6 Supplemental Information

6.1 Installation and requirements:

The code is entirely programmed in C++, and can be downloaded from gitlab.com/Moonfit/MoonLight.

Running Moonfit requires a C++ compiler and the Qt library to be installed. No external library is required (some are included as C++ files directly inside the folder) [3]. It is possible to use boost solvers, in which case the boost library should be installed.

- *In Linux:* g++ is recommanded (sudo apt-get install g++). The QT framework can either be installed as complete version (sudo apt-get install qt-sdk and sudo apt-get install qtcreator), or as core libraries one by one (qtbase5-dev, libqt5svg5, libqt5printsupport5, qtcreator). If boost is wanted, (sudo apt-get install libboost-dev).

- *In Windows:* A full version of Qt can be downloaded from https://www.qt.io/download, by choosing the Open Source version. More specifically, some offline installers available in https://www.qt.io/offline-installers/ include C++ compilers like MinGW or VisualC++. We recommend the most recent 'Qt for windows 32 (MinGW)', that works on both 32 and 64 bits systems. For boost, download the latest version (http://www.boost.org/users/download/) and unzip it anywhere. See below how to link it.

- *In MAC:* The clang C++ compiler can be installed with brew by running (brew install –with-clang llvm) on command line, and the QT platform can be downloaded from https://www.qt.io/download, by choosing the Open Source version.

Several files have been developed together with the software 'organism' (Developed at CBBP Lund / SLCU Cambridge, see gitlab.com/slcu/teamHJ/organism), which is able to define and simulate and optimize ODEs in a multicellular environment from ODEs defined in a text file, together with mechanical forces and cell population rules (e.g. [21]).

Running examples: The main folders of the code include: Docs for documentation, Extreminator for the built-in optimizers, Framework for the minimal classes to simulate a model, Interface for the graphical interface, NewProject as a guidelines to create new models, and Examples that include small models and respective datasets ready to fit. To run the examples, two ways are possible:

1. *with QtCreator:* open Examples.pro with QtCreator, click on 'configure project' if it is opened for the first time here, and then compile and run the code by CTRL+R or by clicking on the run green button. 'Release' mode is preferred. The interface will open directly.

2. *Through Manual compiling:* to this end, open a terminal in the folder of Examples.pro, and execute the commands: qmake Examples.pro (generates a makefile) and make. The executable file is created in an automatically generated folder ('build...') from the parent folder, and can be run from there. Note: on windows, the compiler might have a different command for make, such as mingw32-make, and it might be necessary to add the path of the compiler binaries into the system path (right-click on 'this computer', 'properties', 'advanced system properties', 'environment variables', and inside the 'path' field, add '; folderOfTheCompiler'.)

When launched, a list of different examples is proposed (see Examples/Examples.pdf), leading to the start of the graphical interface (Figure 2).

[3] The library libSRES [20] is included for SRES optimization, and CMA-ES optimization can be used by uncommenting the content of the CMAES.cc file and installing the C++ shark library. The library QCustomPlot is included in the graphical interface to display plots.