SCIENTIFIC COMMITTEE

SEVENTEENTH REGULAR SESSION

ONLINE MEETING

11-19 August 2021

DEVELOPMENTS IN THE MULTIFAN-CL SOFTWARE 2020-21

WCPFC-SC17-2021/SA-IP-01

24 July 2021

Nick Davies[3], David Fournier[2], Fabrice Bouyé[1], and John Hampton[1]

[1] Oceanic Fisheries Programme, The Pacific Community, Noumea, New Caledonia

[2] Otter Research Ltd, North Saanich, British Columbia, Canada

[3] Te Takina Ltd, Whangarei, New Zealand

Table of Contents

Executive summary 5

1 Introduction 6

2 Development overview 6

2.1 Team 6

2.2 Calendar 6

2.3 Collaboration and versioning 7

2.4 Compilation framework and Source code repository 7

2.4.1 Compilation framework 7

2.4.2 Compilation of dependent libraries 7

2.4.3 Compilation of Linux executable 8

2.4.4 Compilation of Mac OS executable 8

2.4.5 Visual Studio 2019 Windows compilation 8

2.4.6 Development version 9

2.4.7 Source code repository 9

2.5 Developer's workshops 9

2.6 Benchmark testing during 2020-21 10

2.6.1 Version 2.0.8.0 10

2.7 Postings to website 11

2.8 Independent Peer Review of the 2011 bigeye tuna stock assessment 11

2.9 Tool development 12

2.9.1 R4MFCL 12

2.9.2 Testing framework 12

2.9.3 Viewer 13

2.9.4 Condor parallel processing facility 13

2.10 User's guide ..... 13~~14~~

3 Training workshop 14

4 New features 14

4.1 Catch-conditioned fishing mortality calculation 14

4.1.1 Rationale 14

4.1.2 Methods and Testing 14

4.1.2.1 Initial population at equilibrium 15

4.1.2.2 Newton-Raphson estimation of fishing mortalities 15

4.1.2.3 Fishing mortality levels and effort relationship 15

4.1.2.4 Survey fisheries that provide indices of relative abundance 16

4.1.2.5 Preliminary testing ..... 16~~17~~

4.1.2.6 Further work required 18

4.2 Positive Definite Hessian (PDH) diagnostics 18
4.2.1 Rationale 18
4.2.2 Methods and Testing 18
4.2.2.1 Positivised Eigenvalue Diagnostic (PED) 18
4.2.2.2 Empirical evidence of the influence of negative eigenvalues 19
4.3 Parallel Hessian calculation – "stitching" the components 19
4.4 Higher precision calculations 20
4.4.1 Rationale 20
4.4.2 Method and experimentation 20
4.5 New movement parameterisations 21
4.6 Natural mortality at age – minimization stability 21
4.7 Pre-minimisation conversion of recruitment parameterization ..... 21~~22~~
5 Enhancements and bug fixes 22
5.1 Revised content for PDH diagnostic report 22
5.2 Correction to variance calculations 22
5.3 Additional dependent variables in the *.var ..... 22~~23~~
5.4 Orthogonal-polynomial parameterisation 23
5.5 Predicted tag recaptures – added constant for robustness 23
5.6 Maturity-at-length conversion to age calculations 23
5.7 Parameter scaling 23
5.8 Missing both catch and effort records 23
5.9 Fish_flags sanity checks ..... 23~~24~~
5.10 Age-length data input sanity check 24
5.11 Penalty to movement coefficients 24
5.12 Enhancements/corrections for the multi-species/sexes/stocks feature 24
5.12.1 Fishing impact calculations using the recruitment multiplier 24
5.13 Bug fixes 24
5.13.1 Fix to unfished adult female biomass 24
5.13.2 Fix to the plot.rep section for the Kalman filter 24
5.13.3 Fix to region_pars 24
5.13.4 Fish to .par input of fish_pars 24
6 Application of new features 24
6.1 Spatial variation in growth and movement rates 24
6.2 Stock assessments for SC17 25
7 Future work 25
8 Discussion 26
9 References 27
10 Tables 28

11 Figures 33

EXECUTIVE SUMMARY

This paper summarises the developments made within the MULTIFAN-CL software project as carried out by the team at the Oceanic Fisheries Programme (OFP, The Pacific Community, Noumea, New Caledonia) from August 2020 to July 2021, and updates the report of Davies et al. (2020).

The progress made on implementing new features related largely to improving its capability for stock assessments. Highlights during 2020-21 include:

- Fishing mortality estimates conditioned in respect of observed catches (the catch-conditioned method).
- Diagnostics of the Hessian solution with methods that evaluate the implications of negative eigenvalues.
- Capability for compilation for undertaking higher (128-bit) precision calculations.

Various enhancements were made including: alternative movement parameterisations, improved stability in estimating age-specific natural mortality, and a pre-minimisation conversion from the mean+deviate parameterisation of recruitment to the orthogonal-polynomial form. Collectively, these improvements aim to enable analysts to obtain well-determined solutions through reducing model complexity; providing the diagnostic tools to effectively address confounding among the independent variables and for assessing the implications of non-positive definite Hessian solutions.

Development of the catch-conditioned method for estimating fishing mortality has been in progress since early-April 2020 (Davies et al. 2020), and has been a main focus during 2020-21. In brief, the development status is that trial implementations have been done using a range of tuna and billfish examples, that produce converged solutions, some of which are well-determined (positive definite Hessian). In the past 6 months, the feature has been further refined to deal with spurious input data; and, aspects have been revised including: the assumptions for the fishing mortality:effort regression in respect of fisheries with standardised indices, and the addition of a dedicated likelihood term for relative abundance indices (CPUE) supplied by survey fisheries. The feature has reached a sufficient standard to be used for experimental model runs, but with caveats in respect of aspects yet to be completed and tested. These include: the initial equilibrium population calculations when doing fishing impact analysis, excluding survey fisheries from the fishing mortality:effort regression penalty function, and application for multi-species/sexes/stocks. The method is described, with an example that demonstrates a transition from a catch-errors to a catch-conditioned model.

The Hessian diagnostics enables assessment of the implications of several near-zero negative eigenvalues on the interpretations possible from a non-positive definite solution, and making such diagnostics was made more logistically feasible. In one example, the heuristic was satisfied that there was practically no influence upon the management advice taken from the non-PDH solution. It is hoped that these diagnostic tools, and other new tools added for transitioning more easily to less complex parameterisations, will enable analysts to strategically explore the spectrum of model complexities for a particular problem, and to progress towards a plausible set of well-determined solutions upon which management advice may be offered.

Ianelli et al. (2012) reported thirteen recommendations from an independent peer review panel specifically relating to MULTIFAN-CL. During 2020-21 resources were focused on the new features mentioned above, which assumed high priority, especially the catch-conditioned feature. Many of the incomplete items from 2020-21 are retained in the work plan, and the two remaining recommendations of Ianelli et al. (2012), yet to be completed, are included.

The 2021 on-site training workshop for stock assessment analysts to be held at Nouméa, was unfortunately cancelled due to travel restrictions associated with the COVID-19 pandemic. However, an on-line workshop was held during Sep.-Oct. 2020 which proved to be successful.

In respect of future work, the proposed workplan for 2021-22 seeks to consolidate all the new features, especially the catch-conditioned method, and to draft the support documentation, as follows:

- Testing the implementation of examples that employ all the new features and refine the I/O and diagnostic reports.
- Achieving a final "baseline" MULTIFAN-CL version 2.0.8.#.
- From Jul. to Dec. 2021, no further large-scale new developments are planned. Time will be spent to tidy the code; complete a backlog of bug fixes; attend to the long outstanding 2 tasks from the bigeye tuna independent review panel recommendations; and any "small-scale" requests in the tasks list.
- Provide support for MSE requirements and improvements.
- Catch up on the remaining documentation required for updating the MFCL Manual.

Consequently, the proposed items for new features and enhancements will be retained in the MULTIFAN-CL development list, but will be fit within the context of the above workplan for 2021-22.

1 INTRODUCTION

MULTIFAN-CL is a statistical, age-structured, length-based model routinely used for stock assessments of tuna and other pelagic species. The model was originally developed by Dr David Fournier (Otter Research Ltd) and Dr John Hampton (The Pacific Community) for its initial application to South Pacific albacore tuna (Fournier et al. 1998). It has since provided the basis for undertaking stock assessments in the Western and Central Pacific Ocean.

The MULTIFAN-CL model is described in detail in the User's Guide (Kleiber et al. 2018). It is typically fitted to total catch, catch rate, size-frequency and tagging data stratified by fishery, region and time period. For example, recent tuna and billfish assessments (e.g. Castillo-Jordan et al. 2021, Ducharme-Bath et al. 2021) encompass long time periods, e.g. 1952 to 2019 in quarterly time steps, and model multiple separate fisheries occurring in up to 9 spatial regions. The main parameters estimated by the model include: initial numbers-at-age in each region (usually constrained by an equilibrium age-structure assumption), the number in age class 1 for each quarter in each region (the recruitment), growth parameters, natural mortality-at-age (if estimated), movement, selectivity-at-age by fishery (constrained by smoothing penalties or splines), catch (unless using the catch-conditioned catch equation), effort deviations (random variations in the effort-fishing mortality relationship) for each fishery, initial catchability, and catchability deviations (cumulative changes in catchability with time) for each fishery (if estimated). Parameters are estimated by fitting to a composite (integrated) likelihood comprised of the fits to the various data types, and penalized likelihood distributions for various parameters.

Each year the MULTIFAN-CL development team works to improve the model to accommodate changes in our understanding of the fishery, to fix software errors, and to improve model features and usability. This document records changes made since August 2020 to the software and other components of the MULTIFAN-CL project both for the current release version (2.0.7.0), and the current unreleased development version, and updates the report for the previous period, 2019-20 (Davies et al. 2020).

2 DEVELOPMENT OVERVIEW

2.1 Team

The senior developer of MULTIFAN-CL is Dr David Fournier, of Otter Research Ltd, (Canada). Assisting with programming is Nick Davies. Other tasks include testing and debugging (ND, John Hampton, and Fabrice Bouyé (SPC)); documentation (ND); and planning and coordination (DF, ND, and JH). Related project software are developed or managed by FB (MULTIFAN-CL Viewer, Condor, GitHub, Jenkins), ND, and Robert Scott (R4MFCL, FLR4MFCL).

2.2 Calendar

August – November: Testing, planning and ongoing code development, Developer's workshop

December – February: Testing and ongoing code development

March-April: Training and Developer's workshops

May-July: Testing, ongoing code development and support for stock assessments

2.3 Collaboration and versioning

The repository and overall development are coordinated via the GitHub website on GitHub.com at https://github.com/PacificCommunity/ofp-sam-mfcl which is administered by Fabrice Bouye (fabriceb@spc.int) (section 2.4.7).

Problems with MULTIFAN-CL operation or compilation have been reported to the project management website so as to maintain a list of desired enhancements, and to allocate tasks among the project team. A "master" branch exists for the MULTIFAN-CL source code from which release versions are posted, and development branches ("ongoing-dev", "mac-dev") have been created for holding development versions of the source undergoing development and testing. A formal testing procedure has been designed before source code is merged from the branch to the trunk, and a manual for the testing of new compilations, standardizing the source code compilation procedure, and posting of executables is maintained.

2.4 Compilation framework and Source code repository

2.4.1 Compilation framework

A continuous integration facility allows for automatic nightly compilations of the MULTIFAN-CL source on the GitHub repository "master" branch. This automation is done using the software called Jenkins (https://jenkins-ci.org/), an Open Source continuous integration tool that comes bundled with a web server used for administration. This software is now installed on a Linux Virtual Machine (VM) that is dedicated to MULTIFAN-CL development, and administers the compilations over the OFP network.

In this tool, we've added a custom scheduled task that automatically retrieves the MULTIFAN-CL source code out of the GitHub code repository (master branch); it also retrieves required libraries for the compilation. When done, our task compiles both debug and optimized versions of the software. We've also configured this task to produce code documentation out of the source code and to run some C++ code quality checking.

Doing a nightly compilation allows us to find out more quickly whether issues have been included in the source code repository without being solved by the developer. It also helps us identify issues in the makefile configurations that may prevent the compilation of MULTIFAN-CL on some more neutral environment (i.e.: on a machine that is different from the one of the developer's).

During 2018-19 this facility was extended to support automated builds of the Windows (Visual Studio 2019) and the macOS release executables. The Windows10 VM used for undertaking the benchmark testing framework (see section 2.6) and the Mac Mini provided the platforms for undertaking the routine compilation administered by Jenkins (see section 2.4.4). These automated builds were maintained throughout 2020-21.

It is also intended to add to the Jenkins tool the running of automated tests using example fish model data, and, in the future, unit tests for the software.

A directory structure on the dedicated VM was used that is mirrored on all the developer's platforms in respect of source code **Projects/**, associated libraries **libs/**, and **Testing/** directories. This ensures portability of source and makefiles among the developers and the automated build software.

2.4.2 Compilation of dependent libraries

For compilation of the dependent **OpenBLAS** library, the "dynamic architecture feature" is included to the routine compilations that builds several kernels for various processor types, and allows selection of the appropriate kernel at run-time. This may avoid the case where a MULTIFAN-CL executable that was compiled with OpenBLAS on a platform having a very recent processor, fails upon execution because function calls to the OpenBLAS library are attempted on platforms having relatively older processors. This compilation method results in a substantial increase (22 MB) in the executable size. However, it was noted that OpenBLAS libraries are important for the calculations used for the eigenvalues and eigenvectors of the Hessian and also aspects of the self-scaling size composition likelihood.

In order for the MULTIFAN-CL project to be completely portable, three shell scripts automate the compilation of all the dependent libraries, before compiling MULTIFAN-CL. These scripts apply different options for OpenBLAS, QD and compilation flags for MULTIFAN-CL. The script “build_openblas4mfcl.sh” builds 3 options of this library: "default", "generic", and "dynamic", where the "dynamic architecture feature" builds several kernels for various processor types, and allows them to be selected at run-time. Similarly, the script “build_qd4mfcl.sh” builds 4 options of the QD library: "default", "O3", "O3fma", and "native". Given the various combinations of compilation options among the dependent libraries, ADMB and MULTIFAN-CL, compilations of 25 different executables may be produced. For a single option, it compiles in total: 49 minutes 5 seconds. This facilitates the portability of the entire MULTIFAN-CL compilation project including the dependent libraries, such that the complete project may be constructed and compiled with one step.

It is now possible to include in the automated compilation administered by Jenkins, compilation of the dependent libraries QD and OpenBLAS. With the exception of a couple of manual steps required to configure particular options, the integrated compilation of the entire project is now undertaken within the Jenkins routine compilation procedure.

