 or 0.05 in the case of *Burr III*, or p=<0.2 in the case of mixtures).

While the numerical instability issues associated with the *Burr III* distribution yields low convergence (Figure 32), this is only in the context of defining lack of convergence to also include the situation when one or more of the parameters are at the bounds required for numerical stability (see Section 2.1). Indeed, the proportion of true non-convergence for the *Burr III* distribution is actually very low (Table 6, 1% or less). While "convergence" of the two statistical mixture distributions is relatively low (Figure

32), this is due to the default settings of `ssdtools` at the time simulations were run, which set a lower bound of 0.2 on the mixing proportion. A mixture distribution was deemed not to have converged when a parameter estimate was at this bound. This default behaviour has been changed (see Appendix I: Additional analyses assessing the recommended distribution set). Using the 24 example datasets from the `ssddata` package. we compared the difference in AICc based model weights and HCx estimates when the lower bound for the *log-normal-log-normal* mixing parameter was 0.2 (pmix = 0.2) and the distribution was deemed not to have converged if the pmix estimate hit this bound (`at_boundary_ok = FALSE`) (the default settings used in the earlier simulations studies) with the results when the lower bound on pmix was set to 0 (pmix = 0) and the distribution was retained in the set even when at the boundary condition (`at_boundary_ok = TRUE`, see Appendix I) While there were slight differences for some datasets in the weights for the mixture distribution (Table I-1), ultimately there was very little difference in the estimated HCx values (Figure I-6). Allowing the mixing parameter (pmix) to go to 0 and setting `at_boundary_ok = TRUE` did result in a slightly lower proportion of successful bootstrap iterations in some cases, meaning that to obtain confidence bands on the HCx the argument the required proportion of successful bootstrap iterations (`min_pboot`) had to be lowered from the default 0.99 (Table I-2). Further investigation is required to understand and resolve this issue in the current development version of `ssdtools`.

The only other distribution that frequently did not converge for our simulated data was the *Weibull* (Figure 32). Given that there are no currently implemented boundary conditions for either the *Weibull* or the *Gompertz* distributions, the source of this lack of convergence warrants further investigation before their inclusion into the default distribution set should be considered. Investigations into convergence issues for both of these distributions indicated that the original `fitdistplus` version of `ssdtools`, while also failing to converge reliably for the Gompertz distribution, works well for the *Weibull* distribution. Troubleshooting the TMB based development version of `ssdtools` for the *Weibull* distribution indicated there were issues with finding appropriate starting values and these have now been resolved using a more sophisticated mathematical algorithm rather than arbitrary numerical values.

Table 6. The proportion of datasets/iterations for which the `ssdtools` -fitted *Burr III* distribution has parameter estimates at one (or more) of the bounds (At bounds), failed to converge entirely (True non-convergence), or was able to successfully converge without reaching one of the bounds (Converged and not at bounds). Results are summarized across both the Study 1 and Study 2 simulated datasets.

Sample size	At bounds	True non-convergence	Converged and not at bounds
8	0.75	0.04	0.21
16	0.67	0.01	0.33
32	0.61	0.00	0.39
49	0.55	0.01	0.44
64	0.58	0.00	0.42
100	0.53	0.01	0.46
128	0.56	0.00	0.44

How best to include distributions into the model averaging framework in the case where one (or more) parameter estimates meet boundary conditions is an unresolved issue. For the *Burr III* distribution,

meeting the boundary for either the shape1 and shape2 parameters suggests that the underlying distribution is one of the two limiting forms of the *Burr III* (*inverse Pareto*, or *inverse Weibull*). If these limiting distributions are in the default set of distributions over which model averaging is to occur, there are two options: 1) acknowledge that the *Burr III* is at one of its limiting forms, deem it to be non-converged and leave it out of the final set used in the model average; or 2) allow the bounded distribution to also remain in the set, and assume that the extra parameter count will ensure there is insufficient weight such that it will have limited influence over the model-averaged HC estimate. We examined the relative weight of the *Burr III*, *inverse Pareto* and *log-gumbel* (*inverse Weibull*) distributions as fit using `ssdtools` across all simulated datasets, to determine how the weight of the *Burr III* changes with both sample size and depending on whether any of the parameters were at the bounds (Figure 33).

Across the 2,632 simulated datasets from Simulation Study 1 there were very few instances where the *Burr III* distribution had high weight when one or more of the estimated parameters were at the bounds (left hand panel, Figure 33). The exception was for data generated using the *log-logistic* distribution (left hand panel, Figure 33). However, this result merely reflects the fact that the *log-Gumbel* (*inverse Weibull*) or the *inverse Pareto* provide poor representations of data generated from a *log-logistic* distribution. Were the *log-logistic* to also be included in this model set (as would be done in the usual model-averaging approach) the *Burr III* would be unlikely to have a high relative weight (see Figure 27). When data are generated using the *inverse Weibull* distribution (*log-gumbel*), the weights for this distribution are high and are near one when the *Burr III* fit has one or more parameters at the boundary. For larger sample sizes, a properly converged *Burr III* fit that is not at a boundary can still have substantial weight for data generated using the *inverse Weibull* (left-hand panel, Figure 33).

For the Simulation Study 2 results based on the Johnson family for which none of the included distributions is the parent distribution, weights for the *Burr III* in the case of when one or more parameters are at the boundary are consistently very low (right-hand panel, Figure 33). This outcome suggests that there will be very little material difference between which of the two options above are selected, and any final decision is largely philosophical.

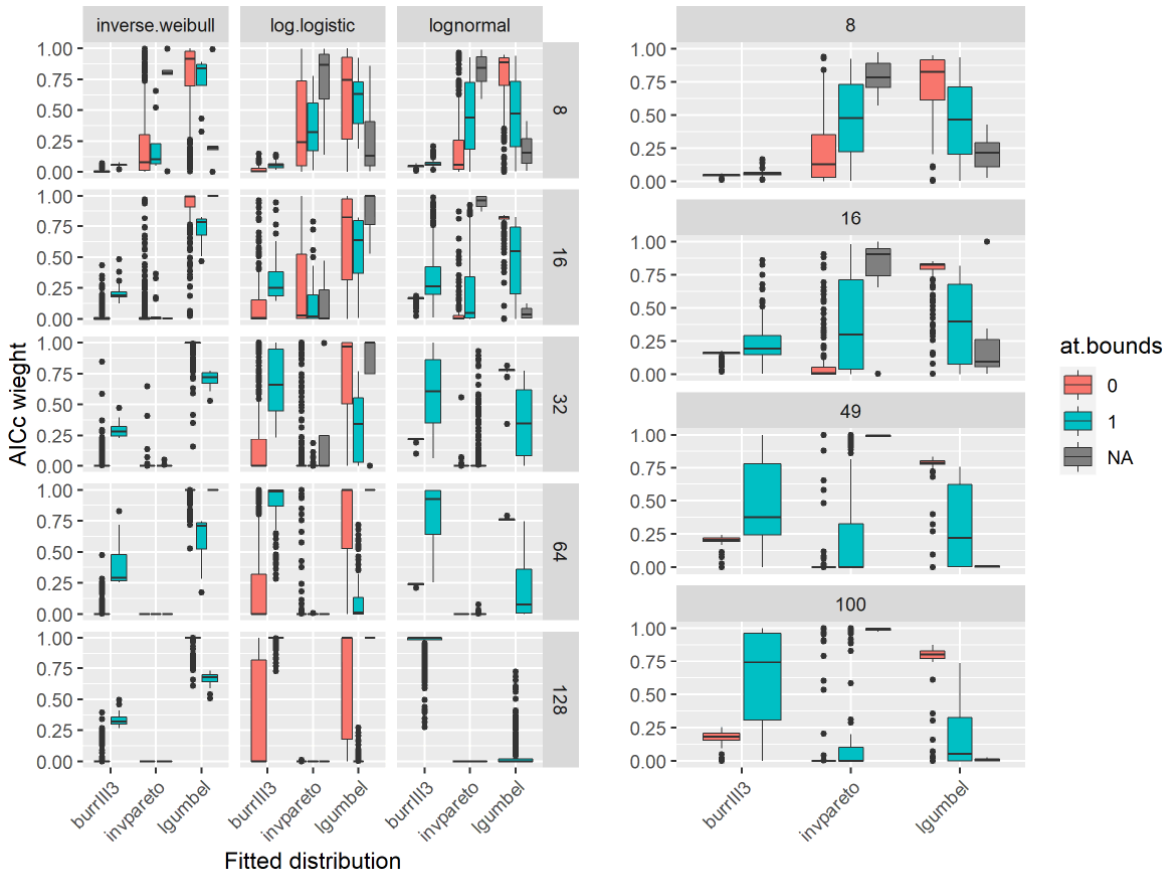


Figure 33. AICc based weights for the *Burr III* (burrIII3), *inverse Pareto* (invpareto) and *log-Gumbel* (lgumbel) distributions fitted using `ssdtools` for the Simulation Study 1 (left-hand plot) and 2 (right-hand plot) datasets. Plot rows show the data sample size. For Simulation Study 1, plot columns show the parent distributions. Boxplot colours show the convergence status of the *Burr III* distribution for the model fit, including 0 (parameters were at one of the defined bounds of shape1 or shape2 = 20 or 0.05), 1 (successful convergence, parameters not at bounds) or NA (non-convergence due to issues unrelated to bounds).

4.4.2 Incorporation of the *inverse Pareto* as a candidate SSD.

While the *inverse Pareto* distribution is implemented in the `Burrlioz 2.0` software, it is important to understand that it is done so only as a limiting distribution. The *inverse Pareto* is not offered as a stand-alone candidate SSD in the `Burrlioz 2.0` software. We have spent considerable time and effort in this report exploring the properties of the *inverse Pareto* distribution, including deriving bias correction equations and alternative methods for deriving confidence intervals. This work has substantial value for improving the current `Burrlioz 2.0` method, and our bias corrections should be adopted when deriving HCx estimates from the *inverse Pareto* where parameters have been estimated using maximum likelihood.

As is the case with the `Burrlioz 2.0` software, we have decided not to include the *inverse Pareto* distribution as a default SSD model but instead reserving its use as a limiting form of the *Burr III* distribution when the numerical calculations suggest this is necessary with the current `Burrlioz` like framework. As a stand-alone SSD, the *inverse Pareto* distribution presents several difficulties. This includes the requirement to make a choice between the so-called 'European' and 'American' versions of the *inverse Pareto* distribution – the first being unbounded and the second being bounded to the

right (which is problematic in the context of ecotoxicology); and potential problems with bootstrapping estimation and inference. In addition, we have observed counter-intuitive results when fitting the inverse Pareto distribution to certain data sets (for example, the ccme_boron dataset in `ssddata`) whereby the likelihood (and hence model-average weight) is high but both a visual inspection of the fitted distribution and the goodness-of-fit statistics reveal a demonstrably poorer fit when compared to other 'less likely' distributions. This issue would require further investigation before this distribution could be confidently included as a standalone distribution in the recommended set.

The European version of the *inverse Pareto* distribution could potentially be added to `ssdtools` and indeed it is already available in the R via the `actuar` package, however this was deemed to be unwarranted. Our experience with *actual* toxicity data sets indicated that there was little difference in the fits provided by the gamma distribution (already in the candidate list) and the European *inverse Pareto* distribution. Further, mathematical theory dictates the use of the American version of the *inverse Pareto* distribution as the limiting form of the *Burr III* under conditions outlined in Section 2.2. It would therefore be confusing to use both versions of the *inverse Pareto* distribution in the `ssdtools` software.

4.4.3 Parsimony – the minimum "optimal" set

Minimising the number of candidate distributions in the default set will also reduce computational time and potentially avoid overweighting. In addition, secondary considerations suggest that the candidate distributions should not be overly complex nor unrealistic in the context of ecotoxicology (an example of the latter is the triangular distribution which has no left or right tails).

Even if a purely theoretical comparison of the performance of different default distribution sets was possible, this would still not extinguish the overarching problem of identifying an 'optimally' minimal set of distributions. This is because there is an infinite number of theoretical distributions from which this selection could be made. When confronted with such problems, statisticians often impose additional constraints to limit the scope of the problem – for example, by only considering distributions that belong to a certain class or 'family' such as the exponential class of distributions. While the exponential class encompasses a wide variety of distributions such as the *normal*, *log-normal*, *logistic*, *log-logistic* etc., it does not include the *Burr* family of distributions which we believe are important for SSD modelling. In any event, a theoretical comparison of default distribution sets is an entirely new proposition and research activity that is not within the scope of the current project. Our assessment of different options for the default distribution set is thus necessarily limited to an examination of the results of computer simulation studies combined with best professional judgement. While this approach has yielded useful results and insights, it is recognised that it is imperfect and subject to experimenter bias. Clearly, our results pertain only to the small number of scenarios that we chose. Other experts may disagree with our choices, methodology, and interpretations. This is inevitable but not pathological. To minimise systematic bias in our evaluation of candidate sets we developed the following three-tiered approach for the inclusion of test datasets:

Round 1: Approximately 2,000 synthetic datasets covering a wide variety of skewness-kurtosis combinations generated from distributions included in the various software tools being evaluated. These were the *log-normal*, *log-logistic*, and *inverse Weibull*.

Round 2: A small number of 'replica' datasets that were generated using the Johnson family of distributions (Johnson 1949). Six member datasets from the `ssddata` collection were used as the basis of the 'replica' data by determining the member of the Johnson family that had the same first four moments as those of the `ssddatasets`. This approach allowed us to replicate the characteristics of real ecotox datasets while providing access to the true, theoretical HCx values.

Round 3: A small number of "mixture" datasets based on a *log-normal* and *log-logistic* that were used to evaluate how the weighting of mixtures changes with sample size and assess their practical usability.

Overall, we found the best outcomes in terms of both coverage of confidence intervals for and bias in estimated HCx were obtained using all the available distributions - including the mixture distributions. However, in the interests of parsimony at least some discussion regarding reducing this set is warranted. In the context of avoiding the potential issue of "over-weighting" (Fox et al. 2021) it is prudent to not include distributions that are extremely similar.

Two of the most common distributions used in SSD modelling are the *log-normal* and *log-logistic* distributions. Although these distributions have some common properties (for example, both are always positively skewed) as well as tending to produce similar fits, we observed enough instances where the fits were very different and for this reason both were retained in the default set. This is supported by our first simulation study that showed you can have very low coverage for data generated by each of these distributions, when fitted with each other (Figure 14), and also by the fact that these distributions differ substantially when considered using a Cullen-Frey plot, which shows a distribution's kurtosis (a numerical measure of 'peakedness') as a function of its skewness (Figure 34).

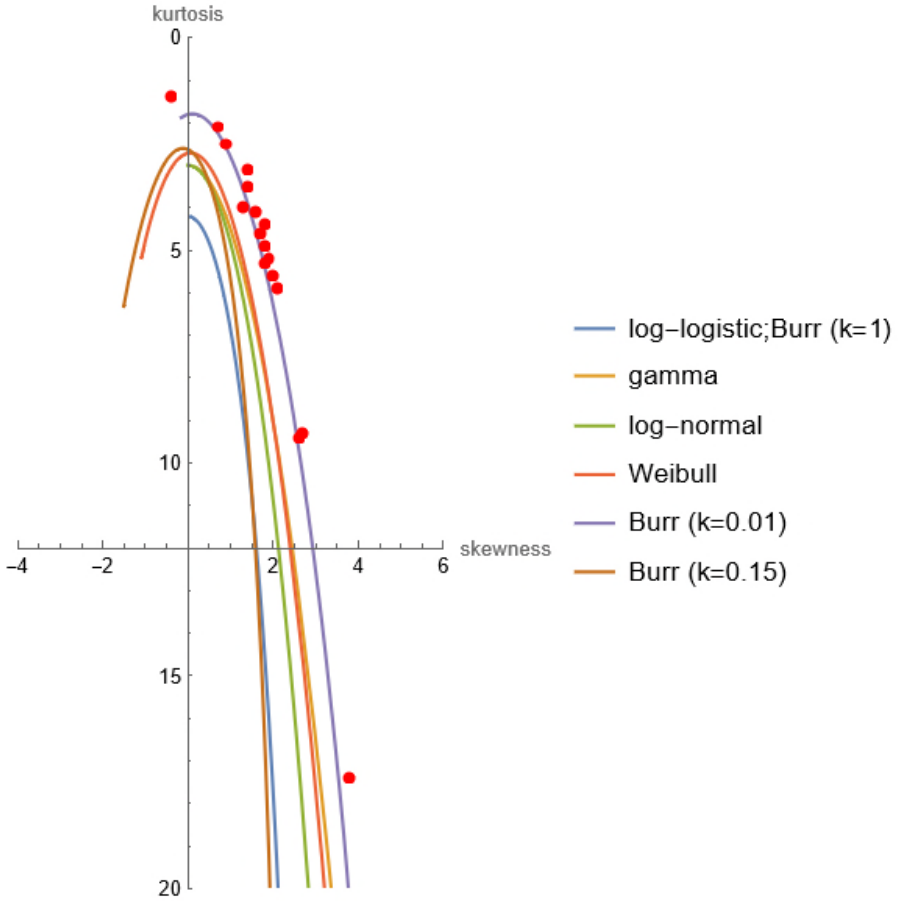


Figure 34. Cullen-Frey plot of four distributions used in SSD modelling together with special cases of the Burr III distribution. The plot shows a distribution's kurtosis (a numerical measure of 'peakedness') as a function of its skewness. Empirical {skewness, kurtosis} pairs for benchmark data sets are also indicated (solid red circles).

It is evident from Figure 34 that the *log-normal* and *log-logistic* distributions cover different parts of the skewness-kurtosis plane and for this reason, both have been included in the default distribution set. However, at this point in time, we recommend that, of the two mixtures currently implemented in `ssdtools`, only the *log-normal-log-normal* mixture be included in the default set. While further investigation into a range of mixtures (including mixtures composed of different statistical distributions, such as the *log-normal* and *log-logistic*) may be warranted, the sample sizes typically available for SSD modelling in practice would likely be insufficient to clearly differentiate between alternative mixtures. We have therefore chosen to use the mixture of *log-normals* by virtue of the simplicity and mathematical tractability of this distribution. Figure 34 also clearly shows that skewness-kurtosis profile lines of the 'standard' distributions considered are bounded by the profile lines of the special cases of the Burr III distribution – again suggesting the Burr III distribution can model a wide variety of both 'standard' and 'non-standard' distributional forms. Perhaps more intriguing, and something for which we currently have no explanation, is the way in which the points corresponding to *actual* benchmark toxicity data sets (available in `ssddata`) lie almost perfectly on the Burr III (*k=0.01)* profile line. It is tempting, if not somewhat fanciful, to conjecture that this observation provides evidence of the existence of a fundamental SSD in ecotoxicology.

We see merit in including distributions that are routinely used in ecotoxicology as this is likely to encourage broader uptake of model-averaging methods and harmonisation of approaches across different jurisdictions. However, including distributions which suffer from convergence and boundary issues may pose a problem in model averaging, as the fitted candidate set will not be consistent among analyses. It seems clear that until convergence issues are resolved, the *Gompertz* distribution cannot be considered in the recommended set of candidate distributions. Convergence issues with the *Weibull* distribution have now been resolved (see above), and this distribution can therefore be considered in the default set. Given the Weibull distribution is one of the few two parameter distributions able to model negatively skewed data its inclusion is clearly warranted.

The long history of use of the *Burr III* in SSD modelling in Australia along with the high flexibility associated with this distribution which can cover a large portion of the skewness-kurtosis space (Figure 34) means that inclusion of the *Burr III* distribution in the candidate model average set would be desirable. As discussed above, there remain unresolved issues associated with the use of limiting forms of the Burr III distribution in the context of model averaging. While our simulations show that this behaviour may be able to be accommodated in a model averaging context providing these limiting distributions form part of the model set (see Section 4.4.1, Sensitivity and numerical stability issues), there are several reasons not to include the *inverse Pareto* (see Section 4.4.2, one of these two limiting forms) as a distribution in its own right in the candidate set. For these reasons, it seems best to exclude the *Burr III* from the recommended candidate set of distributions for the time being. Further work would be required to understand how best to accommodate the *Burr III* into model averaging, given its boundary condition behaviour.

With these considerations, this leaves the *log-normal*, *log-logistic*, *Gamma* (the current default set) plus the *log-Gumbel* (*inverse Weibull*), *Weibull* and *log-normal-log-normal* mixture in the potential candidate set. We explored the outcome on our simulation results of restricting the candidate set to these six distributions, compared to our original analysis using "all" of those available in the TMB version of `ssdtools`. We found that the results (at least for Simulation Study 1) were similar to using all of the available distributions, with only very marginal increases in bias and loss of coverage (see Appendix I: Additional analyses assessing the recommended distribution set). Even under the worst-case scenario based on data simulated using a *Burr III* distribution (which is not in the recommended set), the recommended set performs reasonably well. Overall, it appears that the recommended candidate set of distributions is sufficiently large to provide reasonable results for a wide range of data shapes, whilst ensuring that only numerically stable and defensible distributions are retained.

Based on the results of our comprehensive testing and analysis and in view of the foregoing discussion, we make the following major recommendations:

1. Software

That Australian-New Zealand and Canadian jurisdictions:

- adopt the R-package `ssdtools` (and its on-line implementation `shiny ssdtools`) as the default software tool for fitting SSDs to toxicity data for the purpose of deriving predicted no effects concentration of chemicals in natural aquatic environments.

2. Default distribution set

A. The default list of candidate distributions in `ssdtools` should be comprised of the following distributions:

1. The *log-normal* distribution;
2. The *log-logistic* distribution;
3. The gamma distribution;
4. The *inverse Weibull* (log-Gumbel) distribution;
5. The *Weibull* distribution;
6. The mixture of *two log-normal* distributions

B. That Australian-New Zealand and Canadian jurisdictions:

- agree on a default set of distributions to use with `ssdtools` whether it be those identified in Recommendation 2A above or some other set as may be defined from time to time.

3. Implementation

That Australian-New Zealand:

- encourage and facilitate an expeditious transition to ssdtools and the model averaging method. A period of overlap may be required whereby the results of either Burrlioz or ssdtools can be used and reported. In pursuit of this objective, it is further recommended that the responsible government departments in Australia and New Zealand provide support for education and training initiatives associated with the use of model averaging, ssdtools and the R computing environment.

4. Periodic review and on-going collaboration

That Australian-New Zealand and Canadian jurisdictions:

- Australian-New Zealand and Canadian jurisdictions agree to establish a framework to continue the R&D collaboration on SSD modelling between the two countries that has been initiated by this project. This framework should also provide oversight of periodic reviews, technical evaluations, and resolution of end-user issues.

5 REFERENCES

Aldenberg T, Slob W (1993) Confidence limits for hazardous concentrations based on logistically distributed NOEC toxicity data. Ecotoxicology and environmental safety 25:48-63

ANZECC/ARMCANZ (2000) Australian and New Zealand guidelines for fresh and marine waters. Australian and New Zealand Environment and Conservation Council & Agriculture and Resource Management Council of Australia and New Zealand, Canberra.

ANZG (2018) Australian and New Zealand Guidelines for Fresh and Marine Water Quality. Australian and New Zealand Governments and Australian state and territory governments, Canberra ACT, Australia

ANZG (2020) Toxicant default guideline values for aquatic ecosystem protection: Metolachlor in freshwater. Australian and New Zealand Guidelines for Fresh and Marine Water Quality. Australian and New Zealand Governments and Australian state and territory governments, Canberra, ACT, Australia

Belanger S, Barron M, Craig P, Dyer S, Galay-Burgos M, Hamer M, Marshall S, Posthuma L, Raimondo S, Whitehouse P (2017) Future needs and recommendations in the development of species sensitivity distributions: Estimating toxicity thresholds for aquatic ecological communities and assessing impacts of chemical exposures. Integrated Environmental Assessment and Management 13:664-674

Burnham KP, Anderson DR (2002) Model selection and multimodel inference; A practical information-theoretic approach, 2nd edition. Springer, New York

Burr IW (1942) Cumulative frequency functions. The Annals of mathematical statistics 13:215-232

Carr G, Belanger S (2019) SSDs revisited: Part I—a framework for sample size guidance on species sensitivity distribution analysis. Environmental Toxicology and Chemistry 38:1514-1525

Chapman PF RM, Hart A, Roelofs W, Aldenberg T, Solomon K,, Tarazona J LM, Byrne P, Powley W, Green J, Ferson S, Galicia H. (2007) Methods of uncertainty analysis. In: A H (ed) EUFRAM Concerted Action to Develop a European Framework for Probabilistic Risk Assessment of the Environmental Impacts of Pesticides, Vol 2, Detailed Reports on Role, Methods, Reporting and Validation

Charles S, Veber P, Delignette-Muller ML (2018) MOSAIC: a web-interface for statistical analyses in ecotoxicology. Environmental Science and Pollution Research 25:11295-11302

Cullen AC, Frey HC (1999) Probabilistic Techniques in Exposure Assessment: A Handbook for Dealing with Variability and Uncertainty in Models and Inputs. Plenum Press, USA:181-241

Dalgarno S (2021) shinyssdtools: A web application for fitting Species Sensitivity Distributions (SSDs). Journal of Open Source Software 6:2848

Delignette-Muller ML, Dutang C (2015) fitdistrplus: An R package for fitting distributions. Journal of Statistical Software 64:1-34

ECETOC (2014) Estimating toxicity thresholds for aquatic ecological communities from sensitivity distributions 11-13 February 2014, Amsterdam, the Netherlands Workshop Report No. 28. European Centre for Ecotoxicology and Toxicology of Chemicals.

Fisher R, Thorley J (2021) ssddata: Species Sensitivity Distribution Data. R package version 1.0.0.

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 (2019) Key issues in the derivation of water quality guideline values: a workshop report. Australian Institute of Marine Science Report, Crawley, WA, Australia. 57 pp.

Forbes VE, Calow P (2002) Species sensitivity distributions revisited: a critical appraisal. Human and Ecological Risk Assessment 8:473-492

Fox D, van Dam R, Fisher R, Batley G, Tillmanns A, Thorley J, Schwarz C, Spry D, McTavish K (2021) Recent developments in species sensitivity distribution modeling. Environmental Toxicology and Chemistry 40:293-308

Fox DR (1999) A robust method for water quality monitoring and assessment. A discussion paper prepared for ANZECC review committee. CSIRO, Perth, Western Australia
Fox DR (2016) Chapter 2 - Contemporary Methods for Statistical Design and Analysis. In: Blasco J, Chapman PM, Campana O, Hampel M (eds) Marine Ecotoxicology. Academic Press
Hosking J (2006) On the characterization of distributions by their L-moments. Journal of Statistical Planning and Inference 136:193-198
Hosking JR (1990) L-moments: Analysis and estimation of distributions using linear combinations of order statistics. Journal of the Royal Statistical Society: Series B (Methodological) 52:105-124
Ismail NHB, Khalid ZBM EM algorithm in estimating the 2-and 3-parameter Burr Type III distributions. Proc Proceedings of the 21st National Symposium on Mathematical Sciences (SKSM21) AIP Conference Proceedings. American Institute of Physics
Johnson NL (1949) Systems of frequency curves generated by methods of translation. Biometrika 36:149-176
Kjeldsen TR, Smithers J, Schulze R (2002) Regional flood frequency analysis in the KwaZulu-Natal province, South Africa, using the index-flood method. Journal of Hydrology 255:194-211
Kleiber C, Kotz S (2003) Statistical size distributions in economics and actuarial sciences, Vol 470. John Wiley & Sons, New York
Kroll CN, Vogel RM (2002) Probability distribution of low streamflow series in the United States. Journal of Hydrologic Engineering 7:137-146
Lim YH, Lye LM (2003) Regional flood estimation for ungauged basins in Sarawak, Malaysia. Hydrological Sciences Journal 48:79-94
Newman MC, Ownby DR, Mezin LCA, Powell DC, Christensen TRL, Lerberg SB, Anderson BA (2000) Applying species-sensitivity distributions in ecological risk assessment: Assumptions of distribution type and sufficient numbers of species. Environmental Toxicology and Chemistry 19:508-515
Posthuma L, Suter II GW, Traas TP (2001) Species sensitivity distributions in ecotoxicology. 1st Edition. CRC Press, Boca Raton, FL, USA. 616 pp
Posthuma L, Traas TP, Suter GW (2002) General introduction to species sensitivity distributions. In: Posthuma L, Traas TP, Suter GW (eds) Species Sensitivity Distributions in Ecotoxicology
CRC Press, Boca Raton, FL, USA
Shao Q (2000) Estimation for hazardous concentrations based on NOEC toxicity data: an alternative approach. Environmetrics 11:583-595
Tadikamalla PR (1980) A look at the Burr and related distributions. International Statistical Review/Revue Internationale de Statistique:337-344
Thorley J, Schwarz C (2018) ssdtools: Species Sensitivity Distributions. https://CRAN.R-project.org/package=ssdtools.
van Dam JW, Trenfield MA, Streten C, Harford AJ, Parry D, van Dam RA (2018) Water quality guideline values for aluminium, gallium and molybdenum in marine environments. Environmental Science and Pollution Research 25:26592-26602
van den Brink PJ, Brock TC, Posthuma L (2001) The value of the species sensitivity distribution concept for predicting field effects:(Non-) confirmation of the concept using semifield experiments. In: Traas TP, Suter GW (eds) Species sensitivity distributions in ecotoxicology. CRC Press, Boca Raton, FL, USA
Zajdlik B. 2005. Statistical analysis of the SSD approach for development of Canadian water quality guidelines. Report for CCME Project Number 354-2005. Zajdlik and Associates, Rookwood, ON, Canada.

6 ACKNOWLEDGEMENTS

The following individuals and organisations have been closely involved and instrumental in the preparatory technical work leading up to this Australian-Canadian collaboration:

Dr. Rick van Dam (*WQadvice*); Dr. Graeme Batley (*CSIRO Land and Water*); Dr. Angeline Tillmanns (*British Columbia Ministry of Environment and Climate Change Strategy*); and Dr Doug Spry and Kathleen McTavish (*Environment and Climate Change Canada, Gatineau, Quebec, Canada*).

Mr. Mathew Bow, Ms Amy Lea, and Mr. Alex Baker of the Department of Agriculture, Water and the Environment have, and continue to provide administrative and contractual support.

Dr Simon Barry of CSIRO provided valuable assistance with replicating the `BurrliOz` software in R for use in the simulation studies.

