31st International Conference Nuclear Energy for New Europe
NENE 2022 September 12-15 PORTOROŽ SLOVENIA

Towards Automation of Fusion Reactor Design Optimization – Neutronic Optimization in Simple Parametric Models

Aljaž Čufar
Jožef Stefan Institute
Jamova cesta 39
SI-1000, Ljubljana, Slovenia
aljaz.cufar@ijs.si

Anže Gabrijel

Jožef Stefan Institute
Jamova cesta 39
SI-1000, Ljubljana,
Slovenia

and

Faculty of Electrical engineering
University of Ljubljana
Tržaška cesta 25
SI-1000, Ljubljana, Slovenia
ag6791@student.uni-lj.si

ABSTRACT

Fusion reactor design and optimization can be a time-consuming task. One of the challenges in expediting this process is the time needed to perform analyses that test the performance of proposed designs. In neutronics analyses often the most time-consuming part is the preparation of the geometry. Typically, both the design of systems and their integration in the reactor can undergo multiple iterations and the preparation of new geometric models can significantly slow down the rate of iteration. Therefore, the design optimization can be slow and only a limited number of possible design solutions can be explored.

In some cases, the solution to speed up the design modifications is parametric modelling where models are prepared in such a way that changing their parameters, e.g. dimensions or number of some features, result in a useable model for some useful range of these parameters. While the preparation of parametric models from scratch is typically more time consuming than preparing a single model in nonparametric fashion, the use of the same foundations for several models can prove to be more efficient. This is especially effective when simplified models or models with significant number of repeated structures defined by a small number of parameters can be used. The use of tools such as PARAMAK or STOK which further simplify the generation of such models using pre-prepared parametric reactor components can additionally speed up the process of setting up the groundwork for preparation of considerable number of different models.

If such models can be easily used in analyses, then they can also be used in automatic design optimization schemes. In this paper we investigate different schemes for automatic design optimization of neutronic performance in some simple fusion-relevant models. It is expected that experience with optimization algorithms, selection of initial designs, and a choice of fitness function used in such optimizations will translate into optimization of more complex models useful for more realistic studies and design optimizations. A fitness function is a function that quantifies the fitness of a design as a single number. In practical terms the optimization process thus consists of minimizing or maximizing the fitness function.

1 INTRODUCTION

1.1 Reactor design

Designing a fusion reactor includes finding many compromises in design choices to make sure that all the necessary conditions are met. While automatic design optimization is routinely used in various industries a barrier to entry for widespread use in nuclear analyses are difficulties related to reliable automatic production of geometry models used in such analyses and computational intensiveness of these simulations. However, available computational resources are increasing and are already sufficient to enable the use of automated design optimization in nuclear aspect for at least some cases.

1.2 Optimization

Design optimisation is generally time consuming especially in terms of analyst time. Often a design is optimized through multiple optimization rounds that include design modifications proposed by analysts who base their suggestions on experience and results of nuclear analyses like particle pathways identified in analyses or through trial and error with different modifications to the model. While this approach is efficient in terms of computational resources it is very time intensive for analysts. We argue that for some cases the process could be automated and that due to development of both tools and computer hardware the number of these cases will continue to grow. With this in mind we think it is worthwhile to start investigating these cases and to build workflows that can be used in increasingly divers sets of cases.

2 PARAMETRIC MODELLING

One way to rapidly generate models for automatic design optimization schemes is through parametric modelling. Such models are generated by defining rules that connect input parameters, e.g. various dimensions or number of certain features, to the generated body. What is usually challenging is the fact that body volumes should not overlap. Rules for preparation of parametric models thus must consider possible conflicts between bodies and include ways to resolve them. Parametric modelling can be done in a way that parametric models are directly created in a format suitable for neutronic simulations, e.g., in constructive solid geometry (CSG) native to the code of interest, or in CAD format that is subsequently converted to formats suitable for neutronic analyses. This latter step can also be done in two separate ways. One way is to use a code such as SuperMC [1] to convert CAD geometry into CSG description of the geometry for the code of interest. This generally works well even for geometries of significant complexities such as DEMO [2] or ITER [3] but is unfortunately not reliable enough to produce models entirely unsupervised. Generally, at least some user input is required as CSG description of the geometry is overly sensitive to even small errors in geometry description. The second way to produce models is to export CAD based models in a format suitable for neutronics analyses, e.g., in *.stl. Currently this is still a novel way to describe the geometry for neutronics analyses and some challenges are expected, e.g., higher memory usage. However, due to the potential simplicity of unsupervised generation of models this seems to be the way forward both in neutronics analyses and especially in efforts to automate design optimizations.

2.1 Constructive solid geometry

While modelling in CSG is widely used in neutronic simulations it can be challenging to use for parametric modelling in generalised fashion. However, in some cases parametric CSG is an appropriate way to produce models. One of such cases was a model of a remote handling system used in simulations in support of the neutron yield calibration at JET [4] where a model of the remote handling system was prepared in such a way that its various positions and orientations of components were reproduced using transformations.

The upsides of CSG usage can be in the speed of model preparation and computational efficiency of simulations. However, the time needed to produce scripts that can reliably build models of increasing complexity does not scale well with complexity of the models. The use of parametric CSG modelling is most suited for cases when a limited number of parameters is being varied within predetermined bounds that make sure that there are no geometry conflicts.