2.4.3 Compilation of Linux executable

The Linux version currently used for compilations is Ubuntu 18.04, with the gcc compiler version 7.3.0. The version of glibc installed with ubuntu18.04 is glibc 2.27, and provides better (faster) Math functions than the earlier versions. It is intended to upgrade to Ubuntu 20.04 and gcc 9.3 or 9.4 during 2021-22.

2.4.4 Compilation of Mac OS executable

During 2018-19, routine macOS compilations of the “master” and “development” branches were added to the compilation framework of the MULTIFAN-CL project.

The version 2.0.8.0 compilation was done on the MULTIFAN-CL Testing PC (Mac Mini) that has the macOS Mojave installed (“macOS 10.14.6 Mojave”). The PC has two compilation directories:

- Local compilation - stand-alone directory for testing development versions
- Jenkins compilation - for routine automated Jenkins compilations of the master branch (checked out from the repository, see section 2.4.1)

A Software ID certificate was assigned to the macOS compilation using an Apple Developer ID certificate for SPC. The macOS version is signed “Developer ID Application: The Pacific Community” issued by Apple. No differences were detected in the computations or performance among the signed and un-signed compilations.

Apple has now officially announced that they are ceasing production with Intel’s CPUs in favour of their own RISC ARM-based CPUs. The macOS compilation of MULTIFAN-CL will ultimately need to accommodate this change. A draft strategy for changing to the ARM-mac compilation for MULTIFAN-CL may entail:

- Following the 2021 Apple conference decide on a machine purchase to be made by May 2022
- Explore the potential for compiling on the existing Intel-mac with output target set to the new ARM-CPU using flags in the most up-to-date or the next version of XCode (the Apple dev tools/compiler)
- In 2021, maintain a careful watch of the capability of the Rosetta 2 emulator for running the MULTIFAN-CL executable (compiled for the Intel CPU) on an ARM-mac; this could offer a "breathing space" for our switch to the ARM-mac compilation
- Consider the merits of upgrading the mac Mini from Mojave to Catalina or Big Sur
- Potentially consider the lead developers purchasing an ARM-mac (external of SPC) for developing the compilation
- Aim for making the switch to the ARM-mac compilation for MULTIFAN-CL in mid-late 2022

2.4.5 Visual Studio 2019 Windows compilation

Few difficulties were encountered with the VS2019 compilations during 2019-20, and generally the MULTIFAN-CL code was compatible among the three compilation platforms. The VS2019 compiler treats differently the casting of constant variables passed as formal arguments, and this was readily addressed using the tool: ADUNCONST(dvar_len_fish_stock_history,fsh).

A Software ID certificate was assigned to the VS2019 compilation using a Windows Developer ID certificate for SPC. The Windows version is signed: "The Secretariat of the Pacific Community" issued by DigiCert. No differences were detected in the computations or performance among the signed and un-signed compilations.

2.4.6 Development version

Upon completing benchmark testing of a development version, the source code in the repository development branch is merged to the master branch and tagged with a release version number. At this point the development branch is created afresh for implementing any subsequent code developments. Another point where a new development version is made is immediately preceding each developer's workshop at which large scale code changes are rapidly made. These are then added to the development branch after the workshop and following preliminary testing. Points during 2020-21 where development versions were created included:

- Following the benchmark testing of version 2.0.8.0
- Preceding the ongoing 2021 developer's workshop

The main developments currently implemented in the development version since version 2.0.8.0 that have not yet been benchmark tested include:

- Enhancements and corrections to the catch-conditioned fishing mortality calculation
- Implementation for a CPUE likelihood term
- Additional dependent variables in the *.var
- Predicted tag recaptures – added constant for robustness
- Maturity-at-length conversion to age calculations - corrections
- Parameter scaling - revisions
- Optional penalty for movement coefficients
- Various corrections (section 5.13)

2.4.7 Source code repository

The MFCL project is hosted on GitHub.com at:

- https://github.com/PacificCommunity/ofp-sam-mfcl

This site is only accessible to registered members of the OFP-SAM team. In order to better coordinate developments within components of the project, separate repositories were created for the:

- User's Guide: https://github.com/PacificCommunity/ofp-sam-mfcl-manual
- ADMB dependent library: https://github.com/PacificCommunity/ofp-sam-admb

The branches of the repository are managed such that following benchmark testing, the development version that has tested positive and held in either of the "mac-dev" or "ongoing-dev" branches, is then merged to the "master" branch. This creates a clear node in the "master" branch tagged as being the next release version. At that point a new development version is created in one of the "mac-dev" or "ongoing-dev" branches for undertaking the next phase of developments. This approach was followed for each of the versions during 2020-21, with the current development version being maintained in the "ongoing-dev" branch (version 2.0.8.2).

A total of 30 source code commits were made to the master and development branches (entire source files have been modified), including merges to the master branch preceding the release of version 2.0.8.0., since the posting of version 2.0.7.0. Substantial commits have been made during the ongoing developer's workshops during 2020-21.

2.5 Developer's workshops

The format of workshops among the primary developer Dr David Fournier and Mr. Nick Davies was changed during 2020-21. This was due to the COVID-19 pandemic (since early 2020) under which travel was not possible. Rather than staging two separate discrete workshops, the collaborative developments were performed as an ongoing continuation of the method used for the April-May 2020 workshop, i.e. using an "on-line" workshop. This entails a Linux **Nomachine** remote desktop protocol server on a mainframe, that enables

both developers to share, with equal participation, a Linux session simultaneously, thus allowing collaborative real-time work on the same desktop. With a concurrent Skype session for verbal and visual communication, this facilitates a "virtual" workshop with considerable success. Rather than staging two workshops, this facility was employed for on average 3-4 days per week for between 2-5 hours per workshop, throughout the year. As such, all the new features and enhancements described in Sections 4 and 5 were achieved using this collaboration method.

2.6 Benchmark testing during 2020-21

The benchmark testing framework is described in section 2.9.2, and one set of benchmark tests was undertaken in 2020-21. A brief description of the test, and the features tested, is provided in this section.

2.6.1 Version 2.0.8.0

In February 2021, and following the on-going on-line developer's workshops to that time, a benchmark test was done of the development version (to become version 2.0.8.0) versus the release version 2.0.7.0. Substantial developments and bug fixes had been made since the previous testing in February 2020. Highlights of new and enhanced features in the development version are described in more detail in section 4:

- **Catch-conditioned method –** implementation for fishing mortality calculation, including incidents with missing catch, and the calculation of initial population conditions based upon assumed average fishing mortalities in the initial time periods.
- **PDH diagnostics** – a set of heuristics to evaluate the importance of negative Hessian eigenvalues (i.e. a non-positive definite Hessian solution).
    - **PED** – adding fixed values that "positivises" the eigenvalue, and to recalculate the st.dev of dependent variables
    - **"Positivizing" the Hessian** – a method to obtain empirical evidence for the relative influence of negative eigenvalues
- **"Stitching" parallel Hessian calculations** – reassembles the component *.hes files created from a parallel Hessian calculation into a single *.hes file.
- **128-bit precision** – an alternative compilation option implementing 128-bit precision calculations.
- **Conversion among movement parameterisations** – facilitates transitioning among the parameterisations with any input **.par** file.
- **Natural mortality** – adjustments to the age-specific estimation to ensure minimisation stability
- **Pre-minimisation conversion of recruitment parameterisation** – to convert from an existing solution obtained using the "standard" mean+deviate parameterisation for recruitments, and to derive the corresponding orthogonal-polynomial parameters.

Note: all the benchmark testing was done using the compilation of the development version for the "standard" 64-bit precision, as this ensures comparability with the benchmark version 2.0.7.0.

Single evaluation tests for single-species and multi-species data, with or without gradient calculations and a minimisation step, indicated no differences among the versions or compilation platforms.

Single evaluation tests using multi-sex data without gradient calculations and a minimisation step, produced identical model quantities and objective function values among the versions for the Linux and mac OS platforms. For the development version Windows executable, minor differences (0.01%) were obtained for one dependent variable. This is likely to be due to small differences in the floating-point calculations on Windows machines, and is unrelated to code changes in devvsn13.

Deterministic and stochastic projections of a single species example exhibited differences among versions within platforms of around 1% for $SB_{current}$. Predicted catches mid-way through the projection period in three of the fisheries were different among the versions, and consequently slight differences in the predicted

total population numbers occurred. The cause of the difference in the fishing mortality calculation among versions was due to a change made for the Newton-Raphson (N-R) calculation for predicted catch in the devvsn13 code that changed the default threshold maximum total mortality. Setting `rmax` using age_flags(116) to ensure equivalent values among the versions produced essentially identical results (see section 10 Annex 2).

Doitall fit tests

Among the versions but within platforms: For the Linux platform executable, negligible differences (~1% or less) in model quantities and objective function terms were obtained among the versions for three of the test data sets (SKJ2016, YFT2017, ALB2018; Table 5). Moderate differences (~20%) were obtained for the ALB2015 and SWO2017 test data sets, and substantial differences for the STM2012 and BET2017 test data sets.

Among the platforms but within versions, and among the versions and the platforms: the differences among the platforms were either minor (4% or less) or moderate (up to 16%) for more complex models. The differences generally being larger for the Windows executables. The reason for these differences most likely does not reflect differences in the code, but rather a difference in the floating-point calculations, particularly of the Windows executable that results in slightly different minimisation paths being taken and with several (or more) different solutions being possible given the data. Consequently, these differences were not due to the code changes made to the development version. This is most pronounced in the complex test data sets, or for ill-determined models. Similar patterns in the differences among versions and platforms were found for the multi-species and multi-sex testing data sets.

Cross-validation of the bigeye testing example demonstrated that each version replicates almost exactly the solution of the other version, however slight differences arise due to the manner in which the movement coefficients are applied. In the development version, the movement coefficient conversion facility among the alternative parameterisations very slightly altered the movement matrix calculation, resulting in minor differences in the tagging likelihood and the gradient. These changes will affect the derivative calculations, hence the gradient, and will alter the minimisation path taken among the versions. This causes different solutions to be obtained for poorly-determined or poorly-converged examples.

Tests of the development version were sufficiently consistent with respect to the benchmark version 2.0.7.0, as all existing features remained intact, and therefore the development version was advanced to the new MULTIFAN-CL version **2.0.8.0**. This version has not yet been posted to the MULTIFAN-CL website for public release (section 2.7).

2.7 Postings to website

There have been no postings of the MULTIFAN-CL release versions to the website since July 2020.

2.8 Independent Peer Review of the 2011 bigeye tuna stock assessment

An outcome of an independent peer review of the 2011 bigeye tuna stock assessment (Ianelli et al. 2012) was a set of recommendations for improvements and developments to the MULTIFAN-CL software. These aim not only to improve the software's application in the context of the bigeye assessment specifically, but also its stock assessment application more generally. These recommendations have been the basis of MULTIFAN-CL developments since the review, and an outline of the status in fulfilling these recommendations is provided.

At the beginning of 2020-21, of the thirteen recommendations, 11 had been implemented and tested, and 2 remained yet to be developed. The recommendation "d", Tags inform movement only, had reached the point of application to pooled and unpooled tags for single species; but testing of its implementation for the multi-species case was yet to be done. The other recommendations that were identified to be undertaken for 2020-21 included (Table 1):

- Non-uniform size bins (recommendation "b")
- Long-term and initial tag loss (recommendation "c")

No progress was made on these recommendations during 2020-21, because higher priority was given to other tasks (some of which were unforeseen) that arose: the catch-conditioned method for estimating fishing mortality; and, the Hessian diagnostics. These are described in section 4.

2.9 Tool development

2.9.1 R4MFCL

The R scripts for working with MULTIFAN-CL, developed by OFP are maintained on a GitHub repository and have been partially updated to adapt to the recent MULTIFAN-CL release version file formats. These scripts are used to manipulate the input files, so that submitting model runs can be automated from R. Other scripts can be used to read in the output files, analyze the results, and generate plots and tables. Only limited and minor improvements were made to this aspect of the project during 2020-21.

2.9.2 Testing framework

The testing framework for MULTIFAN-CL compilations first developed in 2011-12, was applied during 2020-2021 for the benchmark testing of version 2.0.8.0 (section 2.6.1). This framework ensures the repeatability and traceability of testing by streamlining the process for new source code developments through a system of model testing procedures and directories. The testing criterion is based upon pair-wise comparisons of model run results obtained using an existing MULTIFAN-CL compilation (usually the current release version) versus those from a development version compilation. Tests are undertaken over multiple processor platforms (64-bit architecture only), with application to multiple input testing data sets, and with various options for the MULTIFAN-CL operation, viz. single or multiple model evaluations, or full doitall model fits to convergence. This ensures a thorough integrity-check of model quantities and components of the objective function prior to the distribution of new versions.

Since March 2013, the MULTIFAN-CL source code has undergone substantial developments, and those have been described in earlier reports (e.g. Davies et al. 2020), and the recent developments in 2020-21 are described in Sections 4 and 5.

Following the addition of these new features to the development version, regular testing of this versus the release version aims to ensure the integrity of existing operations. Known as “benchmark tests”, those undertaken in 2020-21 are described in section 2.6. The development version was last tested in February 2021 versus the release version 2.0.7.0. The positive result then defines the development version as being the **benchmark** source code, then posted as version 2.0.8.0. Subsequent development versions will then be tested relative to the benchmark to establish their integrity, after which they may be defined as the new benchmark development version. The testing framework entails two levels of tests.

1. Establish the accepted development version

The first level of testing ensures the integrity of existing model features by undertaking tests using a range of single-species data including: ALB2012, ALB2015, BET2011, BET2014, BET2017, YFT2011, SKJ2011, STM2012, SWO2013, SWO2017, YFT2014, YFT2017, SKJ2014 and SKJ2016; to conclude that single model evaluations and the fitted solutions are sufficiently close to regard the development version estimates as being essentially similar to the benchmark version. This indicates integrity of the development version for undertaking single-species model evaluations. Results are compared among the versions and operating systems, to confirm that the development and release versions produced identical solutions. When differences are found, which can be attributable to improvements in the development version, these are accepted.

Tests using multi-species data disaggregated among species are done which entails comparing the fitted solutions of the development version code versus those solutions obtained using the corresponding data for each species fitted individually. These tests concluded that the operations applying to each population in the disaggregated model have integrity and effectively emulate the solutions obtained when each population is modelled individually. Note that species-specific fisheries data were supplied to the models in the test data examples used. Testing was not conducted using test data for which all fisheries data were aggregated among species (or sexes).

Similarly, tests are done for deterministic and stochastic projections with the pair-wise comparisons among versions and operating systems being made.

