Accelerating a QSP Model of Nonalcoholic Steatohepatitis (NASH) Using the Julia Language

Matthew McDaniel[1], Grant Genereaux[2], Francisco Huizar[1], Corey Berry[1], Scott Q. Siler[1]
[1]Quantitative Systems Pharmacology Solutions, Simulations Plus, Inc., Research Triangle Park, NC
[2]Bristol Myers Squibb, Princeton, NJ (current affiliation; was an employee of Simulations Plus at the time of this work)
Contact: matt.mcdaniel@simulations-plus.com

OBJECTIVE

NAFLDsym® is a quantitative systems pharmacology (QSP) platform that simulates progression and treatment of nonalcoholic fatty liver disease (NAFLD) and nonalcoholic steatohepatitis (NASH)[1]. The current NAFLDsym version (v2A) was designed in MATLAB, a choice motivated by MATLAB's ubiquity in numerical computing and its support for graphical user interface (GUI) development. MATLAB also enables code obfuscation via "p-files"[2], enabling protection of proprietary code while maintaining public-facing QSP model equations that users can modify for simulation. However, MATLAB apps incur significant time overhead due to runtime compilation. This cost can be problematic when running large population simulations (>1000 patients) at the longer time scales of NASH treatment. Our objective, therefore, was to make NAFLDsym more computationally efficient.

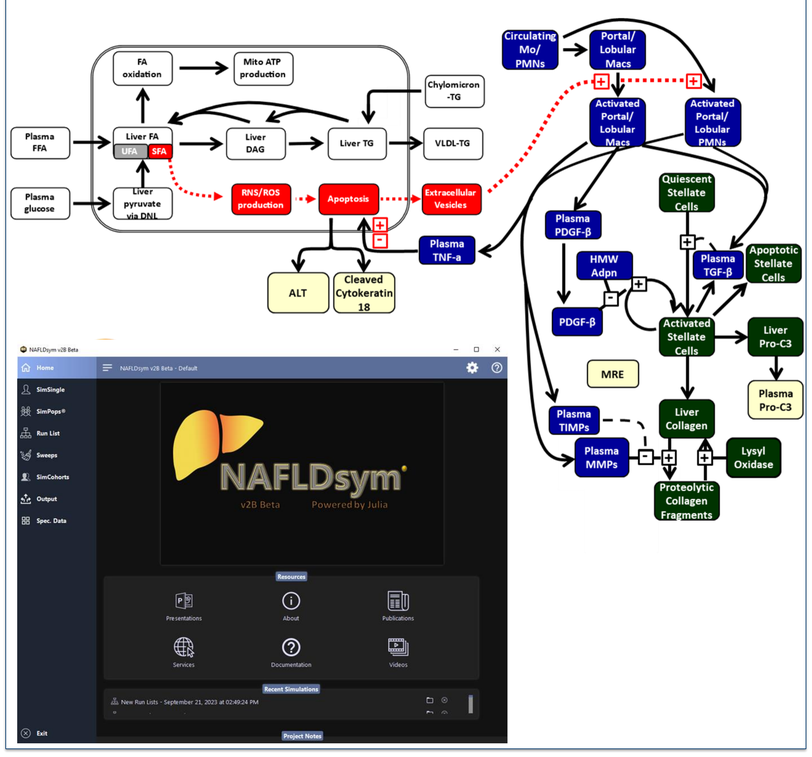
METHODS

The NAFLDsym GUI functionality was first migrated into a separate C++/Qt application. Doing so produced both a faster and a more visually appealing interface. All other engineering and model-related code was then converted to Julia. Julia was selected because it is a dynamic language that was designed for highly performant numerical computing[3]. This emphasis on performance is particularly evident in the Julia Scientific and Machine Learning (SciML)[4] kit, which maintains a library of highly configurable differential equation solvers and optimization tools. Furthermore, Julia code can be compiled to a binary executable via the Package Compiler library[5]. Using this library enabled the continued separation of the public QSP model from the private proprietary code with the added benefit of minimizing runtime compilation. Taken together, the C++/Qt frontend and Julia backend comprise the new NAFLDsym v2B.

