SIMSOPT: A flexible framework for stellarator optimization

Matt Landreman[1], **Bharat Medasani**[2], **Florian Wechsung**[3], **Andrew Giuliani**[3], **Rogerio Jorge**[1], **and Caoxiang Zhu**[2]

1 Institute for Research in Electronics and Applied Physics, University of Maryland, College Park **2** Princeton Plasma Physics Laboratory **3** Courant Institute of Mathematical Sciences, New York University

DOI: 10.21105/joss.03525

Software

- Review
- Repository
- Archive

Editor: Dan Foreman-Mackey

Reviewers:

- @ZedThree
- @StanczakDominik

Submitted: 21 June 2021
Published: 22 July 2021

Summary

A stellarator is a magnetic field configuration used to confine plasma, and it is a candidate configuration for fusion energy, as well as a general charged particle trap. A stellarator's magnetic field is typically produced using electromagnetic coils, and the shaping of the field and coils must be optimized to achieve good confinement. SIMSOPT is a collection of software components for carrying out these optimizations. These components include

- Interfaces to physics codes, e.g. for magnetohydrodynamic (MHD) equilibrium.
- Tools for defining objective functions and parameter spaces for optimization.
- Geometric objects that are important for stellarators – surfaces and curves – with several available parameterizations.
- Implementations of the Biot-Savart law and other magnetic fields, including derivatives.
- Tools for parallelized finite-difference gradient calculations.

Statement of need

To effectively confine plasmas for the goal of fusion energy, the three-dimensional magnetic field of a stellarator has to be carefully designed. The design effort is essentially to vary the magnetohydrodynamic (MHD) equilibrium to meet multiple metrics, for example, MHD stability, neoclassical transport, fast-ion confinement, turbulent transport, and buildable coils. This process involves calling several physics codes and cannot be done manually. A software framework is needed to connect these physics calculations with numerical optimization algorithms.

Although the idea of stellarator optimization is several decades old, there are limited codes available to use. The two most commonly used codes are STELLOPT (Hirshman et al., 1998; Lazerson et al., 2021; Spong et al., 1998) and ROSE (Drevlak et al., 2018). ROSE is closed-sourced, and STELLOPT has the disadvantage that it is written in Fortran and couples all the codes explicitly, meaning that it requires modification of multiple core STELLOPT source files to write an interface for a new module. The goal of SIMSOPT is to flatten the learning curve, improve the flexibility for prototyping new problems, and enhance the extendibility and maintainability. To achieve these goals, SIMSOPT is written in object-oriented Python and incorporates software engineering best practices like continuous integration. Modern tools are used in SIMSOPT to manage the documentation and unit tests.

Structure

The components of SIMSOPT that are not performance bottlenecks are written in python, for flexibility and ease of use and development. In components where performance is critical, compiled C++ code is interfaced to python using the `pybind11` package (Jakob, 2021). As examples, the infrastructure for defining objective functions and optimization problems is written in python, whereas the Biot-Savart law is implemented in C++.

Some of the physics modules with compiled code reside in separate repositories. Two such modules are the VMEC (Hirshman & Whitson, 1983) and SPEC (Hudson et al., 2012; Qu et al., 2020) codes, for MHD equilibrium. These Fortran codes are interfaced using the `f90wrap` package (Kermode, 2020), so data can be passed directly in memory to and from python. This is particularly useful for passing MPI communicators for parallelized evaluation of finite-difference gradients. Another module in a separate repository is BOOZ_XFORM (Landreman, 2021), for calculation of Boozer coordinates. This latter repository is a new C++ re-implementation of an algorithm in an older fortran 77 code of the same name.

A variety of geometric objects and magnetic field types are included in SIMSOPT. Several discretizations of curves and toroidal surfaces are included, since curves are important both in the context of electromagnetic coils and the magnetic axis, and flux surfaces are a key concept for stellarators. One magnetic field type represents the Biot-Savart law, defined by a set of curves and the electric current they carry. Other available magnetic field types include Dommaschk potentials (Dommaschk, 1986) and the analytic formula for the field of a circular coil, and magnetic field instances can be scaled and summed. All the geometric and magnetic field classes provide one or two derivatives, either by explicit formulae, or by automatic differentiation with the `jax` package (Bradbury et al., 2018). Caching is done automatically to avoid repeated calculations.