A positive test result is when the benchmark tests conclude that the development version conserves the existing features, and so can either be advanced as the new release version, or accepted for the new benchmark development version.

2. Establishing integrity of new features

This second level of testing entails a detailed examination of new features. The inputs and model configuration are customized for the new features and the operation of the new algorithms are evaluated in respect of the original formulations. During 2020-21 this level of testing was done for the new features (see section 4), to ensure the correct calculations and the expected results.

Review of Testing Framework

In January 2016 the testing framework was reviewed by project members with the following agreed tasks for improvements:

a) Tidy up the testing framework functions and utilities so as to be as automated as possible and more user-friendly with a view to including other team members in running the tests.
b) Upgrade testing framework functions and utilities for applicability to both single-sex and multi-sex file formats, with portability over condor.
c) Integrate the testing framework functions and utilities into the R4MFCL package and ensure compatibility with all assessment modelling applications.
d) Create a GitHub repository for the testing framework functions, utilities, and testing data.
e) Consolidate the R4MFCL GitHub repository with Rob Scott as the lead developer, and add access levels to Nick Davies as a support developer.
f) Construct a suite of routine tests for the R4MFCL package to be run following each revision to the repository, and load the updated R4MFCL package to the testing framework.
g) Construct a single routine MULTIFAN-CL test operation (e.g. single-evaluation of a fitted test model solution) to be conducted daily and directly from the Jenkins compilation utility that returns an exit status value, with an email report sent to the project developers.

Little action has been taken on these tasks and is also unlikely in the remaining part of 2021. It has been identified as a concern, and that they be included in the 2021 work plan for the MULTIFAN-CL project.

With the routine compilation and development of a macOS executable now firmly implemented in the project, the testing framework includes the MacMini host, and the macOS executable within tests among platforms and versions. The framework therefore conducts testing upon all 3 platforms simultaneously over the Condor network. The test analyses perform pair-wise comparisons among versions and over three platforms: Linux, Windows, and macOS.

2.9.3 Viewer

A new version of the MULTIFAN-CL viewer has been released that can display the results of a multi-species or multi-sex application updated to display the new output added to the MULTIFAN-CL report files.

2.9.4 Condor parallel processing facility

The Condor (www.condor.wisc.edu) facility has been used routinely for managing multiple MULTIFAN-CL model runs on a grid currently numbering more than 40 computers; being windows or Linux platforms. Support for 32-bit architecture has been discontinued since MULTIFAN-CL executables in this architecture have not been produced since version 1.1.5.9. This grid enables intensive model runs for: benchmark testing MULTIFAN-CL development versions; undertaking stock assessments that entail multiple model runs (e.g. sensitivity analyses and structural uncertainty analyses), and for management strategy evaluation. During 2020-21, additional Linux Virtual Machines were added to the grid to increase the number of model runs possible using the Linux development version executable.

2.10 User's guide

A revision to the MULTIFAN-CL User's Guide (Kleiber et al. 2018) has not yet been completed to include the developments since version 2.0.5.1. Proposed future revisions include: incorporating the suggestions arising from the earlier Training workshops; and the recent features added to versions 2.0.6.0, 2.0.7.0 and 2.0.8.0. The revised version will be posted on the website http://www.multifan-cl.org/.

3 TRAINING WORKSHOP

Typically, training workshops for users of MULTIFAN-CL are scheduled for April of each year to be held at the Oceanic Fisheries Programme (Pacific Community) in Nouméa, New Caledonia. As in 2020, however, due to the COVID-19 pandemic and the related travel restrictions, staging the workshop was not possible. Rather, on-line training was held for one of the stock assessment scientists, with 1 hourly sessions held weekly for the months of September-October 2020 using **MS Teams**. The format of this personal training proved to be very effective, with the final few sessions focusing on aspects most relevant to the trainee's needs for a forthcoming stock assessment.

4 NEW FEATURES

All new features implemented into MULTIFAN-CL source code have firstly been added to the development version. Once these features have been tested for their integrity, with no impacts on existing features, then the development version is merged into the release version of the code. The current release version posted is 2.0.7.0. Most of the developments described below are currently implemented in this version, while others have been made to the development version, (since February 2020 and as listed in section 2.4.6), and will be merged to the next release version upon the completion of the benchmark testing.

4.1 Catch-conditioned fishing mortality calculation

Davies et al. (2020) described the development of the catch-conditioned method for estimating fishing mortalities in MULTIFAN-CL. So, only a brief outline is provided here to provide the context for the subsequent advances made and the example presented for its implementation.

4.1.1 Rationale

The existing method for estimating fishing mortalities in MULTIFAN-CL is called the "catch-errors" approach that entails the estimation of a range of effort-related parameters: fishing incident-specific deviates on effort and catchability, and fishery-specific average catchability. Depending upon the model complexity in terms of numbers of fisheries and time periods, this fishing mortality parameterisation can result in a large number of parameters being estimated. For example, a recent complex tuna model, bigeye (Ducharme-Barth et al. 2020) entailed the estimation of around 11,400 independent variables. While the catch-errors approach is effective, and especially in cases where effort is better observed than total catch, an alternative formulation with fewer variables could be an improvement, both in terms of convergence characteristics and ultimately run time. The development of an approach that calculates fishing mortality conditioned upon the observed catches was formulated and implemented in MULTIFAN-CL during 2020, and improvements and testing have since been made.

4.1.2 Methods and Testing

There is a fundamental difference in the dynamic model calculations when using the catch-conditioned method compared to the catch-errors method. With the latter, all the estimated parameters for calculating fishing mortality (catchability, effort deviates etc.) are available at the beginning of the model calculation period, and therefore the numbers-at-age calculation is a linear sequence starting from the first model year. In contrast for the catch-conditioned method, none of the fishing mortalities are available, and these are calculated sequentially for each fishing incident with each time period starting in the first year. The numbers-at-age calculation will therefore include both the dynamic age-matrix and the fishing mortality calculations in each time period. This entailed a fundamental structural development to the model and hence substantial changes to the code.

The method entails four components:

- a method for deriving the initial population at exploited equilibrium, based upon the initial fishing mortalities as specified;
- the internal fitting procedure to estimate fishing mortalities among the fishery incidents within each model time period;

- an internal regression of the estimated relationship between fishing mortality levels and observed effort; and,
- predictions to account for incidents having missing catch and to allow for missing effort.

4.1.2.1 Initial population at equilibrium

Since the average initial total mortality, and hence the survival rate, is not known at the beginning of year 1, a method was developed to allow for the estimation of the survival rate to be used for calculating the equilibrium population within the overall integrated model fit. As described by Davies et al. (2020), a spline formulation was used for the age-specific estimated survival rates with the nodes being the independent variables estimated within the model fit. No substantial modifications were made to this aspect of the feature, aside from rationalising the flags that control the penalty contribution to the function value.

4.1.2.2 Newton-Raphson estimation of fishing mortalities

No substantial changes were made to this component of the feature, except to ensure the selectivity data structures used in Newton-Raphson are robust to the time-variant parameterisation for selectivity.

4.1.2.3 Fishing mortality levels and effort relationship

An orthogonal-polynomial regression model formulated using a Gram-Schmidt (G-S) design matrix and a robustified function is used to fit a fishery-specific relationship between the fishing mortality levels (estimated from the N-R) for those incidents where catch is available, to the observed effort (denoted the **fm_level_rltnshp**). The relationship's parameters are estimated within the overall integrated model fit. The method entails the following algorithm:

- Obtain the fishery and grouping specifications and construct the fm_level_rltnshp parameters
- Build the G-S design matrix for each fishery/grouping – store
- Enter the fitting procedure
- Obtain the first model iteration values for the fm_level_rltnshp parameters
- For incidents with observed catches use the N-R to calculate the fishing mortality levels
- Using the fm_level_rltnshp parameters, the G-S design matrix, and observed effort, predict the fishing mortality levels for the incidents with missing catch
- Calculate the robust mixture distribution function for the fm_level_rltnshp and add to the integrated model fit
- Enter next model iteration
- Repeat iterations until convergence

The first stage of this algorithm was fool-proofed against incorrect fisheries data inputs, and linearly dependent effects in the G-S design matrix. Also, checks were installed for the fm_level_rltnshp regressions to ensure cases of zero observations, and the grouped or ungrouped fisheries are all dealt with in a rational manner. The following examples were explored (grouped and ungrouped cases in each):

- Full effort data; some fisheries have no season effect
- Some fisheries have zero effort; some fisheries have no season effect
- Some fisheries have sparse effort (only 3 or 4 observations); some fisheries have no season effect
- Several fisheries have data in the 1st quarter only; no season effect
- Several fisheries have data in the 1st and 2nd quarters only; season effect estimated

The fitted fm_level_rltnshp was made robust to all these cases, as well as for cases where implausible fishery groupings are attempted. Also, the presence of survey fisheries (see 4.1.2.4) required regulating the Gram-Schmidt design matrix construction to exclude the associated fishery records for which the fm_level_rltnshp is not estimated.

The main improvement made to this component was the formulation of the vectorised student-t regression function. As with the other regression options, this function accommodates the case of grouped fisheries. It is the preferred formulation for these data, and appears to converge more quickly than the mixture distribution formulations.

4.1.2.4 Survey fisheries that provide indices of relative abundance

Typically, certain fisheries (called survey fisheries) provide indices of relative abundance, usually expressed as catch-per-unit-effort (CPUE) and in many fisheries modelling software it is fitted via a dedicated likelihood term. For the MULTIFAN-CL catch-errors model, these indices are expressed as standardized effort, and are included in the integrated model fit via the effort-deviates penalty with the assumption of constant catchability. A different approach is required for the catch-conditioned model, and this was developed during 2020-21.

The survey fishery data are input as a "normal" fishery, but are controlled in how the data are treated in the model based on an activation flag. Two possible configurations of a survey fishery are managed in respect of the how the catch and effort data are employed:

1. Extraction fishery

Catch data are treated as for a normal fishery having substantial fishing mortalities which are estimated via the Newton-Raphson procedure. Effort data are standardised such that the CPUE are indices of relative abundance given assumed constant catchability for the fishery. While the fishing mortalities are substantial, the fishery acts as a survey fishery in that it supplies concurrent indices of relative abundance via the standardised effort data. The standardised effort data are fitted assuming constant catchability within the fm_level_rltnshp regression function.

2. Non-extraction fishery

Catch data are for a "true" survey fishery in that they are negligible, e.g. 1 fish. As such, the fishery is excluded from the Newton-Raphson procedure for estimation of fishing mortalities and the fm_level_rltnshp. Since the catch is arbitrary and low, the model is fitted to the observed CPUE index in a dedicated likelihood function.

In both configurations, the selectivity patterns are estimated via the observed size composition data.

For the first configuration, the fm_level_rltnshp plays the fundamental role in estimating fishing mortality for fisheries with standardised indices of relative abundance. This aspect of the feature was reviewed and various options for the relative weight of the regression for survey fisheries were explored, and an example is presented. The sigma (maximum likelihood estimate of the standard deviation) for the regression is calculated and reported for each fishery.

For the second configuration, the concentrated CPUE likelihood for a given fishery is:

$$0.5\sum_i \log(\gamma_i) + 0.5n\log\left(\sum_i \frac{(O_i - P_i)^2}{n\gamma_i}\right) + 0.5\sum_i \frac{(O_i - P_i)^2}{\gamma_i \hat{\sigma}^2}$$

where

$$\hat{\sigma}^2 = \sum_i \frac{(O_i - P_i)^2}{n\gamma_i}$$

and O_i and P_i are the observed and predicted indices, respectively, for the i^{th} observation, n is the number of observations, and γ_i is the standard error of O_i. The γ_i are input via the effort penalty weights in the fishery data (in the *.frq file). This method has been tested in proof-of-concept for an example having five survey fisheries, that has produced a stable minimisation, and a converged well-determined solution. However, it is not presented here given that further work on this component of the feature is required.

4.1.2.5 Preliminary testing

This feature has been applied to a range of examples: skipjack (Vincent et al. 2019); bigeye (Ducharme-Bath et al. 2020); and, albacore (Castillo-Jordan et al. 2021), for trialling and debugging the feature's development. However, for complex examples it is time consuming to work through the data structures. A simpler example was selected that will expedite the development having the following aspects:

- only 2 regions and a low number of fisheries
- uses the standard catch-errors method for fishing mortality estimation (for the purpose of comparison)
- several index fisheries having CPUE time series
- is a well-determined solution, i.e. positive definite hessian (PDH) solution

The Southwest Pacific swordfish example (Takeuchi et al. 2017) was selected, having all these aspects. Using the example, three variations of the model parameterisations are presented to illustrate the effect of the catch-conditioned method and exploring models with relatively fewer parameters.

- **catch_errs** – fishing mortality calculation using catch-errors method (the original example)
- **ccond** – fishing mortality calculation using catch-conditioned method, with the parameterisation:
    - catch-conditioned parameters:
        - fm_level_rltnshp: students-t distribution; degrees of polynomial = 5
        - spline degrees = 7 for initial equilibrium survival function
        - Penalties applied on higher-level fm_level_rltnshp and the initial population
    - Recruitment: mean+deviates (standard method)
- **ccond_orthp** – fishing mortality calculation using catch-conditioned method (as for ccond), with the parameterisation:
    - Recruitment: orthogonal-polynomial parameteristion, with polynomial orders for the year and region levels only

Each model was fitted to convergence (gradient tolerance of 1.e-06), and the Hessian calculated. Diagnostics of the Hessian were performed and the number of negative eigenvalues determined.

Catch-conditioned method diagnostics

Diagnostics of the catch-conditioned method, confirm it is working as intended. The predicted catches for incidents with observed catches are accurate (Figure 1), and for incidents having missing catches the predictions from the fm_level_rltnshp are plausible. The estimated fm_level_rltnshp regressions have reasonable error, are consistent with the observed trends and with estimated seasonality (Figure 2). Estimated error for the fisheries with standardized indices are relatively high (sigma is 0.3 to 1.2), which is most likely due to the assumption of constant catchability and observation error. The degree of observation error is illustrated in the comparison of the predicted and observed CPUE values (Figure 3). Nevertheless, the link between the fm_level_rltnshp and fishery-specific catchability is demonstrated using this example.

The estimated initial survival rates for the four movement periods are reasonably consistent with the actual values calculated over the 5-year period specified (Figure 4).

Example model results

The estimated recruitments of the two catch-conditioned models (ccond, ccond_orth) differ from the catch_errs model, with those for the ccond_orth illustrating the “simplifying” effects of the orthogonal-polynomial parameterisation (Figure 5).