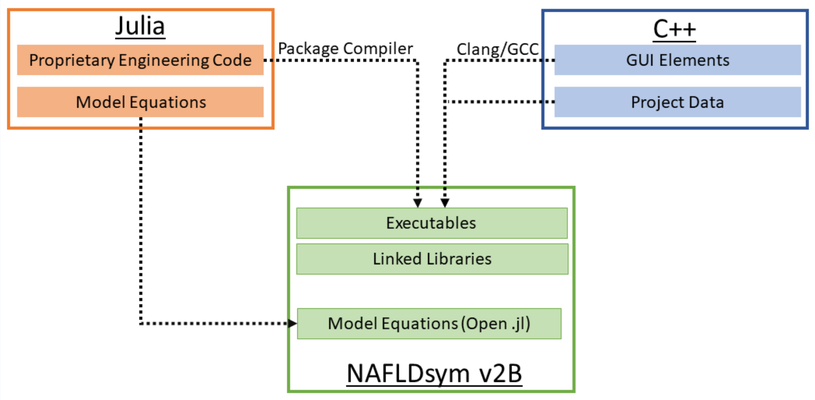
Three simulation configurations were run to compare the performance of NAFLDsym v2A and v2B: All NAFLDsym SimPops® patients (n = 1673) for 24 hours, NASH SimCohorts™ of 100 patients for 1 year, and NASH SimCohorts of 50 patients for 3 years; simulated patients were untreated. All simulations were run on a Windows 10 system with 20 cores (Intel Xeon 2.4 GHz) and 32 GB of RAM.