Dr Rick van Dam, Dr Graeme Batley, Dr Joost van Dam and Dr Jenny Stauber provided valuable advice and review of the example datasets collated through the `ssddata` R package.

7 APPENDICES

Appendix A DETAILED TASK LIST

Appendix B *INVERSE PARETO* DISTRIBUTION

Appendix C UNBIASED ESTIMATION FOR THE *INVERSE PARETO* DISTRIBUTION

Appendix D R-CODE FOR FITTING MIXTURES

Appendix E *L*-MOMENT ESTIMATORS FOR THE BURR III DISTRIBUTION

Appendix F ESTIMATION AND INFERENCE FOR THE BURR III DISTRIBUTION

Appendix G A NOTE ON RE-SCALING SSDs

Appendix H RECONCILIATION OF HC ESTIMATES FOR THE *INVERSE PARETO* DISTRIBUTION

Appendix I: ADDITIONAL ANALYSES ASSESSING THE RECOMMENDED DISTRIBUTION SET

Appendix A: Detailed task list

Category	Task	Software	Ranked Priority (CAN)	Priority(Aus /Can)	More information
Statistical Investigations					
Distribution-fitting	1a Assess numerical instability issues with Burr III and decide whether to include Burr III	ssdtools	1	Aus	Numerical stability issues: All of the candidate distributions in ssd tools are 'well-behaved' with respect to parameter estimation with the exception of the Burr III distribution which often fails to converge to a unique solution. This is thought to be primarily a function of the very small sample sizes used in ecotoxicology, although further work is needed to fully understand issues of multicollinearity among parameter estimates and convergence issues. The outcomes will inform subsequent considerations as to whether the Burr III distribution is a useful inclusion in the suite of candidate SSDs. While the Burr family has proved to be very useful in the ecotox context over the last 20+ years and is the cornerstone of the Australian methodology, the justification for its continued use may be somewhat diminished by virtue of (i) on-going numerical issues; and (ii) the move to model-averaging using a selection of better 'behaved' distributions.
Distribution-fitting	1b (merged with 1a)	ssdtools			
Distribution-fitting	1c Determine convergence criteria	ssdtools	1	Aus + Can	Determine the criteria by which convergence will be tested: method used by fitdistplus or other.
Distribution-fitting	1d Identify benchmark datasets		1	Aus + Can	Identify a suite of benchmark datasets that will serve as a reference standard for all SSD-fitting and evaluation. This is in keeping with the recommendation in Fox et al. (*in prep.*)
Distribution-fitting	1e Evaluate SSD methodologies based on datasets		1	Aus + Can	Fully explore, evaluate and document SSD methodologies (including estimation strategies, numerical stability issues, HCx estimation, confidence interval determination, identification of initial values for iterative parameter estimation strategies (e.g. maximum likelihood) and computational efficiency using the datasets in (d) and other simulated datasets.
Distribution-fitting	1f Refine mixture modelling with view to incorporating	ssdtools	2	Aus + Can	Continue development and refinement of statistical mixture modelling (SMM) methodologies with a view to incorporating this capability as an option within the ssd tools shiny app. The statistical ramifications and requirements in terms of issues previously identified above will also be explored in the context of SMM.
Distribution-fitting	1g Investigations into Burr alternatives. E.g. ggLogis + other	ssdtools		Aus	
Sub-total					
HCx and CI estimation	2a CI for HCx methods - alternatives to bootstrapping		2	Aus + Can (lower priority)	Investigate and document outcomes and recommendations associated with the 'best' methods of estimating an HCx (and FA) post-distribution fitting. We would expect to address issues of invertability of the fitted cdf versus bootstrapping. Importantly, further research is required to undertake a comprehensive evaluation of methods of determining confidence intervals around the estimated HCx. Current strategies are based on either bootstrapping or the 'delta' method. Fox, Thorley and Etterson (USEPA) are currently exploring the potential of profile-likelihood based confidence intervals as a more robust and defensible strategy.
inalization of default distributions	3a Finalize default distributions		1	Aus + Can	Further consideration and justification of the default set of distributions (given this is the core feature of model averaging, and there have been some changes to the default set since the Nov workshop in BC). These considerations will be substantially informed by the outputs and outcomes of 1 and 2 above.
Statistical Sub-total					
Software development					
Functionality	4 Add individual model outputs to shiny apps	shinyssdtools	done?	Can	Add output of individual model weight, HC_x estimates, and CIs to the Shiny App. ECCC and Ontario use tables (similar to the below) in guideline documents but need to use R in order to populate.
Functionality	5 Add functionality for french plots	shinyssdtools	1	Can	Format the X axis so that the thousands separator is a space and not a decimal ex. 20 000 instead of 20,000. Make this option available in the Shiny App.
Functionality	6 Add geom for censored data	ssdtools	3	Can	Allow species names and symbols to be added so the figure looks the same as without censored data.
Tests	7a Tests	ssdtools	3	Can	Develop a minimal set of structured tests that will be run daily to ensure that package is functioning properly and to alert the package maintainer of changes in dependent packages
Tests	7b Test all OS	ssdtools	3	Can	Develop tests to ensure package gives the same results across all operating systems
Extras	8 Add example code to analyse multiple datasets	ssdtools	3	Can	Add code to run multiple sets of data at the same time (in R package- not Shiny)
Extras	9 Add ability to choose colour and symbols in shinyssdtools	shinyssdtools	3	Can	Ability to choose colour and symbols in Shiny App instead of done alphabetically and with colour sets. Historically for ECCC, "amphibian" is a yellow triangle, "invertebrate" is a red circle, "fish" a blue square, and "algae/plant" a green triangle. Some people would like to maintain these shapes and colours- however with a proper legend this shouldn't be a huge issue. French and English versions of the SSD usually have different symbols using ssdtools due to translations and therefore are not exactly the same.

Appendix B: *Inverse Pareto* distribution

Carl James Schwarz

2021-05-07

- 1 Introduction
  - 1.1 PDF and CDF
  - 1.2 MLE for the Pareto Distribution
  - 1.3 MLE for the Inverse-Pareto distribution
  - 1.4 Numerical estimation
  - 1.5 Consequences for fitting in *ssdtools*
- 2 Implementing the inverse-Pareto distribution using TMB
  - 2.1 Testing the distribution function (log-likelihood with a single observation)
    - 2.1.1 Values of y between 0 and scale parameter
    - 2.1.2 Values of y above the scale parameter
    - 2.1.3 Censored values (0 to y)
    - 2.1.4 Censored values (0 to y) where y is above scale
    - 2.1.5 Censored values y to Inf
    - 2.1.6 Censored values between y and 2y
  - 2.2 MLE using simulated data
    - 2.2.1 Uncensored data
    - 2.2.2 Censored data I
    - 2.2.3 Censored data II
- 3 Real data example
  - 3.1 Summary
- 4 Recommendations
- 5 References

1 Introduction

The inverse-Pareto distribution is one of the limiting forms of the Burr III distribution and so may be included in future version of *ssdtools*.

1.1 PDF and CDF

We are using the b Americanb parameterization of the Pareto distribution (see https://www.casact.org/sites/default/files/database/astin_vol20no2_201.pdf (https://www.casact.org/sites/default/files/database/astin_vol20no2_201.pdf)) which has the pdf and cdf of

$$f_P(y|shape, scale) = \frac{shape \times scale^{shape}}{y^{(shape+1)}}$$

$$F_P(y|shape, scale) = 1 - \left(\frac{scale}{y}\right)^{shape}$$

for $scale < y$, $scale > 0$, and $shape > 0$. Notice that Y is positive and larger than the scale parameter.

The *VGAM* and *ssdtools* packages implement the b Americanb Pareto distribution, but the *actuar* package implements the b Europeanb Pareto distribution.

Using the standard transformation theory, the inverse-Pareto (IP) distribution then has

$$f_{IP|shape,scale} = \frac{shape * scale^{shape}}{(1/y)^{(shape+1)}} \frac{1}{y^2}$$

$$F_{IP}(y|shape, scale) = \left(\frac{scale}{1/y}\right)^{shape}$$

for $y < 1/scale$, $scale > 0$ and $shape > 0$. Notice that Y has restricted range from 0 to $1/scale$.

For convenience, we often invert the scale parameter in the inverse-Pareto distribution, i.e., $scale* = 1/scale$. The pdf and cdf are

$$f_{IP|shape,scale*} = \frac{shape \times scale*^{-shape}}{(1/y)^{(shape+1)}} \frac{1}{y^2}$$

$$F_{IP}(y|shape, scale*) = \left(\frac{y}{scale*}\right)^{shape}$$

for $y < scale*$, $scale > 0$ and $shape > 0$. Notice that Y has restricted range from 0 to $scale*$.

So now, if $Y \sim InversePareto(shape, scale*)$ then $1/Y \sim Pareto(shape, scale = 1/scale*)$.

1.2 MLE for the Pareto Distribution

The MLEs for the Pareto distribution are given by Malik (1970).

$$\widehat{scale}_P = min(Y_i)$$

$$\widehat{shape}_P = \frac{n}{\sum\left(log\frac{y_i}{\widehat{scale}}\right)}$$

because $log(Y_i/min(Y))$ has an inverse exponential distribution.

Because the MLE for the scale parameter is the first order statistic of Y, the usual large sample theory on the limiting distribution of these estimators is not applicable and the standard error cannot be found in the usual ways.

Malik (1970) also shows that the distribution of the estimated scale and estimated shape parameter are independent and gives the exact joint distribution. From this he derived that

$$\widehat{se}(\widehat{shape}_P) = \frac{\widehat{shape}}{\sqrt{n}}$$

(simple proof since the estimate of the scale parameter is the min(Y)) with an estimated standard error of

$$\widehat{se}(\widehat{scale}_P) = \sqrt{\frac{n \times \widehat{shape} \times \widehat{scale}^2}{(n \times \widehat{shape} - 1)^2 (n \times \widehat{shape} - 2)}}$$

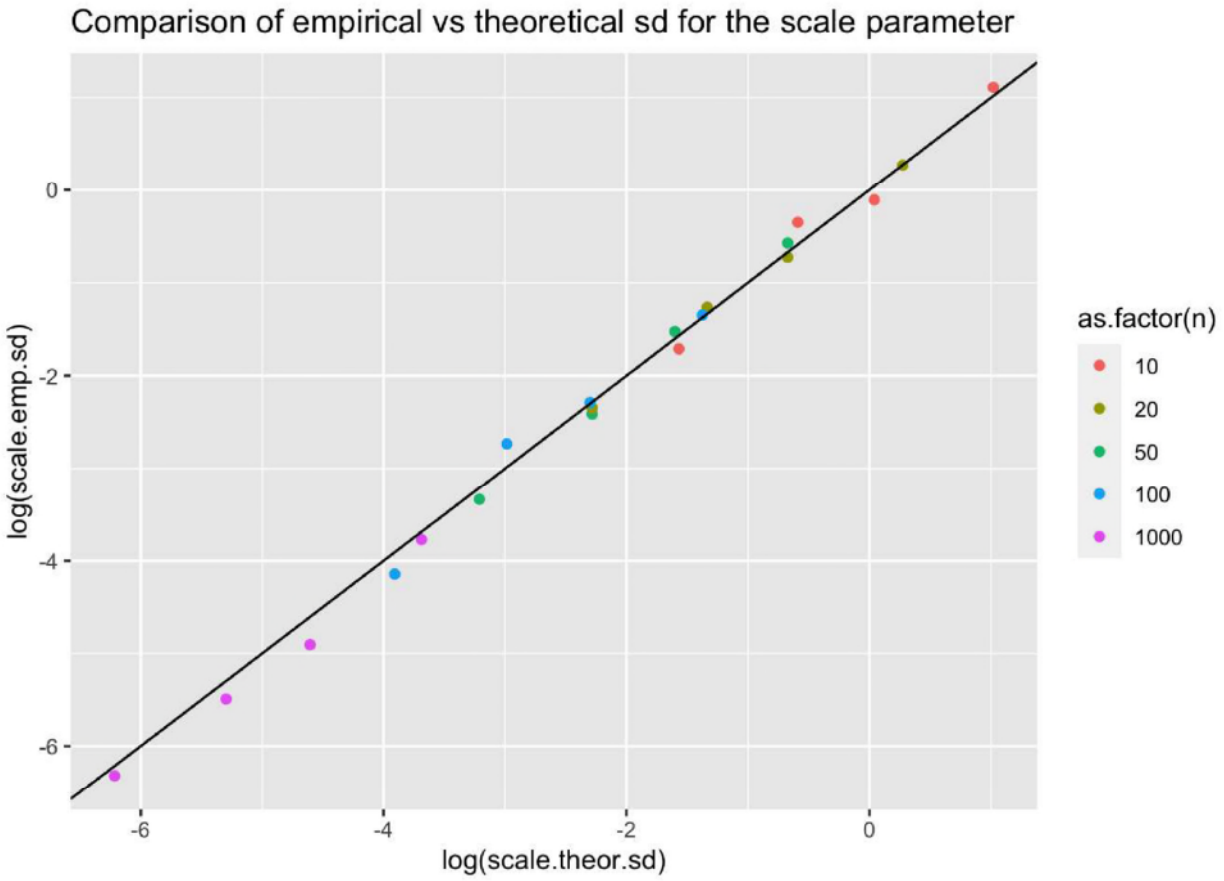
if $n \times shape > 2$; otherwise the standard error infinite for small shape values.

Here is a short simulation study verifying the above results

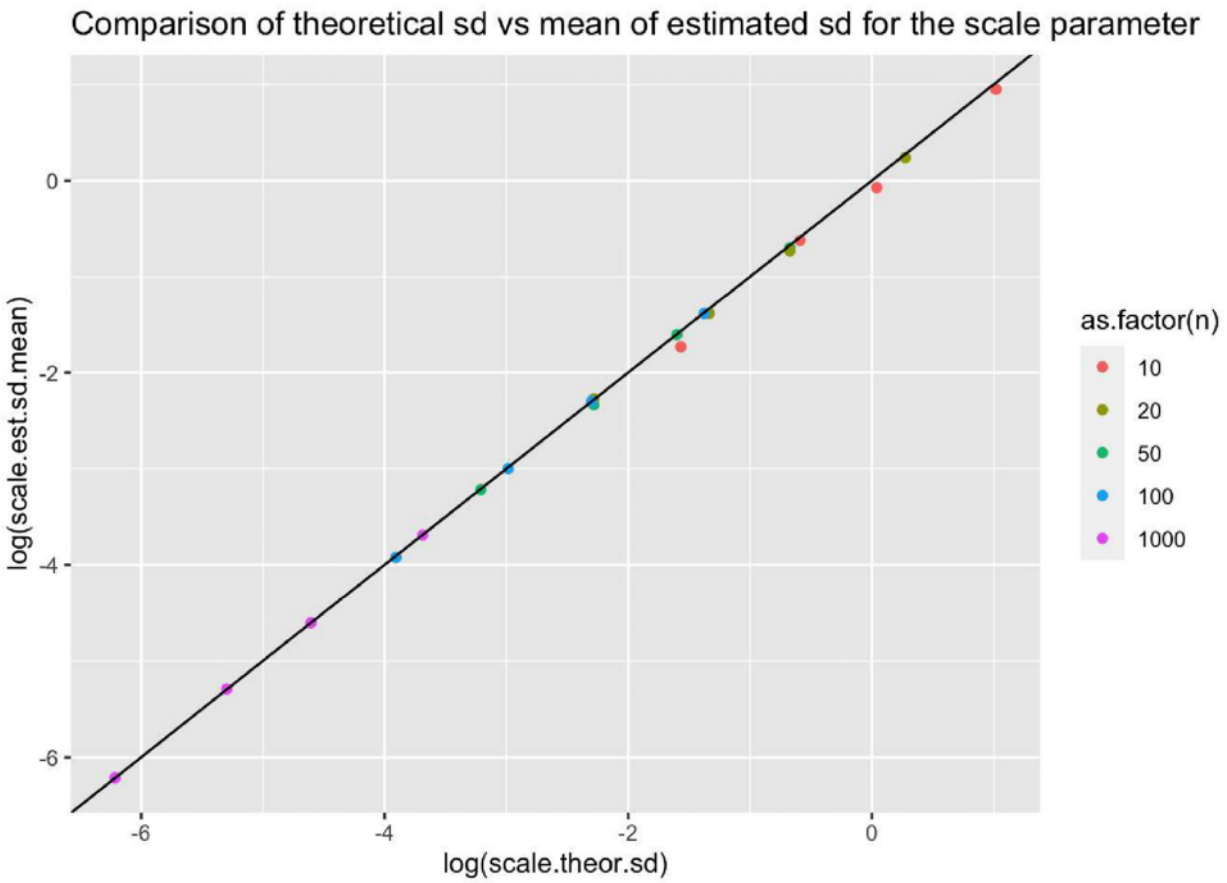
```
# Confirm that se of minimum value from pareto has the proper sd

library(ggplot2)
library(plyr)
library(VGAM)

scenarios <- expand.grid(n=c(10,20,50,100,1000), shape=c(2,5), scale=c(10,50), nrep=100)

res <- plyr::adply(scenarios, 1, function(scenario){
    samples <- plyr::ldply(1:scenario$nrep, function(sim, scenario){
       sample <- VGAM::rpareto(n=scenario$n, scale=scenario$scale, shape=scenario$shape)
       scale.hat <- min(sample)
       shape.hat <- 1/mean(log(sample/scale.hat))
       scale.est.sd <- sqrt(scale.hat^2 * shape.hat * scenario$n /
                            ( (scenario$n*shape.hat-1)^2 * (scenario$n*shape.hat-2)))
       shape.est.sd <- shape.hat/sqrt(scenario$n)

       data.frame(scale.hat=scale.hat, shape.hat=shape.hat, scale.est.sd=scale.est.sd, shape.es
t.sd)
    },scenario)
    #browser()
    # find the se of the scale parameters
    scenario$scale.emp.sd   <- sd(samples$scale.hat)
    scenario$scale.theor.sd <- sqrt(scenario$scale^2 * scenario$shape * scenario$n /
                            ( (scenario$n*scenario$shape-1)^2 * (scenario$n*scenario$shape-2
)))
    scenario$scale.est.sd.mean <- mean(samples$scale.est.sd)

    # find the se of the shape parameters
    scenario$shape.emp.sd   <- sd(samples$shape.hat)
    scenario$shape.theor.sd <- scenario$shape / sqrt(scenario$n)
    scenario$shape.est.sd.mean <- mean(samples$shape.est.sd)

    scenario
})
head(res)
```

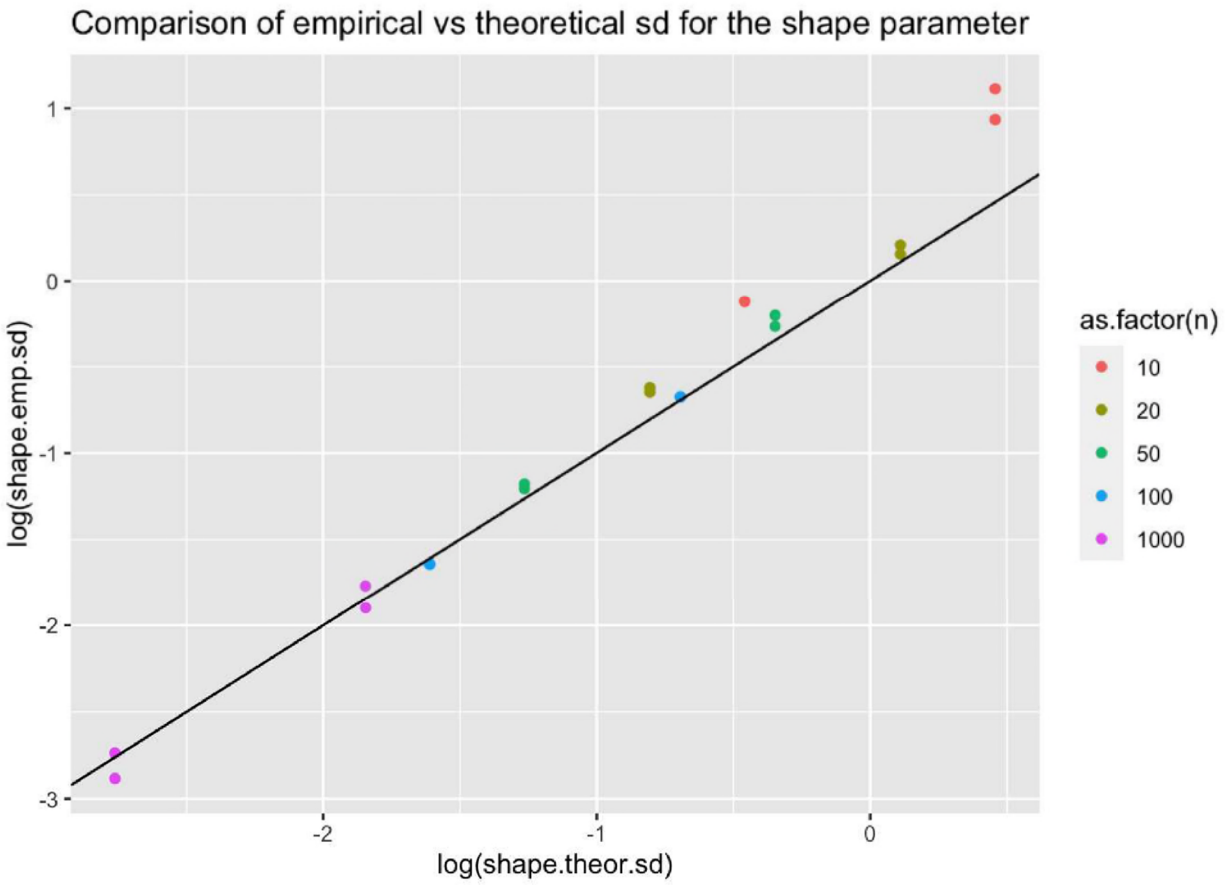
```
##      n shape scale nrep scale.emp.sd scale.theor.sd scale.est.sd.mean
## 1   10     2    10  100  0.705132524    0.554785554        0.53615885
## 2   20     2    10  100  0.282479428    0.263071372        0.25073444
## 3   50     2    10  100  0.089826273    0.102035611        0.09683629
## 4  100     2    10  100  0.065038336    0.050504413        0.04987369
## 5 1000     2    10  100  0.004124351    0.005005004        0.00503092
## 6   10     5    10  100  0.180795463    0.208289944        0.17671406
##   shape.emp.sd shape.theor.sd shape.est.sd.mean
## 1   0.88546043     0.63245553         0.7718429
## 2   0.52412167     0.44721360         0.5037054
## 3   0.29970996     0.28284271         0.3063220
## 4   0.19316857     0.20000000         0.2053208
## 5   0.05596685     0.06324555         0.0630005
## 6   3.04392432     1.58113883         2.1813952
```

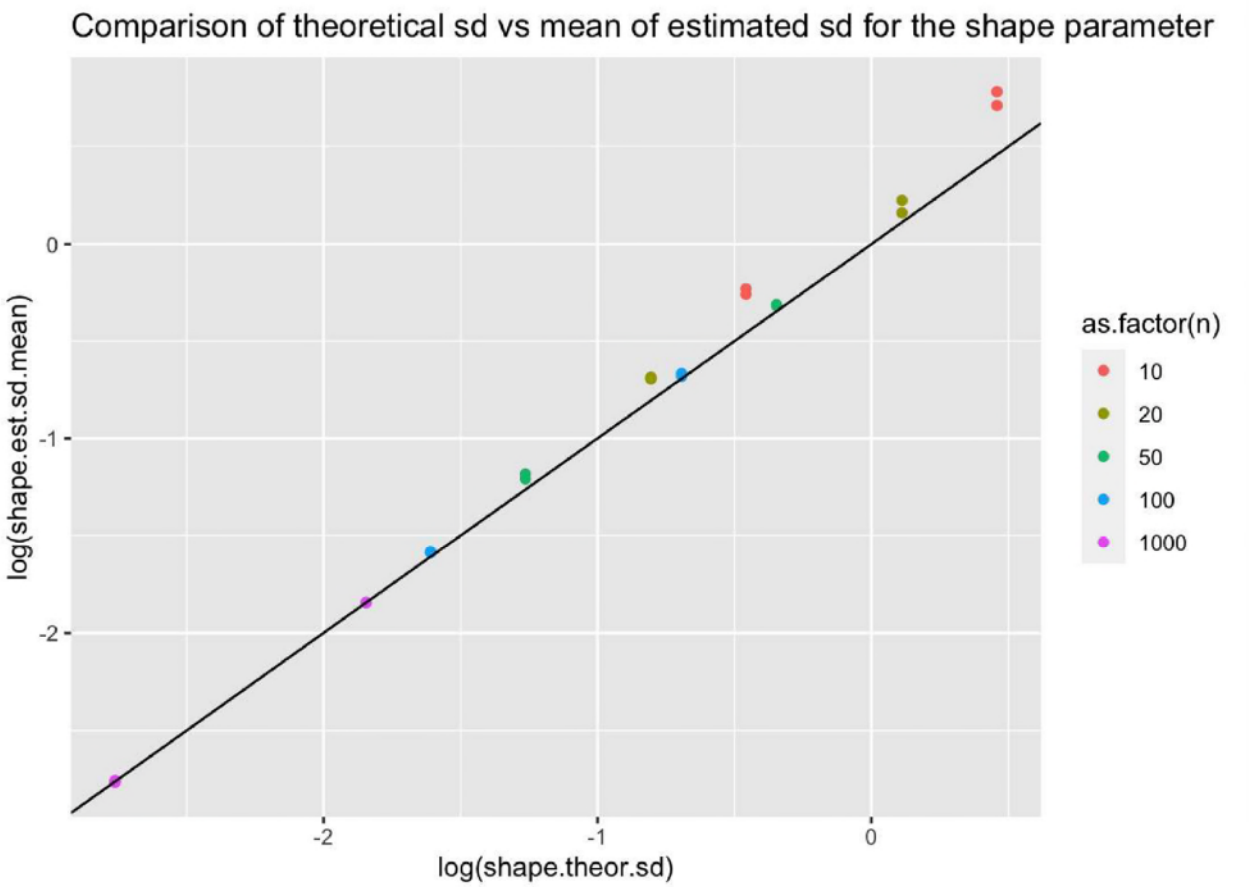
```
ggplot(data=res, aes(x=log(scale.theor.sd), y=log(scale.emp.sd), color=as.factor(n)))+
  ggtitle("Comparison of empirical vs theoretical sd for the scale parameter")+
  geom_point()+
  geom_abline(intercept=0, slope=1)
```




```
ggplot(data=res, aes(x=log(scale.theor.sd), y=log(scale.est.sd.mean), color=as.factor(n)))+
  ggtitle("Comparison of theoretical sd vs mean of estimated sd for the scale parameter")+
  geom_point()+
  geom_abline(intercept=0, slope=1)
```


Comparison of theoretical sd vs mean of estimated sd for the scale parameter
log(scale.est.sd.mean)
log(scale.theor.sd)
0
-2
-4
-6
-6
-4
-2
0
as.factor(n)
10
20
50
100
1000


Comparison of empirical vs theoretical sd for the shape parameter
log(shape.emp.sd)
log(shape.theor.sd)
1
0
-1
-2
-3
-2
-1
0
as.factor(n)
10
20
50
100
1000



1.3 MLE for the Inverse-Pareto distribution

Consequently, the MLEs for the Inverse-Pareto distribution (after the redefinition of the scale) are

$$\widehat{scale*}_{IP} = max(Y)$$

$$\widehat{shape}_{IP} = \frac{n}{\sum\left(log\frac{\widehat{scale*}}{y_i}\right)}$$

Notice that the estimator of the scale for the Pareto distribution uses the first order statistic (the minimum data value), while the estimator for the scale* of the InversePareto distribution uses the last order statistic (the maximum data value).

1.4 Numerical estimation

The key problem with numerical estimation for the Pareto and Inverse-Pareto distribution is the non-differential log-likelihood at the MLE for the scale parameter because of the discontinuity to the left and right of the estimator.

For example consider the log-likelihood function for some randomly generated data from the Pareto distribution

```
library(VGAM)
library(TMB)
library(ggplot2)

set.seed(234324)

n = 10
scale = 10
shape =  2

Y <- VGAM::rpareto(n, scale=scale, shape=shape)
Y
```

```
##  [1] 18.28961 10.42548 14.92527 50.35829 13.60922 10.30869 14.66036 10.90508
##  [9] 11.99338 10.94461
```

```
# MLE are
scalehat <- min(Y)
shapehat <- 1/mean(log(Y/scalehat))
cat("Explicit MLE for data ", shapehat, scalehat, "\n")
```

```
## Explicit MLE for data  2.90846 10.30869
```

Notice that all values of Y are larger than the scale value. The log-likelihood function is easily constructed and evaluated at the computed MLEs

```
logL_P <- function(shapescale, Y, return.nll=FALSE){
   shape <- shapescale[1]
   scale <- shapescale[2]
   ll <- sum(VGAM::dpareto(Y, scale=scale, shape=shape, log=TRUE))
   if(return.nll) ll <- - ll  # do you want the negative of the logL
   ll
}

ll.MLE <- logL_P(c(shapehat, scalehat), Y)
ll.MLE
```

```
## [1] -26.09188
```

Now examine the performance of the log-likelihood function as the scale parameter approaches, reaches, and exceeds the minimum value of Y

```
shapescale <- c(1, min(Y)-.001);
cat(shapescale, logL_P(shapescale, Y), "\n")
```

```
## 1 10.30769 -30.20733
```

```
shapescale <- c(1, min(Y)-.000);
cat(shapescale, logL_P(shapescale, Y), "\n")
```

```
## 1 10.30869 -30.20636
```

```
shapescale <- c(1, min(Y)+.001);
cat(shapescale, logL_P(shapescale, Y), "\n")
```

```
## 1 10.30969 -Inf
```

This cliff makes it difficult to numerically compute the MLE as shown below.

For example, the default use of the *optim()* function leads to:

```
res.optim <- optim( c(1,10), logL_P,  Y=Y, return.nll=TRUE)
res.optim
```

```
## $par
## [1]  2.909235 10.308689
##
## $value
## [1] 26.09188
##
## $counts
## function gradient
##      111       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

Notice that *optim()* obtained the correct scale estimate, but failed to return the correct MLE for the shape parameter. If we try and compute a hessian to find the standard errors, we obtain an error because of the non-differential log-likelihood at the scale parameter.