Table 3 presents the objective function components, the number of independent variables (parameters) estimated, and a selection of the dependent variables for the three example models. A clear advantage gained by the catch-conditioned method is the significantly fewer independent variables (by 83%). This may be reduced further by applying the orthogonal-polynomial form for the recruitment parameterization (by 91%). The differences in the objective function components between the catch-errors and catch-conditioned examples include: no terms for the effort-deviates and total catch likelihoods; and, a term for the fml_effort_reltnshp, in the case of the catch-conditioned examples. In common to both examples are the size composition terms which are estimated to be very similar. The penalty terms for the effort-deviates and fml_effort_reltnshp are substantially different among the examples, and hence the relative influence of each component upon the integrated model fit, the catch-errors and catch-conditioned solutions are not directly comparable. The dependent variables therefore cannot be expected to be similar. Nevertheless, all solutions appeared to be well-determined, with no negative eigenvalues calculated from the Hessian solutions (Table 4).

Comparisons among the dependent variables (model quantities) for the examples that used the two methods is not valid because the solutions are not directly comparable due to the differences in the objective function components. The examples are presented simply to illustrate the implementation of the catch-conditioned method and that well-determined solutions may be obtained.

4.1.2.6 Further work required

The catch-conditioned feature requires further work before it may be used for production assessments. Areas include:

- fishing impact analysis, in particular the initial unfished equilibrium population
- refinements to fitting non-extraction survey fishery CPUE indices:
  - exclusion from the fml_effort_reltnshp regression term
  - reporting of the CPUE predictions from the survey fisheries
- take correct account of the grouping of fisheries for the estimation of the regression when survey fisheries exist
- Pope's approximation that apportions a fraction of the fishing and natural mortality for a given period. This method is more direct, and may be more efficient.

These will be the priority tasks for 2021-22.

4.2 Positive Definite Hessian (PDH) diagnostics

4.2.1 Rationale

The development stages of this feature were described previously (Davies et al. 2019, 2020), and during 2020-21 its development continued.

To provide a brief background, a positive definite Hessian solution (PDH), having no negative eigenvalues, indicates a well-determined solution, that is without substantial confounding among the independent variables, i.e. it has apparently found the minimum and converged. A non-PDH may be a basis for rejecting a model solution as being ill-determined. However, in such a case, one may ask the question: "When having a low number of very small (near-zero) negative Hessian eigenvalues, what interpretation can be made as to whether the model solution is ill-determined or not."

Two heuristics were developed by Dr David Fournier to assess the relative importance of having several small negative eigenvalues. The diagnostic addresses the question by obtaining an indication as to whether the small negative Hessian eigenvalues have a substantial influence upon the dependent variables of interest taken from the model, e.g. management quantities.

4.2.2 Methods and Testing

4.2.2.1 Positivised Eigenvalue Diagnostic (PED)

This is a diagnostic of the variance of the dependent variables of the non-positive definite Hessian solution. The aim is to gauge the relative effect of a negative eigenvalue on the dependent variables of interest for management advice, e.g. SB/SB_{MSY}.

A range of values (say from 1.e-13 to 1.0) are added to the eigenvalues that "positivize" them. Then the covariance matrix and the standard deviations of the dependent variables are re-calculated at each value in the range. An inspection of the relative effect of the added values upon the standard deviation for a given dependent variable (e.g. a reasonable variable to use: "AdultB/AdultBmsy") provides an indication of the relative influence of the associated independent variable estimates. The diagnostic is:

> If for added values <= abs(negative eigenvalue) there is no substantial change in the stdev(dep_var); one can infer low or no relative influence

This is a bit ad hoc and is done to provide some re-assurance, but it is a good check to make sure that those parameters corresponding to the negative eigenvalues are not too influential on the model dependent

variables of interest, i.e. upon which management advice may be based. This exerts some discretion upon the "mandatory" conclusion often made that a non-positive definite Hessian solution lacks utility.

A proof-of-concept test using the complex bigeye tuna model (Ducharme-Batch et al. 2020) indicated there was some influence on the dependent variable: $SB_{current}/SB_{MSY}$, (Figure 6), showing the large change in the standard deviation that occurs between the added values of 1.e-12 and 1.e-11. This indicated that the independent variables related to the negative eigenvalue have some influence.

4.2.2.2 Empirical evidence of the influence of negative eigenvalues

This method aims to demonstrate that alternatives to the original non-PDH solution may be obtained by applying scalars on the independent variables during the minimization. This would generate empirical evidence that a PDH solution can be obtained with added scaling, that has negligible influence upon the dependent variables of interest.

Taking a non-PDH solution having some negative, or very small, eigenvalues, these would be set to some positive value, e.g. 0.001, and the diagonals of the modified Hessian calculated. The minimisation is then repeated with the independent variables being scaled by the diagonals in each iteration, until convergence is achieved. The Hessian of the scaled solution is calculated and the eigenvalues reported. The dependent variables of interest taken from the re-fitted model (PDH) are then compared with those of the original (non-PDH) solution.

The heuristic is: If the solution with added scalars is PDH, and there is no substantial change in the dependent variables of interest, one can infer low, or no, relative influence of the independent variables estimated that are associated with the small negative eigenvalues.

Using the complex bigeye tuna example (Ducharme-Bath, 2020), the method indicated the heuristic to be true (Table 6). That is, a PDH solution was obtained by re-scaling the independent variables, and there was negligible difference in the dependent variable of interest between the original and the scaled solutions. This suggests the parameters contributing to the negative eigenvalues, had negligible influence on the result.

The outcome of this test will certainly be case-specific, and it is not certain that the re-scaled solution will be PDH, or that the heuristic will be satisfied.

4.3 Parallel Hessian calculation – “stitching” the components

Intrinsic to reporting and diagnosing a solution, is the Hessian calculation itself. For complex models, e.g. bigeye tuna (Ducharme-Bath et al. 2020) having ~11,400 parameters, this may entail up to 70 hours calculation on a single processor. Parallel calculation of the Hessian has long been a feature in MULTIFAN-CL, such that multiple processors may be assigned segments of the calculation, that greatly improves efficiency. For this example, parallel calculation over 15 processors will reduce the time required to 4.5 hours – substantially more tractable in a production setting. However, the task remains to re-assemble, or “stitch”, the respective components into a single file containing the entire Hessian solution over the *nvar* dimensions of the independent variables. External tools have been used historically to do this, but it is cumbersome and prone to error.

During 2020-21, a new routine was added to MULTIFAN-CL that easily and rapidly re-assembles the parallel Hessian files into the single binary file needed for subsequent analyses. Firstly, the existing feature for a parallel Hessian calculation is performed that produces a set of *n* component *.hes files. The routine is activated by the Hessian options flag (parest_flags(145)) by setting a value = 11, together with specific input files obtained from the parallel hessian calculations. The routine then combines all the component *.hes files into the single unified Hessian. Scripts have been drafted for submitting the parallel Hessian calculations, and preparing the inputs needed for the stitching routine. For the large example, stitching the components takes < 1 minute.

4.4 Higher precision calculations

4.4.1 Rationale

Sometimes the function minimizer cannot make any more progress and the calculated Hessian approximation at the current value of the parameters is not positive definite. It is tempting to hope that the solution is close enough so that useful information can be extracted from the current value of the parameters, i.e. for all practical purposes the problem is solved. But this may not be the case, and it is difficult to immediately demonstrate this. One solution would be to employ computer code with additional numerical precision so the model could be encouraged to converge to a point with a positive definite Hessian. Even more encouraging would be if there was no practical difference between these new parameters and the original ones, for this might indicate that the extra calculations are not always needed. The function minimizer used in MULTIFAN-CL relies on function values and first derivative information when attempting to fit the model. If the derivatives are not reliable enough it may lead to non-convergence of the model, and since we are interested in the Hessian, the extension of this consideration to second derivatives is also relevant. Therefore, the capability for compilation at 128-bit was undertaken to investigate this.

4.4.2 Method and experimentation

One of the problems with higher precision (more bits) is that to benefit from it, the corresponding math libraries must also have the additional accuracy required. The capability for compiling MULTIFAN-CL for 128-bit precision was implemented using the **boost C++ libraries** (https://www.boost.org/users/history/version_1_74_0.html), requiring also the compilation of the ADMB support library, and adjustments for the quad-double precision calculations previously achieved with the QD support library.

To address the problem posed above, an experiment was undertaken to determine if the executable compiled with higher precision capability, when starting from the point where the standard 64-bit solution failed to progress further, and that produced a non-PDH solution, could continue the minimization and obtain a fully converged solution. Therefore, a pairwise comparison was done of the 64-bit vs 128-bit executables applied to the same example, the bigeye tuna 2020 model (Ducharme-Bath et al. 2020). The single-precision executable is denoted as **std_precsn** and the high precision executable as **lngdbl_precsn**.

The main characteristics of the lngdbl_precsn minimization (Figure 7) were:

- the higher precision enabled the minimization to progress beyond the point where the 64-bit version failed (i.e. the start point of the 128-bit minimization)
- although at a much smaller scale, the function declined gradually, and the gradients were more stable and were lower.

The vertical dashed green lines in Figure 7 indicate PDH solutions, with that at the end of the experiment being positive definite for both the 64-bit and the 128-bit calculation of the Hessian solution (Table 5). The dependent variable: ratio of current adult biomass to unfished adult biomass, differed by only 0.003% among the two solutions. The experiment suggests that for a highly complex and ill-determined problem, the minimisation is sensitive to the precision of the calculations, and that at higher precision, a PDH solution can be found. Noting however, that the std_precsn solution had only three relatively small negative eigenvalues (Table 5). The experiment was interesting in that it revealed for this complex example, that there were minimal implications of a non-PDH solution on the dependent variables relative to that of the high precision solution.

To ensure the code was completely backwardly compatible with the 64-bit compilation in terms of its performance, pair-wise speed trials (over 50 evaluations) were conducted with respect to the preceding version as follows:

- version.2.0.7.3 - 991.77 secs
- devvsn13_wrkshp - 911.43 secs

indicating no impact on the performance of the code at 64-bit calculations.

This feature is not proposed for production implementation, but rather for one-off experimentation given it is highly computationally intensive (Table 5).

4.5 New movement parameterisations

Trials using models with simplified parameterization indicated that movement coefficients often corresponded with negative eigenvalues, and may therefore often be ill-determined in spatially complex models. Dr David Fournier developed an alternative parameterisation for the movement coefficients as follows.

Consider the simple example of two parameters α and β which determine movement between two regions. Let N_{i1} and N_{i2} be the "density" of fish in the two regions at the beginning of period *i*, while N'_{i1} and N'_{i2} are the densities after movement.

$$N'_{i1} = \frac{1}{1+\alpha+\beta}((1+\beta)N_{i1} + \beta N_{i2})$$

$$N'_{i2} = \frac{1}{1+\alpha+\beta}(\alpha N_{i1} + (1+\alpha N_{i2}))$$

If parameters α = β and $N_{i1} = N_{i2}$ then increasing α and β by the same amount will leave the equation unchanged so that if α and β are parameterized by

$$\alpha = \frac{u+v}{2}$$

$$\beta = \frac{u-v}{2}$$

then the derivative of the expression with respect to u will be 0. Now we claim that a good rule of thumb is that when the derivative with respect to a parameter is (almost) 0, it is often a good idea to isolate it such as in this case using u and v instead of α and β. Since α and β must be non-negative, it is more convenient to use the parameterization

$$\alpha = e^{\frac{u+v}{2}}$$

$$\beta = e^{\frac{u-v}{2}}$$

Preliminary trials have indicated better minimization stability with this new parameterization, but further rigorous testing is required.

An extension to this feature was added that provides the facility to alternate between the parameterisations available in MULTIFAN-CL, when starting with a given solution. A routine was added that rationalizes among the parameterisations to obtain the independent variable values in each that correspond to (almost) the same absolute movement process. These values are reported in the solution *.par file so that they are available should the parameterization be altered before undertaking further model evaluations. This improves efficiency when exploring their relative performance (rather than repeating all phases of an entire "doitall" minimization).

4.6 Natural mortality at age – minimization stability

Instances of minimization instability have been apparent when estimating age-specific natural mortality. This was traced to implausible initial values being specified at the outset of the evaluations, or during the initial control phases. This was rectified by initializing the age-specific deviate M_{age} parameters equal to the mean level in log-space. Also, the bounds of these parameters were adjusted to avoid extreme values being trialled during the initial minimization evaluations.

4.7 Pre-minimisation conversion of recruitment parameterization

Davies et al. (2018) described the implementation of the orthogonal-polynomial parameterization for recruitment in version 2.0.4.0 that facilitates reduced model complexity. During 2020-21 a feature was added that makes it convenient and efficient when exploring alternative recruitment parameterizations to readily

convert from the mean+deviates form to the orthogonal-polynomial form. Similar to the approach described in section 4.5. it provides the facility to change the parameterisation when starting with a given solution, rather than repeating the entire "doitall" minimization employing the orthogonal-polynomial parameterization.

New sections were added to the solution parameter file that contain the absolute temporal-spatial recruitments predicted under both parameterisations, such that these are available at the outset of an evaluation. To convert from the mean+deviate parameterisation for recruitments, a routine is invoked that performs an "inner" minimisation to derive the corresponding orthogonal-polynomial parameters that approximate the absolute temporal-spatial recruitments predicted by mean+deviate parameterisation. This provides a "best" set of initial values for undertaking subsequent evaluations with the revised parameterization.

This feature entails a simple quick step, preceding the minimization, to make it easier to explore alternative and less complex parameterisations, while aiming for a well-determined, i.e. PDH, solution.

An example is presented using the swordfish model (Takeuchi et al. 2017) illustrating the effect of the conversion to the orthogonal-polynomial option that entails substantially fewer parameters (22 instead on 61), and therefore a simplified temporal trend (Figure 8). Note, these are the initial values, and the subsequent minimization to convergence will result in the optimized orthogonal-polynomial parameters that will most likely produce higher temporal variability.

5 ENHANCEMENTS AND BUG FIXES

5.1 Revised content for PDH diagnostic report

The feature that generates a report of the Hessian approximation's extreme eigenvalues was expanded to list a wider range of the independent variables corresponding to each eigenvalue. The independent variables for each eigenvalue were sorted in descending order of their absolute "contribution". An example is shown for the bigeye tuna model (2020), where the two numbers in each set of parentheses are: the index of the independent variable (as listed in the xinit.rpt); and, the relative "contribution" to the eigenvalue. This allows for ready identification of the independent variables contributing most to the confounding that is causing the solution to be ill-determined. The analyst is then guided to the area of the model for which the parameterisation is best revised.