To date, SIMSOPT calculations have primarily used optimization algorithms from `scipy` (Virtanen et al., 2020). However, since SIMSOPT provides the objective function (and, for least-squares problems, the inidividual residual terms) as a standard python function handle, it requires minimal effort to connect the SIMSOPT objective to outside optimization libraries.

Presently, MPI and OpenMP parallelism are used in different code components. The parallelized finite-difference gradient capability uses MPI, to support use of multiple compute nodes, and to support concurrent calculations with physics codes like VMEC and SPEC that employ MPI. Biot-Savart calculations are accelerated using SIMD intrinsics (via the xsimd library (xtensor-stack, n.d.)) and OpenMP parallelization.

SIMSOPT does not presently use input data files to define optimization problems, in contrast to STELLOPT. Rather, problems are specified using a python driver script, in which objects are defined and configured. An advantage of this approach is that any other desired scripting elements can be included. One way this capability can be used (which is done in the first example below) is to define a series of optimization steps, in which the size of the parameter space is increased at each step, along with the numerical resolution parameters of the codes. The former is valuable to avoid getting stuck in a local minimum, and the latter improves computational efficiency.

Capabilities

Presently, SIMSOPT provides tools for each of the two optimization stages used for the design of stellarators such as W7-X (Klinger et al., 2017) and HSX (Anderson et al., 1995). In the first stage, the boundary of a toroidal magnetic surface is varied to optimize the physics properties inside it. In the second stage, coil shapes are optimized to approximately produce the boundary magnetic surface that resulted from the first stage.

For the first stage, MHD equilibria or vacuum fields can be represented using the VMEC or SPEC code, or both at the same time. VMEC, which makes the assumption that good nested magnetic surfaces exist, is extremely robust and many other physics codes are able to postprocess its output. In VMEC-based optimizations, a typical objective to minimize is the departure from quasisymmetry, a symmetry in the field strength that provides good confinement (Nührenberg & Zille, 1988). SPEC can provide added value because of its ability to represent magnetic islands, which are undesirable since a large temperature gradient cannot be supported across them. Islands can be eliminated using SIMSOPT by minimizing the magnitude of the residues (Greene, 1979), similar to the method in (Hanson & Cary, 1984). An example of stage-1 optimization including both VMEC and SPEC simultaneously is shown in Figure 1-Figure 2. Here, the shape is optimized to both eliminate an internal island chain, as computed from SPEC, and to achieve quasisymmetry, as computed from VMEC and BOOZ_XFORM. More details of this calculation will be presented elsewhere.

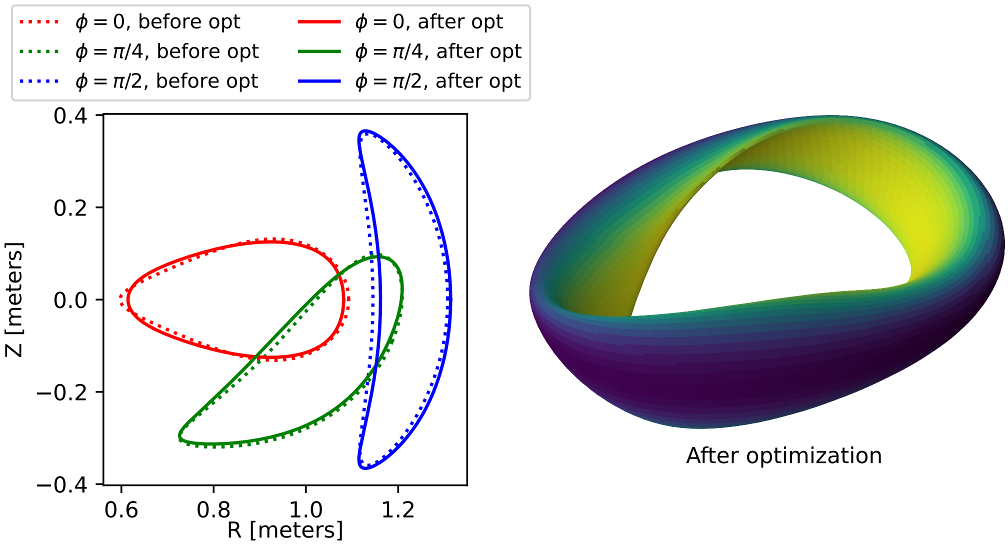

Figure 1: An example of stage-1 optimization using SIMSOPT, in which the shape of a toroidal boundary is optimized to eliminate magnetic islands and improve quasisymmetry. Shown on the left are slices through the surface at different angles ϕ of the initial and the optimized configuratios.