```
res.optim <- optim( c(1,10), logL_P,  Y=Y, return.nll=TRUE, hessian=TRUE)
```

```
## Error in optim(c(1, 10), logL_P, Y = Y, return.nll = TRUE, hessian = TRUE): non-finite finite
-difference value [2]
```

If you are willing to b special caseb the Pareto (and Inverse-Pareto) distribution, you can set the scale parameter to the MLE and keep it fixed with the *lower* and *upper* arguments:

```
res.optim.bounded <- optim( c(1,min(Y)), logL_P,  Y=Y, return.nll=TRUE,
                    upper=c(Inf,min(Y)))
res.optim.bounded
```

```
## $par
## [1]  2.90846 10.30869
##
## $value
## [1] 26.09188
##
## $counts
## function gradient
##        8        8
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

Now *optim()* has converged to the true MLE. But we still cannot get the hessian to estimate the standard errors.

```
res.optim.bounded <- optim( c(1,min(Y)), logL_P,  Y=Y, return.nll=TRUE, hessian=TRUE,
                    upper=c(Inf,min(Y)))
```

```
## Error in optim(c(1, min(Y)), logL_P, Y = Y, return.nll = TRUE, hessian = TRUE, : non-finite f
inite-difference value [2]
```

Because of the joint independence of the estimators for the scale and shape parameter, you can convert the problem into a one-dimensional problem by b fixingb the scale parameter. The *optim()* function still stops a bit early, but close enough. YMMV.

```
logL_P.rev <- function(shape,  scale, Y, return.nll=FALSE){
   ll <- sum(VGAM::dpareto(Y, scale=scale, shape=shape, log=TRUE))
   if(return.nll) ll <- - ll  # do you want the negative of the logL
   ll
}
res.optim.rev <- optim( c(1), logL_P.rev, scale=min(Y), Y=Y, return.nll=TRUE, hessian=TRUE)
res.optim.rev
```

```
## $par
## [1] 2.907813
##
## $value
## [1] 26.09188
##
## $counts
## function gradient
##       28       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##         [,1]
## [1,] 1.18268
```

```
cat("Estiamted se of shape ", sqrt(1/res.optim.rev$hessian), "\n")
```

```
## Estiamted se of shape  0.9195309
```

```
cat("Explicit  se of shape ", res.optim.rev$par/sqrt(n), "\n")
```

```
## Explicit  se of shape  0.9195311
```

1.5 Consequences for fitting in *ssdtools*

So.. when fitting the Pareto or Inverse-Pareto distribution, I think we will have to b specialb case the fitting by fixing the scale at the minimum (for the Pareto fit) or maximum (for the Inverse-Pareto) distribution. This converts the problem to a one-dimensional minimization function.

This will give us a numerical estimate of the standard error for the shape parameter, and the standard error for the scale parameter will need to be estimated outside of the optimization procedure using the formula in an earlier section. Profile likelihood may work? **Need to do a literature search to see if this has been done**

Similarly, we can construct a hessian ourselves because the two estiamtors are independent but cannot get the numerical optimization procedure to do this automatically.

We will only need the standard error (and covariance) terms if we want to estimate the uncertainty of the HCx estimates using the delta method (see the other document on estimating the uncertainty of HCx). After we construct the hessian ourselves, the multivariate methods for finding the uncertainty of HCx can also be applied. The likelihood profile and (parametric) bootstrap methods should work.

2 Implementing the inverse-Pareto distribution using TMB

The C++ template for the inverse-Pareto distribution was implemented in TMB in the usual way.

2.1 Testing the distribution function (log-likelihood with a single observation)

We define our density, CDF, quantile and random number generator in *R* to check our results and for inclusion into *ssdtools*.

```
# define the density function for my inverse pareto
dmyinvpareto <- function(x, shape, scale, log=FALSE){
  if(x==0)return(ifelse(log,log(0),0))
  if(x>scale)return(ifelse(log,log(0),0))
  logpdf <- VGAM::dpareto(1/x, shape=shape, scale=1/scale, log=TRUE)- 2*log(x)
  if(log)return(logpdf)
  pdf <- exp(logpdf)
  pdf
}

# define the cdf function
pmyinvpareto <- function(x, shape, scale, lower.tail=TRUE, log.p=FALSE){
  #browser()
  if(x<=0)    cdf = 0
  if(x >scale)cdf = 1
  if(x<=scale)cdf <- VGAM::ppareto(1/x, shape=shape, scale=1/scale, lower.tail=!lower.tail, log.
p=FALSE)
  if(log.p)cdf=log(cdf)
  cdf
}

# define the quant function
qmyinvpareto <- function(p, shape, scale){
  q <- VGAM::qpareto(1-p, shape=shape, scale=1/scale)
  #browser()
  1/q
}


rmyinvpareto <- function(n, shape, scale){
   x <- VGAM::rpareto(n, shape=shape, scale=1/scale)
   return(1/x)
}
```

Here we test the TMB code. Donb t forget that the TMB model returns the negative of the log-likelihood.

2.1.1 Values of y between 0 and scale parameter

```
# check the function by comparing to invpareto() from R
dyn.load(dynlib("ll_invpareto"))
shape=2
scale=10
y=10

# uncensored values
dmyinvpareto(y, shape=shape, scale=scale, log=TRUE)
```

```
## [1] -1.609438
```

```
VGAM::  dpareto(1/y, shape=shape, scale=1/scale, log=TRUE)- 2*log(y)
```

```
## [1] -1.609438
```

```
invpareto.model <- MakeADFun(
    data= list(left=y, right=y, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

invpareto.model$fn()
```

```
## [1] 1.609438
```

2.1.2 Values of y above the scale parameter

Values above the scale should be b invalidb and return a pdf of 0 (or a log(pdf) of -Inf)

```
shape=2
scale=10
y=10.1

# uncensored values
dmyinvpareto(y, shape=shape, scale=scale, log=TRUE)
```

```
## [1] -Inf
```

```
VGAM::dpareto(1/y, shape=shape, scale=1/scale, log=TRUE)- 2*log(y)
```

```
## [1] -Inf
```

```
invpareto.model <- MakeADFun(
    data= list(left=y, right=y, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

invpareto.model$fn()  # results differ because of penalty function
```

```
## [1] Inf
```

2.1.3 Censored values (0 to y)

```
# check the function by comparing to invpareto() from R
shape=2
scale=10
y=8

pmyinvpareto(y, shape=shape, scale=scale, log.p=TRUE)
```

```
## [1] -0.4462871
```

```
invpareto.model <- MakeADFun(
    data= list(left=0, right=y, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

invpareto.model$fn()
```

```
## [1] 0.4462871
```

2.1.4 Censored values (0 to y) where y is above scale

```
# check the function by comparing to invpareto() from R
shape=2
scale=10
y=12

pmyinvpareto(y, shape=shape, scale=scale, log.p=TRUE)
```

```
## [1] 0
```

```
invpareto.model <- MakeADFun(
    data= list(left=0, right=y, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

invpareto.model$fn()
```

```
## [1] 0
```

2.1.5 Censored values y to Inf

```
# censored values (y to Inf)
shape=2
scale=10
y=8

pmyinvpareto(y, shape=shape, scale=scale, log.p=TRUE, lower.tail=FALSE)
```

```
## [1] -1.021651
```

```
invpareto.model <- MakeADFun(
    data= list(left=y, right=Inf, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

-invpareto.model$fn()
```

```
## [1] -1.021651
```

2.1.6 Censored values between y and 2y

```
# between y and 2y
log(pmyinvpareto(2*y, shape=shape, scale=scale) -
    pmyinvpareto(y, shape=shape, scale=scale))
```

```
## [1] -1.021651
```

```
invpareto.model <- MakeADFun(
    data= list(left=0, right=y, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

invpareto.model$fn()
```

```
## [1] 0
```

2.1.5 Censored values y to Inf

```
# censored values (y to Inf)
shape=2
scale=10
y=8

pmyinvpareto(y, shape=shape, scale=scale, log.p=TRUE, lower.tail=FALSE)
```

```
## [1] -1.021651
```

```
invpareto.model <- MakeADFun(
    data= list(left=y, right=Inf, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

-invpareto.model$fn()
```

```
## [1] -1.021651
```

2.1.6 Censored values between y and 2y

```
# between y and 2y
log(pmyinvpareto(2*y, shape=shape, scale=scale) -
    pmyinvpareto(y, shape=shape, scale=scale))
```

```
## [1] -1.021651
```

```
invpareto.model <- MakeADFun(
    data= list(left=y, right=2*y, weight=1),
    parameters=list(
        log_shape = log(shape),
        log_scale = log(scale)  ),
    DLL="ll_invpareto"
)

invpareto.model$fn()
```

```
## [1] 1.021651
```

2.2 MLE using simulated data

We will generate a (large) sample of data from the inverse-pareto distribution and find the MLE to check that the optimization is working properly.

Notice that we will use the *map* argument to fix the scale parameter during the optimization process.

2.2.1 Uncensored data

```
# Check with simulated data
fit <- NULL

# Uncensored case

shape=3
scale=40

Conc <- rmyinvpareto(10000, shape=shape, scale=scale)
Conc[1:100]
```

```
##   [1] 21.518620 34.797461 37.974426 35.096392 34.718983 22.209360 39.872179
##   [8] 10.419708 24.632182 19.894477 17.581421 35.936850 13.672447 38.603543
##  [15] 28.556547 36.800381 32.370035 34.553205 34.135813 31.259229 36.716228
##  [22] 33.133452 36.968254 24.944995 36.592789  5.537077 22.516535 25.864365
##  [29] 29.507351 30.439227 35.594707 37.942472 34.466118 39.371823 27.229546
##  [36] 25.871472 28.767652 38.846856 36.854223 24.665558 33.393586 29.399337
##  [43] 35.099082 33.494870 21.657060 37.815466 17.201552 21.091997 34.643006
##  [50] 39.697364 39.364193 19.952866 26.712106 17.658459 34.217662 20.920701
##  [57] 31.861488 24.263930 31.364350 21.866944 33.286954 25.117173 25.943420
##  [64] 36.947543 38.781032 38.412255 26.937160 36.501412 22.884104 20.814260
##  [71] 32.649883 21.249206 33.062418 37.430812 28.942029 37.573077 38.344844
##  [78] 39.028018 29.030255 37.580565 26.118043 20.202507 37.639102 26.395162
##  [85] 24.442329  9.539289 12.942417 38.015939 31.036937 32.062752 28.770925
##  [92] 37.277765 13.816573 20.116680 35.446858 34.315986 25.940687 35.928731
##  [99] 39.920894 33.599770
```

```
mean(Conc)
```

```
## [1] 30.03497
```

```
sd(Conc)
```

```
## [1] 7.721216
```

```
max(Conc)
```

```
## [1] 39.99585
```

```
p = seq(.1,.9,.1)
qmyinvpareto(p, shape=shape, scale=scale)
```

```
## [1] 18.56636 23.39214 26.77732 29.47225 31.74802 33.73731 35.51616 37.13271
## [9] 38.61958
```

```
quantile(Conc, prob=p)
```

```
##      10%      20%      30%      40%      50%      60%      70%      80%
## 18.64574 23.51125 26.78218 29.54028 31.80458 33.74866 35.55691 37.10953
##      90%
## 38.58210
```

```
q= p^(1/shape)* scale   # from the definition in the spreadsheet
q
```

```
## [1] 18.56636 23.39214 26.77732 29.47225 31.74802 33.73731 35.51616 37.13271
## [9] 38.61958
```

```
##############################
# Fit the inverse pareto using
# get initial values
# see https://stats.stackexchange.com/questions/27426/how-do-i-fit-a-set-of-data-to-a-pareto-dis
tribution-in-r
# For a pareto distribution, the scale=min(Y) and shape= 1/ mean(log(Y/scale))
invpareto.init.scale <- max(Conc)  # Y values must be < scale value in invpareto distribution
invpareto.init.shape <- 1/mean(log(invpareto.init.scale/Conc))
c(invpareto.init.scale, invpareto.init.shape )
```

```
## [1] 39.995849  3.014136
```

```
# note that for the invpareto, the MLE for the scale is the maximum value and so we fix it to th
e maximum value
# using the map argument
invpareto.model <- MakeADFun(
    data= list(left=Conc,
               right=Conc,
               weight =rep(1, length(Conc))),
     parameters=list(
        log_shape = log(invpareto.init.shape),
        log_scale = log(invpareto.init.scale)),
    map = list(log_scale = factor(NA)),
    DLL="ll_invpareto"
)


fit$myinvpareto <- nlminb(invpareto.model$par, invpareto.model$fn, invpareto.model$gr, invparet
o.model$he,
                          control=list(eval.max=10000, iter.max=1000))
```

```
## outer mgc:  9.320558e-11
```

```
cat("Convergence codes from nlminb ", fit$myinvpareto$convergence, fit$myinvpareto$message, "\n"
)
```

```
## Convergence codes from nlminb  0 both X-convergence and relative convergence (5)
```

```
invpareto.model$he(invpareto.model$par)
```

```
##       [,1]
## [1,] 10000
```

```
sd<-sdreport( invpareto.model , getReportCovariance=TRUE)
```

```
## outer mgc:  9.320558e-11
## outer mgc:  10.005
## outer mgc:  9.995002
## outer mgc:  3.014136
```

```
sd
```

```
## sdreport(.) result
##           Estimate  Std. Error
## log_shape 1.103313 0.009999999
## Maximum gradient component: 9.320558e-11
```

```
summary(sd)
```

```
##            Estimate  Std. Error
## log_shape  1.103313 0.009999999
## shape      3.014136 0.030141356
## scale     39.995849 0.000000000
```

Notice that the reported standard error for the scale parameter is 0. The scale parameter is close to that used in the generation of the data, but Rytgaard (1990) shows the MLEs are biased.

Look what happens if we try and do an ordinary maximization without fixing the scale parameter:

```
# what happens if we don't fix the mle for log_scale using the map parameter
invpareto.model <- MakeADFun(
    data= list(left = Conc,
               right= Conc,
               weight =rep(1, length(Conc))),
     parameters=list(
        log_shape = log(invpareto.init.shape),
        log_scale = log(invpareto.init.scale+1)+1),
    DLL="ll_invpareto"
)

#invpareto.model$env$tracepar <- TRUE

fit$myinvpareto <- nlminb(invpareto.model$par, invpareto.model$fn, invpareto.model$gr, invparet
o.model$he,
                          lower=c(-Inf, log(max(Conc))),
                          control=list(eval.max=10000, iter.max=1000))
```

```
## outer mgc:  30885.7
## outer mgc:  29541.07
## outer mgc:  30417.73
## outer mgc:  30142.61
## outer mgc:  30141.36
```

```
cat("Convergence codes from nlminb ", fit$myinvpareto$convergence, fit$myinvpareto$message, "\n"
)
```

```
## Convergence codes from nlminb  0 both X-convergence and relative convergence (5)
```

```
invpareto.model$he(invpareto.model$par)
```

```
##          [,1]     [,2]
## [1,] 40885.70 30141.36
## [2,] 30141.36     0.00
```

```
# the se cannot be found because the MLE for scale is on the boundary of the boundary space
sd<-sdreport( invpareto.model , getReportCovariance=TRUE)
```

```
## outer mgc:  30141.36
## outer mgc:  30171.52
## outer mgc:  30111.23
## outer mgc:  30141.36
## outer mgc:  30141.36
## outer mgc:  39.99585
```

```
sd
```

```
## sdreport(.) result
##           Estimate Std. Error
## log_shape 1.103313          0
## log_scale 3.688776        NaN
## Warning:
## Hessian of fixed effects was not positive definite.
## Maximum gradient component: 30141.36
```

```
summary(sd)
```

```
##            Estimate Std. Error
## log_shape  1.103313          0
## log_scale  3.688776        NaN
## shape      3.014136          0
## scale     39.995849        NaN
```

The hessian could not be computed at the MLE and so the standard errors for any parameter cannot be computed.

2.2.2 Censored data I

Values below detection limit cause no problems if the detection limit is less than the maximum uncensored value. The MLE for the scale parameter in the inverse-Pareto distribution will still be the largest uncensored value.

If the detection limit is larger than the largest uncensored data value (e.g., largest uncensored data value is 800, but the censored observation is <1000), it is unclear what the MLE for the scale parameter is. **This may be a case where the both parameter optimization will work - see below**

```
# Censored case (we censor 10% of the smallest values)
left  <- Conc
right <- Conc

select <- left < quantile(Conc, prob=.1)
left[ select] <- 0
right[select] <- quantile(Conc, prob=.1)

# Create the model
invpareto.init.scale <- max(right[is.finite(right)])  # Y values must be < scale value in invpar
eto distribution
invpareto.init.shape <- 1/mean(log(invpareto.init.scale/left[left>0]))
c(invpareto.init.scale, invpareto.init.shape )
```

```
## [1] 39.995849  4.054943
```

```
# note that for the invpareto, the MLE for the scale is the maximum value and so we fix it to th
e maximum value
# using the map argument
invpareto.model <- MakeADFun(
    data= list(left=left,
               right=right,
               weight =rep(1, length(Conc))),
     parameters=list(
        log_shape = log(invpareto.init.shape),
        log_scale = log(invpareto.init.scale)),
    map = list(log_scale = factor(NA)),
    DLL="ll_invpareto"
)

fit$invpareto <- nlminb(invpareto.model$par, invpareto.model$fn, invpareto.model$gr, invpareto.m
odel$he,
                    control=list(eval.max=10000, iter.max=1000))
```

```
## outer mgc:  3094.563
## outer mgc:  364.1825
## outer mgc:  6.990795
## outer mgc:  0.002712258
```

```
cat("Convergence codes from nlminb ", fit$invpareto$convergence, fit$invpareto$message, "\n")
```

```
## Convergence codes from nlminb  0 relative convergence (4)
```

```
sd<-sdreport( invpareto.model , getReportCovariance=TRUE)
```

```
## outer mgc:  0.002712258
## outer mgc:  9.007216
## outer mgc:  8.992792
## outer mgc:  3.01743
```

```
summary(sd)
```

```
##            Estimate Std. Error
## log_shape  1.104406 0.01054092
## shape      3.017430 0.03180650
## scale     39.995849 0.00000000
```

2.2.3 Censored data II

We now censor on the right, i.e., some values are recorded as 100+. This may cause many problems because now the estimated scale may be larger than the largest uncensored value. I will try and fit the model with the scale set to the largest non-infinite right-hand value + 1 and let TMB maximize the likelihood. Rather surprisingly it works just fine!

```
# Censored case II (we censor 10% of the smallest values and 10% of the rightmost values)
left  <- Conc
right <- Conc

select <- left < quantile(Conc, prob=.1)
left[ select] <- 0
right[select] <- quantile(Conc, prob=.1)

select <- left > quantile(Conc, prob=.9)
left [select] <- quantile(Conc, prob=.9)
right[select] <- Inf

# Create the model
invpareto.init.scale <- max(right[is.finite(right)])+10  # Y values must be < scale value in inv
pareto distribution
invpareto.init.shape <- 1/mean(log(invpareto.init.scale/left[left>0]))
c(invpareto.init.scale, invpareto.init.shape )
```

```
## [1] 48.582076  2.256978
```

```
invpareto.model <- MakeADFun(
    data= list(left = left,
               right= right,
               weight=rep(1,length(left))),
    parameters=list(
        log_shape = log(invpareto.init.shape),
        log_scale = log(invpareto.init.scale)),
    DLL="ll_invpareto"
)
#invpareto.model$env$tracepar <- TRUE


fit$invpareto <- nlminb(invpareto.model$par, invpareto.model$fn, invpareto.model$gr, invpareto.m
odel$he,
                    control=list(eval.max=10000, iter.max=1000))
```

```
## outer mgc:  17004.87
## outer mgc:  14113.93
## outer mgc:  12971.54
## outer mgc:  9499.966
## outer mgc:  15673.73
## outer mgc:  3732.372
## outer mgc:  643.6433
## outer mgc:  105.8935
## outer mgc:  1.310646
## outer mgc:  8.648778e-05
```

```
cat("Convergence codes from nlminb ", fit$invpareto$convergence, fit$invpareto$message, "\n")
```

```
## Convergence codes from nlminb  0 relative convergence (4)
```

```
sd<-sdreport( invpareto.model , getReportCovariance=TRUE)
```

```
## outer mgc:  8.648778e-05
## outer mgc:  28.74233
## outer mgc:  28.71383
## outer mgc:  802.9554
## outer mgc:  850.5454
## outer mgc:  39.94747
```

```
summary(sd)
```

```
##            Estimate  Std. Error
## log_shape  1.108433 0.011179761
## log_scale  3.687565 0.001166391
## shape      3.029606 0.033870273
## scale     39.947471 0.046594386
```

Rather surprisingly this works just fine.

3 Real data example

This is chemical D from the Burrlioz set provided by DF. The Burrlioz output is

```
CALL:  fit(dataframe = dataframe, ldensity = inverse.pareto.density,
    optim.trace = 0, start2 = c(1, -log(max(data) - 1)))

MODEL:  inverse.pareto
Parameter estimates:
log(alpha) -0.493423025402734
log(bound) -9.3758548104575

Log Lik: - 109.64324498392
   concen chemical
1    8944        d
2   10000        d
3    1650        d
4    2700        d
5   11800        d
6    4800        d
7     140        d
8     229        d
9     236        d
10    550        d
11   2226        d
12   5530        d
File name:  \\Hp-spectre\s\Environmetrics Australia\SSD R&D - Documents\Australia-Canada collabo
ration\Data Repository\chemical_d.csv
Date read:  Mon May 03 11:04:43 2021
```

It turns out that for numerical reasons, the data should be scaled in advance because the likelihood has component proportional to $shape * log(scale)$ which can overwhelm other parts of the log-likelihood. **This initial rescaling should ALWAYS be done to get the concentration values around 1.**

```
Conc <- c(8944, 10000,  1650,  2700, 11800,  4800,   140,   229,   236,   550,  2226,  5530)
Conc <- Conc/1000

invpareto.init.scale <- max(Conc)  # Y values must be < scale value in invpareto distribution
invpareto.init.shape <- 1/mean(log(invpareto.init.scale/Conc))
c(invpareto.init.scale, invpareto.init.shape )
```

```
## [1] 11.8000000  0.5318113
```

```
log(c(invpareto.init.scale, invpareto.init.shape ))
```

```
## [1]  2.4680995 -0.6314666
```

```
invpareto.model <- MakeADFun(
    data= list(left  = Conc,
               right = Conc,
               weight =rep(1, length(Conc))),
     parameters=list(
        log_shape = log(invpareto.init.shape),
        log_scale = log(invpareto.init.scale)),
    map = list(log_scale = factor(NA)),
    DLL="ll_invpareto"
)

#do.call(optim, invpareto.model)
fit$myinvpareto <- nlminb(invpareto.model$par, invpareto.model$fn, invpareto.model$gr, invparet
o.model$he,
                          control=list(eval.max=10000, iter.max=1000))
```

```
## outer mgc:  0
```

```
cat("Convergence codes from nlminb ", fit$myinvpareto$convergence, fit$myinvpareto$message, "\n"
)
```

```
## Convergence codes from nlminb  0 both X-convergence and relative convergence (5)
```

```
fit$myinvpareto$par
```

```
##  log_shape
## -0.6314666
```

```
sd<-sdreport( invpareto.model , getReportCovariance=TRUE)
```

```
## outer mgc:  0
## outer mgc:  0.012006
## outer mgc:  0.011994
## outer mgc:  0.5318113
```

```
summary(sd)
```

```
##             Estimate Std. Error
## log_shape -0.6314666  0.2886751
## shape      0.5318113  0.1535207
## scale     11.8000000  0.0000000
```

```
cat("log-likelihood value ", -invpareto.model$fn(invpareto.model$par), "\n")
```

```
## log-likelihood value  -26.6304
```

The scale estimate matches, but the TMB estimate of the $log(shape)$ parameter differs considerably from that provided by Burrlioz. This is a consequence of Burrlioz using the *optim()* function to maximize the likelihood as a function of both parameters and the *optim()* function fails to converge properly.

The likelihood value is also quite different because Burrlioz did not converge properly as well. This will have major impact if you try and use model averaging with the Burrlioz package.

3.1 Summary

The current implementation of Burrlioz does not fit the inverse-Pareto distribution properly because it tries to maximize the likelihood over both parameters and the *optim()* function gets b stuckb with the likelihood estimator at the largest data value. YMMV.

4 Recommendations

- Scale the data to get the concentrations around 1b ish and make the numerical methods more stable.
- Donb t rely on optimization of the 2 parameters in the Pareto or Inverse-Pareto distribution because the likelihood is not differentiable at the maximum.
- In the case of no-censoring and some cases of censoring, estimate the scale parameter using the order statistics and convert the problem to a one-dimensional fit.
- Estimates of HCx are obtainable using the MLEs; estimates of uncertainty for the HCx values canbe derived using the delta method or the multi-variate method (see companion papers) only if we estimate the se of the scale parameter separately from the optimization method. The likelihood profile and parametric bootstrap methods should still work without any b special casingb .

Appendix C: Unbiased estimation for the *inverse Pareto* distribution

David R. Fox

PREAMBLE

Simulation studies have confirmed that the *actual* percent coverage for HC_x confidence intervals based on the *inverse Pareto* distribution decreases as the sample size decreases. This is true for both the `Burrlioz` bootstrapped CIs and for the newly developed alternative method due to Fox (2021). For example, we observed that the actual coverage of 95% confidence intervals generated from a specified *inverse Pareto* distribution was very close to the nominal 95% (typically ~ 94%) for *n=100*. However, this coverage decreased uniformly with decreasing sample size whereby for *n=8* the actual coverage was about 80%. The reasons for this are explained next.

BIASED CONFIDENCE INTERVALS

The source of the problem described above is somewhat unique to the Pareto / *inverse Pareto* distributions by virtue of their constrained ranges (which are functions of the scale parameter).

PARETO DISTRIBUTION

Let *Y* have the Pareto(α, β) distribution given by Equation C.1.

$$f_Y(y;\alpha,\beta) = \alpha\beta^{\alpha} y^{-(\alpha+1)} \;;\; y > \beta;\; \alpha,\beta > 0 \tag{C.1}$$

INVERSE PARETO DISTRIBUTION

If *Y* has the distribution given by Equation C.1, then the distribution of $X = \frac{1}{Y}$ has the *inverse Pareto* distribution given by Equation C.2.

$$g_X(x;\alpha,\beta) = \alpha\beta^{\alpha} x^{\alpha-1} \;;\; 0 \le x \le \frac{1}{\beta};\; \alpha,\beta > 0 \tag{C.2}$$

Thus, we see that *Y* is bounded below by β and *X* is bounded above by $\frac{1}{\beta}$.

PARAMETER ESTIMATION

It is well known (e.g. Malik, 1970) that the MLEs for α and β are:

$$\hat{\beta} = \min\{Y_i\} \tag{C.3a}$$

$$\hat{\alpha} = \left[\ln\left(\frac{g}{Y_1}\right)\right]^{-1} \tag{C.3b}$$

where $g(\cdot)$ is the geometric mean of the *Y*-data i.e. $g = \left(\prod_{i=1}^{n} y_i\right)^{\frac{1}{n}}$. Furthermore, $\hat{\alpha}$ and $\hat{\beta}$ are *statistically independent*.

SAMPLING DISTRIBUTIONS OF $\hat{\alpha}$ AND $\hat{\beta}$

We next develop expressions for the sampling distributions of the MLEs for α and β .

$\hat{\alpha}$

Let $\lambda = n\alpha$; $r = n-1$ and $\hat{\alpha}' = \frac{1}{\hat{\alpha}}$. Then, it can be shown that:

$$g(\hat{\alpha}') = \frac{\lambda^r}{\Gamma(r)}\hat{\alpha}'^{r-1}e^{-\lambda\hat{\alpha}'} \;;\; \hat{\alpha}' > 0 \tag{C.4}$$

Equation C.4 is recognised as the $\text{gamma}(r,\lambda)$ *pdf* and therefore $\hat{\alpha}'$ has an inverse gamma distribution:

$$h(\hat{\alpha}) = \frac{\lambda^r}{\Gamma(r)}\left(\frac{1}{\hat{\alpha}}\right)^{r+1} e^{-\frac{\lambda}{\hat{\alpha}}} \;;\; \hat{\alpha} > 0 \tag{C.5}$$

Using standard results for the inverse gamma distribution we have:

$\mathrm{E}[\hat{\alpha}] = \frac{\lambda}{r-1}$ for $r > 1 \Rightarrow n > 2$ and hence:

$$\mathrm{E}[\hat{\alpha}] = \frac{n\lambda}{n-2} \;;\; n > 2 \tag{C.6}$$

Equation C.6 shows that the estimator $\hat{\alpha}$ given by Equation C.3b is a *biased* estimator of α since $\mathrm{E}[\hat{\alpha}] \neq \alpha$ although clearly the modified estimator:

$$\hat{\alpha}^* = \frac{n-2}{n}\hat{\alpha} \tag{C.7}$$

Is unbiased for α . Furthermore, since $\lim_{n\to\infty}\{\mathrm{E}[\hat{\alpha}]\} = \alpha$, the bias in $\hat{\alpha}$ becomes negligible for 'large' *n.*

Additionally, it is evident from Equation C.6 that $\hat{\alpha}$ *always* over-estimates the true value, α . For example, with $n = 8$ this bias is +33%.

OTHER USEFUL PROPERTIES

It is also readily shown that:

$$Var[\hat{\alpha}] = \frac{\lambda^2}{(r-1)^2(r-2)} \text{ for } r > 2 \;\Rightarrow\; n > 3 .$$

and so:

$$SE[\hat{\alpha}] = \frac{n\alpha}{n-2}\frac{1}{\sqrt{n-3}}$$

Our estimate of this standard error is thus:

$$\widehat{SE[\hat{\alpha}]} = \frac{n\hat{\alpha}^*}{n-2}\frac{1}{\sqrt{n-3}} = \frac{n}{n-2}\frac{n-2}{n}\hat{\alpha}\frac{1}{\sqrt{n-3}}$$

That is:

$$\widehat{SE[\hat{\alpha}]} = \frac{\hat{\alpha}}{\sqrt{n-3}} \tag{C.8}$$

Note, we use $\hat{\alpha}$ given by Equation C.3b in Equation C.8.