Smallest eigenvalues

-7.7578e-08 (30 -0.373) (31 0.358) (33 -0.355) (28 -0.353)…

4.5226e-12 (198 0.45) (197 -0.395) (119 -0.325) (140 -0.32)…

4.5228e-12 (184 -0.404) (140 0.352) (198 0.31) (183 -0.306)…

5.2 Correction to variance calculations

It was noticed that in very few cases, negative values were being calculated from the Hessian inverse and matrix multiplications for the standard deviation of the dependent variables. It was elected to replace the SVD library routines previously used for this operation, with the Lapack (solve) routines available in the OpenBLAS support library. These are more numerically stable, resulting in no negative values, and just as fast in completing the calculations. A pair-wise comparison with respect to the previous version indicated the: maximum percentage difference in the standard deviation estimates was 0.453%; and, the parameter estimates were identical.

5.3 Additional dependent variables in the *.var

While the variance estimates for individual dependent variables obtained from the fished and unfished (fishing impact) evaluations were available in the *.var report, ratio estimates were not. The dependent variable ratio SB_{latest} /$SB_{F0\_10year\_average}$ is the most recent adult biomass relative to the mean of the unfished values over the 10-year period prior to the latest. This is a significant quantity of interest for management advice. It was

therefore included in the calculations, and the estimates and variances were reported to higher level of precision.

5.4 Orthogonal-polynomial parameterisation

The specification of the number of degrees for the polynomials at each of the four levels in the orthogonal-polynomial recruitment parameterization was different from the total number of parameters being estimated by 1. This was because the mean was being estimated if the number of degrees was 0. This could cause confusion or errors in how this feature may be employed, so the polynomial degrees were transformed to match the number of parameters, with the correction for the degrees being made within the function.

Another potential for confusion was the default case where parest_flags(155) will apply to all four levels if the associated flags for those levels are left at the default value of 0. The degrees of each level must now be set respectively by the corresponding activation flag.

5.5 Predicted tag recaptures – added constant for robustness

During minimization, trial values for the independent variables may result in very small or zero predictions for the predicted tag recaptures. Taking the logarithm of these predictions was causing minimization instability, and therefore a very small constant was added: log(**1.e-5**+mfexp(tagnum_fish(it,ir,ip)). In a deterministic comparison using an example with a large amount of tagging data, it made negligible difference to the tag likelihood. However, it may cause differences in the minimization path taken to convergence, compared to that of earlier versions, for very complex or ill-determined problems.

5.6 Maturity-at-length conversion to age calculations

Part of the feature that facilitates the conversion from maturity-at-length to maturity-at-age internally within the model, allows for spline functions to be used. In very rare cases if the nodes have zero values, the cubic splines may produce very slightly negative values which is counter-intuitive in a biological sense. Therefore, a very small constant (1.e-8) was applied to prevent this instance, and makes no impact on the nodes having values larger than near-zero.

Similarly, for calculation of percent maturity-at-age (pmature) the normalizing to between 0 and 1 used a “smoothed” maximum routine. In some instances, this could produce values slightly larger than 1, which again is counter-intuitive in a biological sense. The `smax1()` function was therefore replaced with the `posfun()` function and which ensured pmature is <0.0 and > 1.0.

5.7 Parameter scaling

The scaling of certain independent variables prior to their input to the minimization routine were altered because it improved the relativity among the gradients. However, this sometimes causes differences in the minimization path taken to convergence for very complex or ill-determined problems. The difference may complicate comparisons among previous and current assessment models. Therefore, the capability to re-instate the scaling to that used in earlier versions (i.e. version 2.0.7.0) was enabled via an activation flag.

5.8 Missing both catch and effort records

A routine was added that checks for spurious records in the fisheries data (*.frq) having both catch and effort missing, i.e. values = -1.

5.9 Fish_flags sanity checks

Several routines were added that review the settings applied to fish_flags for their validity in respect of the assumed grouping among fisheries as specified for a range of features. Screen error messages are displayed and the execution is stopped.

5.10 Age-length data input sanity check

A check was added to identify instances where age-length data is being input (file "age_length.*") with particular samples having no matching fishing incidents in the fisheries input data (file "*.frq"). A screen output error message is displayed and the execution is stopped.

5.11 Penalty to movement coefficients

To improve minimization stability for spatially-structured models with estimation of movement coefficients, a penalty function was added. This prevents trial values for the coefficients decreasing to implausibly low values that may cause minimization instability. It is optionally activated by a flag setting.

5.12 Enhancements/corrections for the multi-species/sexes/stocks feature

5.12.1 Fishing impact calculations using the recruitment multiplier

The feature that applies a multiplier to the recruitments predicted by the stock-recruitment relationship when undertaking fishing impact analyses (zero fishing mortality), was extended for the multi-species/sexes/stocks cases.

5.13 Bug fixes

5.13.1 Fix to unfished adult female biomass

For multi-sex models it was noticed that the calculation of the adult unfished biomass ONLY for the report "plotN0.rep" was not using the multi-species data structure for pmature. This was corrected.

5.13.2 Fix to the plot.rep section for the Kalman filter

When employing the catch-conditioned feature, the section of plot.rep file: "# Kalman filter smoothed catchability by realization (across) by fishery (down)" was containing blank spaces that caused errors when opening the file within the Viewer. A correction was made that doesn't write to this section, and avoids reporting the `grouped_catchability_coffs` that are not estimated for the catch-conditioned option.

5.13.3 Fix to region_pars

In the instance where a fixed value of zero is assigned to the parameter: mean proportion of total recruitments in a region, this caused an arithmetic error. A very small constant (1.e-12) was added to the initial value to prevent execution failure.

5.13.4 Fish to .par input of fish_pars

During the April 2020 workshop, the fish_pars dimension was increased to 250 with a view to storing the implicit_fm_level_regression_pars there. This was later reversed with the constructor restored to 50, however, it was omitted to restore the input for backward compatibility to 50. This was corrected.

6 APPLICATION OF NEW FEATURES

6.1 Spatial variation in growth and movement rates

SC15 reviewed progress for the research recommendations from SC14 for bigeye growth and noted that the following research issue needed to be addressed further, after classifying these research items as short-term (preferably before SC16) and long-term (preferably before the scheduled 2023 stock assessment).

a) *Develop MULTIFAN-CL functionality that can accommodate spatial variation in growth rates and movement between western and eastern Pacific to consider the appropriateness of delineating the two stocks at 150˚W (long-term).*

The capability for employing spatial variation in growth rates was implemented in an example that was presented to the workshop held on 4 – 8 Nov. 2019 Center for Advancement of Population Assessment

Methodology (CAPAM), the discussions of which contributed to a review of the essential features for integrated fisheries stock assessment software (Punt et al. 2020). A multi-stock model was developed using Pacific-wide bigeye tuna data, with unique growth rates estimated for two stocks in the eastern and western parts, respectively (Figure 9).

Biological:

stock-specific growth (retained with movement among regions)

recruitment of each stock occurs in its defined region

movement occurs between the regions (populations in a region are a mix of both stocks)

This implementation used the data structures in MULTIFAN-CL that facilitates multiple species, stocks or sexes such that a unique growth rate can be applied (or estimated) for each partition. Stock 1 was defined as the western stock, and stock 2 as the eastern stock. Unique growth was estimated for each stock (Figure 10). Unique movement (Figure 11), with for stock 1 - around 40% movement out of region 1 into region 2; and for stock 2 - almost all remains in region 2. The purpose of developing this model was to demonstrate in proof-of-concept that such an approach was possible and feasible within MULTIFAN-CL. The movement and growth estimates are not presumed to be accurate, and there has been no diagnostic evaluation of their plausibility. Nevertheless, unique growth and movement estimates were estimated, and this model offers a basis for a closer examination of this approach towards obtaining realistic parameters for a multi-stock and Pacific-wide model.

6.2 Stock assessments for SC17

The features for Hessian diagnostics described in section 4 were employed for developing the albacore and swordfish stock assessments undertaken at SPC-OFP in 2021. During model development, the solutions were assessed for positive definite Hessian approximations, and particularly for albacore, both diagnostics for assessing the implications of negative eigenvalues were applied. The application of these features for each assessment are described in context by Ducharme-Barth et al. (2021), and Castillo-Jordan et al. (2021).

7 FUTURE WORK

The future work plan for the development of new features in MULTIFAN-CL in 2021-22 is presented in Table 2, which is largely the same as provided by Davies et al. (2020) for 2020-21. This is because of the attention given to the new features in 2020-21 as presented in section 4, such that the items in Table 2 were not addressed. Items relating to the support areas of the project have also received little attention. This situation has been considered, and the proposed workplan for 2021-22 presented to the 2021 pre-assessment workshop (Hamer et al. 2021) seeks to consolidate all the new features, and to draft the support documentation. Specifically, it includes:

- Testing the implementation of examples that employ all the new features and refine the I/O and diagnostic reports.
- Achieving a final "baseline" MULTIFAN-CL version 2.0.8.# completed by 30 June.
- From Jul. to Dec. 2021, no further large-scale new developments are planned. Time will be spent to tidy the code; complete a backlog of bug fixes; attend to long outstanding tasks from the bigeye tuna independent review panel recommendations; and any "small-scale" requests in the tasks list.
- Providing support for MSE requirements and improvements.
- Catch up on the remaining documentation required for updating the MFCL Manual.

Consequently, the items in Table 2 will be retained, but will be fit within the context of the above workplan for 2021-22.

The primary focus in consolidating new features will be to complete the catch-conditioned fishing mortality method in conjunction with fitting survey fishery CPUE indices. Other notable developments are: informing growth estimation with tagging data, constraining recruitment and effort deviates, and allowing for long-term and initial tag-loss in tagging data (see section 2.8).

Important developments that have not been addressed in previous years, and that fit with this workplan, relate to the simulation mode for its application to MSE. These are:

- improving the efficiency for using MULTIFAN-CL as an estimation model (EM); and,
- achieving realistic pseudo-observations by incorporating forms of process error such as – autocorrelation in recruitments, random selectivity deviates, and, overdispersion in the probability of tag recaptures.

8 DISCUSSION

The highlights of 2020-21 are several innovative and promising developments from the lead developer, Dr David Fournier, with the broad theme being to: enable analysts to obtain well-determined solutions through reducing model complexity; providing the diagnostic tools to effectively address confounding among the independent variables, and for assessing the implications of non-positive definite Hessian solutions.

The catch-conditioned method offers the most efficient means to reduce model complexity over that of the catch-errors method, with examples demonstrating up to 83% reductions in the number of independent variables. While this method was largely developed and implemented during early 2020, it required revisions, refinements, fool-proofing, and testing on a wide range of examples to reveal the corrections needed. The integration of survey fishery indices of relative abundance within the integrated model while using the catch-conditioned approach has been the focus of developments in recent weeks. This offers the most insightful approach to fitting these data, rather than within the fishing mortality levels:observed effort regression where the relative influence of each set of indices is less obvious. This has been implemented successfully in the code, and applied using the swordfish example having 5 survey fisheries, and which produced plausible solutions. However, as described in section 4.1.2.6 there remain several areas, before it may be presented or used in a production situation. These final work areas will be the priority for the first part of 2021-22.

The challenge of obtaining a well-determined solution (PDH) with a spatially complex population model having a very large number of independent variables (up to 11,400 in recent assessments) cannot be under-estimated. Consequently, the development and testing done in 2020-21 focused on developing, refining and applying the methods for reducing complexity, viz. catch-conditioned approach, orthogonal-polynomial recruitments, and alternative movement parameterizations that improve minimization stability. Cumulatively, these features reduced the number of independent variables considerably, by 91% in one example. Despite these approaches, in many of the examples trialled, non-PDH solutions prevailed. Hence, the development focus shifted to assessing the implications of several near-zero negative eigenvalues, and making such diagnostics more logistically feasible. In one example, the heuristic was satisfied that, there was practically no influence upon the management advice taken from the non-PDH solution. It is hoped that these diagnostic tools, and other tools for transitioning more easily to less complex parameterisations, will enable analysts to strategically explore the spectrum of model complexities for a particular problem, to progress towards a plausible set of well-determined solutions upon which management advice may be offered.

No improvements to the support structures of the MULTIFAN-CL project were made during 2020-21 because the priority was the ongoing development workshop. The lack of attention to this aspect of the project has been consistent over the past 2-3 years due to the rapid pace of new developments, such that: the User's Guide lacks documentation for many of the new features; versions 2.0.7.0 and 2.0.8.0 have not yet been posted on the website; and, few updates to R4MFCL have been made. Also, testing of some of the new features (e.g. new movement parameterization) is incomplete, which makes their utility relative to existing methods unclear. The proposed workplan presented to the 2021 pre-assessment workshop (Hamer et al. 2021) seeks to address this situation, by consolidating the new features in version 2.0.8.0, and to update the supporting documentation (section 7).

Short-term priorities for work in 2021-22 are to complete the primary items currently in progress: the catch-conditioned approach; integrating the survey fishery CPUE likelihood; and, refining the I/O and diagnostic reports for the Hessian diagnostics. Other immediate priorities in the support aspect of the project include updating the User Guide that lacks adequate descriptions for many of the new features in versions 2.0.6.0, 2.0.7.0 and 2.0.8.0.

9 REFERENCES

Castillo-Jordan, C., et al. 2021. Stock assessment of South Pacific albacore. WCPFC-SC17-2021/SA-WP-02, Electronic Meeting, 11–20 August 2021.

Davies, N., Fournier, D., Bouyé, F., and Hampton, J. 2020. Developments in the MULTIFAN-CL software 2019-20. WCPFC-SC16-2020/SA-IP-01

Ducharme-Barth, N., Vincent, M., Hampton, J., Hamer, P., Williams, P. and Pilling, G. 2020. Stock assessment of bigeye tuna in the western and central Pacific Ocean. WCPFC-SC16-2020/SA-WP-03 [REV3], Online, 11–20 August 2020.

Ducharme-Barth, et al. 2021. Stock assessment of Southwest Pacific swordfish. WCPFC-SC17-2021/SA-WP-04, Electronic Meeting, 11–20 August 2021.

Fournier, D.A., Hampton, J., and Sibert, J.R. 1998. MULTIFAN-CL: a length-based, age-structured model for fisheries stock assessment, with application to South Pacific albacore, *Thunnus alalunga*. *Can. J. Fish. Aquat. Sci.* **55**:2105-2116

Hamer, P., Castillo Jordan, C., Ducharme-Bath, N., and Vidal Cunningham, T. 2021. Report from the SPC pre-assessment E-workshop, Noumea, March 2021. WCPFC-SC17-2021/SA-IP-02, Online, 11–20 August 2021

Ianelli, J., Maunder, M., and Punt, A. 2012. Independent review of 2011 WCPO bigeye tuna assessment. WCPFC-SC8-SA-WP-01

Kleiber, P., Fournier, D., Hampton, J., Davies, N., Bouyé, F., and Hoyle, S. 2018. MULTIFAN-CL User's Guide. http://www.multifan-cl.org/

Punt, A.E., Dunn, A., þór Elvarsson, B., Hampton, J., Hoyle, S.D., Maunder, M.N., Methot, R.D., and Nielsen, A. 2020. Essential features of the next-generation integrated fisheries stock assessment package: A perspective. Fish.Res.229 (2020) 105617

Takeuchi, Y., Pilling, G., and Hampton, J. 2017. Stock assessment of swordfish (*Xiphias gladius*) in the southwest Pacific Ocean. WCPFC-SC13-2017/SA-WP-2013, Rarotonga, Cook Islands, 9-17 August 2017.