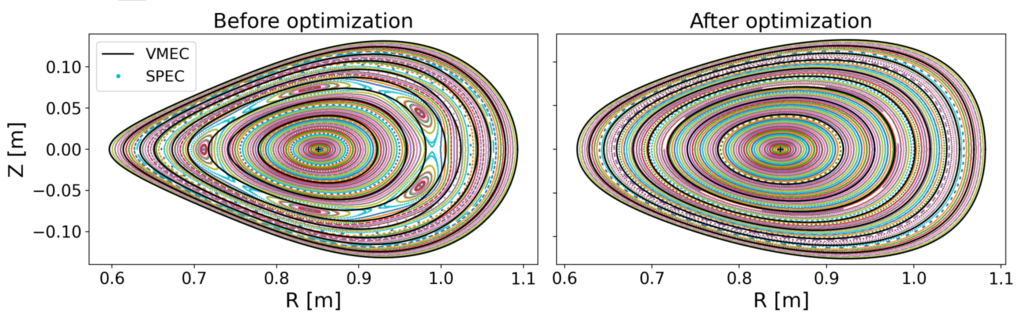

Figure 2: VMEC flux surfaces (black lines) and Poincare plot computed from the SPEC solution (colored points) for the initial and optimized configurations in Figure 1. The initial configuration contains an island chain, whereas the optimized configuration has nested flux surfaces.

The curve and magnetic field classes in SIMSOPT can then be used for the second optimization stage, in which coil shapes are designed. Varying the shapes of the coils, derivative-based optimization can be used to minimize the normal component of the magnetic field on the target surface, similar to the FOCUS code (Zhu et al., 2018).

One can also use SIMSOPT for other optimization problems that differ from the above two-stage approach. For instance, SIMSOPT is presently being used for a single-stage derivative-based method in which coil shapes are varied to optimize directly for quasisymmetry near the magnetic axis (Giuliani et al., 2020). Figure 3 shows an example in which stochastic optimization is applied to find a configuration in which the quasisymmetry is relatively insensitive to errors in the coil shapes. This example will be described in more detail in a separate publication.

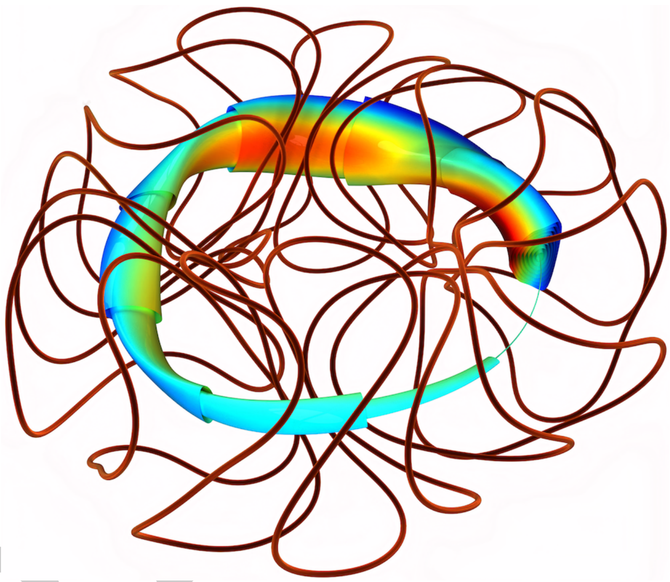

Figure 3: A stellarator obtained using stochasic optimization with `Curve` and `BiotSavart` classes from SIMSOPT, with magnetic surfaces computed using `Surface` classes.

Acknowledgements

We gratefully acknowledge discussions with and assistance from Aaron Bader, Antoine Baillod, David Bindel, Benjamin Faber, Stuart Hudson, Thomas Kruger, Jonathan Schilling, Georg Stadler, and Zhisong Qu. This work was supported by a grant from the Simons Foundation (560651, ML). BM and CZ are supported by the U.S. Department of Energy under Contract No. DE-AC02-09CH11466 through the Princeton Plasma Physics Laboratory.