$\hat{\beta}$

For $\hat{\beta}$ it can be shown that $\hat{\beta}$ has the Pareto distribution given by Equation C.9.

$$h\left(\hat{\beta}\right) = \lambda \beta^{\lambda} \hat{\beta}^{-(\lambda+1)} \;\; ; \;\; \hat{\beta} > \beta \tag{C.9}$$

It is immediately evident from Equation C.9 that $\hat{\beta}$ is a biased estimator of β since $\hat{\beta} > \beta$ which implies $\mathrm{E}\left[\hat{\beta}\right] > \beta$ i.e., $\hat{\beta}$ always over-estimates β. The magnitude of this bias is readily established:

$$\mathrm{E}\left[\hat{\beta}\right] = \begin{cases} \infty & \text{for } \lambda \le 1 \;\Rightarrow\; n\alpha \le 1 \;\Rightarrow\; \alpha \le \dfrac{1}{n} \\ \dfrac{\lambda\beta}{\lambda - 1} & \text{for } \lambda > 1 \;\Rightarrow\; \alpha > \dfrac{1}{n} \end{cases}$$

Thus, for $\alpha > \dfrac{1}{n}$:

$$\mathrm{E}\left[\hat{\beta}\right] = \frac{\lambda}{\lambda - 1}\beta \;=\; \frac{n\alpha}{n\alpha - 1}\beta \tag{C.10}$$

Two things are obvious from Equation C.10:

1. The bias in $\hat{\beta}$ is a function of both the sample size, *n* and the shape parameter, α .
2. As $n \to \infty$, $bias \to 0$.

As was done for $\hat{\alpha}$, we can modify $\hat{\beta}$ to obtain an unbiased estimator for β. Hence:

$$\hat{\beta}^{*} = \frac{n\alpha - 1}{n\alpha}\hat{\beta} \;=\; \left[1 - \frac{1}{n\alpha}\right]\hat{\beta} \tag{C.11}$$

Now, α (the *true* parameter value) is unknown and so we replace α in Equation C.11 with our unbiased estimator, $\hat{\alpha}^{*}$ (Equation C.7). Thus:

$$\hat{\beta}^{*} = \left[1 - \frac{1}{n\hat{\alpha}^{*}}\right]\hat{\beta}$$

but, $n\hat{\alpha}^{*} = n\dfrac{n-2}{n}\hat{\alpha} = (n-2)\hat{\alpha}$ and so:

$$\hat{\beta}^{*} = \left[1 - \frac{1}{(n-2)\hat{\alpha}}\right]\hat{\beta} \tag{C.12}$$

where $\hat{\alpha}$ and $\hat{\beta}$ are the MLEs given by Equations C.3b and C.3a respectively.

OTHER USEFUL PROPERTIES

It is readily established that:

$$Var\left[\hat{\beta}\right] = \begin{cases} \infty & \text{for } \lambda \le 2 \Rightarrow n \le \dfrac{2}{\alpha} \\ \dfrac{\lambda\beta^2}{(\lambda-1)^2(\lambda-2)} & \text{for } \lambda > 2 \Rightarrow n > \dfrac{2}{\alpha} \end{cases}$$

Now, for $n > \dfrac{2}{\alpha}$:

$$Var\left[\hat{\beta}\right] = \frac{n\alpha\beta^2}{(n\alpha-1)^2(n\alpha-2)}$$

And so:

$$SE\left[\hat{\beta}\right] = \frac{\beta}{(n\alpha-1)} \cdot \sqrt{\frac{n\alpha}{(n\alpha-2)}} \qquad \text{(C.13)}$$

Replacing α and β in Equation C.13 with their unbiased estimators, our estimate of $SE\left[\hat{\beta}\right]$ becomes:

$$\widehat{SE\left[\hat{\beta}\right]} = \frac{\hat{\beta}^*}{(n\hat{\alpha}^*-1)} \cdot \sqrt{\frac{n\hat{\alpha}^*}{(n\hat{\alpha}^*-2)}}$$

and thus:

$$\widehat{SE\left[\hat{\beta}\right]} = \frac{\hat{\beta}}{n\hat{\alpha}} \cdot \frac{1}{\sqrt{1-\dfrac{2}{n\hat{\alpha}}}} \quad ; \; n > \frac{2}{\hat{\alpha}} \qquad \text{(C.14)}$$

where $\hat{\alpha}$ and $\hat{\beta}$ are the MLEs given by Equations C.3b and C.3a respectively.

THE CASE OF SHRINKING COVERAGE

As noted earlier, the *actual* coverage of confidence intervals for α and β decreases as $n \to 0$. `Burrlioz` uses non-parametric bootstrapping to generate its confidence intervals for an HC_x. Briefly, the procedure is as follows:

From the original sample: $\{X_1, X_2, \ldots, X_n\} = \Omega_X$ obtain the *i*th bootstrap sample by randomly sampling (*with replacement*) *n* values from Ω_X. We denote this bootstrapped sample as $\{W_{1i}, W_{2i}, \ldots, W_{ni}\}$ with $W_{ij} \in \Omega_X \ \forall \ i, j$. The scale parameter estimated from the *i*th bootstrap sample is $\tilde{\beta}_i = \min_j \left\{ Y_{ij} = \frac{1}{W_{ij}} \right\} = \frac{1}{\max_j \{W_{ij}\}}$. Thus $\tilde{\beta}_i \geq \hat{\beta} \ \forall \ i$ bootstrap samples where $\hat{\beta}$ is the estimate of β from Ω_X.

AN ALTERNATIVE (RAPID) METHOD

Instead of resampling from the data (non-parameteric bootstrapping) or from data generated from the *inverse Pareto* distribution whose parameters have been estimated from the sample, Ω_X (parametric bootstrapping), an alternative approach is to generate $\{\tilde{\alpha}, \tilde{\beta}\}$ pairs from their respective sampling distributions (Equations C.5 and C.9). For each simulated pair $\{\tilde{\alpha}_j, \tilde{\beta}_j\}$ an HC_p value is computed using Equation C.15.

$$\tilde{\psi}_p = \frac{p^{\frac{1}{\tilde{\alpha}}}}{\tilde{\beta}} \qquad \text{(C.15)}$$

To implement this strategy, the parameters in Equations C.5 and C.9 must be estimated from the sample. We have established that the MLEs for both α and β are biased and thus any data generated from distributions using these parameter estimates will also be biased. To counter this bias (and hence 'corruption' of the assumed confidence), we use the unbiased estimators given by Equations C.7 and C.12 in Equation C.15.

Appendix D: R-code for fitting mixtures

Joseph L Thorley, Carl Schwarz

Lnorm_lnorm

TBM code

```
Type ll_lnorm_lnorm(objective_function<Type>* obj) // normal with parameters mu an
d log(sigma)
{
  // Data
  DATA_VECTOR( left  );  // left and right values
  DATA_VECTOR( right );
  DATA_VECTOR( weight);  // weight

  // The order of these parameter statements determines the order of the estimates
in the vector of parameters
  PARAMETER( meanlog1 ); // first distribution
  PARAMETER( log_sdlog1    );
  PARAMETER( meanlog2 ); // second distribution
  PARAMETER( log_sdlog2    );
  PARAMETER( logit_pmix         );  // mixing proportion

  Type sdlog1 = exp(log_sdlog1);    // Convert to the [0,Inf] range
  Type sdlog2 = exp(log_sdlog2);
  Type pmix      = 1/(1+exp(-logit_pmix));// Convert to the [0,1] range


  Type nll = 0;  // negative log-likelihood
  int n_data    = left.size(); // number of data values
  Type pleft;    // probability that concentration < left(i)  used for censored da
ta
  Type pright;   // probability that concentration < right(i) used for censored da
ta

  //vector<Type> mynll(n_data); //(for debugging)

  // Probability of data conditional on parameter values for uncensored data
  // pdf of log(normal) obtained from pdf(normal) using the standard transformatio
n theory
  for( int i=0; i<n_data; i++){
     if(left(i) == right(i)){   // uncensored values
       if(left(i)>0){
        nll -= weight(i)*(log(pmix * dnorm(log(left(i)), meanlog1, sdlog1, false )
/left(i) +
                              (1-pmix) * dnorm(log(left(i)), meanlog2, sdlog2, fals
e )/left(i)));   // log likelihood for uncensored values
       };
     };
     if(left(i) < right(i)){    // censored values; no builtin function so we code
the cdf directly
        pleft = 0;
        if(left(i)>0){ pleft=pmix * pnorm(log(left(i)), meanlog1, sdlog1)+
                           (1-pmix) * pnorm(log(left(i)), meanlog2, sdlog2);};
        pright=pmix * pnorm(log(right(i)), meanlog1, sdlog1)+
                                       (1-pmix)* pnorm(log(right(i)), meanlog2, sd
log2);
```

```
        nll -= weight(i)*log(pright-pleft);
      };
  };

  ADREPORT(sdlog1);
  REPORT  (sdlog1);
  ADREPORT(sdlog2);
  REPORT  (sdlog2);
  ADREPORT(pmix);
  REPORT  (pmix);

  return nll;
}
```

Exported functions

ssd_plnorm_*lnorm*: Probability Distribution Function for Log-Normal/Log-Normal Mixture Distribution

```
ssd_plnorm_lnorm <- function(q, meanlog1 = 0, sdlog1 = 1,
                             meanlog2 = 1, sdlog2 = 1, pmix = 0.5,
                             lower.tail = TRUE, log.p = FALSE) {
  pdist("lnorm_lnorm", q = q, meanlog1 = meanlog1, sdlog1 = sdlog1,
        meanlog2 = meanlog2, sdlog2 = sdlog2, pmix = pmix,
        lower.tail = lower.tail, log.p = log.p)
}
```

ssd_qlnorm_*lnorm*: Cumulative Distribution Function for Log-Normal/Log-Normal Mixture Distribution

```
ssd_qlnorm_lnorm <- function(p, meanlog1 = 0, sdlog1 = 1,
                             meanlog2 = 1, sdlog2 = 1, pmix = 0.5,
                             lower.tail = TRUE, log.p = FALSE) {
  qdist("lnorm_lnorm", p = p, meanlog1 = meanlog1, sdlog1 = sdlog1,
        meanlog2 = meanlog2, sdlog2 = sdlog2, pmix = pmix,
        lower.tail = lower.tail, log.p = log.p)
}
```

ssd_rlnorm_*lnorm*: Random Generation for Log-Normal/Log-Normal Mixture Distribution

```
ssd_rlnorm_lnorm <- function(n, meanlog1 = 0, sdlog1 = 1,
                             meanlog2 = 1, sdlog2 = 1, pmix = 0.5, chk = TRUE) {
  rdist("lnorm_lnorm", n = n, meanlog1 = meanlog1, sdlog1 = sdlog1,
        meanlog2 = meanlog2, sdlog2 = sdlog2, pmix = pmix, chk = chk)
}
```

Other supporting functions

```
slnorm_lnorm <- function(data, pars = NULL) {
  if(!is.null(pars)) return(pars)

  x <- mean_weighted_values(data)

  x <- sort(x)
  n <- length(x)
  n2 <- floor(n / 2)
  x1 <- x[1:n2]
  x2 <- x[(n2+1):n]
  s1 <- slnorm(data.frame(left = x1, right = x1, weight = 1))
```

```
  s2 <- slnorm(data.frame(left = x2, right = x2, weight = 1))
  names(s1) <- paste0(names(s1), "1")
  names(s2) <- paste0(names(s2), "2")
  logit_pmix <- list(logit_pmix = 0)
  c(s1, s2, logit_pmix)
}

blnorm_lnorm <- function(x, min_pmix, ...) {
  list(lower = list(meanlog1 = -Inf, log_sdlog1 = -Inf, meanlog2 = -Inf, log_sdlog
2 = -Inf, logit_pmix = logit(min_pmix)),
       upper = list(meanlog1 = Inf, log_sdlog1 = Inf, meanlog2 = Inf, log_sdlog2 =
Inf, logit_pmix = logit(1 - min_pmix)))
}

plnorm_lnorm_ssd <- function(q, meanlog1, sdlog1, meanlog2, sdlog2, pmix) {
  if(sdlog1 <= 0 || sdlog2 <= 0 || pmix <= 0 || pmix >= 1) {
    return(NaN)
  }
  pmix * plnorm_ssd(q, meanlog1, sdlog1) + (1 - pmix) * plnorm_ssd(q, meanlog2, sd
log2)
}

qlnorm_lnorm_ssd <- function(p, meanlog1, sdlog1, meanlog2, sdlog2, pmix) {
  if(sdlog1 <= 0 || sdlog2 <= 0 || pmix <= 0 || pmix >= 1) {
    return(NaN)
  }

  f <- function(x) {
    plnorm_lnorm_ssd(x, meanlog1, sdlog1, meanlog2, sdlog2, pmix) - p
  }
  stats::uniroot(f, lower = 0, upper = 10, extendInt = "yes")$root
}

rlnorm_lnorm_ssd <- function(n, meanlog1, sdlog1, meanlog2, sdlog2, pmix) {
  if(sdlog1 <= 0 || sdlog2 <= 0 || pmix <= 0 || pmix >= 1) {
    return(rep(NaN, n))
  }
  dist <- stats::rbinom(n, 1, pmix)
  dist <- as.logical(dist)
  x <- rep(NA_real_, n)
  x[dist] <- rlnorm_ssd(sum(dist), meanlog = meanlog1, sdlog = sdlog1)
  x[!dist] <- rlnorm_ssd(sum(!dist), meanlog = meanlog2, sdlog = sdlog2)
  x
}
```

llogis_llogis

TMB code

```
Type ll_llogis_llogis(objective_function<Type>* obj) // normal with parameters mu
and log(sigma)
{
  // Data
  DATA_VECTOR( left  );  // left and right values
  DATA_VECTOR( right );
  DATA_VECTOR( weight);  // weight

  // The order of these parameter statements determines the order of the estimates
in the vector of parameters
```

```
  PARAMETER( locationlog1 ); // first distribution
  PARAMETER( log_scalelog1    );
  PARAMETER( locationlog2 ); // second distribution
  PARAMETER( log_scalelog2    );
  PARAMETER( logit_pmix         );  // mixing proportion

  Type scalelog1 = exp(log_scalelog1);    // Convert to the [0,Inf] range
  Type scalelog2 = exp(log_scalelog2);
  Type pmix      = 1/(1+exp(-logit_pmix));// Convert to the [0,1] range


  Type nll = 0;  // negative log-likelihood
  int n_data    = left.size(); // number of data values
  Type pleft;    // probability that concentration < left(i)  used for censored da
ta
  Type pright;   // probability that concentration < right(i) used for censored da
ta

  //vector<Type> mynll(n_data); //(for debugging)

  // Probability of data conditional on parameter values for uncensored data
  // pdf of log(normal) obtained from pdf(normal) using the standard transformatio
n theory
  for( int i=0; i<n_data; i++){
     if(left(i) == right(i)){   // uncensored values
       if(left(i)>0){
        nll -= weight(i)*(log(   pmix * dlogis( log(left(i)), locationlog1, scalel
og1, false )/left(i) +
                              (1-pmix) * dlogis( log(left(i)), locationlog2, scalel
og2, false )/left(i)));   // log likelihood for uncensored values
       };
     };
     if(left(i) < right(i)){    // censored values; no builtin function so we code
the cdf directly
        pleft = 0;
        if(left(i)>0){ pleft=pmix   * 1/(1+exp(-(log(left(i))-locationlog1)/scalel
og1))+
                           (1-pmix)* 1/(1+exp(-(log(left(i))-locationlog2)/scalel
og2));};
        pright=pmix    * 1/(1+exp(-(log(right(i))-locationlog1)/scalelog1))+
                                    (1-pmix)* 1/(1+exp(-(log(right(i))-location
log2)/scalelog2));
        nll -= weight(i)*log(pright-pleft);
     };
     // mynll(i) = nll;  // for debugging
  };

  ADREPORT(scalelog1);
  REPORT  (scalelog1);
  ADREPORT(scalelog2);
  REPORT  (scalelog2);
  ADREPORT(pmix);
  REPORT  (pmix);


  //REPORT( mynll);  //for debugging
  return nll;
}
```

Exported functions

ssd_pllogis_*llogis*: Cumulative Distribution Function for Log-Logistic/Log-Logistic Mixture Distribution

```
ssd_pllogis_llogis <- function(q, locationlog1 = 0, scalelog1 = 1,
                               locationlog2 = 1, scalelog2 = 1, pmix = 0.5,
                               lower.tail = TRUE, log.p = FALSE) {
  pdist("logis_logis", q = q, location1 = locationlog1, scale1 = scalelog1,
        location2 = locationlog2, scale2 = scalelog2, pmix = pmix,
        lower.tail = lower.tail, log.p = log.p, .lgt = TRUE)
}
```

ssd_qllogis_*llogis*: Cumulative Distribution Function for Log-Logistic/Log-Logistic Mixture Distribution

```
ssd_qllogis_llogis <- function(p, locationlog1 = 0, scalelog1 = 1,
                               locationlog2 = 1, scalelog2 = 1, pmix = 0.5,
                               lower.tail = TRUE, log.p = FALSE) {
  qdist("logis_logis", p = p, location1 = locationlog1, scale1 = scalelog1,
        location2 = locationlog2, scale2 = scalelog2, pmix = pmix,
        lower.tail = lower.tail, log.p = log.p, .lgt = TRUE)
}
```

ssd_rllogis_*llogis*: Random Generation for Log-Logistic/Log-Logistic Mixture Distribution

```
ssd_rllogis_llogis <- function(n, locationlog1 = 0, scalelog1 = 1,
                               locationlog2 = 1, scalelog2 = 1, pmix = 0.5, chk = TRUE
) {
  rdist("logis_logis", n = n, location1 = locationlog1, scale1 = scalelog1,
        location2 = locationlog2, scale2 = scalelog2, pmix = pmix, .lgt = TRUE, ch
k = chk)
}
```

Other supporting functions

```
sllogis_llogis <- function(data, pars = NULL) {
  if(!is.null(pars)) return(pars)

  x <- mean_weighted_values(data)
  x <- sort(x)
  n <- length(x)
  n2 <- floor(n / 2)
  x1 <- x[1:n2]
  x2 <- x[(n2+1):n]
  s1 <- sllogis(data.frame(left = x1, right = x1, weight = 1))
  s2 <- sllogis(data.frame(left = x2, right = x2, weight = 1))
  names(s1) <- paste0(names(s1), "1")
  names(s2) <- paste0(names(s2), "2")
  logit_pmix <- list(logit_pmix = 0)
  c(s1, s2, logit_pmix)
}

bllogis_llogis <- function(x, min_pmix, ...) {
  list(lower = list(locationlog1 = -Inf, log_scalelog1 = -Inf, locationlog2 = -Inf
, log_scalelog2 = -Inf, logit_pmix = logit(min_pmix)),
       upper = list(locationlog1 = Inf, log_scalelog1 = Inf, locationlog2 = Inf, l
```

```
og_scalelog2 = Inf, logit_pmix = logit(1 - min_pmix)))
}

plogis_logis_ssd <- function(q, location1, scale1, location2, scale2, pmix) {
  if(scale1 <= 0 || scale2 <= 0 || pmix <= 0 || pmix >= 1) {
    return(NaN)
  }
  pmix * plogis_ssd(q, location1, scale1) + (1 - pmix) * plogis_ssd(q, location2,
scale2)
}

qlogis_logis_ssd <- function(p, location1, scale1, location2, scale2, pmix) {
  if(scale1 <= 0 || scale2 <= 0 || pmix <= 0 || pmix >= 1) {
    return(NaN)
  }
  f <- function(x) {
    plogis_logis_ssd(x, location1, scale1, location2, scale2, pmix) - p
  }
  stats::uniroot(f, lower = 0, upper = 10, extendInt = "yes")$root
}

rlogis_logis_ssd <- function(n, location1, scale1, location2, scale2, pmix) {
  if(scale1 <= 0 || scale2 <= 0 || pmix <= 0 || pmix >= 1) {
    return(rep(NaN, n))
  }
  dist <- stats::rbinom(n, 1, pmix)
  dist <- as.logical(dist)
  x <- rep(NA_real_, n)
  x[dist] <- rlogis_ssd(sum(dist), location = location1, scale = scale1)
  x[!dist] <- rlogis_ssd(sum(!dist), location = location2, scale = scale2)
  x
}
```

Appendix E: *L*-Moment estimators for the *Burr III* distribution

David R. Fox

Introduction

- Brief review of use of Burr distributions for SSDs including development of `Burrlioz` software
- Characteristics and properties of Burr distributions – recap known statistical results
- Estimation strategies – issues with maximum likelihood

L-Moments

The concept of L-moments was introduced by Hosking (1990) and they have found to be particularly useful for describing probability distributions. Unlike conventional moments (which are also widely used to describe and fit probability distributions), L-moments have several unique properties. Among these is the fact that any distribution with finite mean is uniquely determined by its L-moments.

Although the use of conventional moments for estimating the parameters of a probability distribution (the so-called method-of-moments or MoM estimation) has a long history, their use in this context has largely given way to likelihood-based approaches (maximum likelihood estimates or MLEs).

Maximum likelihood estimators are generally more accurate than conventional MoM estimators and they enjoy several desirable statistical properties not shared by MoM estimators such as asymptotic convergence to the MVUBE and asymptotically normality. Furthermore, for some monotonic function $g(\cdot)$, the estimate $g\left(\hat{\theta}\right)$ is the mle of $g\left(\theta\right)$ where $\hat{\theta}$ is the mle of θ.

Advantages of L-moment estimators are:

- They are more robust than conventional moments to outliers in the data;
- they enable more secure inferences to be made from small samples about an underlying probability distribution;
- they can yield more efficient parameter estimates than maximum likelihood estimates;
- they characterise a wider range of distributions than is possible with conventional moments;
- small sample bias of L-moment estimators is generally less than conventional moment-based estimators;
- L-moment estimators can usually be used when MLEs are either unavailable or difficult to compute; and
- L-moments can be used to specify a distribution even when some of its conventional moments do not exist.

Since their introduction 30 years ago, L-moments have been widely used by hydrologists for flood-frequency analyses (Kjeldsen et al. 2002; Kroll and Vogel, 2002; Lim and Lye, 2003).

DEFINITION AND PROPERTIES OF L-MOMENTS

The following derivation largely follows that given by Hosking (1990).

Suppose X is a real-valued random variable with cumulative distribution function (*cdf*) $F_X(x)$ and probability density function (*pdf*) $f_X(x)$. Let the r^{th} order statistic from a sample of size *n* $\{X_1, X_2, \ldots, X_n\}$ be denoted as $X_{r:n}$. The *L-moments* of *X* are defined as:

$$\lambda_r \equiv r^{-1}\sum_{k=0}^{r-1}(-1)^k \binom{r-1}{k} E[\![X_{r-k:r}]\!] \qquad r = 1, 2, \ldots \tag{E.1}$$

where $E[\![X_{r-k:r}]\!]$ denotes the expected value of the indicated order statistic and

$$E[\![X_{j:r}]\!] = \frac{r!}{(j-1)!(r-j)!}\int x\left[F_X(x)\right]^{j-1}\left[1-F_X(x)\right]^{r-j} f_X(x)\,dx \tag{E.2}$$

Using Equation 1 it is readily established that the first four L-moments are:

$$\begin{aligned}
\lambda_1 &= E[\![X]\!] \\
\lambda_2 &= \tfrac{1}{2}E[\![X_{2:2} - X_{1:2}]\!] \\
\lambda_3 &= \tfrac{1}{3}E[\![X_{3:3} - 2X_{2:3} + X_{1:3}]\!] \\
\lambda_4 &= \tfrac{1}{4}E[\![X_{4:4} - 3X_{3:4} + 3X_{2:4} - X_{1:4}]\!]
\end{aligned} \tag{E.3}$$

As with conventional moments, the first four L-moments of a distribution are measures of *location*, *dispersion*, *skewness* and *kurtosis* respectively and that $\lambda_1 = \mu_X$ (the conventional mean of *X*).

Hosking (2006) showed that the characterisation of a distribution by its L-moments is nonredundant, meaning that if one L-moment is dropped from the set the distribution cannot be determined from the remaining L-moments. Hosking (2006) further suggested that the information contained in the r^{th} L-moment is maximally independent of the information given in the first *r-1* L-moments thereby making the L-moments particularly suited as summary statistics of a distribution.

The *L-moment ratios* $\tau_r \equiv \lambda_r / \lambda_2 \quad (r = 3, 4, \ldots)$ are dimensionless quantities that will also prove useful in characterising probability distributions and are more readily interpreted than their conventional moment counterparts by virtue of the fact $|\tau_r| < 1, \quad r = 3, 4, \ldots$. While κ (the conventional measure of kurtosis) has no unique interpretation, it is invariably associated with a distribution's 'peakedness'. Hosking (1990) notes that the L-kurtosis, τ_4 is equally difficult to interpret uniquely and is best thought of as a measure like κ but giving less weight to the extreme tails of the distribution.

L-MOMENTS SAMPLE ESTIMATES

Unbiased estimation of the population L-moments is readily achieved using the sample L-moments, l_r defined as follows:

$$l_r = \binom{n}{r}^{-1} \sum\sum\cdots\sum_{1 \le i_1 < i_2 < \ldots < i_r \le n} r^{-1} \sum_{k=0}^{r-1}(-1)^k \binom{r-1}{k} x_{i_{r-k}:n}, \quad r = 1, 2, \ldots, n \tag{E.4}$$

where $x_{1:n} \le x_{2:n} \le \ldots \le x_{n:n}$ is the ordered sample.

Using Equation E.4 we obtain the following expressions for the first four sample L-moments:

$$
\begin{aligned}
l_1 &= n^{-1}\sum_i x \\
l_2 &= \frac{1}{2}\binom{n}{2}^{-1}\sum_{i>j}\sum\left(x_{i:n}-x_{j:n}\right) \\
l_3 &= \frac{1}{3}\binom{n}{3}^{-1}\sum_{i>j>k}\sum\sum\left(x_{i:n}-2x_{j:n}+x_{k:n}\right) \\
l_4 &= \frac{1}{4}\binom{n}{4}^{-1}\sum_{i>j>k>l}\sum\sum\sum\left(x_{i:n}-3x_{j:n}+3x_{k:n}-x_{l:n}\right)
\end{aligned}
\qquad \text{(E.5)}
$$

We next develop explicit expressions for the population L-moments of the *Burr III* family of densities.

L-Moments of the *Burr III* distribution

The *cdf* and *pdf* for the *Burr III* distribution are given respectively by Equations 8 and 9.

$$F_X(x;b,c,k) = \frac{1}{\left[1+\left(\frac{b}{x}\right)^c\right]^k} \;;\; x>0;\;\; b,c,k>0 \qquad \text{(E.6)}$$

$$f_X(x;b,c,k) = \frac{bck\left(\frac{b}{x}\right)^{c-1}}{x^2\left[1+\left(\frac{b}{x}\right)^c\right]^{k+1}} \;;\; x>0 \;;\; b,c,k>0 \qquad \text{(E.7)}$$

From Equations (1) and (2) we have:

$$\lambda_r = \frac{1}{r}\; l_r = \frac{1}{r}\sum_{m=0}^{r-1}(-1)^m\binom{r-1}{m}E\left[\!\left[X_{r-m:r}\right]\!\right] \qquad \text{(E.8)}$$

where

$$
\begin{aligned}
E\left[\!\left[X_{j:r}\right]\!\right] &= \frac{r!}{(j-1)!(r-j)!}\int_0^\infty x\,F(x)^{j-1}\left[1-F(x)\right]^{r-j}\,f(x)\,dx \\
&= \frac{r!}{(j-1)!(r-j)!}\int_0^\infty x\left\{\left[1+\left(\frac{b}{x}\right)^c\right]^{-k}\right\}^{j-1}\left\{1-\left[1+\left(\frac{b}{x}\right)^c\right]^{-k}\right\}^{r-j} bck\left(\frac{b}{x}\right)^{c-1} x^2\left[1+\left(\frac{b}{x}\right)^c\right]^{-(k+1)} dx
\end{aligned}
\qquad \text{(E.9)}
$$

Letting I be the integral in Equation E.9, we have

$$I = bck\int_0^\infty \frac{1}{x}\left[1+\left(\frac{b}{x}\right)^c\right]^{-k(j-1)}\left\{1-\left[1+\left(\frac{b}{x}\right)^c\right]^{-k}\right\}^{r-j}\left(\frac{b}{x}\right)^{c-1}\left[1+\left(\frac{b}{x}\right)^c\right]^{-(k+1)} dx$$

and making the substitution $u = \dfrac{1}{1+\left(\dfrac{b}{x}\right)^c}$ we obtain

$$I = bk\int_0^1 u^{k(j-1)+k+1/c-1}\ (1-u)^{-1/c}\left(1-u^k\right)^{r-j}\ du \tag{E.10}$$

Hence

$$\frac{I}{bk\,B(\alpha,\beta)} = \frac{1}{B(\alpha,\beta)}\int_0^1 u^{\alpha-1}\ (1-u)^{\beta-1}\left(1-u^k\right)^{r-j}\ du \tag{E.11}$$

where $\alpha = kj + 1/c$; $\beta = 1 - 1/c$ and $\mathrm{B}(\cdot,\cdot)$ is the standard beta function.

It is immediately apparent from Equation E.11 that $\dfrac{I}{bk\,B(\alpha,\beta)} = E\left[\!\left[\left(1-U^k\right)^{r-j}\right]\!\right]$ with

$U \sim beta(\alpha,\beta)$. From standard statistical theory, $E\left[\!\left[U^r\right]\!\right] = \dfrac{B(r+\alpha,\beta)}{B(\alpha,\beta)}$ and thus Equation E.10

becomes:

$$I = bk\sum_{i=0}^{r-j}(-1)^{r-j-i}\binom{r-j}{i}B\left[k(r-i)+1/c;1-1/c\right] \tag{E.12}$$

Substituting Equation 14 back into Equation 10 we have:

$$\begin{aligned}\lambda_r &= bk\sum_{m=0}^{r-1}\sum_{i=0}^{m}\left\{\frac{1}{r}(-1)^{2m-i}\binom{r-1}{m}\frac{r!}{(r-m-1)!m!}\binom{m}{i}B\left[k(r-i)+1/c;1-1/c\right]\right\}\\ &= bk\sum_{m=0}^{r-1}\sum_{i=0}^{m}\left\{(-1)^i\binom{r-1}{m}^2\binom{m}{i}B\left[k(r-i)+1/c;1-1/c\right]\right\}\end{aligned} \tag{E.13}$$

Using Equation E.13, L-moment estimation of the parameters b, c, k for the distribution given by Equation E.7 is simply a matter of setting $\hat{\lambda}_r = l_r$, $r = 1,2,3$ and solving the resulting system of 3 non-linear Equations. Selecting suitable initial estimates for this iterative procedure is facilitated by a preliminary inspection of the *L-moment ratios* as described in the following Section.