Vincent, M. T., Pilling, G. M. and Hampton, J. 2019. Stock assessment of skipjack tuna in the western and central Pacific Ocean. WCPFC-SC15-2019/SA-WP-05, Pohnpei, Federated States of Micronesia, 12--20 August 2019.

10 TABLES

Table 1. New features added to MULTIFAN-CL with respect to their state of completion as of July 2021.

Peer review recommendations		
Task	Description	Status of completion
b. Non-uniform size bins	Allow the length bins to be of different widths. One might, for example, want many narrow length bins for the smaller lengths, but fewer but wider length bins for the larger lengths.	Development 0%
c. Long-term tag loss	Allow for long-term and initial tag-loss. Currently initial tag-loss is implemented by reducing the number of animals tagged when inputting data to the model and no account can be taken of long-term tag-loss.	Development 0%
d. Tags inform movement	Include an option which allows the tagging data to inform movement only rather than movement and mortality.	Development 100%; Testing 90%
Other developments		
Task	Description	Status of completion
Catch-conditioned	Catch-conditioned fishing mortality with survey fishery CPUE likelihood, extended for application to multi-species.	Development 90%; Testing 80%
Hessian diagnostics	Reports of eigenvalues, diagnostics of the implications of negative eigenvalues	Development 100%; Testing 100%
128-bit precision	Capability for calculations at higher 128-bit precision	Development 100%; Testing 100%
Recruitment random effects	Estimate the autocorrelation and variance of recruitment deviates as random effects rather than as independent parameters. Attempt to extend this functionality also to effort deviates.	Development 60%; Feasibility has been assessed
Constrain deviates	Apply constraints on the recruitment deviates such that the $\bar{x}$ = 0. Apply also to the effort deviates for fisheries with missing effort data for the complete time series.	Development 10%; Testing 0%
Binned likelihood for tagging data	Implement a censored Gamma likelihood for tagging data that better deals with large numbers of zero observations.	Development 100%; Testing 100%
Tags inform growth	Development of a feature that incorporates size data from tag recaptures to inform growth estimation.	Development 0%; Testing 0%
Self-scaling multinomial with random effects	Consolidate the self-scaling methods for fitting composition data - in particular the revised formulation for SSMULT_like.	Development 100%; Testing 100%
Length-structured model	Simulation project - design experiment; undertake simulations; draft report; assess feasibility for implementation in MULTIFAN-CL code	Development 100%; Testing 100%
Outstanding testing of existing features	Catch-conditioned estimation of fishing mortality	Development 70%; Testing 50%
EM fit only projection pars	For the “assessment” estimation model (EM) embedded in a management procedure, only fit parameters relating to the “new” data provided for the projection time periods, i.e. for effort devs, catchability devs, recruitment	Development 80%; Testing 0%

	devs, while holding all other parameters fixed at the initial values.	
OM size comps	Generate a report of the OM size compositions for projection period without error at the end of the projection period as required for deriving economics-based indicators.	Development 0%; Testing 0%
Turing test	Ensure the quality of pseudo-observations to be made more realistic by: • Including the sel_dev_coffs and eff_devs estimates in applying process error in projection size compositions and effort • Including over-dispersion error in tagging data	Development 0%; Testing 0%
Stochastic projection functionality	• Implement process error in future recruitments with application of the derived autocorrelation coefficient in historical recruitment estimates • Fix a bug in generating inputs for stochasticity in $N_{terminal}$ (more stable method is to use terminal year less 5 as the period for obtaining variance) and eff_devs	Development 0%; Testing 0% Development 100%; Testing 90%

Table 2. New features to be added to MULTIFAN-CL in 2020-21 and subsequent years, and those for which implementation and testing is to be completed.

Peer review recommendations		
Task	Description	Implementation
b. Non-uniform size bins	Allow the length bins to be of different widths. One might, for example, want many narrow length bins for the smaller lengths, but fewer but wider length bins for the larger lengths.	2021-22
c. Long-term and initial tag loss	Allow for long-term and initial tag-loss. Currently initial tag-loss is implemented by reducing the number of animals tagged when inputting data to the model and no account can be taken of long-term tag-loss.	2021-22
d. Tags inform movement	Testing for the case of multi-species/stocks/sexes.	Development 100%; Testing 90%
Other developments		
Task	Description	Implementation
Length-structured model	A length-structured model with a differentiable growth transition matrix – assess feasibility for implementation in MULTIFAN-CL	2020-21
Recruitment random effects	Report on the feasibility of its implementation in MULTIFAN-CL.	2020-21
Constrain deviates	Apply constraints on the recruitment deviates such that the $\bar{x}$ = 0. Apply also to the effort deviates for fisheries with missing effort data for the complete time series.	2020-21
Tags inform growth	Development of a feature that incorporates size data from tag recaptures to inform growth estimation.	2021-22
Self-scaling multinomial with random effects	Complete draft paper for peer review.	2020-21
Recruitment correlates	Region-specific environmental recruitment correlates estimated within the orthogonal polynomial parameterisation for recruitments	2020-21
Movement correlates	Add a time-series structure (e.g. random walk, time blocks or using environmental correlates) to movement coefficients	2020-21

Recruitment deviate penalties	Allow for time-variant penalties on recruitment deviate estimates	2021-22
Tagging multi-sex	Account for instances of differences between size composition the tag releases and the sex-specific populations	2021-22
Region specific SRR	Allow that region-specific spawning biomass is responsible for recruitment within that region. This is consistent with the assumption that stocks may not be truly panmictic. This would estimate region-specific SRRs.	2021-22
Outstanding testing of existing features	- Catch-conditioned fishing mortality for multi-species - Reinstate the original estimation of length-based selectivities as activated by fish_flags(i,26)=3 that integrates the distribution of lengths-at-age in calculating the selectivity-at-age. (single species only) - Multi-sex model projections	Complete development 2020-21 2021-22
Report comments	Add comment descriptions of the selectivity parameter configurations in the output .par and .rep reports	2021-22
EM fits only projection parameters	For the "assessment" estimation model (EM) embedded in a management procedure, only fit parameters relating to the "new" data provided for the projection time periods, i.e. for effort devs, catchability devs, recruitment devs, while holding all other parameters fixed at the initial values.	2020-21
OM size comps	Generate a report of the OM size compositions for projection period without error at the end of the projection period as required for deriving economics-based indicators.	2020-21
Stochastic projection functionality	- implement process error in future recruitments with application of the derived autocorrelation coefficient in historical recruitment estimates - consolidate the generation of stochasticity in effort_dev_coffs	2020-21 2020-21
Turing test	Ensure the quality of pseudo-observations to be made more realistic by: - Including the sel_dev_coffs and effort_dev_coffs estimates in applying process error in projection size compositions and effort - Including over-dispersion error in tagging data	2020-21

Table 3. Comparison among model quantities between the catch-errors (catch_errs) and catch-conditioned models (ccond and ccond_orth). The percentage difference with respect to the catch_errs model is shown (%diff)

Model quantity	catch_errs	ccond	ccond_orthp	%diff_ccond	%diff_ccond_orthp
MSY	8848	16270	15730	83.88	77.78
Ccurr.MSY	1.229	0.668	0.693	-45.66	-43.60
Fmsy	0.161	0.161	0.160	-0.50	-1.12
Fmult	1.165	3.962	3.801	240.09	226.27
Fcurr.Fmsy	0.858	0.252	0.263	-70.60	-69.35
B0	137300	261500	253200	90.46	84.41
Bmsy	54810	101300	98520	84.82	79.75
Bcurr	68907	224694	218852	226.08	217.61
SB0	77940	148900	144100	91.04	84.89
SBmsy	16670	31670	30870	89.98	85.18
SBcurr	23709	105041	100802	343.05	325.17
Bcurr.Bmsy	1.257	2.218	2.221	76.43	76.69
SBcurr.SBmsy	1.422	3.317	3.265	133.21	129.60
obj_bhsteep	11.705	10.068	5.298	-13.98	-54.73
obj_effdev	1101.113	-	-	-	-
obj_totcatch	3.466	-	-	-	-
obj_fmleffort	-	2792.105	2859.372	-	-
obj_lencomp	-16936.827	-16975.141	-16972.378	0.23	0.21
obj_wtcomp	-10361.322	-10349.144	-10340.331	-0.12	-0.20
Obj	26060.426	24380.154	24312.060	-6.45	-6.71
No. parameters	2268	396	206	-82.54	-90.92
gradient	0.0000011	0.0000035	0.0000012	229.62	11.86

Table 4. Comparison of the smallest eigenvalues calculated from the Hessian solutions among the catch-errors (catch_errs) and catch-conditioned models (ccond and ccond_orth).

	catch_errs	ccond	ccond_orthp
Smallest eigenvalue	5.1064e-08	5.3664e-10	1.5847e-10

Table 5. Minimisation characteristics and Hessian diagnostics of the standard 64-bit precision (std_precsn) and the high 128-bit precision (lngdbl_precsn) executables for the complex bigeye tuna example.

	std_precsn	lngdbl_precsn
Negative eigenvalues	3 (-6.8291e-06; -7.54e-08; -5.682e-10)	0
Objective function	2.13083025570223e+06	2.13083025599597e+06
Max.gradient	9.70099350508258e-05	8.00167636660917e-05
Evaluations	-	15936
Iterations	-	11765
Performance per evaluation	7.05 seconds	2.46 minutes

Table 6. Original and re-scaled solution eigenvalues that demonstrates empirical evidence for the influence of negative eigenvalues for the bigeye example.

	Original solution	Re-scaled solution
Smallest eigenvalues	-6.0072e-08	3.3122e-08
	-4.9418e-08	1.6562e-07
	-2.1069e-09	1.6817e-06
% difference SB/SB_{MSY}		0.003%

Table 7. The minimisation solutions for the SKJ2019 example fitted using alternative methods for estimating fishing mortality: the catch-errors (catch_err) and catch-conditioned (catch_cond) methods; with the relative percentage difference with respect to the catch_err model.

	catch_err	catch_cond	%diff
Objective function	-272830.1	-271741.3	-0.40
Maximum gradient	9.795e-04	7.817e-04	-20.19
No. parameters	7602	2351	-69.07
Evaluations in final phase	5942	4459	-24.96

11 FIGURES

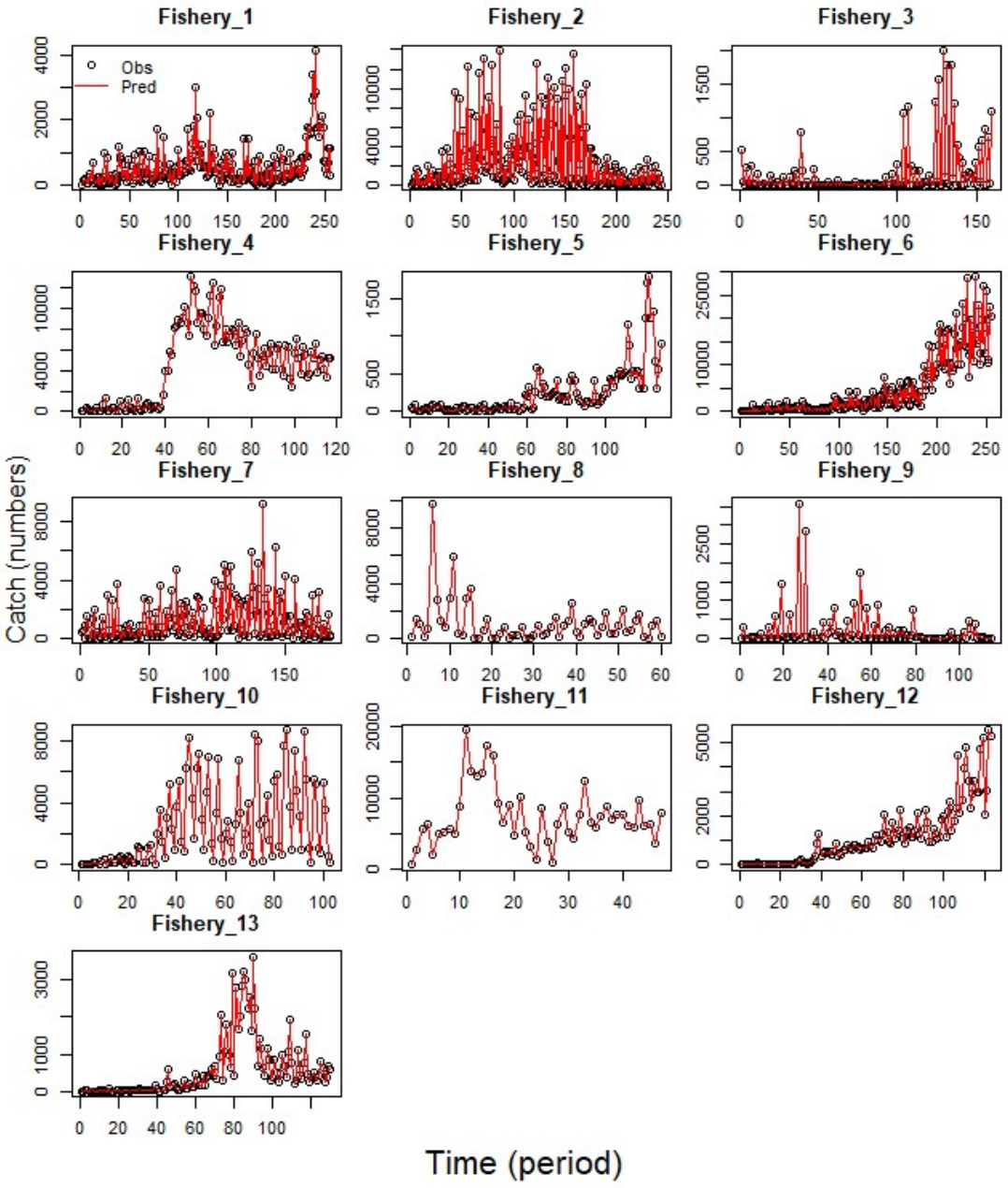

Figure 1. Observed and predicted catches for each fishery or grouping for the model employing the catch_cond feature (ccond).

