Fisheries Integrated Modeling System (FIMS) Test Plan

Integration tests for FIMS version 0.0.0.9

13 March, 2023

Contents

1 Integration test: model comparison project case 0 operating model 1

1.1 Test ID

integration_test.mcp_c0_om1_works

1.2 Test objectives

The objective of the integration test is to validate the FIMS C++ operation as a whole. At the conclusion of testing, the project team will have a high level of confidence that FIMS population module will work according to user requirements. The integration test of the FIMS (version 0.0.0.9) should validate from the requirements below:

- FIMS C++ modules run together correctly.
- FIMS C++ implementation work on operating systems Windows-2019, Mac OS-latest, and Ubuntu-latest.
- The current FIMS can replicate age-structured modeling requirements from Li et al. (2021) operating model and FIMS deterministic run can produce reliable outputs (e.g., spawning biomass and numbers at age).

1.3 Approach

- Prepare expected values using R script FIMS_integration_test.R from Age Structured Stock Assessment Model Comparison repository (ASSAMC).

- Run `FIMS::setup_gtest()` to create om_input1.json and om_output1.json files and save the files under `FIMS/tests/integration/inputs/` folder.

- Write test cases in `FIMS/tests/gtest/integration_test_populatin.cpp` to validate components to be tested.

- Run C++ tests locally using ctest executable from CMake test driver program or `FIMS::run_gtest()` in R. Make sure other existing C++ unit tests still work after making changes to FIMS code repository.

- Push tests to the working branch and run tests through GitHub Actions.

- Submit a pull request for code review.

Currently, FIMS only runs one deterministic integration test case 0 (C0). Here the case 0 sets standard deviations of recruitment in log space to 0.0. All other operating model settings are the same as case 0 in Li et al. (2021).

This workflow (Figure 1) uses automated testing to quickly execute test cases. The workflow includes converting operating model data (.RData) from Age Structured Stock Assessment Model Comparison repository to .json files using setup_gtest() function in FIMS, running FIMS population module, and validating FIMS output values with "true" values from the operating model.

```
                          ╔═══════╗
                          ║ Start ║
                          ╚═══╤═══╝
                              │
┌─────────────────────────────────────┐
│ ASSAMC GitHub Repository             │
│                                      │
│ • ASSAMC/FIMS_integration_test_data  │
│ • FIMS_integratin_test.R             │
└─────────────────────────────────────┘

                    run OM

┌──────────────────┐  ┌──────────────────┐
│ C0               │  │ C1               │
│                  │  │                  │
│ • OM1.RData      │  │ • OM1.RData      │
│ • OM2.RData      │  │ • OM2.RData      │
│ • ...            │  │ • ...            │
│ • OM100.RData    │  │ • OM100.RData    │
└──────────────────┘  └──────────────────┘

        ┌──────────────────────────┐
        │ FIMS GitHub Repository   │
        │                          │
        │ • FIMS/R/setup_gtest.R   │
        │ • setup_gtest()          │
        └──────────────────────────┘

                convert .RData to .json

┌─────────────────────────────────────┐  ┌─────────────────────────────────────┐
│ C0                                  │  │ C1                                  │
│                                     │  │                                     │
│ • om_input1.json om_output1.json    │  │ • om_input1.json om_output1.json    │
│ • om_input2.json om_output2.json    │  │ • om_input2.json om_output2.json    │
│ • ...                               │  │ • ...                               │
│ • om_input100.json om_output100.json│  │ • om_input100.json om_output100.json│
└─────────────────────────────────────┘  └─────────────────────────────────────┘

                run FIMS population module

        ┌──────────────────────────────┐
        │ FIMS and OM Comparison       │
        │                              │
        │ • Unfished numbers at age    │
        │ • Unfished biomass           │
        │ • Unfished spwaning biomass  │
        │ • Biomass                    │
        │ • Spawning biomass           │
        │ • Fishing mortality at age   │
        │ • Total mortality at age     │
        │ • Expected catch             │
        │ • Expected index             │
        │ • Unfished numbers at age    │
        └──────────────────────────────┘
```
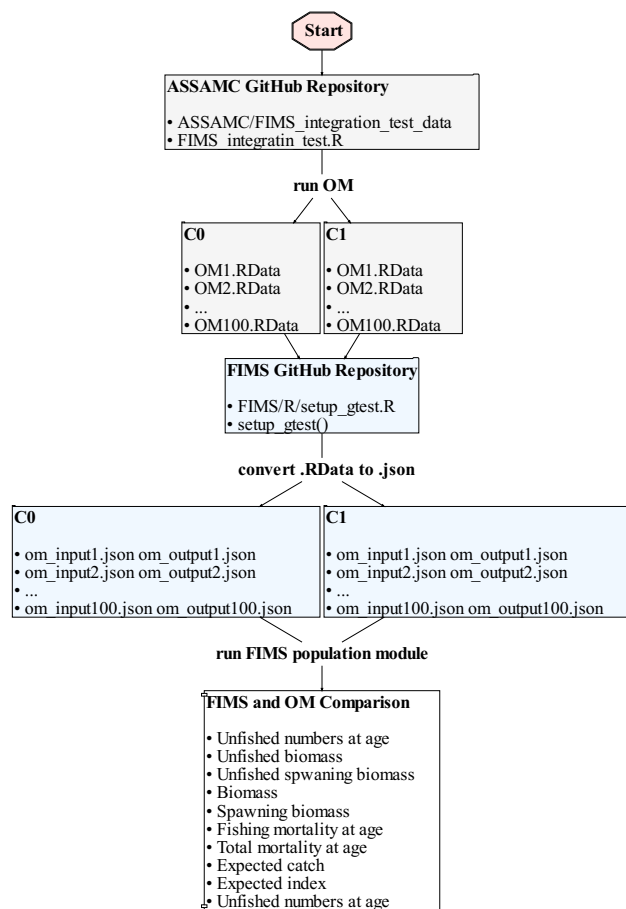
Figure 1: FIMS integratino test workflow.