L-moment parameter estimation for the *Burr III* distribution

As noted earlier, the L-moment ratios τ_r are dimensionless quantities which makes them useful for characterising and comparing distributions. A common tool used for this purpose is an L-moment ratio plot in which values of the L-moment kurtosis are plotted on the vertical scale and values of the L-moment skewness on the horizontal scale. For the *Burr III* distribution given by Equation E.7, this L-moment ratio plot is independent of the parameter *b*. It is therefore possible to estimate *c* and *k* directly from contour plots of $\tau_3(c,k)$ and $\tau_4(c,k)$ by finding $\left\{\hat{c},\hat{k}\right\}$ such that $\tau_3(\hat{c},\hat{k})$ and $\tau_4(\hat{c},\hat{k})$

match their sample values. The estimate $\hat{b}$ of the parameter *b* is readily estimated once $\{\hat{c},\hat{k}\}$ have been determined. However, using the third and fourth moments in this way does not guarantee $\lambda_2\left(\hat{b},\hat{c},\hat{k}\right)=l_2$. For the 3-parameter Burr distribution we believe it is more important to match the first three moments rather than L-moments 1,3, and 4. This is readily achieved with a slight modification to the procedure just described.

INITIAL ESTIMATES FOR L-MOMENT MATCHING

The L-moment ratios τ_r were defined by Hosking (1990) for $r=3,4,\ldots$. We introduce a new quantity *LCV* which is similar to the conventional moment-based coefficient of variation.

$$LCV=\frac{\lambda_2}{\lambda_1} \tag{E.14}$$

As with the regular coefficient of variation, *LCV* is a dimensionless, scaled measure of dispersion. Initial estimates of *c* and *k* can be found by finding the $\{c,k\}$ pair for which *LCV* and τ_3 are roughly equal to their sample values. The procedure is illustrated with the use of 2 examples.

Example 1: SSD-fitting to lead in marine waters data.

The data in Table E1 are concentrations (microgram per litre) of lead found in a sample of 16 marine species.

The first four sample L-moments for these data are $\{232.406,164.71,95.872,53.827\}$ from which it is readily established that the sample values of *LCV* and τ_3 are 0.709 and 0.582 respectively.

Plots of *LCV* and τ_3 as functions of *c* and *k* are shown in Figures 1(a) and 1(b). Superimposing these 2 plots we find that an approximate solution for $\{c,k\}$ is found at the interSection of the 0.709 contour of Figure 1(a) and the 0.581 contour of Figure 1E(b) (these contour lines were determined by the plotting software) (Figure E2).

Table E1. Concentrations (micrograms per litre) of lead in marine species (Data courtesy Dr. Graeme Batley, CSIRO).

concen	Name	Species
251	Dunaliella tertiolecta	Green alga
1234	Phaeodactylum tricornutum	Diatom
29.4	Skeletonema costatum	Diatom
11.9	Champia parvula	Macroalga
397	Tisbe battagliai	Copepod
48	Strongylocentrus pupuratus	Sea urchin
119	Paracentrotus lividus	Sea urchin
250	Dendraster excentricus	Sea urchin
10	Heliocideris tubiculata	Sea urchin
7	Americamysis bahia	Mysid
51	Mytilus galloprovincialis	Mussel
9	Mytilus trossolus	Mussel
931	Crassotrea gigas	Oyster
95.9	Neanthes arenaceodantata	Polychaete
230	Cyprinodon variegatus	Fish
44.3	Atherinops affinis	Fish

From Figure E2, we obtain the approximate solution $\hat{c} = 2$ and $\hat{k} = 0.2$. The L-moment estimator of $\hat{b} = 463.7$ is readily found using Equation E.15.

$$\hat{b} = \frac{l_1}{\hat{k}B\left(\hat{k} + \frac{1}{\hat{c}}, 1 - \frac{1}{\hat{c}}\right)} \tag{E.15}$$

With these initial values of $\{b, c, k\}$ convergence is rapid and yields the L-moment estimators of $\left\{\hat{b} = 479.396;\ \hat{c} = 2.006; \hat{k} = 0.192\right\}$.

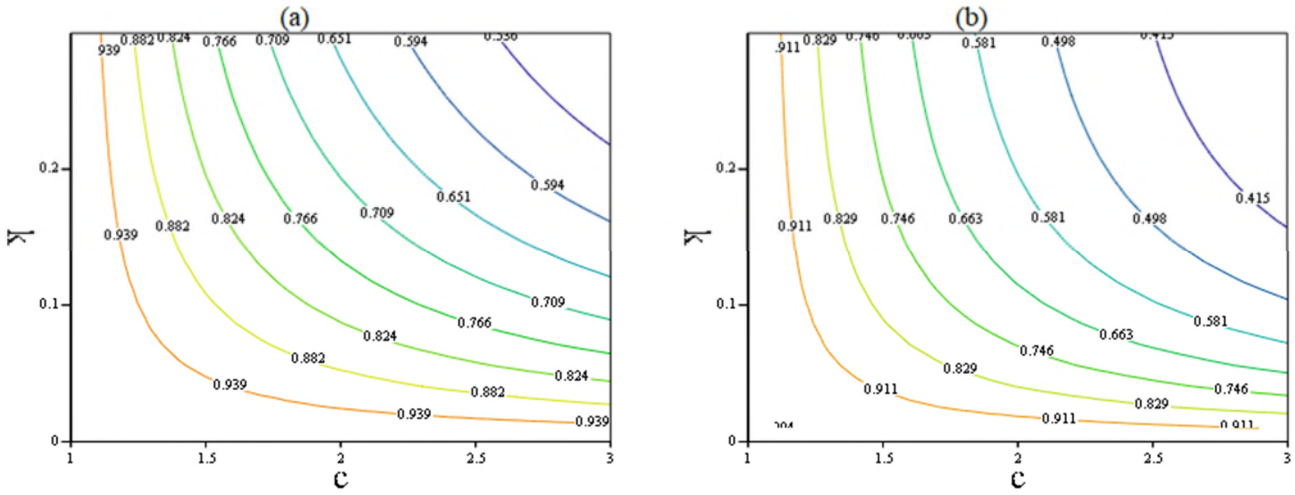


Figure E1. Plots of (a) LCV and (b) τ3 for lead data as a function of c and k in Equation 9.

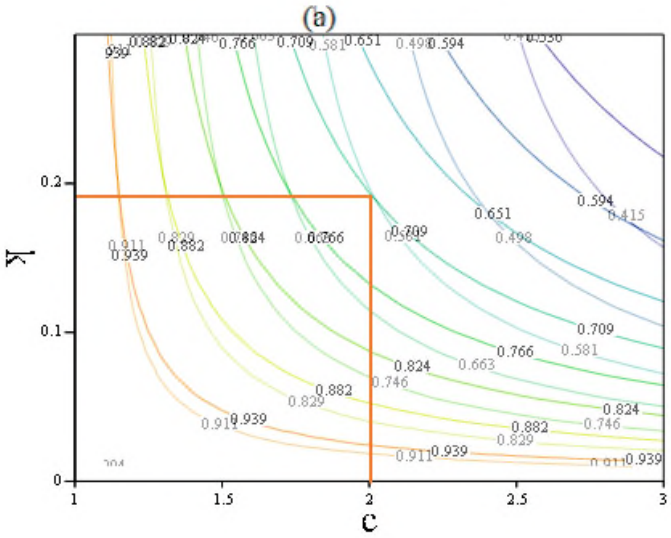


Figure E2. Figures 1(a) and 1(b) overlaid on single plot

A plot of the empirical and fitted *cdfs* is shown in Figure E3.

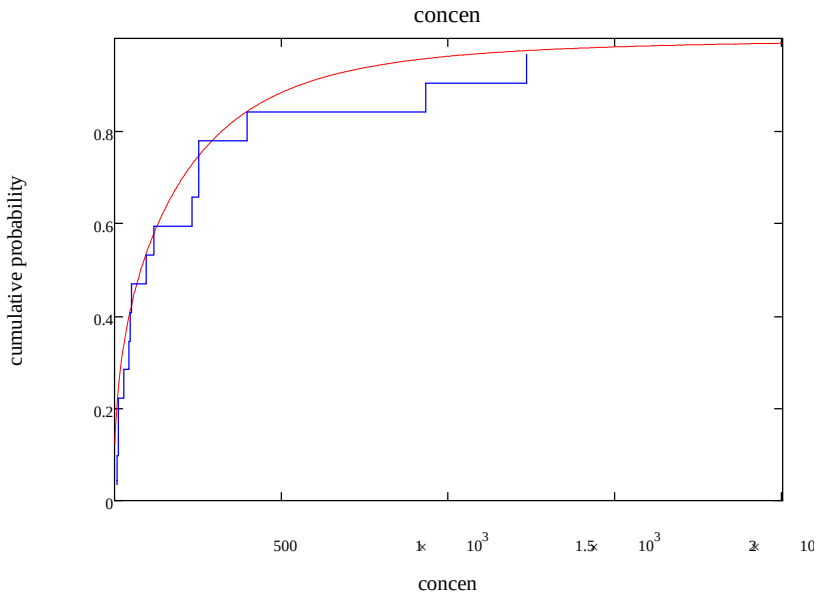


Figure E3. Comparison of fitted (red curve) and empirical (blue curve) cdfs for Pb in marine waters example.

Example 2: SSD-fitting to Metolachlor in freshwater.

The data in Table E2 are concentrations (microgram per litre) of metolachlor found in a sample of 21 freshwater species.

Table E2. Summary of single chronic toxicity values, all species used to derive default guideline values for metolachlor in freshwater (Source: ANZG, 2020).

Taxonomic group (Phylum)	Species	Life stage	Duration (d)	Toxicity measure [a]	Test endpoint	Final toxicity value (µg/L)
Diatom (Bacillariophyta)	*Achnanthidium minutissimum* [d]	Exponential growth phase	4	Chronic EC10	Cell density	6 528
Blue–green alga (Cyanobacteria)	*Anabaena flosaquae*	Not stated	5	Chronic EC50	Biomass yield, growth rate, AUC [c]	240
Macrophyte (Tracheophyta)	*Ceratophyllum demersum* [d]	Not stated	14	Chronic EC50	Wet weight	14
Green alga (Chlorophyta)	*Chlamydomonas reinhardtii* [d]	Not stated	4	Chronic EC50	Chlorophyll-a content	228
Green alga (Chlorophyta)	*Chlorella pyrenoidosa* [d]	Exponential growth phase	4	Chronic NOEC	Chlorophyll-a content	1
Diatom (Bacillariophyta)	*Craticula accomoda* [d]	Exponential growth phase	4	Chronic EC10	Chlorophyll-a content	4 016
Diatom (Bacillariophyta)	*Cyclotella meneghiniana* [d]	Exponential growth phase	4	Chronic EC10	Cell density	925
Macroinvertebrate (Arthropoda)	*Daphnia magna*	<24 hour old	21	Chronic EC10	Young per female	224
Macrophyte (Tracheophyta)	*Elodea canadensis* [d]	Not stated	14	Chronic EC50	Wet weight	471

Taxonomic group (Phylum)	Species	Life stage	Duration (d)	Toxicity measure [a]	Test endpoint	Final toxicity value (µg/L)
Diatom (Bacillariophyta)	*Encyonema silesiacum* [d]	Exponential growth phase	4	Chronic EC10	Chlorophyll-a content	1 048
Diatom (Bacillariophyta)	*Fragilaria capucina* var *vaucheriae* [d]	Not stated	4	Chronic EC10	Chlorophyll-a content	90
Diatom (Bacillariophyta)	*Gomphonema gracile* [d]	Exponential growth phase	7	Chronic NOEC	Live cell density	1
Diatom (Bacillariophyta)	*Gomphonema parvulum*	Exponential growth phase	4	Chronic EC10	Chlorophyll-a content	6 384
Macrophyte (Tracheophyta)	*Lemna gibba*	Stage 3 (3 fronds/plants)	14	Chronic NOEL	Frond number	8.4
Diatom (Bacillariophyta)	*Mayamaea fossalis*	Exponential growth phase	4	Chronic EC10	Chlorophyll-a content	863
Macrophyte (Tracheophyta)	*Najas* sp.	Not stated	14	Chronic EC50	Wet weight	48.4
Diatom (Bacillariophyta)	*Navicula pelliculosa* [d]	Not stated	5	Chronic EC50	Biomass yield, growth rate, AUC [c]	76
Fish (Chordata)	*Pimephales promelas*	Early life stage	35	Chronic LOEC	Mortality	640
Green alga (Chlorophyta)	*Pseudokirchneriella subcapitata* [b]	Not stated	3	Chronic NOEC	Cell density	27.4
Green alga (Chlorophyta)	*Scenedesmus vacuolatus*	Exponential growth phase	2	Chronic EC50	Cell density	0.53
Diatom (Bacillariophyta)	*Ulnaria ulna* [d]	Exponential growth phase	4	Chronic EC10	Chlorophyll-a content	27

Given the wide range of toxicity values in Table E2, we will work with (natural) log-transformed data. The transformed data now range from -0.635 to 8.784. The *Burr III* distribution cannot be fitted directly to these transformed data given the presence of negative values and the negative value of l_3. To overcome both issues we further transform the data by subtracting the log-transformed values from 10. Thus, in terms of the original data (*x*), the transformed values (*y*) are obtained as $y_i = 10 - \ln(x_i)$. The first four sample L-moments for the y values are 5.296, 1.611, 0.175, and 0.147. The sample *LCV* value is 0.304 and the value of τ_3 is 0.109. Following the graphical procedure as previously described (Figures E4 and E5) we obtain the initial L-moment estimates of c and k as 5.8 and 0.26 respectively. Substituting these values into Equation E.15 gives $\hat{b} = 7.925$. The final L-moment estimates are $\left\{\hat{b} = 7.909,\ \hat{c} = 5.768,\ \hat{k} = 0.262\right\}$

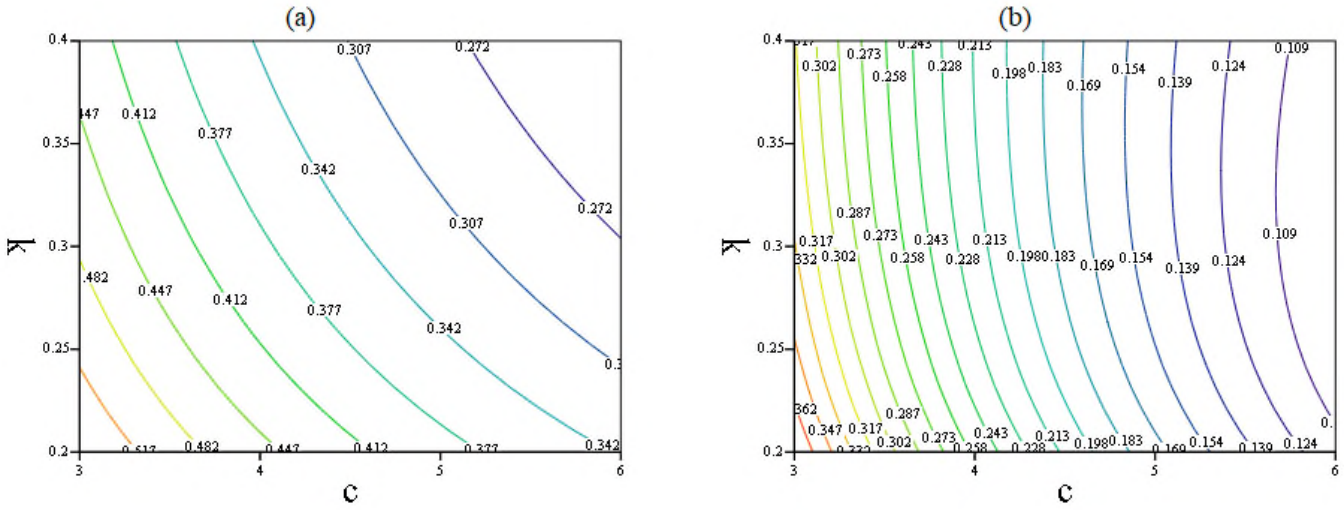


Figure E4. Plots of (a) LCV and (b) τ3 for metolachlor data as a function of c and k in Equation 9.

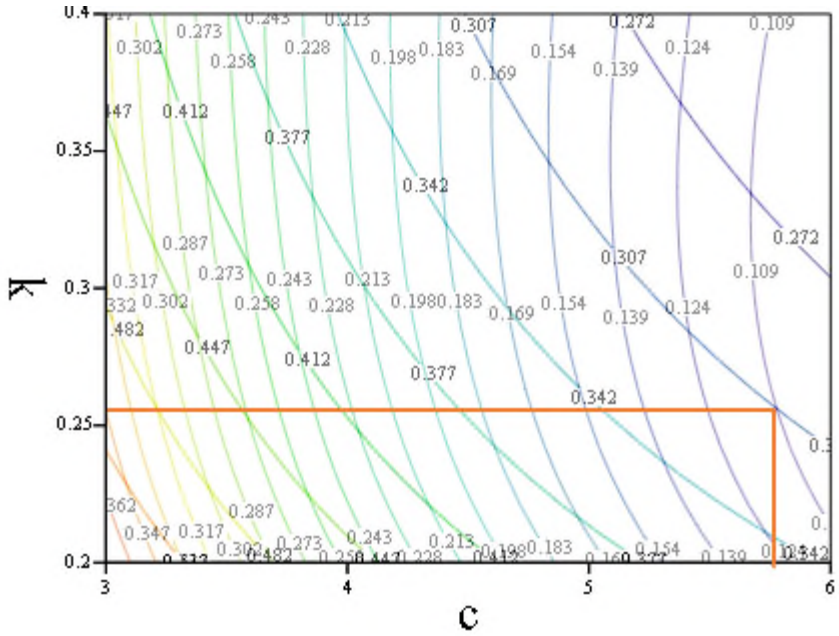


Figure E5. Figures 4(a) and 4(b) overlaid on single plot.

A plot of the empirical and fitted *cdfs* is shown in Figure E6.

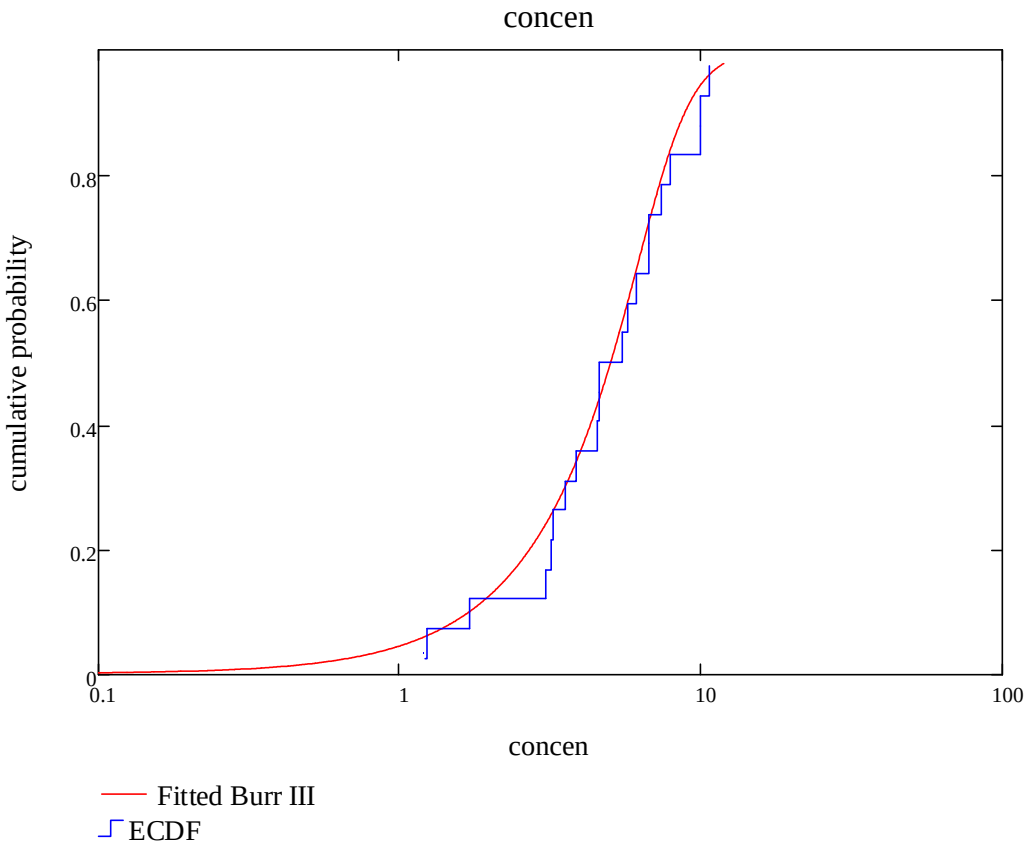


Figure E6. Comparison of fitted and empirical cdfs for metolachlor in freshwater example.

For untransformed data, HC_θ values are readily obtained using Equation E.16:

$$HC_\theta = \hat{b}\left[\left(\frac{1}{\theta}\right)^{\frac{1}{\hat{k}}} - 1\right]^{\frac{-1}{\hat{c}}} \qquad \text{(E.16)}$$

However, in Example 2 we used the transformation $Y = a - \ln(X)$ where the constant *a* was chosen to be 10. Let HC_p^X denote the HC_p for the untransformed data and HC_p^Y denote the HC_p for the transformed data. Then it is straightforward to show that

$$HC_p^X = \exp(a - HC_{1-p}^Y) \qquad \text{(E.17)}$$

Using $\left\{\hat{b} = 7.909,\ \hat{c} = 5.768,\ \hat{k} = 0.262\right\}$ and $\theta = 0.95$ in Equation E.16 we obtain $HC_{0.95}^Y = 10.314$ and upon substitution in Equation E.17 we obtain $HC_{0.05}^X = \exp(10 - 10.314) = 0.731$. Similarly, we obtain $HC_{0.01}^X = \exp(10 - 13.873) = 0.021$.

ANZG (2020) reported an $HC_{0.05}^X$ of 0.46 and an $HC_{0.01}^X$ of 0.0084.

Model-averaged estimates using a *log-normal* and a *Weibull* distribution (Figure E7) gave an $HC_{0.05}^{X}$ of 0.762 and an $HC_{0.01}^{X}$ of 0.0838.

Clearly, there's considerable variability in these estimates although the ANZG estimates would appear to be overly conservative.

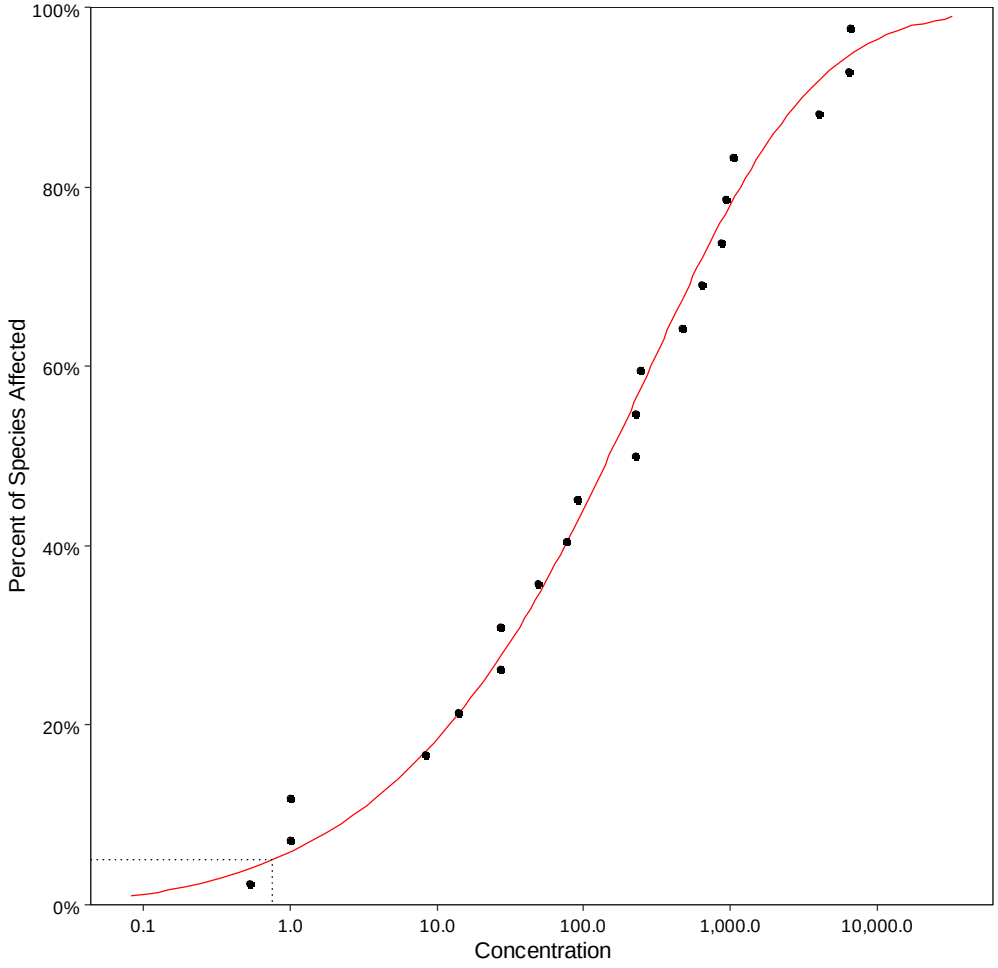


Figure E7. Model-averaged fit of lognormal and Weibull distributions for metolachlor in freshwater example.

Discussion and Conclusions

The use of L-moment estimation for fitting SSDs should be explored further. Results presented herein, while preliminary, are nevertheless encouraging and suggest:

- Distributions can be fitted with ease – we have shown how to obtain very good parameter estimates using an entirely graphical approach;
- L-moments always exist and provided the SSD has a finite first moment (which will always be the case for the distribution used for SSD modelling), moment estimators can be uniquely determined.
- Method appears robust to outliers and has good small-sample properties.
- Computations are very fast.

- The calculations are readily coded in R and make use of package `lmom`

REFERENCES

Hosking, JRM. 1990. L-moments: analysis and estimation of distributions using linear combinations of order statistics. *J. Roy. Statist. Soc.* B, 52:105-124.

Kjeldsen, TR, Smithers, JC, Schulze, RE. 2002. Regional flood frequency analysis in the KwaZulu-Natal province, South Africa, using the index-flood method. *J. Hydrol.* 255:194–211.

Kroll, CN., Vogel, RM. 2002. Probability distribution of low streamflow series in the United States. *J. Hydrol. Eng*. 7:137–146.

Lim,YH, Lye, LM. 2003. Regional flood estimation for ungauged basins in Sarawak, Malaysia. *Hydrol. Sci. J.* 48:79–94.

ANZG 2020. Toxicant default guideline values for aquatic ecosystem protection: Metolachlor in freshwater. Australian and New Zealand Guidelines for Fresh and Marine Water Quality. Australian and New Zealand Governments and Australian state and territory governments, Canberra, ACT, Australia.

Appendix F: Estimation and inference for the burr iii distribution

David R. Fox

In this Appendix we derive theoretical results for the *Burr III* distribution which will admit a closed-form expression for the covariance matrix of parameter estimates and in turn, allow the computation of a standard error for an estimated HC_x.

PRELIMINARIES

The *pdf* for the *Burr III* distribution is:

$$f_X(x;b,c,k) = \frac{bck\left(\frac{b}{x}\right)^{c-1}}{x^2\left[1+\left(\frac{b}{x}\right)^c\right]^{k+1}} \; ; x,b,c,k > 0$$

By letting $Y = \left(\frac{b}{X}\right)^c$ we obtain:

$$g_Y\left(y;k\right) = k\left[1+y\right]^{-k-1} \;\; ; \;\; y > 0 \tag{F.1}$$

An alternative parameterisation of the *Burr III* distribution is given by Equation F.2.

$$h_X\left(x;\alpha,\tau,\theta\right) = \frac{\alpha\tau\left(\frac{x}{\theta}\right)^{\tau}}{x\left[1+\left(\frac{x}{\theta}\right)\right]^{\alpha+1}} \;\; ; \;\; x,\alpha,\tau,\theta > 0 \tag{F.2}$$

whose (raw) moments are:

$$\mathrm{E}\left[X^r\right] = \frac{\theta^r\,\Gamma\left(\frac{r}{\tau}+1\right)\Gamma\left(\alpha-\frac{r}{\tau}\right)}{\Gamma\left(\alpha\right)} \;\; ; \;\; -\tau < r < \tau$$

Now, since $g_Y\left(y\right) = h\left(y;k,1,1\right)$ we have immediately that:

$$\mathrm{E}\left[Y^r\right] = \frac{\Gamma\left(r+1\right)\Gamma\left(k-r\right)}{\Gamma\left(k\right)} \;\; ; \;\; -1 < r < k \tag{F.3}$$

THE INFORMATION MATRIX FOR THE BURR III DISTRIBUTION

It is well known that the MLE $\hat{\Theta}$ is asymptotically normally distributed with mean Θ and variance given as the inverse of the *information matrix*, $\mathrm{I} = -\mathrm{E}\left[\frac{\partial^2 l(\Theta)}{\partial \Theta^2}\right]$ where, for the *Burr III* distribution, $\Theta = \{b, c, k\}$ and

$l(\Theta) = \log\left[f_X(x; b, c, k)\right]$. With $k_{ij} = \mathrm{E}\left[\frac{\partial^2 l(\Theta)}{\partial \theta_i \partial \theta_j}\right]$ we next develop theoretical expressions for the 6 unique k_{ij} elements. Some of these can be quite complex and involve powers of the quantity $\frac{b}{x}$.