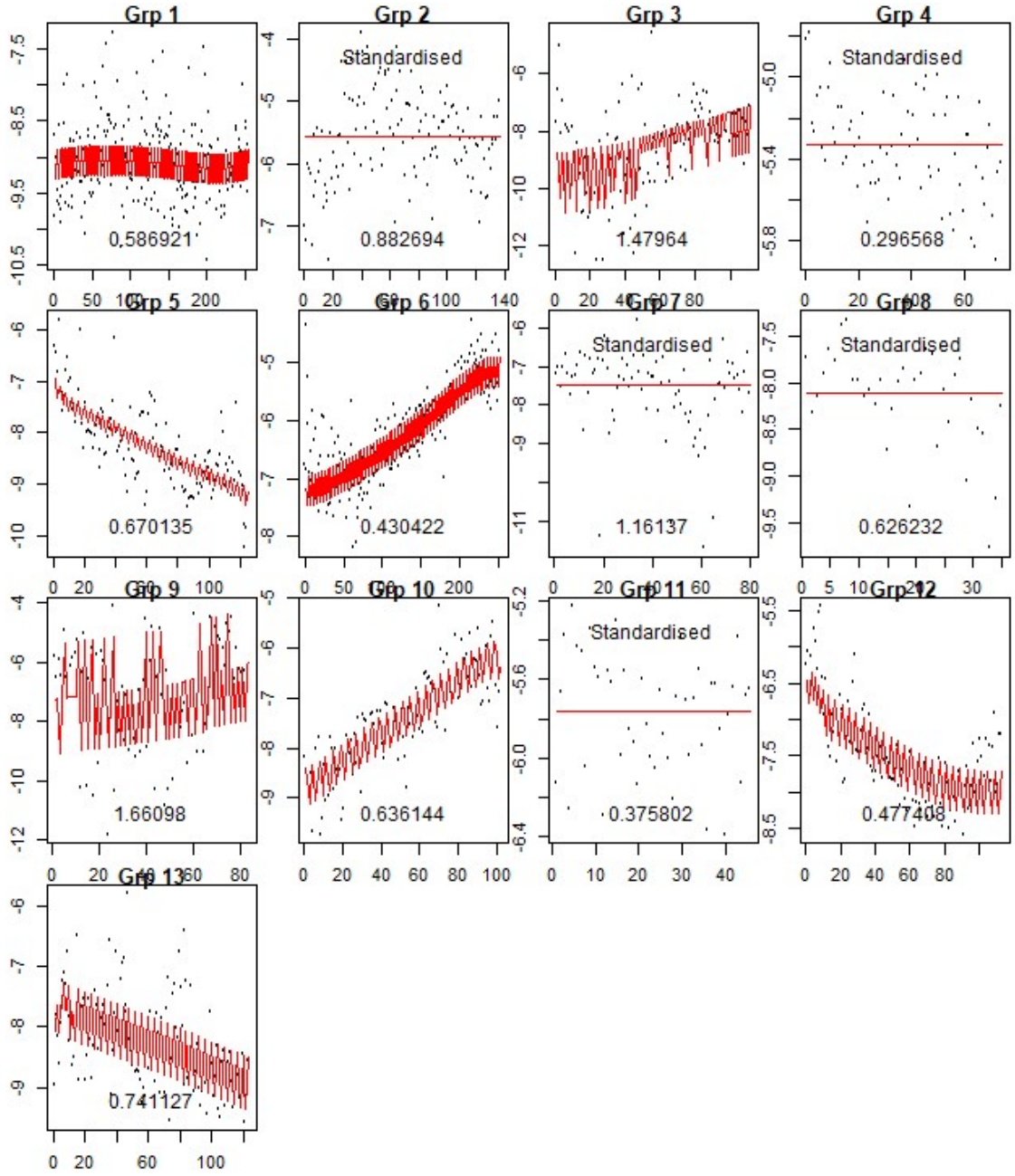

Figure 2. The fitted fm_level:effort relationship specific to each fishery (group) for the model employing the catch conditioned feature (catch_cond), where the red lines are the predictions. Those panels for those fisheries for which standardized effort is available are annotated "Standardised" for which constant catchability was assumed. The sigma value for the relationship is shown in each panel.

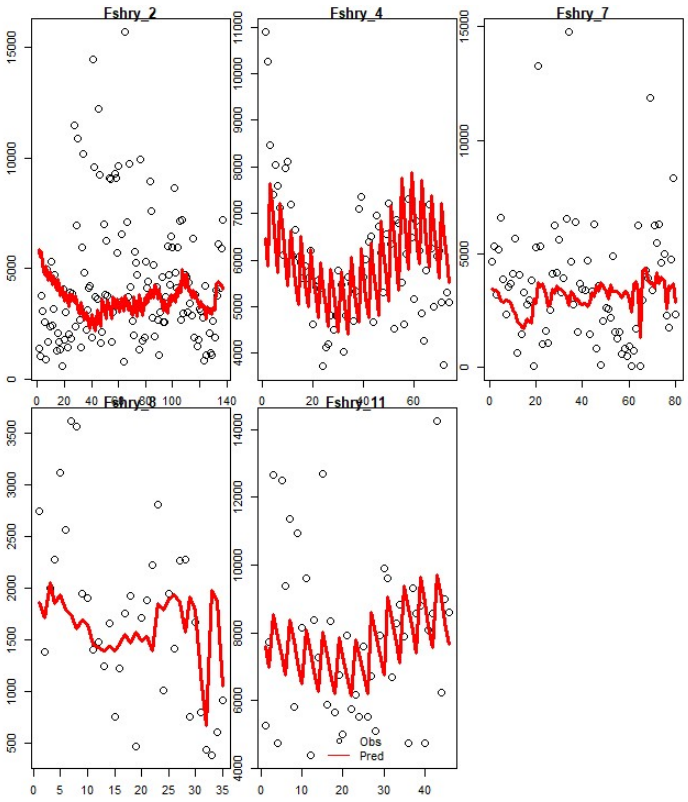

Figure 3. A comparison of the observed (black circles) and predicted CPUE (red lines) trends for each fishery having standardised effort taken from the model employing the catch conditioned feature (ccond).

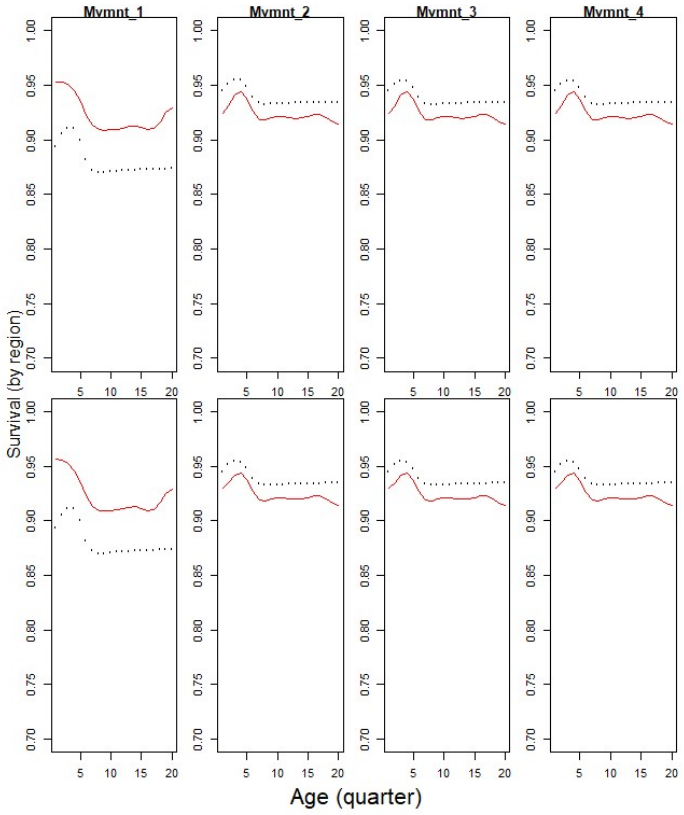

Figure 4. Comparison of the kludged initial survival parameters (splines, red lines) and the actual initial survival as calculated (black points), for the model employing the catch conditioned feature (ccond).

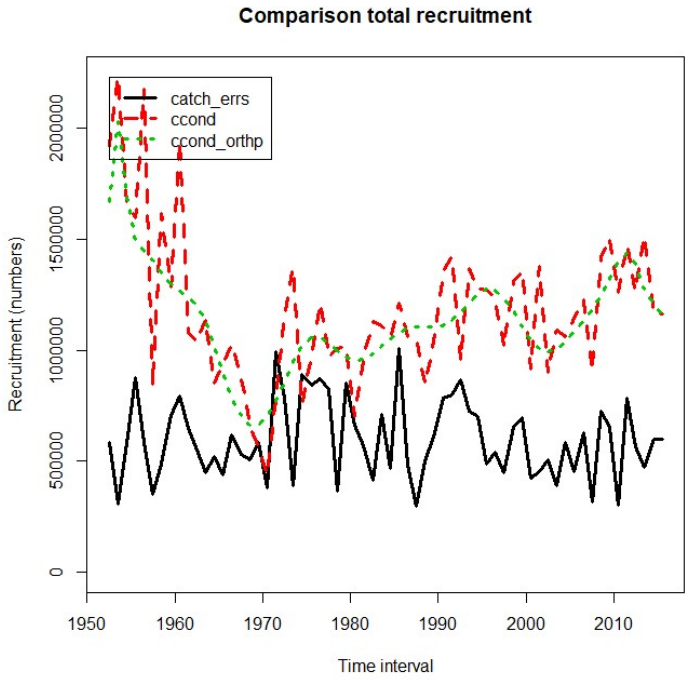

Figure 5. Comparison of annual recruitments between the catch-errors model (catch_errs), and the models employing the catch conditioned feature (ccond, ccond_orth).

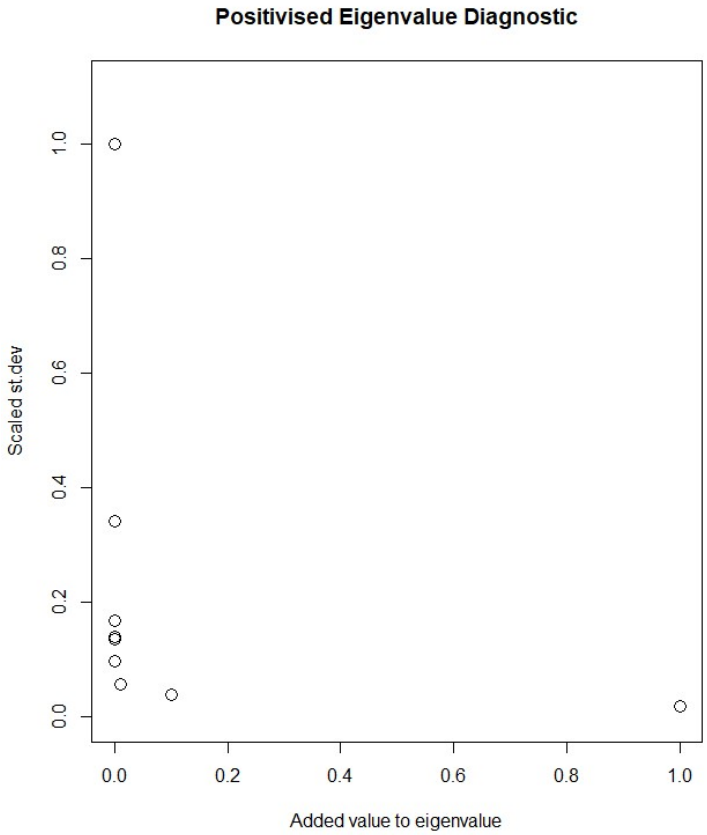

Figure 6. The positivised eigenvalue diagnostic showing the scaled standard deviation of the dependent variable SB/SB_{MSY} as obtained when fixed values are added to the associated negative eigenvalues.

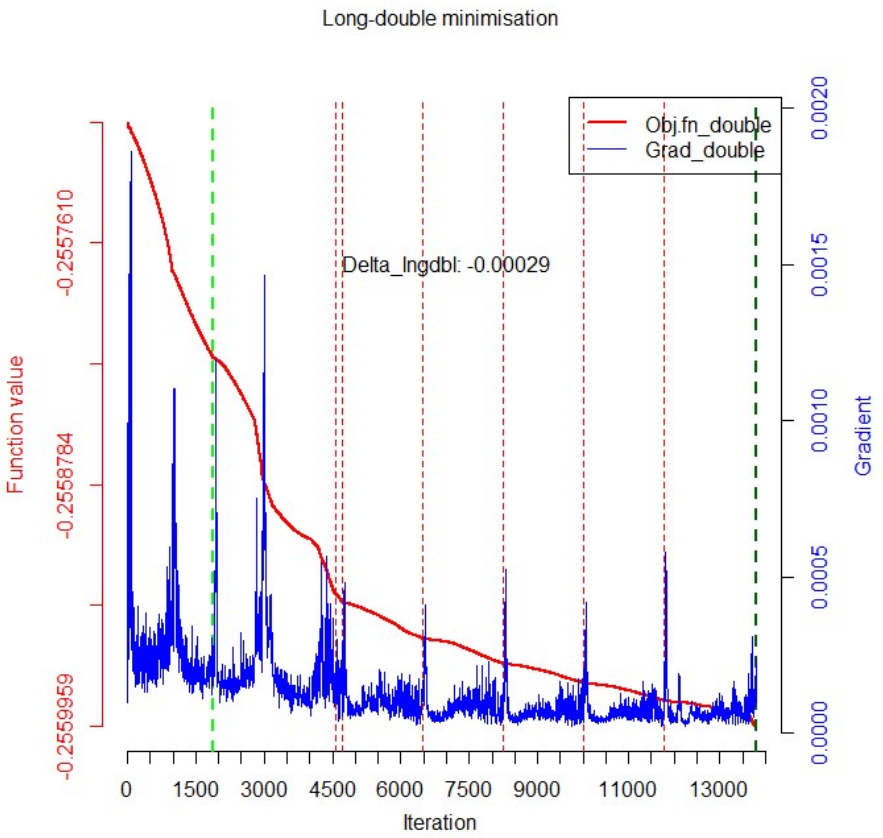

Figure 7. Minimisation characteristics of the 128-bit executable obtaining a solution for an initially ill-determined model, showing at each iteration the objective function value (red line), and the maximum gradient (blue line). The vertical dashed lines indicate points at which the Hessian solution was evaluated for negative eigenvalues. The absolute difference in the objective function value from that at the start of the minimisation is shown (Delta_lngdbl).