1.4   Evaluation criteria

The integration test validates following components (Table 1)

- Unfished numbers at age
- Unfished biomass
- Unfished spawning biomass
- Numbers at age
- Biomass
- Spawning biomass
- Fishing mortality at age
- Total mortality at age
- Expected catch
- Expected index

1.5   Test deliverables

- Test logs can be found on GitHub Actions page. All tests have passed.

- C++ test coverage from codecov: `codecov 65%`

  - FIMS modules under FIMS/inst/include have been 100% tested by either C++ unit test or itegration test.
  - 45.6% of code from FIMS/tests/integration has been tested.
    * Only 39.71% of code that is from `rapidjson` has been tested. It can be removed from FIMS code coverage calculation since `rapidjson` is a third-party library.
    * `integration_class.hpp` has a code coverage of 72.02%. The test coverage can be improved by adding another integration test using the double logistic fleet data from Li et al. (2021).

- Current integration test only covers one iteration run of one case of the operating model from Li et al. (2021). Value-parameterized testing can be used to test FIMS code with different parameters without writing multiple copies of the same test.

- Current test data files are stored in FIMS. An R data package may need to be developed to conduct integration tests.

  - The data package can be developed for the purpose of distributing test data, not only for FIMS, but also a broader stock assessment model development community. It can be used to share test data across multiple tools.
  - It can store data in multiple formats (e.g., .RData and .json).
  - It is a useful technique for getting relatively large and static files out of FIMS, which is a more function-oriented package that might require more frequent updates.
  - The test data can be better documented following R data pacakge documentation guides.
  - The origin story of data can be preserved in the data package. The code used to make the cleaned up version of raw data can be included in the data package. This makes it easy to update or reproduce various version of the test data.

Table 1: Evalation criteria for FIMS testing components.

Component	Test
Unfished numbers at age (nyears + 1) x nages	- Individual elements (including numbers at age from nyear + 1) are greater than 0 - Individual elements are less than unfished recruitment from OM (OM only has an unfished recruitment value)
Unfished biomass nyears + 1	- Individual elements (including numbers at age from nyear + 1) are greater than 0 (No corresponding value from OM)
Unfished spawning biomass nyears + 1	- Individual elements (including numbers at age from nyear + 1) are greater than 0 - Individual elements are less than unfished spawning biomass from OM
Numbers at age (nyears + 1) x nages	- Absolute relative error of inidividual elements (|FIMS - OM| / OM x 100) is less than 1.0% - Absolute error of individual elements (|FIMS - OM|) is less than 50 fish - Individual elements (including numbers at age from nyears + 1) are greater than 0
Biomass nyears + 1	- Absolute error (|FIMS - OM|) of individual elements is less than 1.5 mt - Absolute relative error (|FIMS - OM| / OM x 100) of individual elements is less than 1.0% - Individual elements (including numbers at age from nyears + 1) are greater than 0)
Spawning biomass nyears + 1	- Absolute error (|FIMS - OM|) of individual elements is less than 1 mt - Absolute relative error (|FIMS - OM| / OM x 100) of individual elements is less than 1.0% - Individual elements (including numbers at age from nyears + 1) are greater than 0
Fishing mortality at age nyears x nages	- Absolute error (|FIMS - OM|) of individual elements is less than 0.0001 - Individual elements are greater than 0
Total mortality at age nyears x nages	- Absolute error (|FIMS - OM|) of individual elements is less than 0.0001 - Individual elements are greater than 0
Expected catch nyears	Fishing fleet: - Absolute error (|FIMS - OM|) of individual elements is less than 1 mt - Absolute relative error (|FIMS - OM| / OM x 100) of individual elements is less than 1.0% - Individual elements are greater than 0 Survey fleet: - Individual elements equal to 0
Expected index nyears	Fishing fleet: - Individual elements of catchability equal to 1 - Individual elements of expected index are greater than 0 Survey fleet: - Absolute error (|FIMS - OM|) of individual elements is less than 0.0001 - Absolute relative error (|FIMS - OM| / OM x 100) of individual elements is less than 5.0% - Absolute error (|FIMS - OM|) of individual elements of catchability is less than 1.0e-07 - Individual elements of expected index equal to 0