For example, $k_{11} = \mathrm{E}\left[\left[-\frac{c}{b^2}\left\{\frac{\left[\left(\frac{b}{x}\right)^c - k\left(\frac{b}{x}\right)^{2c} + c\left(\frac{b}{x}\right)^c - k\left(\frac{b}{x}\right)^c + ck\left(\frac{b}{x}\right)^c + 1\right]}{\left[1+\left(\frac{b}{x}\right)^c\right]^2}\right\}\right]\right]$.

It will be convenient to use the transformation $Y = \left(\frac{b}{x}\right)^c$ and so k_{11} for example becomes:

$$k_{11} = -\frac{c}{b^2}\mathrm{E}\left\{\frac{Y}{(1+Y)^2} - k\frac{Y^2}{(1+Y)^2} + c\frac{Y}{(1+Y)^2} - k\frac{Y}{(1+Y)^2} + ck\frac{Y}{(1+Y)^2} + \frac{1}{(1+Y)^2}\right\}$$
$$= -\frac{c}{b^2}\left\{(ck + c - k + 1)\mathrm{E}_Y\left[\frac{Y}{(1+Y)^2}\right] - k\mathrm{E}_Y\left[\frac{Y^2}{(1+Y)^2}\right] + \mathrm{E}_Y\left[\frac{1}{(1+Y)^2}\right]\right\} \quad \text{(F.4)}$$

It turns out that all 6 unique k_{ij} elements involve linear combinations of the following quantities:

$$I_1 = \mathrm{E}_Y\left[\frac{Y}{(1+Y)^2}\right]; \quad I_2 = \mathrm{E}_Y\left[\frac{Y^2}{(1+Y)^2}\right]; \quad I_3 = \mathrm{E}_Y\left[\frac{1}{(1+Y)^2}\right]$$
$$I_4 = \mathrm{E}_Y\left[\frac{Y\ln^2(Y)}{(1+Y)^2}\right]; \quad I_5 = \mathrm{E}_Y\left[\frac{1}{(1+Y)}\right]; \quad I_6 = \mathrm{E}_Y\left[\frac{Y\ln(Y)}{(1+Y)^2}\right]$$
$$I_7 = \mathrm{E}_Y\left[\frac{Y\ln(Y)}{(1+Y)}\right].$$

Evaluation of explicit expressions for $I_1, \ldots I_7$ tends to be repetitive and involves integrals of varying complexity. An illustrative example of a simpler case is given for I_1.

I_1 : $\mathrm{E}_Y\left[\dfrac{Y}{(1+Y)^2}\right]$

Now,
$$\mathrm{E}_Y\left[\frac{Y}{(1+Y)^2}\right]=\int_0^\infty \frac{y}{(1+y)^2}\cdot g_Y(y)\,dy$$
$$=\int_0^\infty \frac{y}{(1+y)^2}\cdot\frac{k}{(1+y)^{k+1}}\,dy$$
$$=\int_0^\infty y\cdot\frac{k}{(1+y)^{k+3}}\,dy$$

Letting $k'=k+2$ we have $\dfrac{I_1}{k}\cdot k'=\int_0^\infty y\cdot\dfrac{k'}{(1+y)^{k'+1}}\,dy$. We recognise this integral as the expected value of *Y* where *Y* has the *pdf* $g_Y(y;k')$ with $g_Y(\cdot)$ given by Equation F.1.

Now from Equation F.3: $\mathrm{E}\left[Y^r\right]=\dfrac{\Gamma(r+1)\Gamma(k-r)}{\Gamma(k)}$ and therefore

$\dfrac{I_1}{k}\cdot k'=\mathrm{E}\left[Y\,|k=k'\right]=\dfrac{\Gamma(k'-1)}{\Gamma(k')}$ and hence: $I_1=\dfrac{k}{k'}\dfrac{\Gamma(k'-1)}{\Gamma(k')}=\dfrac{k\,\Gamma(k+1)}{\Gamma(k+3)}=\dfrac{k}{(k+1)(k+2)}$.

Thus:

$$I_1=\mathrm{E}_Y\left[\frac{Y}{(1+Y)^2}\right]=\frac{k}{(k+1)(k+2)}$$

We will not labour through all the tedious calculations associated with various expectations that will be needed to evaluate the information matrix. Instead, we provide results in Table F1 for various combinations of q,r,s,t in the more general expression: $\mathrm{E}\left\{\dfrac{Y^r(\ln Y)^s\left[\ln(1+Y)\right]^q}{(1+Y)^t}\right\}$.

NB: not all the results in Table F1 are used to construct the information matrix. Entries for which q=1 are related to the expectation of third derivatives of the log-likelihood function and are useful for computing the Hessian matrix when using a Newton-Raphson method to solve the likelihood Equations.

Table F1. Expectations of various functions of Y. Notation: $\Psi(\cdot)$ *is the digamma function;* $\Psi^{(m)}(\cdot)$ *is the polygama function of order m;* γ *is the Euler–Mascheroni constant; H(k) is the Harmonic number (*$=\sum_{n=1}^{k}\frac{1}{n}$ *for integer k>0;* $=\Psi(k+1)+\gamma$ *for* $k\in\mathbb{R}^{+}>0$ *); and* $\zeta(\cdot)$ *is the Riemann zeta function.*

q	***r***	***s***	***t***	$\mathrm{E}\left[\frac{\mathrm{Y}^{\mathrm{r}}(\ln Y)^{s}\left[\ln(1+Y)\right]^{q}}{(1+\mathrm{Y})^{\mathrm{t}}}\right]$
0	0	0	1	$\frac{k}{k+1}$
0	0	0	2	$\frac{k}{k+2}$
0	0	0	3	$\frac{k}{k+3}$
0	0	1	0	$-\left[\gamma+\Psi(k)\right]$
0	0	1	1	$\frac{-k}{k+1}\left[\gamma+\Psi(k+1)\right]$
0	0	1	3	$\frac{-k}{k+3}\left[\gamma+\Psi(k+3)\right]$
0	0	2	0	$\Psi^{2}(k)+2\gamma\Psi(k)+\gamma^{2}+\Psi^{(1)}(k)+\frac{\pi^{2}}{6}$
0	0	3	0	$\frac{1}{2}\left\{-2\gamma^{3}-\gamma\pi^{2}-\Psi(k)\left[\left[6\gamma^{2}+\pi^{2}+2\Psi(k)\left[3\gamma+\Psi(k)\right]\right]\right]-6\left[\gamma+\Psi(k)\right]\Psi^{(1)}(k)-2\Psi^{(2)}(k)-4\zeta(3)\right\}$
0	1	0	1	$\frac{1}{k+1}$
0	1	0	2	$\frac{k}{(k+1)(k+2)}$
0	1	0	3	$\frac{k\Gamma(k+2)}{\Gamma(k+4)}$
0	1	1	1	$\frac{1}{(k+1)}\left[1-\Psi(k)-\gamma\right]$

q	*r*	*s*	*t*	$E\left[\left[\frac{Y^{r}(\ln Y)^{s}\left[\ln(1+Y)\right]^{q}}{(1+Y)^{t}}\right]\right]$
0	1	1	2	$\frac{k}{(k+1)(k+2)}\left[1-\Psi(k+1)-\gamma\right]$
0	1	1	3	$\frac{-k\Gamma(k+2)\left[\gamma+\Psi(k+2)-1\right]}{\Gamma(k+4)}$
0	1	2	2	$\frac{k}{(k+1)(k+2)}\left\{2(\gamma-1)\Psi(k+1)+\gamma(\gamma-2)+\Psi^{(1)}(k+1)+\Psi^{2}(k+1)+\frac{\pi^{2}}{6}\right\}$
0	1	2	3	$\frac{k\Gamma(k+2)}{\Gamma(k+4)}\left\{\Psi^{(1)}(k+2)+\left[\gamma+\Psi(k+2)-1\right]^{2}+\frac{\pi^{2}}{6}-1\right\}$
0	1	3	3	$\frac{-k}{2(k+2)(k+3)}\left\{\begin{array}{l}-\pi^{2}+\gamma\left[\left[2(\gamma-3)\gamma+\pi^{2}\right]\right]+\left[\left[2(\gamma-3)\gamma+\pi^{2}+2H(k+1)\left[\gamma-3+H(k+1)\right]\right]\right]\Psi(k+2)\\-6\Psi^{(1)}(k+2)+6H(k+1)\Psi^{(1)}(k+2)+2\Psi^{(2)}(k+2)+4\zeta(3)\end{array}\right\}$
0	2	0	2	$\frac{2k\Gamma(k)}{\Gamma(k+3)}$
0	2	0	3	$\frac{2k\Gamma(k+1)}{\Gamma(k+4)}$
0	2	1	2	$\frac{2}{(k+2)(k+1)}\left[\frac{3}{2}-\Psi(k)-\gamma\right]$
0	2	1	3	$\frac{-2k\Gamma(k+1)}{\Gamma(k+4)}\left[\gamma+\Psi(k+1)-\frac{3}{2}\right]$
0	2	2	2	$\frac{2}{(k+2)(k+1)}\left\{\gamma^{2}-3\Psi(k)-3\gamma+\Psi^{2}(k)+2\gamma\Psi(k)+\Psi^{(1)}(k)+\frac{\pi^{2}}{6}+1\right\}$
0	2	2	3	$\frac{2k\Gamma(k+1)}{\Gamma(k+4)}\left\{\Psi^{(1)}(k+1)+\left[\gamma+\Psi(k+1)-\frac{3}{2}\right]^{2}+\frac{\pi^{2}}{6}-\frac{5}{4}\right\}$

q	r	s	t	$\mathrm{E}\left[\frac{\mathrm{Y}^{\mathrm{r}}(\ln Y)^{s}\left[\ln(1+Y)\right]^{q}}{(1+\mathrm{Y})^{\mathrm{t}}}\right]$
0	2	3	3	$\frac{k}{2(k+1)(k+2)(k+3)}\left\{3\pi^{2}+18\Psi^{(1)}(k+1)-2H(k)\left[6+\pi^{2}+H(k)\left[2H(k)-9\right]+6\Psi^{(1)}(k+1)\right]-4\Psi^{(2)}(k+1)-8\zeta(3)\right\}$
0	3	0	2	$\frac{6k\Gamma(k-1)}{\Gamma(k+3)}$
0	3	0	3	$\frac{6k\Gamma(k)}{\Gamma(k+4)}$
0	3	1	3	$\frac{6}{(k+3)(k+2)(k+1)}\left[\frac{11}{6}-\Psi(k)-\gamma\right]$
1	0	0	2	$\frac{k}{(k+2)^{2}}$
1	1	0	2	$\frac{k(2k+3)}{\left(k^{2}+3k+2\right)^{2}}$
1	1	1	2	$\frac{k}{\left(k^{2}+3k+2\right)^{2}}\left\{(1-\gamma)(3+2k)-\frac{(k+1)(k+2)}{k^{2}}-(2k+3)\Psi(k+1)+(k+1)(k+2)\Psi^{(1)}(k)\right\}$
1	2	0	2	$\frac{2\left(3k^{2}+6k+2\right)}{k\left(k^{2}+3k+2\right)^{2}}$

The elements of the information matrix can be expressed in terms of $\{I_1,\ldots,I_7\}$ as follows:

$$k_{11} = -\frac{c}{b^2}\left[(ck + c - k + 1)I_1 - kI_2 + I_3\right]$$

$$k_{22} = -\frac{1}{c^2}\left[2I_1 + I_2 + I_3 + (k+1)I_4\right]$$

$$k_{33} = -\frac{1}{k^2}$$

$$k_{12} = \frac{k+1}{b}(I_5 - I_6) - \frac{k}{b}$$

$$k_{13} = \frac{c}{b}(I_5 - 1)$$

$$k_{23} = -\frac{1}{c}I_7$$

Expressions for $\{I_1,\ldots,I_7\}$ can be found by making the appropriate substitution of *q,r,s,*and *t* in Table F1. Doing so, we finally obtain the elements of the information matrix as:

$$\mathrm{I} = \begin{pmatrix} \frac{ck^2}{b^2(k+2)} & \frac{k}{b(k+2)}\left[1-\Psi(k+1)-\gamma\right] & \frac{-c}{b(k+1)} \\ \frac{k}{b(k+2)}\left[1-\Psi(k+1)-\gamma\right] & \frac{1}{c^2}\left\{1+\frac{k}{k+2}\left[2(\gamma-1)\Psi(k+1)+\gamma(\gamma-2)+\Psi^{(1)}(k+1)+\Psi^2(k+1)+\frac{\pi^2}{6}\right]\right\} & \frac{1-\Psi(k)-\gamma}{c(k+1)} \\ \frac{c}{b(k+1)} & \frac{1-\Psi(k)-\gamma}{c(k+1)} & \frac{1}{k^2} \end{pmatrix}$$

(F.5)

ESTIMATION AND INFERENCE FOR THE HCx

The results of the previous Section can be used to obtain an estimate of the standard error of an estimated HC_x value and hence a $(1-\alpha)100\%$ confidence interval.

ESTIMATING THE HCx

The *pdf* and *cdf* for the *Burr III* distribution are given by Equations F.6 and F.7 respectively.

$$f_X\left(x;b,c,k\right)=\frac{bck\left(\frac{b}{x}\right)^{c-1}}{x^2\left[1+\left(\frac{b}{x}\right)^{c}\right]^{k+1}} \tag{F.6}$$

$$F_X\left(x;b,c,k\right)=\frac{1}{\left[1+\left(\frac{b}{x}\right)^{c}\right]^{k}} \tag{F.7}$$

It is straightforward to obtain an expression for the p^{th} quantile of the *Burr III* by a simple re-arrangement of the terms in Equation F.7:

$$q\left(p,b,c,k\right)=\frac{b}{\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{\frac{1}{c}}} \quad = \quad HC_p \tag{F.8}$$

STANDARD ERROR OF THE HCx

We use the delta method to obtain an estimate of the standard error of the HCx as follows. Let $\mathbb{Q}\left(\Theta\right)=q\left(p;\Theta\right)$ for some *p* and $\Theta=\left\{b,c,k\right\}$ and $q\left(\cdot\right)$ is the function given by Equation F.8. Now, for a sample of size *n*, the information matrix is $\mathrm{I}\left(\Theta\right)=-n\,\mathrm{E}\left[\frac{\partial^2\Theta}{\partial\theta_i\partial\theta_j}\right]$. The MLE, $\hat{\Theta}$ is asymptotically normally distributed with:

$$Cov\left[\hat{\Theta}\right]=\mathrm{I}\left(\Theta\right)^{-1}=\Omega_{\Theta}$$

from which we have:

$$\widehat{Cov\left[\hat{\Theta}\right]}=\mathrm{I}\left(\hat{\Theta}\right)^{-1}=\Omega_{\hat{\Theta}} \tag{F.9}$$

The elements of $\mathrm{I}\left(\hat{\Theta}\right)$ are obtained using Equation F.5 with $\hat{\Theta}=\left\{\hat{b},\hat{c},\hat{k}\right\}$.

Using the delta-method, the approximate variance of $\mathbb{Q}\left(\hat{\Theta}\right)$ is given by Equation F.10.

$$Var\left[\mathbb{Q}\left(\hat{\Theta}\right)\right]\simeq\nabla\left(\hat{\Theta}\right)^{T}\Omega_{\hat{\Theta}}\nabla\left(\hat{\Theta}\right)^{T} \tag{F.10}$$

where $\nabla_{\Theta}\left(\cdot\right)$ is the *gradient* vector having elements given by Equation F.11.

$$\nabla_{\Theta}\left(\Theta\right)^{T}=\nabla_{b,c,k}\left(b,c,k\right)^{T}=\left\{\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{\frac{-1}{c}} \;;\; \frac{b}{c^2}\ln\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{-\frac{1}{c}} \;;\; \frac{b}{ck^2}\ln\left(\frac{1}{p}\right)\left(\frac{1}{p}\right)^{\frac{1}{k}}\left[\left(\frac{1}{p}\right)^{\frac{1}{k}}-1\right]^{-\frac{1}{c}-1}\right\} \tag{F.11}$$

Thus, the estimated standard error of the HC_p is:

$$SE\left[\mathbb{Q}\left(\hat{\Theta}\right)\right] = \sqrt{Var\left[\mathbb{Q}\left(\hat{\Theta}\right)\right]} \tag{F.12}$$

EXAMPLES

1. *Cadmium dataset*

MODEL: Burr type III
Parameter estimates:
log(b) -6.73785268926432
log(c) -0.760252173401116
log(k) 3.40142003849742

$n = 36$

$\Rightarrow \hat{b} = 0.00119 \; ; \; \hat{c} = 0.468 \; ; \; \hat{k} = 30.007$

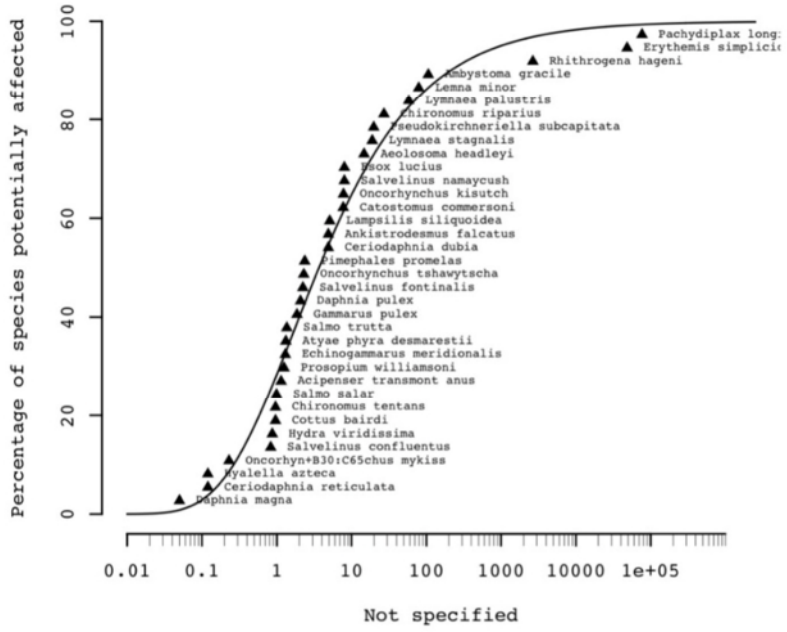
Burrlioz 2.0 report

Toxicant:
Input file: \\Hp-spectre\s\Environmetrics Australia\SSD R&D - Documents\Australia-Can
Time read: Mon May 03 10:44:30 2021
Units: Not specified
Model: Burr type III

Protection level information

Protect. level	Guideline Value	lower 95% CI	upper 95% CI
99%	0.055	0.017	0.22
95%	0.15	0.066	0.47
90%	0.26	0.13	0.76
80%	0.58	0.32	1.4

notes:



Using Equation F.9 we have:

$$I\left(\hat{\Theta}\right)^{-1} = \begin{pmatrix} 5.272\times10^{-4} & 1.814\times10^{-3} & -5.706 \\ 1.814\times10^{-3} & 9.557\times10^{-3} & -19.018 \\ -5.706 & -19.018 & 6.189\times10^{4} \end{pmatrix}$$

Aside: The estimated parameter correlation matrix, $\mathbb{R}$ has the following elements:

$$\mathbb{R} = \begin{pmatrix} 1.0 & 0.808 & -0.999 \\ 0.808 & 1.0 & -0.782 \\ -0.999 & -0.782 & 1.0 \end{pmatrix}$$

The highlighted entry in red is the correlation between $\hat{b}$ and $\hat{k}$. This shows that these two parameter estimates are almost perfectly (negatively) correlated. This is the cause of the instability issues with the MLE for the *Burr III* distribution.

With *p=0.05* in Equation F.11 we obtain:

$$\nabla_{\Theta}\left(\hat{\Theta}\right)^{T} = [124.049, -1.516, 0.011]$$

and substituting this into Equation F.10 we obtain:

$$Var\left[\hat{\Theta}\right] \simeq 3.955 \times 10^{-3}$$
$$\Rightarrow SE\left[\hat{\Theta}\right] \simeq 0.063$$

Using Equation F.8 we have $\widehat{HC}_5 = 0.147$ and thus an approximate 95% confidence interval is:

$$\widehat{HC}_5 \pm t_{n-3,\left(1-\frac{\alpha}{2}\right)} \cdot SE\left[\widehat{HC}_5\right] = \{0.019; 0.275\}$$

Relevant output from `Burrlioz` is shown below.

```
Protection level information
Protect. level     Guideline Value     lower 95% CI     upper 95% CI
99%                0.055               0.017            0.22
95%                0.15                0.066            0.47
90%                0.26                0.13             0.76
80%                0.58                0.32             1.4
```

From `Burrlioz` we obtain: $\widehat{HC}_5 = 0.15$ and a boostrapped 95% confidence interval $\{0.066; 0.47\}$.

2. *Chloride dataset*

MODEL: Burr type III
Parameter estimates:
log(b) 7.2527402821831
log(c) 0.56853202527791
log(k) -0.534793562523389

$n = 28$

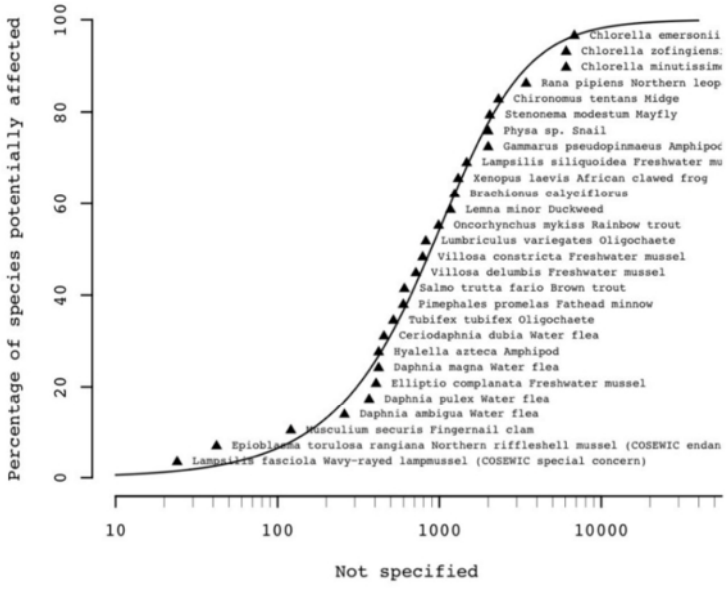
$\Rightarrow \hat{b} = 1,411.969 ; \hat{c} = 1.766 ; \hat{k} = 0.58579$

Burrlioz 2.0 report

Toxicant:
Input file: \\Hp-spectre\s\Environmetrics Australia\SSD R&D - Documents\Australia-Ca
Time read: Mon May 03 10:48:34 2021
Units: Not specified
Model: Burr type III

Protection level information

Protect. level	Guideline Value	lower 95% CI	upper 95% CI
99%	16	1.5	217
95%	78	20	314
90%	154	59	393
80%	309	162	557



A comparison of confidence intervals obtained from `Burrlioz` and the methods of this report are shown in Table F1. It is evident from this comparison that there are large differences in the lower limits and the upper limits at the 99% and 95% protection levels. This raises an important, and as of this time, unresolved issue of computation of confidence intervals using analytical expressions of the form presented in this report.

Table F1. Comparison of 95% confidence intervals for various HCx *values for chloride data*

Protection Level	GV	SE (Eqn.F.12)	Lower 95% CI (Burrlioz)	Lower 95% CI (this report)	Upper 95% CI (Burrlioz)	Upper 95% CI (this report)
99%	16	16.884	1.5	0	217	55.9
95%	78	45.073	20	0	314	183.5
90%	154	62.976	59	7.1	393	301.3
80%	309	87.412	162	105.2	557	513.5

Aside: How should* HC*x* *confidence intervals be computed?

This question only arises in the context of obtaining confidence intervals by direct computation rather than bootstrapping. The central issue is managing negative lower CI limits. This cannot occur from bootstrapping since, in this case, the CI approximation is via an analysis of simulated HCx values and as we only use distributions defined on the positive real line for SSDs, all the simulated HCx values will be non-negative. However, in the analytical approach, we obtain a confidence interval by adding and subtracting a multiple of the estimated standard error to the estimated HCx value. Clearly, the lower limit will be negative if this multiple is larger than the estimated HCx. One, not unreasonable approach, is to simply relace the negative lower limit with zero as has been done in Table F1 for the 99% and 95% HC values. Difficulties with this approach can and will arise if, as is sometimes the case, the lower CI is to be used as a guideline value when an additional level of protection is required. In such cases, a zero concentration is both operationally and conceptually problematic. From a practical point of view the whole point of the SSD-fitting exercise is to identify a concentration that is deemed to be sufficiently protective to the ecosystem while still allowing the polluting process to proceed – albeit with increased safeguards. Conceptually, at zero concentration, the level of protection is 100% and so zero cannot be, say, an HC5 unless we redefine the level of protection to be a *minimum*.

One way to resolve this conundrum is to compute the CI for *log*-transformed data and then *back-transform* the CI limits by exponentiating to obtain a CI commensurate with the original scale. While this guarantees that the back-transformed limits will *always* be non-negative, the process of exponentiation can result in inflated limits.

Further discussion and contemplation by ecotoxicological researchers is required to come to an agreed position on the preferred approach.

3. Metolachlor dataset

$n = 21$

MODEL: Burr type III
Parameter estimates:
log(b) 4.62751106363125
log(c) 0.350624409945066
log(k) -0.0244160444627527

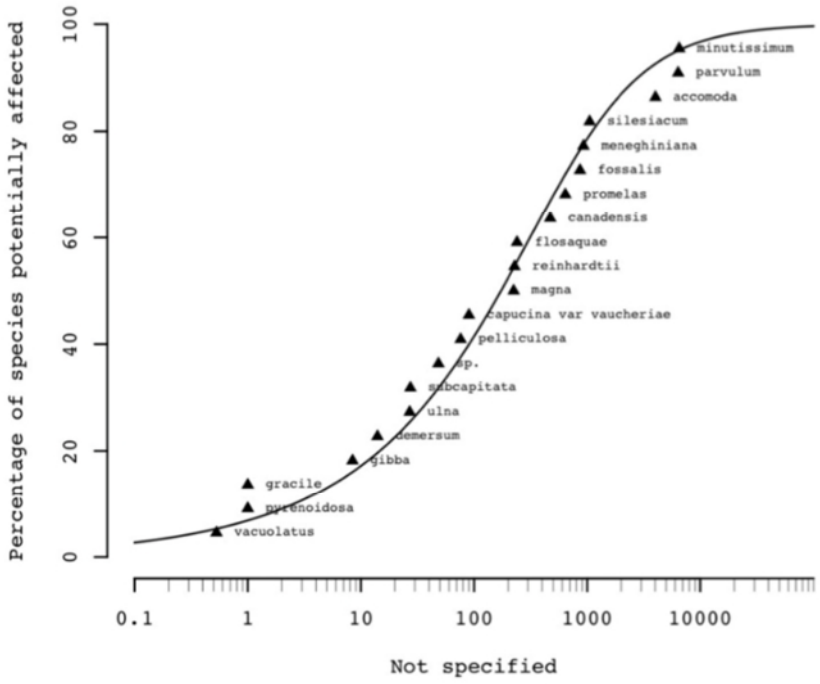
$\Rightarrow \hat{b} = 776.232;\ \hat{c} = 0.970$
$\hat{k} = 0.416$

Burrlioz 2.0 report

Toxicant:
Input file: \\Hp-spectre\s\Environmetrics Australia\SSD R&D - Documents\Australia-Cana
Time read: Mon May 03 11:33:07 2021
Units: Not specified
Model: Burr type III

Protection level information

Protect. level	Guideline Value	lower 95% CI	upper 95% CI
99%	0.0085	0	3.3
95%	0.46	0	11
90%	2.6	0.49	22
80%	15	2.5	61



A comparison of confidence intervals obtained from Burrlioz and the methods of this report are shown in Table F2. Once again, there are large differences between methods – particularly for the upper limits with Burrlioz interval widths up to 50 times wider than those obtained by the analytical methods of this report.

Table F2. Comparison of 95% confidence intervals for various HC values for chloride data.

Protection Level	GV	SE (Eqn.F.12)	Lower 95% CI (Burrlioz)	Lower 95% CI (this report)	Upper 95% CI (Burrlioz)	Upper 95% CI (this report)
99%	0.0085	0.021	0	0	3.3	0.066
95%	0.46	0.665	0	0	11	2.289
90%	2.6	2.664	0.49	0	22	9.908
80%	15	10.093	2.5	0	61	42.414

4. Nickel in freshwater dataset

MODEL: Burr type III
Parameter estimates:
log(b) 3.16013968793554
log(c) 0.334074349292958
log(k) -0.0551718548249647

$\Rightarrow \hat{b} = 23.574;\ \hat{c} = 1.397$
$\hat{k} = 0.946$

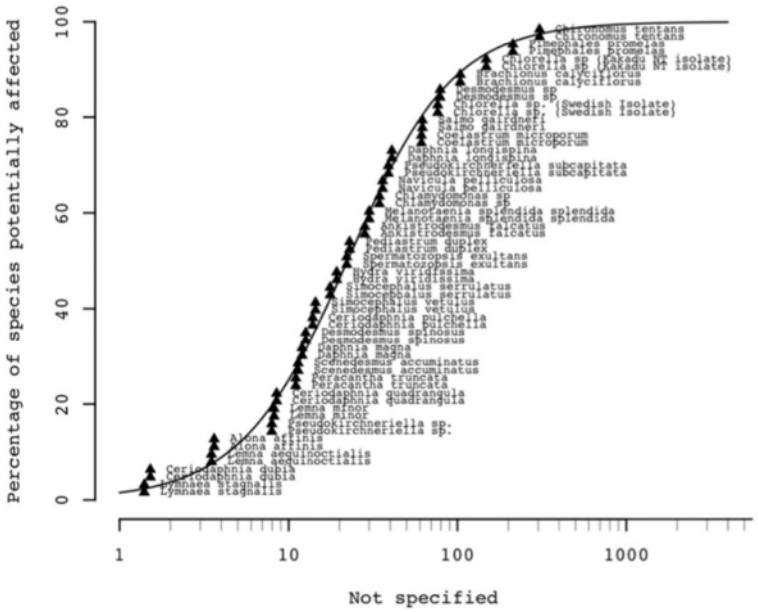
Burrlioz 2.0 report

Toxicant:
Input file: \\Hp-spectre\s\Environmetrics Australia\SSD R&D - Documents\Austr
Time read: Mon May 03 11:30:23 2021
Units: Not specified
Model: Burr type III

Protection level information

Protect. level	Guideline Value	lower 95% CI	upper 95% CI
99%	0.73	0.23	2.3
95%	2.5	1.3	4.8
90%	4.4	2.8	7.1
80%	8.1	5.5	12

notes:



A comparison of confidence intervals obtained from Burrlioz and the methods of this report are shown in Table F3. Unlike the previous examples, we see a much closer level of agreement between the two sets of confidence intervals.