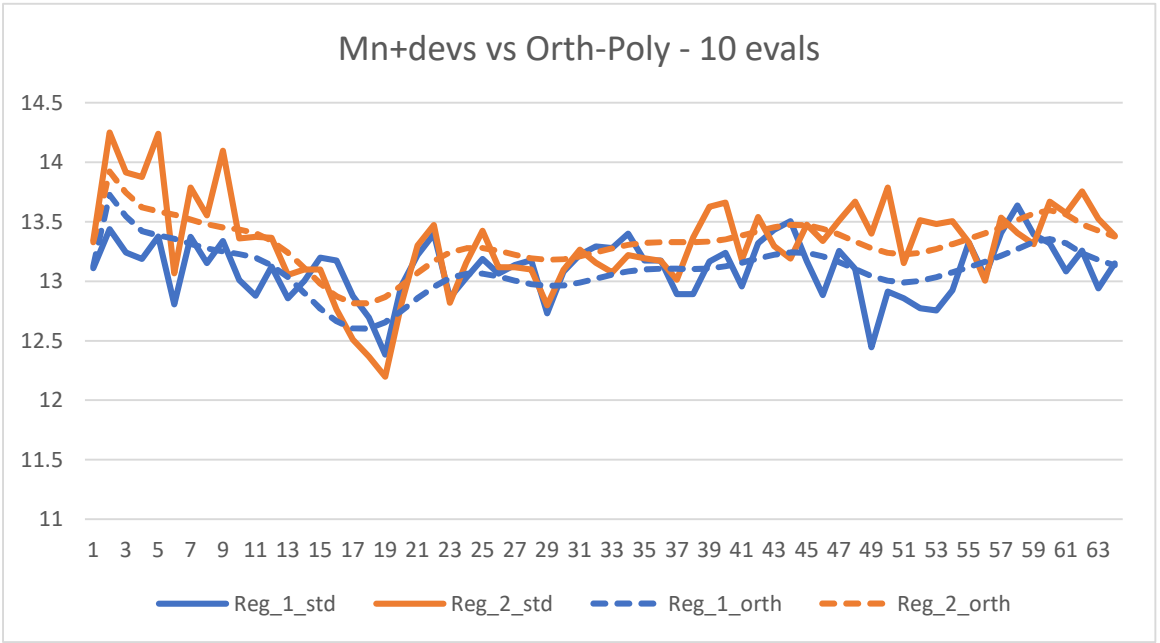

Figure 8. Actual recruitment estimates by model "annual" time period for the mean plus deviate parameterisation (Reg_1_std, Reg_2_std), and that obtained using the pre-minimisation conversion to the orthogonal-polynomial parameterisation ((Reg_1_orth, Reg_2_orth).

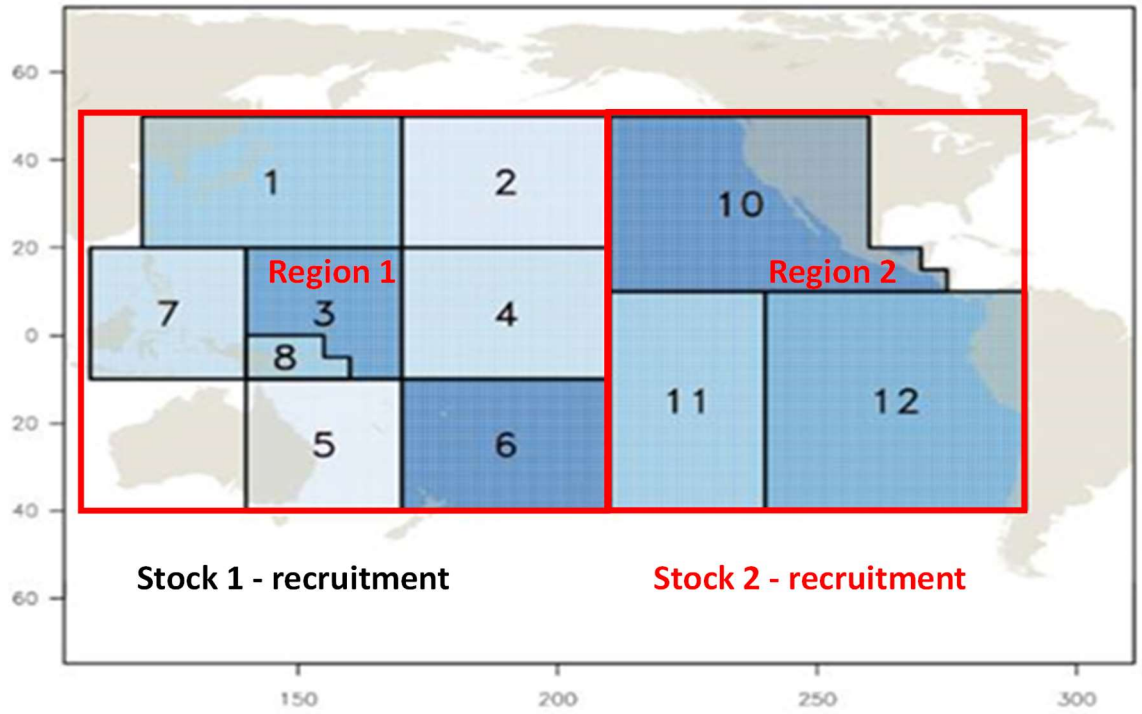

Figure 9. Delineation of Eastern and Western stocks of the Pacific-wide bigeye tuna population as defined in a multi-stock model.

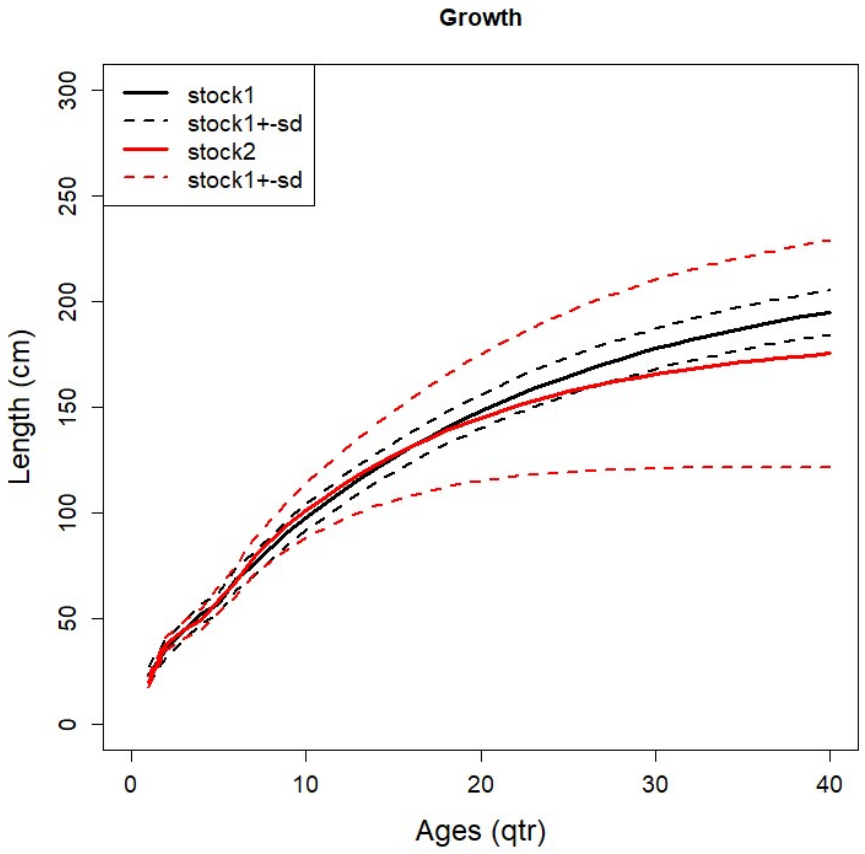

Figure 10. Estimates of growth for the western (stock 1) and eastern (stock 2) stocks obtained from the multi-stock Pacific-wide bigeye tuna model.

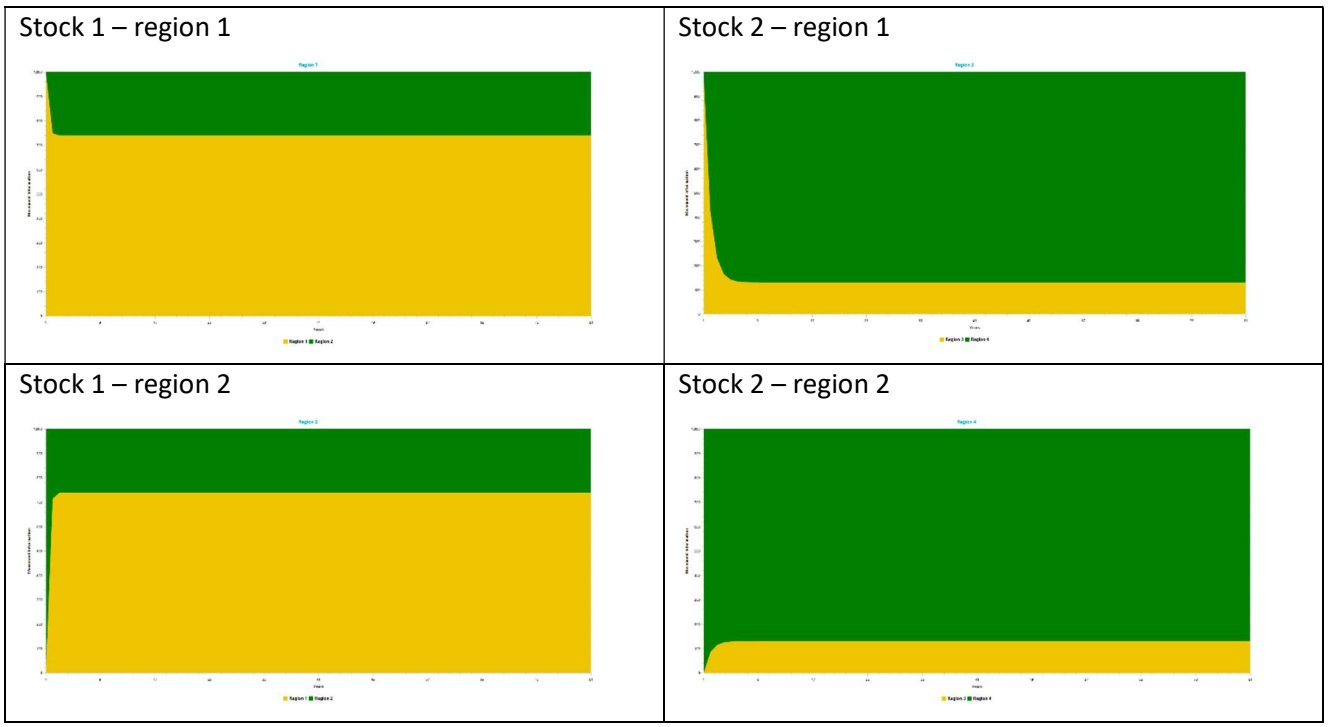

Figure 11. Movement estimates for the multi-stock model. Stock 1: region 1 - yellow; region 2 – green. Stock 2: region 1 - yellow; region 2 – green.

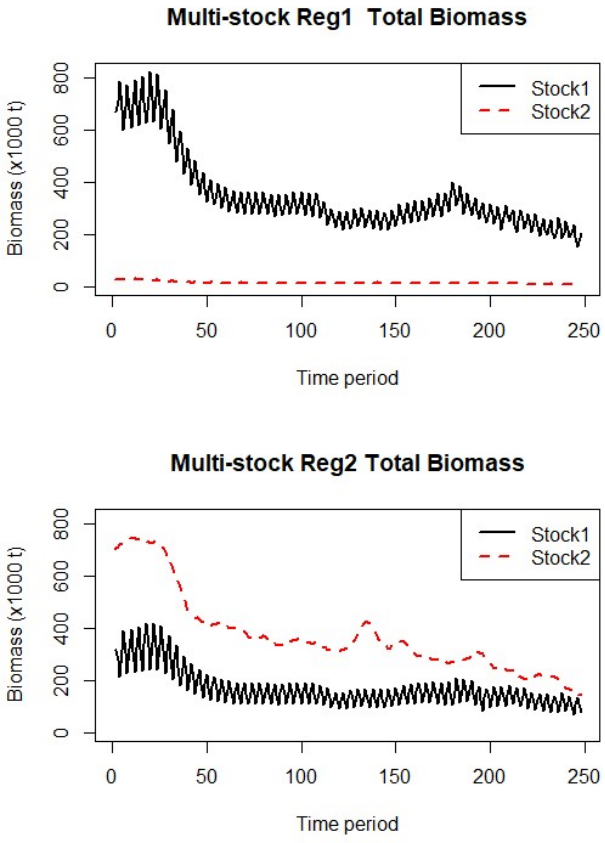

Figure 12. Biomass estimates for the multi-stock model for each of the two stocks in regions 1 and 2.

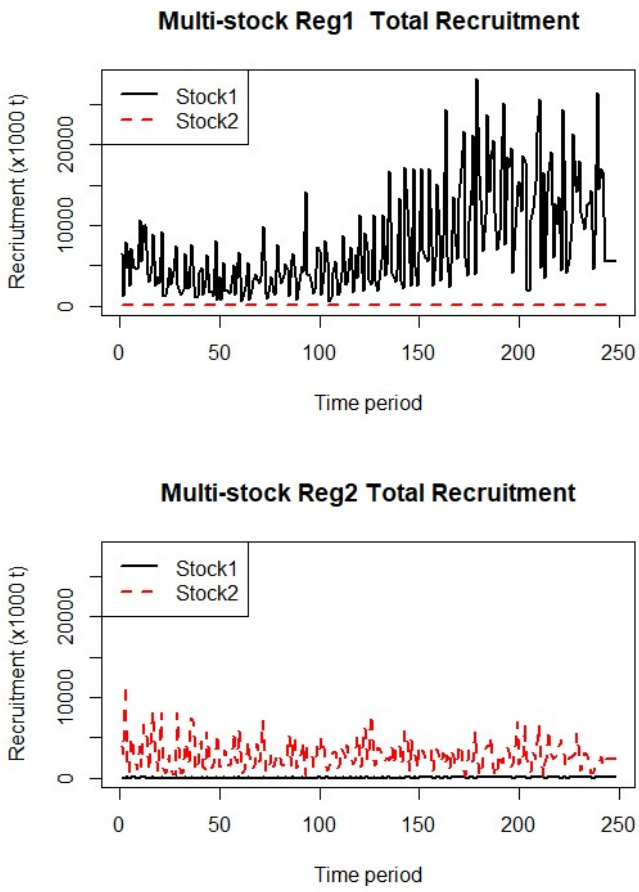

Figure 13. Stock-specific recruitment estimates in each of the two regions of the multi-stock model.