References

Anderson, F. S. B., Almagri, A. F., Anderson, D. T., Matthews, P. G., Talmadge, J. N., & Shohet, J. L. (1995). The helically symmetric experiment, (HSX) goals, design and status. *Fusion Tech.*, *27*, 273.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs*. http://github.com/google/jax

Dommaschk, W. (1986). Representations for vacuum potentials in stellarators. *Comp. Phys. Comm.*, *40*, 203.

Drevlak, M., Beidler, C., Geiger, J., Helander, P., & Turkin, Y. (2018). Optimisation of stellarator equilibria with ROSE. *Nucl. Fusion*, *59*, 016010.

Giuliani, A., Wechsung, F., Cerfon, A., Stadler, G., & Landreman, M. (2020). Single-stage gradient-based stellarator coil design: Optimization for near-axis quasi-symmetry. *arXiv:2010.02033*.

Greene, J. M. (1979). A method for determining a stochastic transition. *J. Math. Phys.*, *20*, 1183.

Hanson, J. D., & Cary, J. R. (1984). Elimination of stochasticity in stellarators. *Phys. Fluids*, *27*, 767.

Hirshman, S. P., Spong, D. A., Whitson, J. C., Lynch, V. E., Batchelor, D. B., Carreras, B. A., & Rome, J. A. (1998). Transport optimization and MHD stability of a small aspect ratio toroidal hybrid stellarator. *Phys. Rev. Lett.*, *80*, 528.

Hirshman, S. P., & Whitson, J. C. (1983). Steepest-descent moment method for three-dimensional magnetohydrodynamic equilibria. *Phys. Fluids*, *26*, 3553.

Hudson, S. R., Dewar, R. L., Dennis, G., Hole, M. J., McGann, M., von Nessi, G., & Lazerson, S. (2012). Computation of multi-region relaxed magnetohydrodynamic equilibria. *Phys. Plasmas*, *19*, 112502.

Jakob, W. (2021). *pybind11 — seamless operability between c++11 and python*. https://github.com/pybind/pybind11

Kermode, J. R. (2020). f90wrap: An automated tool for constructing deep python interfaces to modern fortran codes. *J. Phys. Condens. Matter*, *32*, 305901.

Klinger, T., Alonso, A., Bozhenkov, S., Burhenn, R., Dinklage, A., Fuchert, G., Geiger, J., Grulke, O., Langenberg, A., Hirsch, M., Kocsis, G., Knauer, J., Kramer-Flecken, A., Laqua, H., Lazerson, S., Landreman, M., Maassberg, H., Marsen, S., Otte, M., … the Wendelstein 7-X Team. (2017). Performance and properties of the first plasmas of Wendelstein 7-X. *Plasma Phys. Controlled Fusion*, *59*, 014018.

Landreman, M. (2021). https://github.com/hiddenSymmetries/booz_xform

Lazerson, S., Zhu, C., Schmitt, J., & et al. (2021). *STELLOPT*. https://github.com/PrincetonUniversity/STELLOPT

Nührenberg, J., & Zille, R. (1988). Quasi-helically symmetric toroidal stellarators. *Phys. Lett. A*, *129*, 113.

Qu, Z. S., Pfefferlé, D., Hudson, S. R., Baillod, A., Kumar, A., Dewar, R. L., & Hole, M. J. (2020). Coordinate parameterisation and spectral method optimisation for Beltrami field solver in stellarator geometry. *Plasma Phys. Controlled Fusion*, *62*, 124004.

Spong, D. A., Hirshman, S. P., Whitson, J. C., Batchelor, D. B., Carreras, B. A., Lynch, V. E., & Rome, J. A. (1998). J^* optimization of small aspect ratio stellarator/tokamak hybrid devices. *Phys. Plasmas*, *5*, 1752.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., & others. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, *17*, 261.

xtensor-stack. (n.d.). *Xsimd*. https://github.com/xtensor-stack/xsimd

Zhu, C., Hudson, S., Song, Y., & Wan, Y. (2018). New method to design stellarator coils without the winding surface. *Nucl. Fusion*, *58*, 016008.