Table F3. Comparison of 95% confidence intervals for various HC values for freshwater nickel data.

Protection Level	GV	SE (Eqn.F.12)	Lower 95% CI (Burrlioz)	Lower 95% CI (this report)	Upper 95% CI (Burrlioz)	Upper 95% CI (this report)
99%	0.73	0.638	0.23	0	2.3	1.701
95%	2.5	1.19	1.3	0.706	4.8	4.334
90%	4.4	1.503	2.8	2.117	7.1	6.702
80%	8.1	2.071	5.5	4.9	12	11.21

Finally, we provide code below to implement the methods of this Appendix:

```
require(RBurrlioz)


##  Function to compute HCp and SE

hc.burr<-function(p,model){
  stopifnot(attributes(model[[2]])$name[1]=="Burr type III")
  param<-exp(model$result$par);b=param[1];c=param[2];k=param[3]  # parameter
estimates
  g<- -digamma(1)  # Euler's constant gamma
  n<-length(model$data)
#  kappa matrix
  kappa<-matrix(NA,3,3)
  kappa[1,1]<- -c^2*k/(b^2*(k+2))
  kappa[2,2]<- (-1/c^2)*(1+(k/
(k+2))*(2*(g-1)*psigamma(k+1)+g*(g-2)+psigamma(k+1,1)+psigamma(k+1)^2+pi^2/6))
  kappa[3,3]<- -1/k^2
  kappa[1,2]<- -k/(b*(k+2))*(1-psigamma(k+1)-g); kappa[2,1]<-kappa[1,2]
  kappa[1,3]<- -c/(b*(k+1)); kappa[3,1]=kappa[1,3]
  kappa[2,3]<- (psigamma(k)+g-1)/(c*(k+1)); kappa[3,2]<-kappa[2,3]
  I<- -n*kappa
# gradient vector
  grad<-matrix(NA,3,1)
  tmp<-((1/p)^(1/k)-1)
  grad[1]<-tmp^(-1/c)
  grad[2]<-(b/c^2)*log(tmp)*tmp^(-1/c)
  grad[3]<-b/(c*k^2)*(1/p)^(1/k)*log(1/p)*tmp^-(1/c+1)
#compute variance of HC
  v<-t(grad)%*%solve(I)%*%grad
  hc<-estimate.hc(model,p)
  return(c(hc,sqrt(v)))
}

##  Example  90% CI for HC2  for CSIRO Cadmium data   ###

dat<-read.csv("cadmium.csv")[,1]  # read first column of data (concentrations)
n<-n<-length(dat)  # sample size

##  Obtain parameter estimates from RBurrlioz

fit<-fit(dat)  #  fit Burr III using RBurrlioz
res<-hc.burr(0.02,fit)  # HC2 estimate + standard error
cat("\n","HC2=",res[1],"   SE=",res[2],"\n",
        "90% Confidence limits: {",res[1]-qt(0.95,n-3)*res[2],";
",res[1]+qt(0.95,n-3)*res[2],"}",sep="")

HC2=0.08035998   SE=0.04252525
90% Confidence limits: {0.008391928; 0.152328}
```

Appendix G: A note on re-scaling SSDs

David R. Fox

Let *X* have *pdf* $f_X(x;\alpha,\beta)$ where α is the shape parameter and β is the scale parameter and $x;\alpha;\beta>0$.

Further, let $Y=X-\delta$ where δ is some positive constant. Then $\int_0^{\psi_p} f_X(x)dx=p \Rightarrow \psi_p$ is the p^{th} percentile for *X*.

But $P[X\le\psi_p]=p \Leftrightarrow P[Y+\delta\le\psi_p]=p \Rightarrow P[Y\le\psi_p-\delta]=p$ and so $\psi_p-\delta$ is the p^{th} percentile for *Y*.

Distributions with a scale parameter

Now let $Y=\frac{X}{\delta}$. Then, $G_Y(y)=P[Y\le y]=P\left[\frac{X}{\delta}\le y\right]=P[X\le\delta y]=F_X(\delta y)$.

LOG-LOGISTIC DISTRIBUTION

$F_X(x)=\frac{1}{1+\left(\frac{x}{\alpha}\right)^{-\beta}}$ where the scale parameter is α.

Therefore $G_Y(y)=\frac{1}{1+\left(\frac{\delta y}{\alpha}\right)^{-\beta}}=\frac{1}{1+\left(\frac{y}{k}\right)^{-\beta}}$ where $k=\frac{\alpha}{\delta}$.

We see that the distribution for *Y* is of the same form as the distribution for *X*, but with scale parameter $\frac{\alpha}{\delta}$.

Thus, if $\{\hat{\alpha},\hat{\beta}\}$ are parameter estimates from *X*-data, then the re-scaled *Y* data will have parameter estimates $\left\{\frac{\hat{\alpha}}{\delta},\hat{\beta}\right\}$.

LOGNORMAL DISTRIBUTION

If $X\sim LN(\mu,\sigma)$ then $\ln(X)\sim N(\mu,\sigma^2)$. Now if $Y=\frac{X}{\delta}$, then $\ln(Y)\sim N(\mu-\ln\delta,\sigma^2)$ and hence *Y* has a *log-normal* distribution with parameters $(\mu-\ln\delta,\sigma)$.

INVERSE PARETO DISTRIBUTION

$f_X(x) = \alpha\,\beta^{\alpha}\,x^{\alpha-1} \quad 0 \le x \le \frac{1}{\beta}$ and $F_X(x) = (\beta x)^{\alpha}$.

Therefore, $G_Y(y) = \left(\beta\frac{x}{\delta}\right)^{\alpha} = \left[\left(\frac{\beta}{\delta}\right)x\right]^{\alpha} = (kx)^{\alpha}$ where $k = \frac{\beta}{\delta}$ and so the distribution of *Y* is of the same form as *X*.

INVERSE WEIBULL DISTRIBUTION

$$F_X(x) = \exp\left[-\frac{x^{-\beta}}{\alpha}\right] = \exp\left[-\frac{1}{\alpha}\cdot\left(\frac{1}{x}\right)^{\beta}\right].$$

Now $\frac{1}{X} = \frac{1}{\delta Y}$ and therefore $G_Y(y) = \exp\left[-\frac{1}{\alpha}\cdot\left(\frac{1}{\delta y}\right)^{\beta}\right] = \exp\left[-\frac{1}{\alpha\delta^{\beta}}\cdot\left(\frac{1}{y}\right)^{\beta}\right]$.

Thus the distribution for Y is also *inverse Weibull* with scale parameter $\alpha\,\delta^{\beta}$ and shape parameter β .

Appendix H: Reconciliation of HCx estimates for the inverse pareto distribution

David R. Fox

PRELIMINARIES

The `Burrlioz` software and `actuar` package in R use different parameterisations of the *inverse Pareto* distribution (a.k.a 'European' and 'American' implementations).

This difference has important ramifications for estimation, random data generation, and HCx estimation. Without going into the mathematical detail, the differences can be reconciled as described below.

RELATIONSHIP BETWEEN HC$_X$ VALUES

First, assume the sample data $\{Y_1,\ldots,Y_n\}$ were generated using the `rinvpareto` from the `actuar` package with $shape = a$ and $scale = b$.

If this *Y*-data is used in `Burrlioz` without modification, the shape and scale parameter estimates obtained are *not* estimates of $\{a,b\}$ and consequently, the HCx estimates from `Burrlioz` will *not* be representative of the true HCx values. To avoid this situation, we need to do the following:

Step 1

Transform the *Y*-data to obtain *Z*-values as follows:

$$Z = \frac{1}{b}\left(\frac{Y}{Y+b}\right)$$

Step 2

Supply the *Z*-data to `Burrlioz` and fit the *inverse Pareto* distribution.

Step 3

Denote the p^{th} percentile of the *Y*-data as ξ_p and the p^{th} percentile of the *Z*-data as ψ_p.

Then, the `actuar` percentile (ξ_p) is obtained from the `Burrlioz` percentile (ψ_p) as:

$$\xi_p = \frac{b^2}{\left(\frac{1}{\psi_p - b}\right)}$$

An example follows.

Reconciliation of HCx values from 'American' and 'European' representations of the Inverse Pareto Distribution – An Example

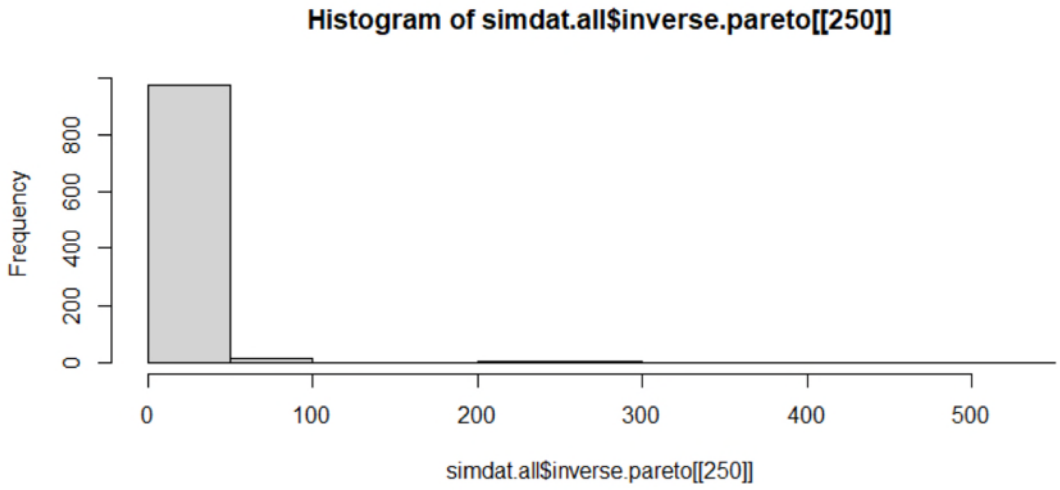
The data

```
simdat.info[2191,]
```

shape	scale	mean	sd	skewness	kurtosis
59.5	49.5	7.519321	30.09381	10.21264	133.2724

distr	qdist	actual.HC1	actual.HC5	actual.HC10	actual.HC20
inverse.pareto	qinvpareto	0.2510446	0.3912279	0.5119949	0.7368017

NB: The actual scale value used to generate these data was 1/49.5



American IP (`Burrlioz`) fitted to European IP (`actuar`) data

```
y<-simdat.all$inverse.pareto[[250]]   # European IP data

> m<-fit(y,ldensity = inverse.pareto.density)
> m
CALL:  fit(dataframe = y, ldensity = inverse.pareto.density)
```

MODEL: inverse.pareto

```
Parameter estimates:
log(alpha) -1.70976853638844 #  => shape estimate = 0.18
log(bound) -6.27300045042321 #  => scale estimate = 0.001887 & 1/0.0019 = 530.06
                             #     Note: max(Y) = 530.0654 = MLE for scale
Log Lik: - 3455.08873685455
```

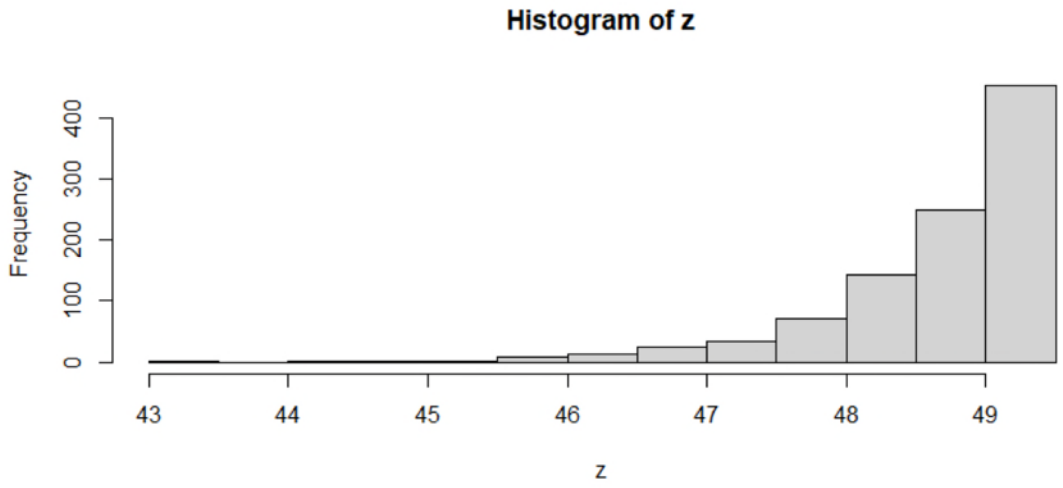
```
> estimate.hc(m)
```

	0.01	0.05	0.1	0.2
	4.666226e-09	3.409177e-05	1.572706e-03	7.255138e-02
(actual)	0.2510446	0.3912279	0.5119949	0.7368017

Data Transformation

```
> b<-1/49.5      # scale parameter
> z<-(y/(yb))/b  #  transformation of Y -> Z
> hist(z)
```

should be y-b



American IP (Burrlioz) fitted to transformed data

```
> ml<-fit(z,ldensity=inverse.pareto.density)
> ml
CALL:  fit(dataframe = z, ldensity = inverse.pareto.density)
```

MODEL: inverse.pareto

```
Parameter estimates:
log(alpha) 4.08087106118778 #  => shape estimate = 59.197 (actual = 59.5)
log(bound) -3.901934557998  #  => scale estimate = 0.0202
                            #     &  1/0.0202 = 49.505 (actual = 49.5)
                            #     &  1/max(Z) = 0.0202 = scale MLE

Log Lik: - 804.178976920753
```

> **HC<-estimate.hc(m1)**

```
> HC
     0.01      0.05       0.1       0.2
45.79343  47.05553  47.60975  48.17050
```

Use HC estimates from transformed data and formula given in paper to estimate HC on original scale

```
>  b^2*(1/HC[1]-b)^-1
      0.01
0.2495891                    #  actual = 0.2510446

>  b^2*(1/HC[2]-b)^-1
      0.05
0.3888848                    #  actual = 0.3912279

>  b^2*(1/HC[3]-b)^-1
       0.1
0.5088286                    #  actual = 0.5119949

>  b^2*(1/HC[4]-b)^-1
       0.2
0.7319586                    #  actual = 0.7368017
```

Appendix I: Additional analyses assessing the recommended distribution set

Rebecca Fisher

Comparing the recommended candidate distribution set to using all distributions

The final recommended set of candidate distributions comprises the *log-normal*, *log-logistic*, *inverse Weibull*, *Weibull*, *Gamma* and the *log-normal-log-normal* mixture distribution. Here we examine the impact of restricting the candidate set to only these recommended distributions compared to using all of the available distributions (*Burr III*, *Gamma*, *Gompertz*, *inverse Pareto*, *log-Gumbel* (*inverse Weibull*), *log-logistic*, *log-logistic-log-logistic* mixture, *log-normal*, *log-normal-log-normal* mixture, and *Weibull*), as was done in many of our initial analyses and comparisons.

The largest set of simulation results are those from Simulation Study 1 (see Section 2.5.2, Simulated datasets). We re-analysed these results using the recommended set of distributions, and compared the bias and coverage relative to the original analyses based on all of the available distributions. We found that there was a slight increase in bias when the distribution set was restricted to the recommended set compared to when all of the available distributions were used, particularly for data generated using a *log-logistic* distribution at low species protection values (0.01, Fig. I-1). While the recommneded set results in slightly lower coverage compared to when all distributions are used, the differences are marginal (Figure I-2) and coverage overall is very high compared to either the original "default"(*log-logistic, log-norma*l and *gamma*), or that obtained via RBurrlioz (see Figure 16).

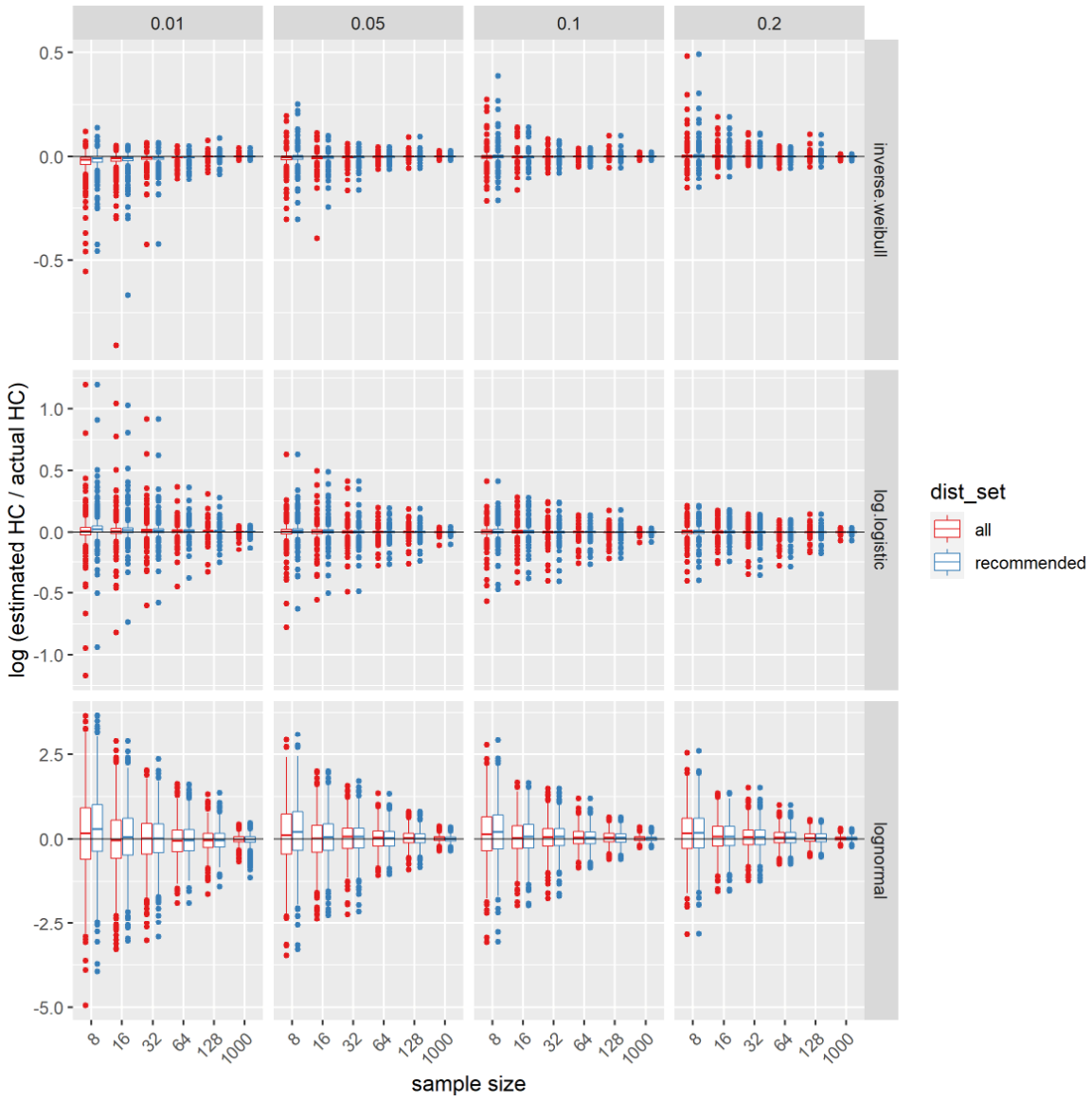


Figure I-1. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the candidate distribution set used for model averaging and includes one using only the distributions recommended in this report (*log-logistic*, *log-normal*, *gamma, inverse Weibull, Weibull,* and *log-normal-log-normal* mixture, recommended) and one using all the available distributions (all).

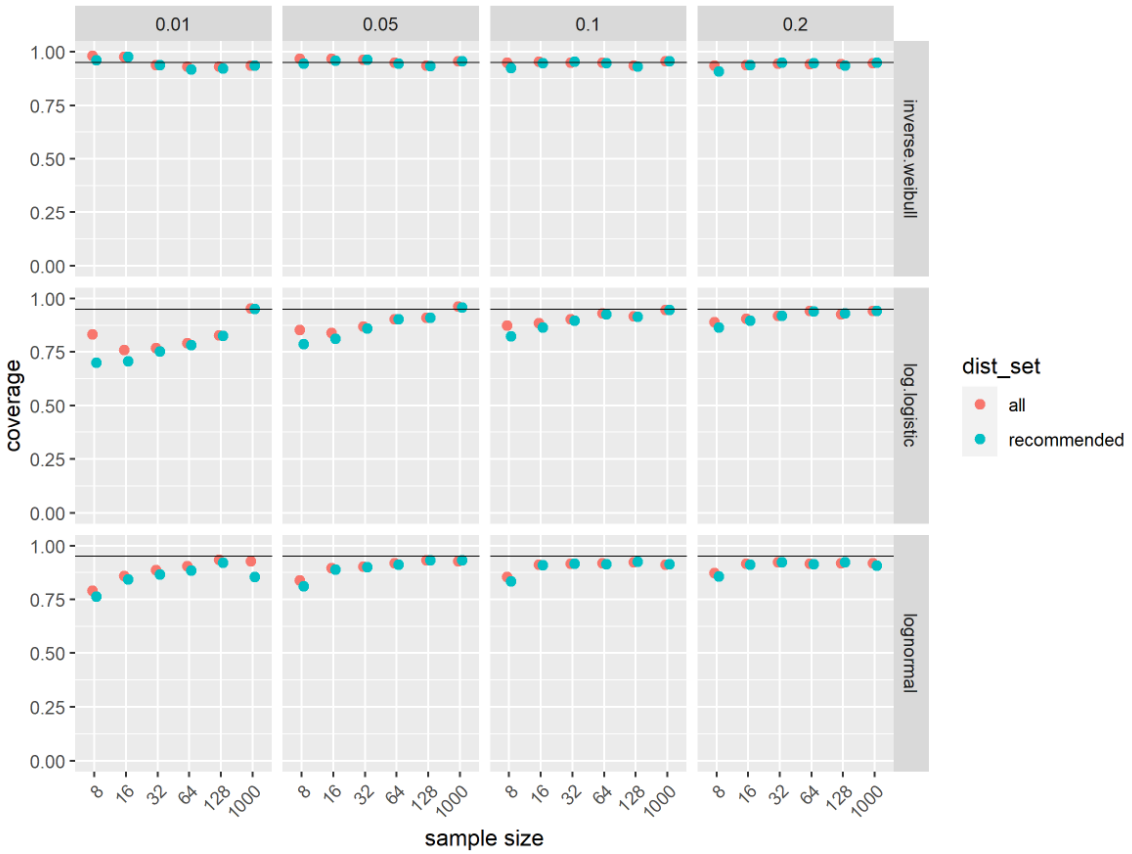


Figure I-2. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different parent distributions (plot rows - *inverse Weibull*, *log-logistic* and *log-normal*). Plots are coloured according to the candidate distribution set used for model averaging and includes one using only the distributions recommended in this report (*log-logistic*, *log-normal*, *gamma, inverse Weibull, Weibull,* and *log-normal-log-normal* mixture, recommended) and one using all the available distributions (all).

Implications of the recommended set for data following a *Burr III* distribution

The *Burr III* distribution is the primary distribution adopted in the `Burrlioz 2.0` method and software. For reasons described in our discussion, our final recommended set does not include this three-parameter distribution in the candidate set for model averaging. While the examination above (Comparing the recommended candidate distribution set to using all distributions) using the Simulation Study 1 data suggests the recommended set should have comparable coverage and accuracy compared to using all the available distributions in `ssdtools`, those simulated data were based on the *log-logistic*, *inverse Weibull* and *log-normal* distributions, all of which are retained in the recommended candidate distribution set. Simulation studies are essential for assessing coverage and accuracy of SSD modelling methods because knowledge of the 'true' parameter values (and corresponding HCx values) is required. However, the choice of distribution used to generate the data in the simulations can have a direct impact on the results. To examine a worst-case scenario where the underlying generating distribution is not one of those contained in the recommended candidate set, we simulated data using the three parameter *Burr III* distribution to assess coverage and bias, as an additional test of the robustness of the recommended final candidate distribution set. The simulated data are shown in Figure I-3, and are based on shape1 values of 0.1, 0.5, 1, 5 and 10, with shape2 and scale both being set at 1 (Figure I-3). We found that there was little difference in bias and coverage between the "all" and our recommended set across the range of shape1 values explored (Figure I-4, Figure I-5). In most cases bias was slightly negative (Figure I-4). However, this bias was relatively similar amongst the two distribution sets, resulting in only marginal loss in coverage for the recommended distribution set when compared to a set using "all" available distributions (Figure I-5).

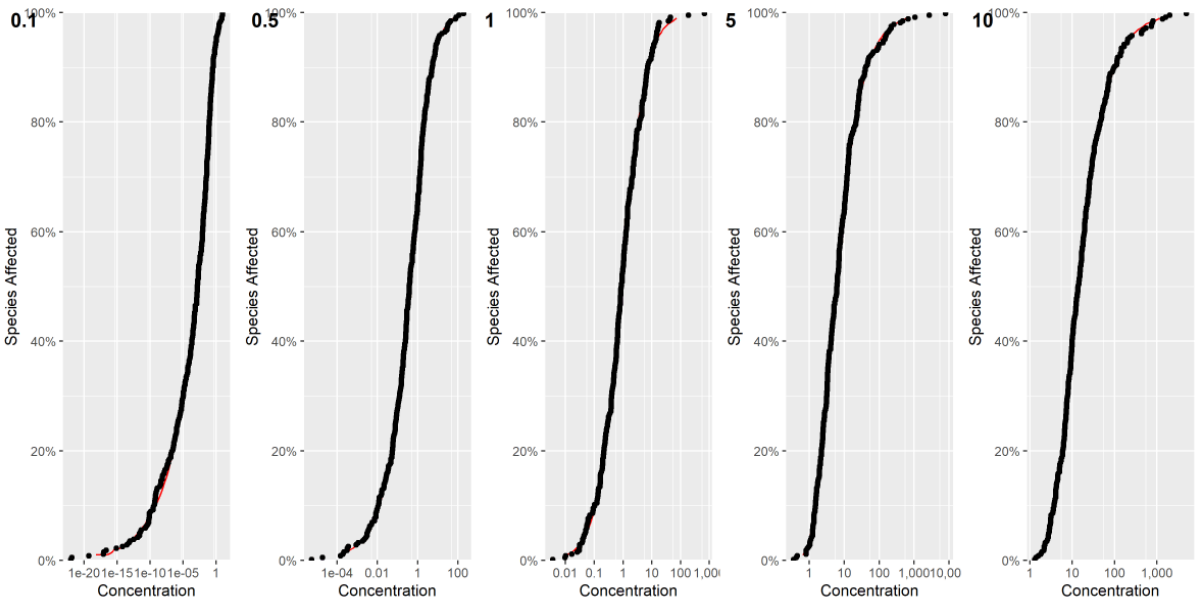


Figure I-3. *Burr III* distributions used in the simulation study to assess coverage and bias estimates using the recommended set of distributions. Data are based on shape1 values of 0.1, 0.5, 1, 5 and 10, with shape2 and scale both being set at 1.

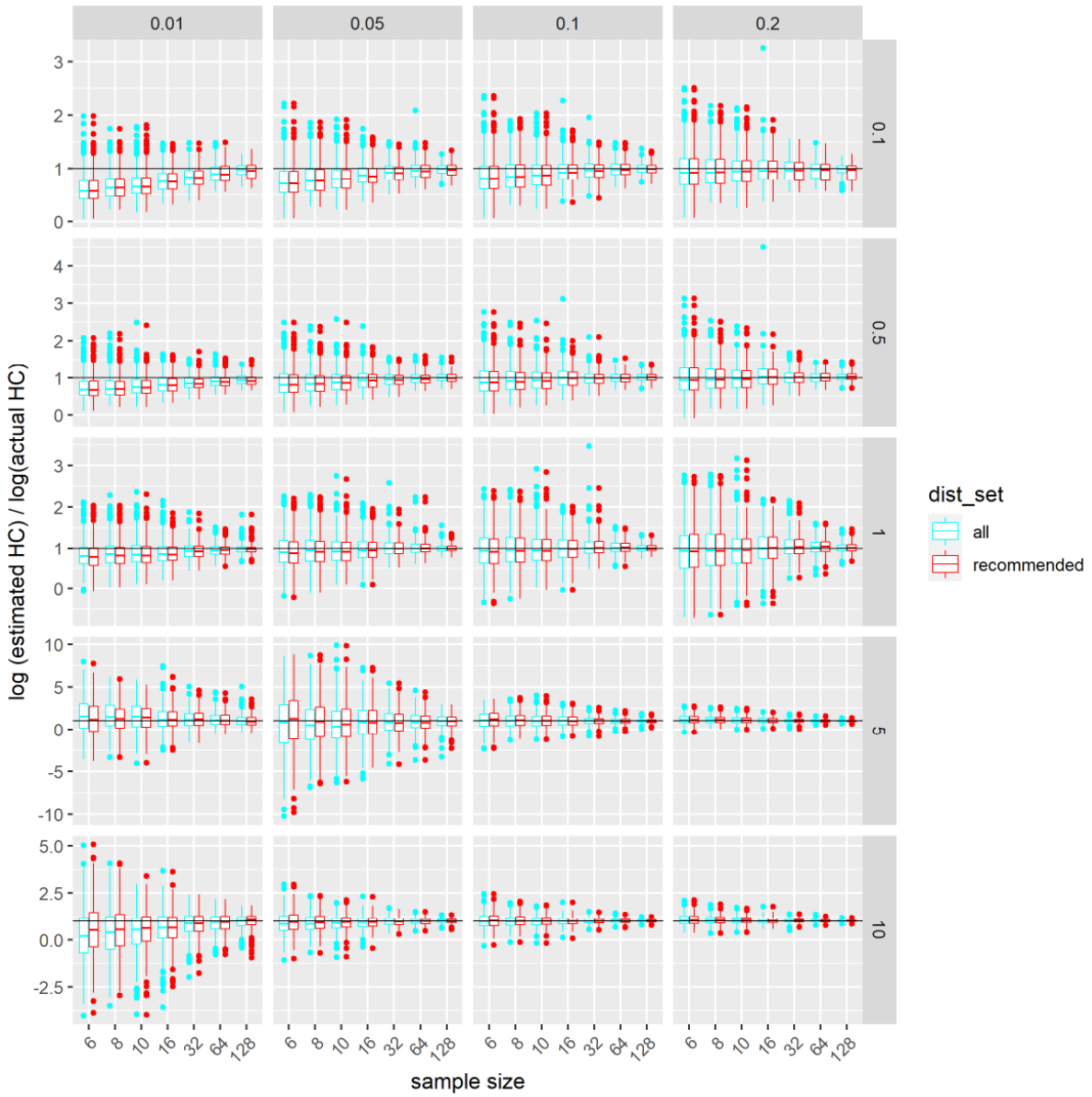


Figure I-4. Bias as measured as the log ratio of the estimated HC value against the actual known value, across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different *Burr III* parent distributions (plot rows - shape1=0.1, 0.5, 1, 5, and 10, see Figure I-3). Plots are coloured according to the candidate distribution set used for model averaging and includes one using only the distributions recommended in this report (*log-logistic*, *log-normal*, *gamma, inverse Weibull, Weibull,* and *log-normal-log-normal* mixture, recommended) and one using all the available distributions (all).