RESULTS

NAFLDsym Overview Diagram and GUI

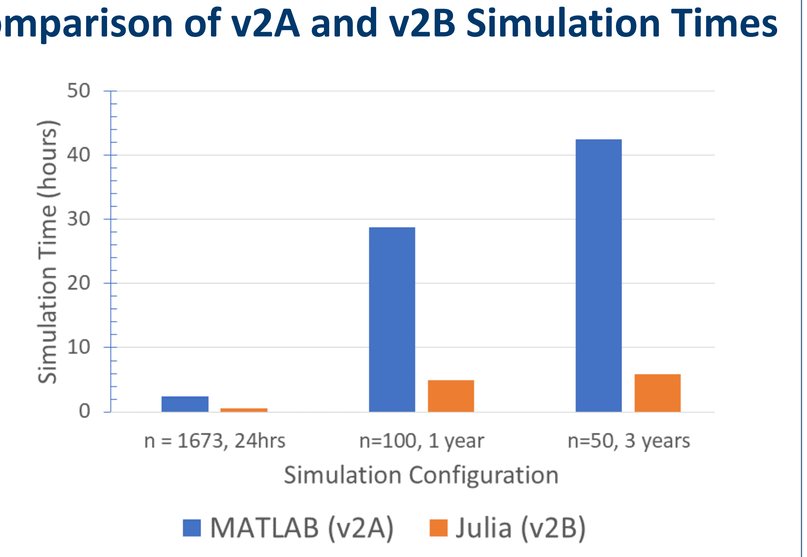


NAFLDsym Version 2B Design Strategy



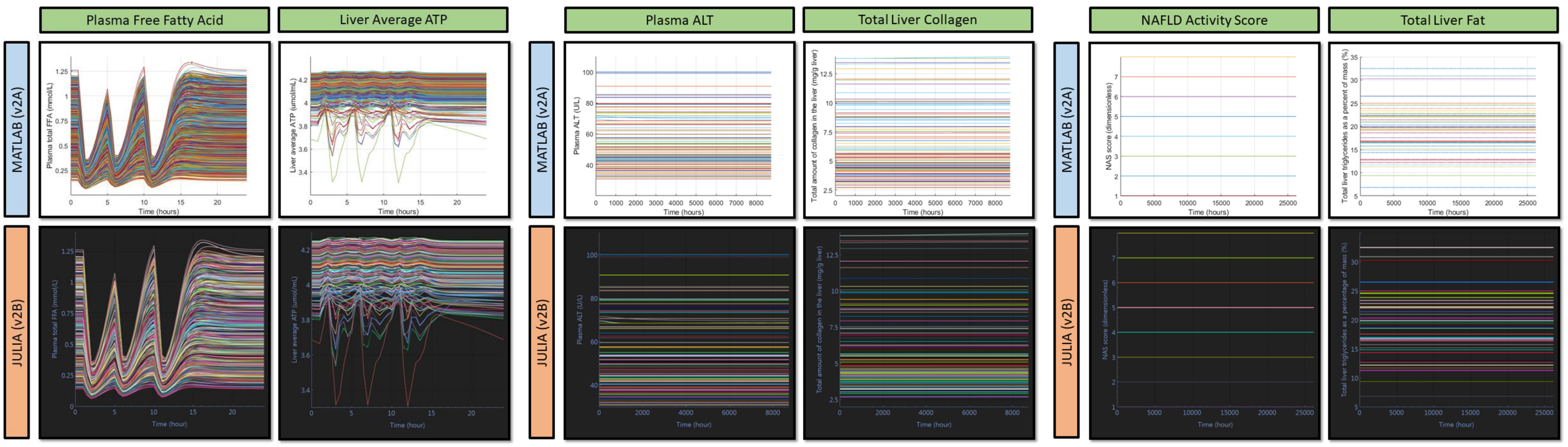
NAFLDsym Version 2B divides visual and non-visual coding elements between Qt (C++) and Julia, respectively. All proprietary engineering code written in Julia is compiled to a binary executable using the Package Compiler library, which is callable from the C++ interface. Public model elements (code, parameters, etc.) are copied to the project folder and can be edited by users. These public files are processed by the compiled application at runtime.

Comparison of v2A and v2B Simulation Times



The execution times of the three configurations in NAFLDsym v2A (MATLAB) were 2.4 hours (n = 1673, 24hrs), 28.7 hours (n=100, 1yr), and 42.5 hours (n=50, 3yrs). The respective times for NAFLDsym v2B (Julia) were 0.6 hours, 4.9 hours, and 5.9 hours. Thus, NAFLDsym v2B exhibits a four-fold to seven-fold improvement in speed over NAFLDsym v2A.

Representative Outputs from NAFLDsym v2A and NAFLDsym v2B for the Three Simulation Configurations



Outputs from NAFLDsym v2A (top row) and v2B (bottom row) are shown for the n=1673, 24 hours simulation (left), the n=100, 52 weeks simulation (middle), and the n=50, 3 years simulation (right). Version 2B results are consistent with Version 2A with respect to both short-term dynamics (e.g., plasma FFA, average liver ATP) and long-term dynamics (e.g., plasma ALT, total liver collagen, NAFLD Activity Score, liver fat). Furthermore, as with Version 2A, Version 2B simulations are stable for up to three years of simulated time. Version v2B plots were generated using a custom Qt/C++ plotting tool that is included with the release version.

CONCLUSION

The efficiency of the NAFLDsym QSP platform has been increased by migrating the model from MATLAB to a Julia/C++/Qt hybrid. Further improvements in efficiency should be possible with ongoing optimization efforts. For instance, substantial progress has been made in resolving instances of type instability, which prevent Julia's compiler from inferring variable types and lead to non-performant code. Reducing the execution time of large-scale simulations from days to hours will streamline NASH treatment simulations, SimPops generation, and additional model development.

REFERENCES

1. Siler, S. Q. Pharm Res. 2022 Aug;39(8):1789-1802.
2. Pcode. https://www.mathworks.com/help/matlab/ref/pcode.html
3. The Julia Language. https://docs.julialang.org/en/v1/
4. SciML: Open Source Software for Scientific Machine Learning. https://sciml.ai/
5. PackageCompiler. https://julialang.github.io/PackageCompiler.jl/dev/