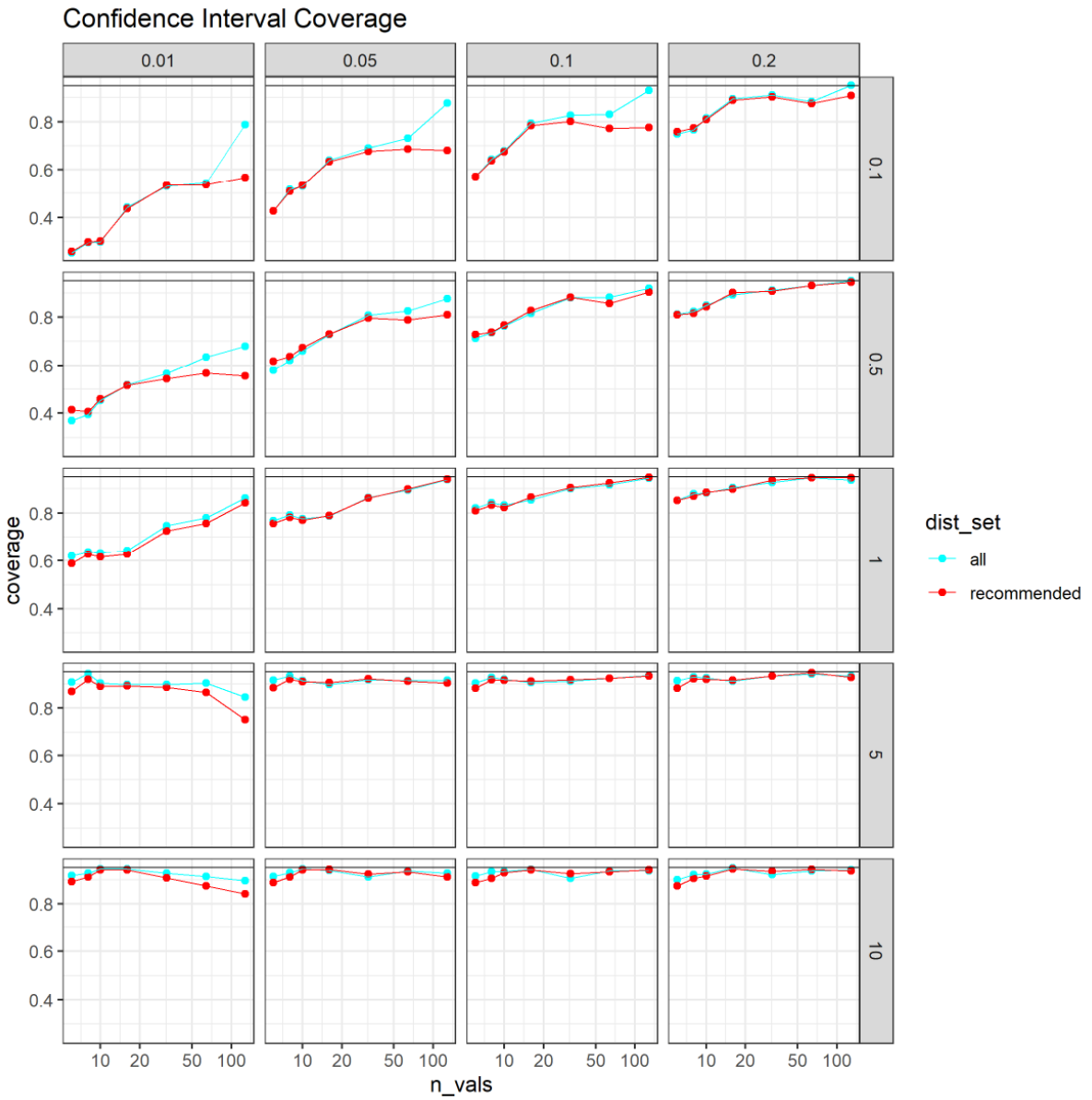


Figure I-5. Approximate coverage estimates for a nominal 95% confidence interval across four different species protection levels (plot columns are p = 0.01, 0.05, 0.1 and 0.2; equivalent to HC1, HC5, HC10 and HC20) for data simulated from three different *Burr III* parent distributions (plot rows - shape1=0.1, 0.5, 1, 5, and 10, see Figure I-3). Plots are coloured according to the candidate distribution set used for model averaging and includes one using only the distributions recommended in this report (*log-logistic*, *log-normal*, *gamma, inverse Weibull, Weibull,* and *log-normal-log-normal* mixture, recommended) and one using all the available distributions (all).

Mixture distribution mixing proportion and boundary conditions

The default arguments in `ssdtools` with respect to the statistical mixture distributions were to restrict the mixing parameter (labelled pmix in `ssdtools`) such that it had to be >0.2, and to exclude the mixture distribution(s) when this parameter is at the boundary (at_boundary_ok = FALSE). These were the conditions used at the time our Simulation Studies 1-3 were carried out.

Because our recommendation is to change these settings (see Finalise default distributions), we examined the resulting impact on model weights for the `ssddata` datasets. Specifically, we compare the original settings of pmix > 0.2 and at_boundary_ok = FALSE (the default at the time the simulations in the body of the report were run) to 0 and at_boundary_ok = TRUE (our recommendations in the current report).

When pmix is restricted to >0.2, nine of the datasets did not include the *log-normal* mixture in the fitted set, presumably due to exclusion of the distribution because pmix is at the boundary (Table I-1). When pmix was allowed to converge to 0 (ie the *log-normal-log-normal* mixture converges to simply a *log-normal*) and be retained even at the boundary, the *log-normal* mixture is included for all 24 example datasets, as would be expected (Table I-1). For two datasets (aims molybdenum marine – unfiltered and anonymous dataset e, Figures I-7g andi) the *log-normal-log-normal* mixture has high weight regardless of the settings used, and both of these datasets show a high level of bi-modality. Of the nine datasets failing to fit the *log-normal-log-normal* mixture with the original default settings, seven had very low weight (Table I-1), as might be expected given the mixture likely converges to a single *log-normal*.

Two datasets for which the *log-normal-log-normal* mixture failed to fit when pmix was restricted to >0.2 gained moderately high weight when this restriction was remove (Table I-1). These were the ccme cadmium dataset (AICc weight = 0.42, Figure I-7o) and the csiro nickel dataset for freshwater (AICc weight =0.32, Figure I-7x). For the cadmium dataset the mixture distribution appears to be modelling some extreme values in the right-hand tail of the dataset (Figure I-7o). For the freshwater nickel dataset, the mixture appears to be modelling two extreme values in the left-hand tail (Figure I-7x).
Overall, despite some small differences in weighting of models based on the two different settings, the estimates of HCx were similar, including estimates of the lower and upper confidence limits (Figure I-6). As such, the recommended changes would be unlikely to alter any of the conclusions drawn from our initial simulation studies based on the default settings.

Table I-1. Relative model weights for the 24 example datasets in `ssddata`, using two alternative arguments for boundary mixing parameter (pmix) and exclusion criteria. The default settings in `ssdtools` at the time simulations were run in the report were pmix = 0.2, at_boundary_ok = FALSE (see column (a). The recommended settings in this report are pmix = 0, at_boundary_ok = TRUE (see column (b). Note some data sets are included more than once with additional filtering.

	pmix=0.2, boundary not ok						pmix=0, boundary ok					
Data	gamma	lgumbel	llogis	lnorm	lnorm_lnorm	weibull	gamma	lgumbel	llogis	lnorm	lnorm_lnorm	weibull
aims aluminium marine (temperate)	0.21	0.12	0.17	0.25	0.00	0.26	0.21	0.12	0.17	0.25	0.00	0.26
aims aluminium marine (tropical)	0.38	0.02	0.12	0.09	0.00	0.40	0.38	0.02	0.12	0.09	0.00	0.40
aims aluminium marine	0.02	0.19	0.32	0.39	NA	0.09	0.02	0.19	0.32	0.39	0.00	0.09
aims gallium marine	0.27	0.09	0.13	0.17	0.00	0.35	0.27	0.09	0.13	0.17	0.00	0.35
aims molybdenum marine (temperate)	0.41	0.04	0.10	0.12	0.00	0.33	0.41	0.04	0.10	0.12	0.00	0.33
aims molybdenum marine (tropical)	0.10	0.46	0.15	0.20	0.00	0.10	0.10	0.46	0.15	0.20	0.00	0.10
aims molybdenum marine	0.05	0.08	0.04	0.08	0.70	0.06	0.05	0.08	0.04	0.08	0.70	0.06
anon a	0.35	0.02	0.10	0.17	0.00	0.35	0.35	0.02	0.10	0.17	0.00	0.35
anon b	0.16	0.15	0.22	0.28	NA	0.20	0.16	0.15	0.21	0.28	0.00	0.20
anon c	0.00	0.60	0.17	0.21	NA	0.02	0.00	0.59	0.17	0.21	0.01	0.02
anon d	0.35	0.06	0.10	0.17	0.00	0.32	0.35	0.06	0.10	0.17	0.00	0.32
anon e	0.20	0.00	0.01	0.01	0.73	0.05	0.20	0.00	0.01	0.01	0.73	0.05
anzg metolachlor fresh	0.19	0.03	0.15	0.25	NA	0.38	0.18	0.02	0.14	0.22	0.10	0.35
ccme boron	0.36	0.01	0.07	0.18	0.03	0.36	0.36	0.01	0.07	0.18	0.03	0.36
ccme cadmium	0.00	0.94	0.06	0.00	NA	0.00	0.00	0.55	0.03	0.00	0.42	0.00
ccme chloride	0.28	0.00	0.25	0.15	NA	0.32	0.26	0.00	0.24	0.14	0.05	0.30
ccme endosulfan	0.22	0.19	0.12	0.24	0.03	0.21	0.22	0.19	0.12	0.24	0.03	0.21
ccme glyphosate	0.02	0.58	0.12	0.23	0.03	0.04	0.02	0.58	0.12	0.23	0.03	0.04
ccme silver	0.08	0.33	0.21	0.27	0.00	0.11	0.08	0.33	0.21	0.27	0.00	0.11
ccme uranium	0.12	0.12	0.22	0.31	NA	0.22	0.12	0.12	0.22	0.31	0.00	0.22
csiro chlorine marine	0.00	0.03	0.63	0.31	0.03	0.01	0.00	0.03	0.63	0.31	0.03	0.01
csiro nickel fresh (temperate)	0.06	0.18	0.29	0.35	NA	0.12	0.06	0.18	0.29	0.35	0.01	0.12
csiro nickel fresh (tropical)	0.21	0.10	0.19	0.31	0.00	0.19	0.21	0.10	0.19	0.31	0.00	0.19
csiro nickel fresh	0.06	0.03	0.35	0.46	NA	0.10	0.04	0.02	0.25	0.33	0.30	0.07

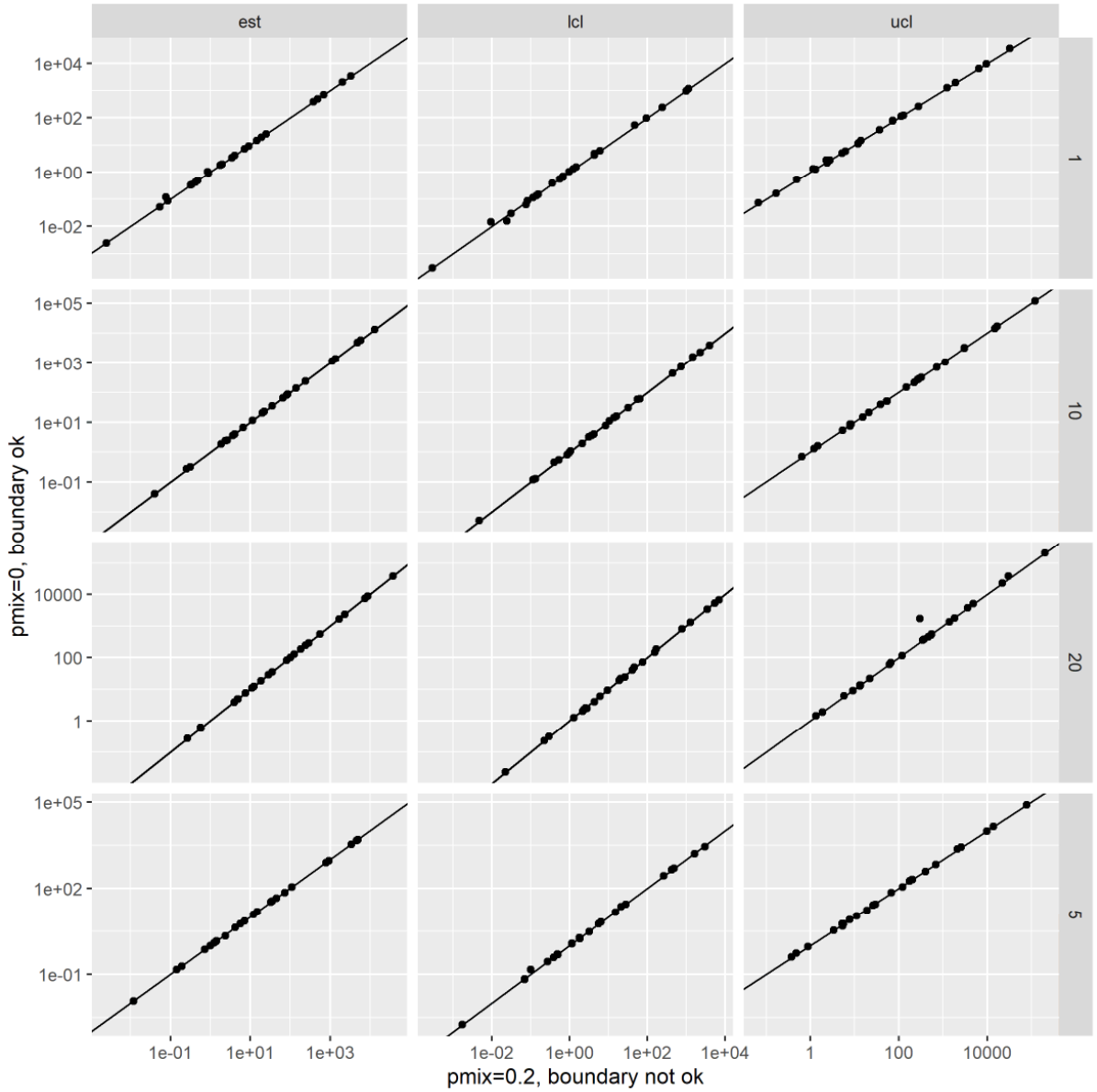


Figure I-6. HC values estimated using `ssdtools` with the recommended distribution set of *log-logistic*, *log-normal*, *Gamma*, *inverse Weibull, Weibull,* and *log-normal-log-normal* mixture. Plotted are fits based on two different default arguments, including pmix = 0.2, at_boundary_ok = FALSE (x-axis) and pmix = 0, at_boundary_ok = TRUE (y-axis). Plot rows show comparisons for 80th, 90th, 95th and 99th protection values (equivalent to HC20, HC10, HC5 and HC1) and plot columns are the actual estimate, as well as the lower and upper confidence bounds.

Allowing the mixing parameter (pmix) to go to 0 did result in a slightly lower proportion of successful bootstrap iterations in some cases, meaning that to obtain confidence bands on the HCx the argument min_pboot had to be lowered from the default 0.99 (Table I-2). The dataset with the lowest proportion was the ccme chloride dataset (at 0.938), although the anzg metolachlor freshwater, ccme cadmium, csiro chlorine marine and csiro nickel freshwater datasets all had proportions lower than the current default in `ssdtools` (min_pboot > 0.99, Table I2). Initial investigations suggest that this is being caused by the TMB being unable to fit the *log-normal-log-normal* distribution to a certain proportion of the parametric bootstrap samples that are generated based on the estimated parameters from the original distribution. Further investigation is required to resolve this issue in the current development

version of ssdtools, although relaxing the current default min_pboot requirement may also be acceptable, given the current setting is very stringent.

Table I-2. Proportion of successful bootstrap iterations for the 24 example datasets in ssddata, using two alternative arguments for boundary mixing parameter (pmix) and exclusion criteria. The default settings in ssdtools at the time simulations were run in the report were pmix = 0.2, at_boundary_ok = FALSE (see column (a). The recommended settings in this report are pmix = 0, at_boundary_ok = TRUE (see column (b). Note some data sets are included more than once with additional filtering.

Data	a) pmix=0.2, boundary not ok	b) pmix=0, boundary ok
aims aluminium marine (temperate)	1	0.999
aims aluminium marine (tropical)	0.999	0.999
aims aluminium marine	1	1
aims gallium marine	0.999	0.999
aims molybdenum marine (temperate)	0.998	0.999
aims molybdenum marine (tropical)	0.996	0.999
aims molybdenum marine	0.991	0.992
anon a	1	1
anon b	0.999	0.999
anon c	1	1
anon d	1	1
anon e	0.996	0.998
anzg metolachlor fresh	1	0.964
ccme boron	0.995	0.994
ccme cadmium	1	0.94
ccme chloride	1	0.938
ccme endosulfan	0.994	0.991
ccme glyphosate	0.998	0.994
ccme silver	0.999	0.999
ccme uranium	1	1
csiro chlorine marine	0.999	0.941
csiro nickel fresh (temperate)	1	1
csiro nickel fresh (tropical)	1	1
csiro nickel fresh	1	0.964

All ssddata fits using the recommended settings

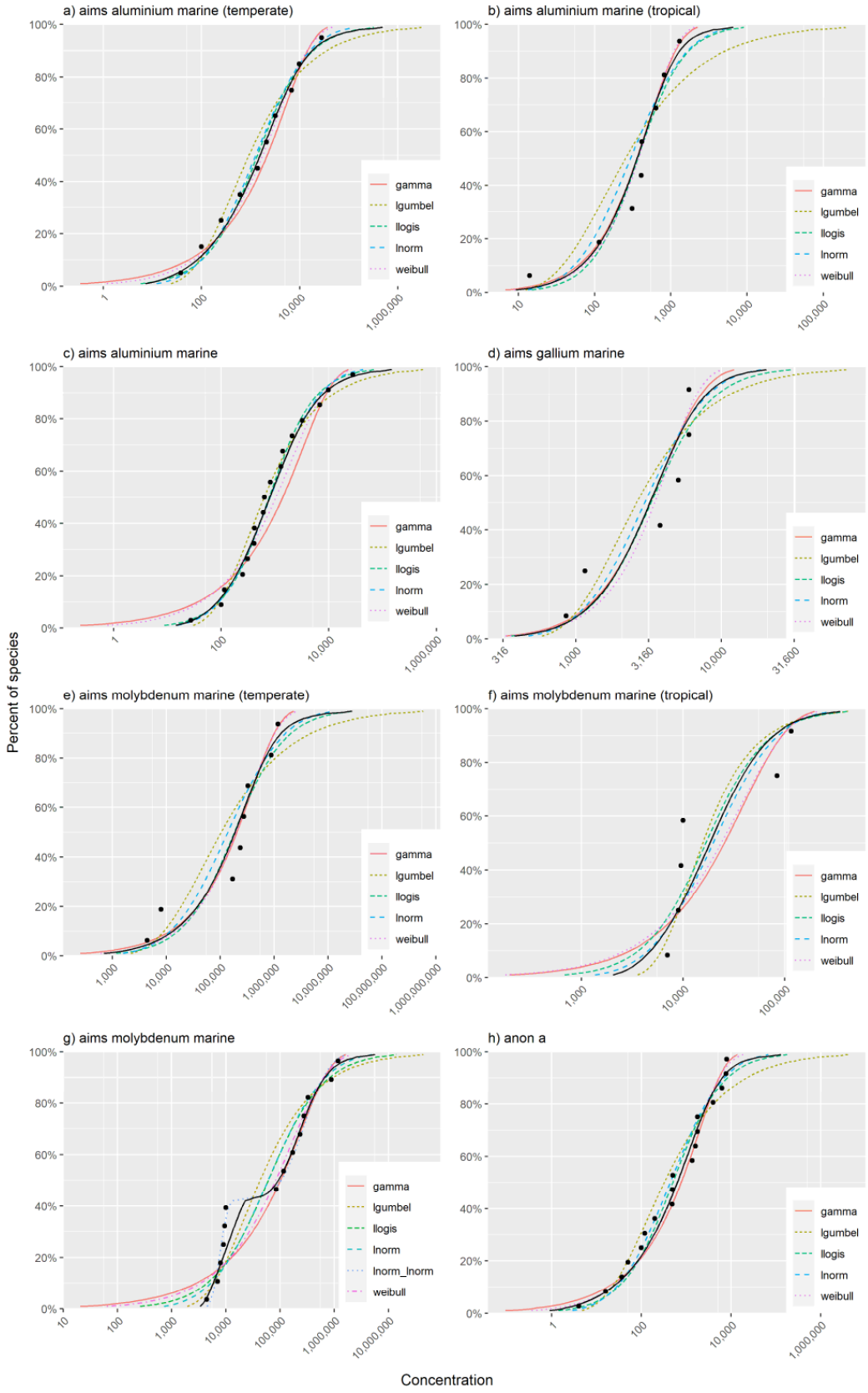


Figure I-7i. All ssddata datasets fitted using the recommended distribution set in `ssdtools`. Note some datasets are also fitted to subsets based on filtering to a specific domain.

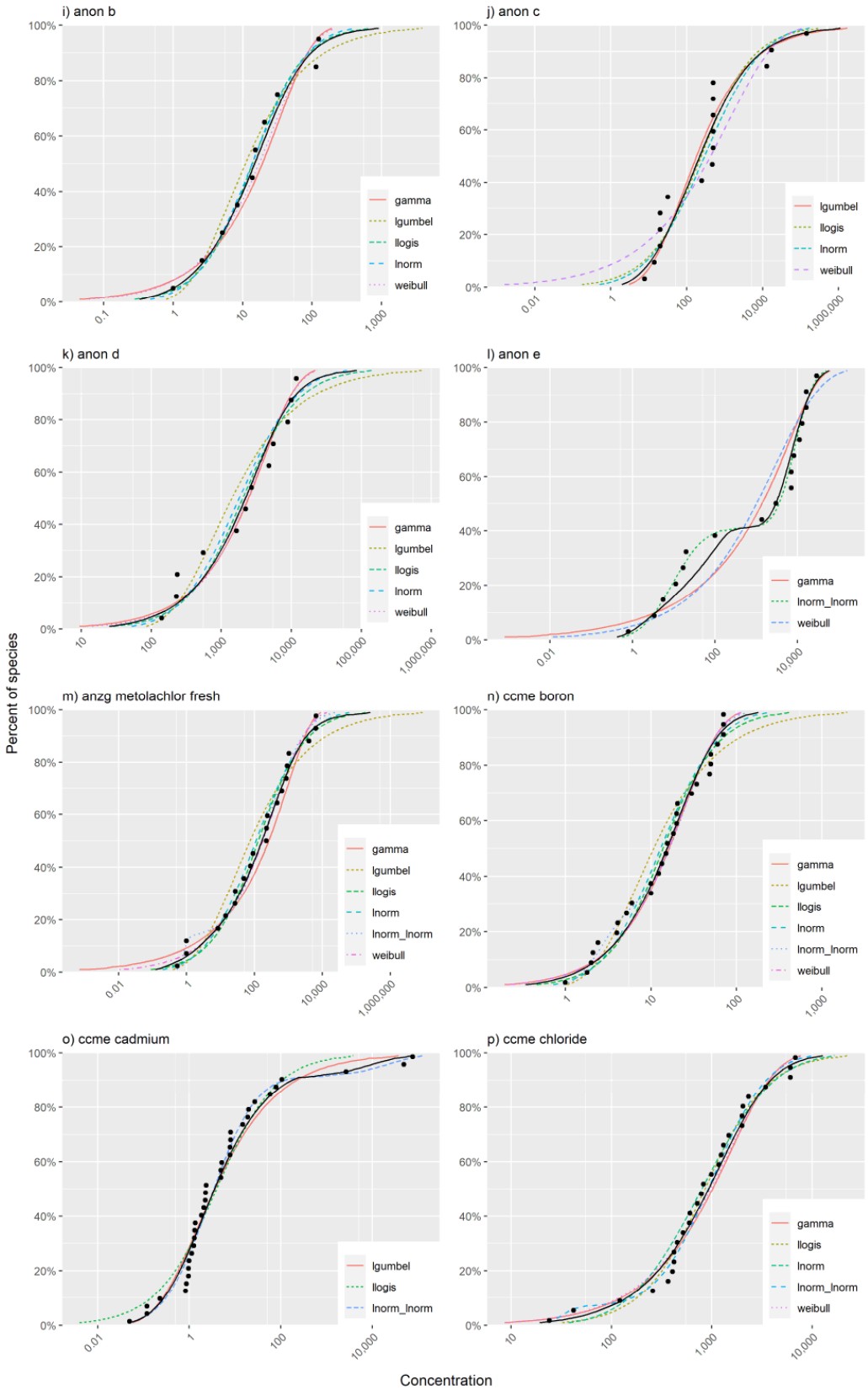


Figure I-7ii. All ssddata datasets fitted using the recommended distribution set in `ssdtools`. Note some datasets are also fitted to subsets based on filtering to a specific domain.

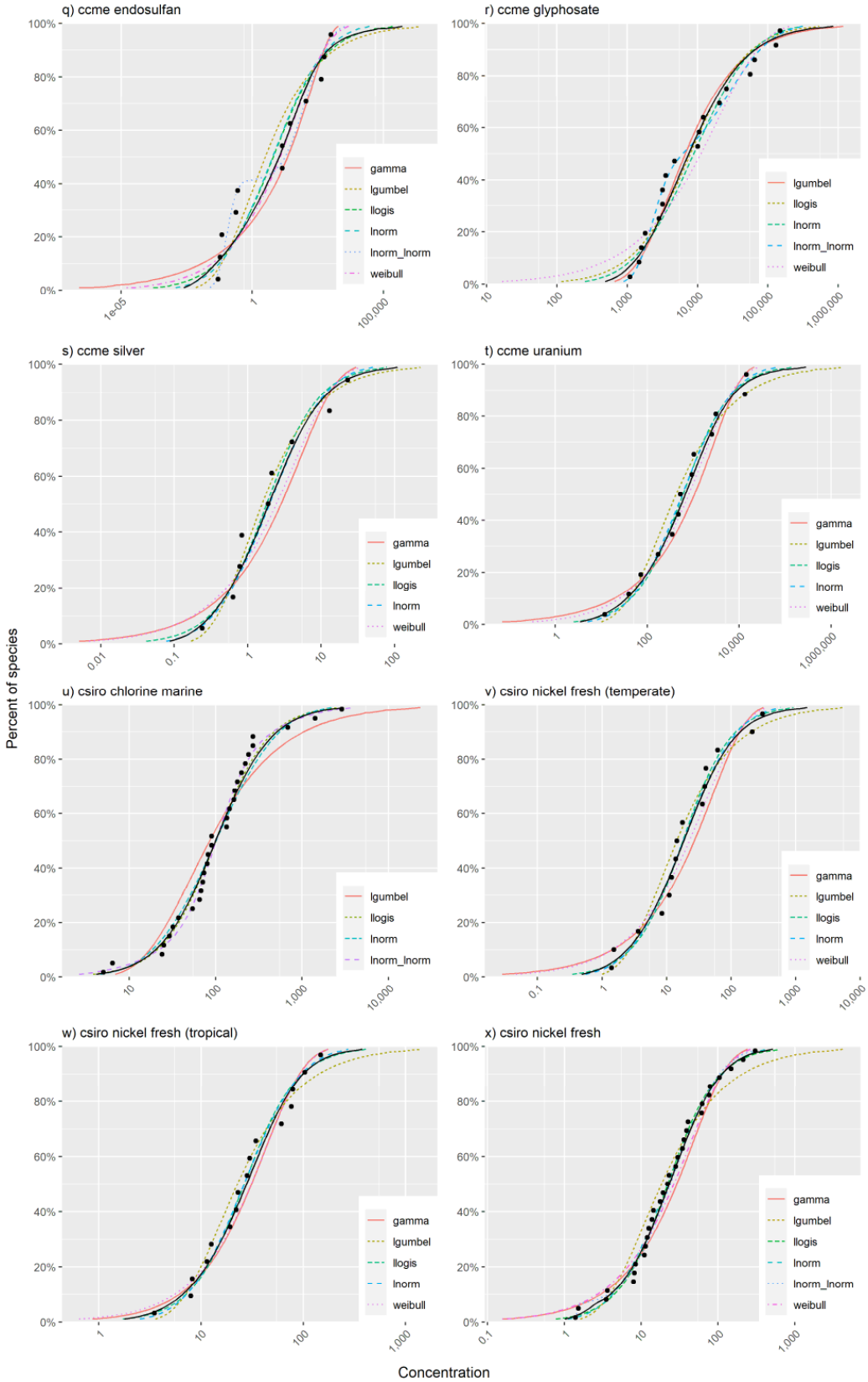


Figure I-7ii. All ssddata datasets fitted using the recommended distribution set in `ssdtools`. Note some datasets are also fitted to subsets based on filtering to a specific domain.