2.2 CAD based alternatives

An alternative to CSG modelling is available in the form of unstructured mesh in MCNP [5], tessellation-based geometry (.stl format) in Serpent 2 [6], and direct accelerated geometry Monte Carlo (DAGMC) [7] based on surface mesh format as found in OpenMC [8] or coupled with other codes like MCNP. CAD based geometry descriptions are compelling as they allow for easier modelling that is more in line with other fields which has the potential to significantly speed up model preparation as well as simplify interfacing of neutronics analyses with analyses of other processes, e.g. thermohydraulic analyses.

As the CAD-based geometry used in simulations is tessellated or meshed in another way it is important to quantify the effect of increasing the complexity of level of detail described by these models. We analysed the times needed to produce geometry and tested the performance of such models in simulations. As expected, we found that values of results do not change when using models with sufficient level of details. However, the level of details has a significant effect on the pre-processing time needed to produce the model and on the computational time needed to complete the simulations [9].

3 OPTIMIZATION

In this section we discuss some of the methods and tests with simple models that were performed to assess the effectiveness of various optimization strategies. As first steps into automation of design optimization these analyses are dealing with simple models and algorithms. This is done to maximize our understanding of what is happening both in simulations and in optimization procedure. Through this we seek to get as much insight into future avenues of this research as possible and are looking into ways to make use of optimizations with simpler models as a way to speed up optimization of more complex models.

3.1 Genetic algorithm

Optimizations in this paper were performed with a simple genetic algorithm as shown in Figure 1. This means that the models used in simulations were produced from strings of numbers, or “chromosomes,” which describe the part of the geometry that can be varied, and each new generation of chromosomes is produced from best performing chromosomes in previous generation.

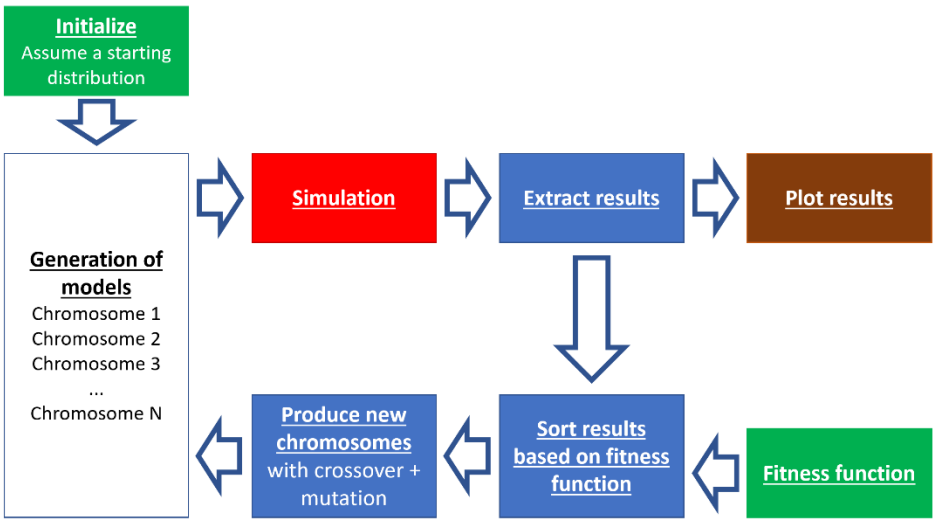

Figure 1: Flowchart of genetic algorithm used in present work.

Initialize: To start off the optimization process a starting generation of chromosomes is produced. For the algorithm to work the starting generation must be sufficiently diverse to cover sufficiently large parts of the space of viable solutions. To do this we randomly generate the chromosomes based on some assumptions/random distribution. These assumptions can significantly speed up the optimization process but can also lead to optimization towards some local minimum instead of global one.

Simulation and data extraction: For present work simulations were performed either analytically or using Monte Carlo codes and results were extracted using Python scripts in such a way that chromosomes were linked to the results of interest.

Sort: The chromosomes were sorted based on the value of a fitness function associated with them. The fitness function is defined depending on the goal of the optimization, e.g., minimization of a particle flux value if we want to maximize shielding.

Production of new chromosomes: From the list of sorted chromosomes, we took one quarter of the best performing chromosomes and performed crossovers between random pairs of chromosomes from this list (Figure *2*). We tested both one-point crossover (where child chromosome was generated from the first half of one and the second half of the second parent chromosome) and multipoint crossover (where for each value in the array representing a chromosome, there was an equal chance that the value would come from first or the second parent chromosome). Furthermore, mutations were performed on the child chromosomes based on a set mutation rate. This means that a random change would be introduced in a random chromosome based on a predefined probability.

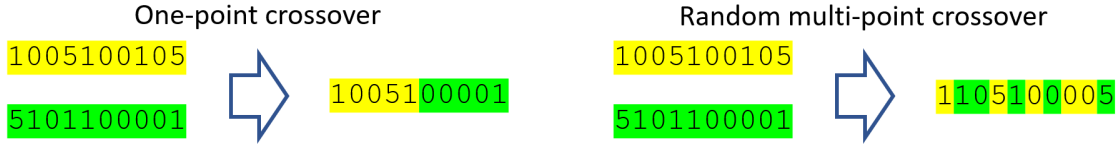

Figure 2: Crossover demonstration.

Model production: Python scripts were used to produce models for use in simulations.

3.2 Simple analytical model

Our first tests were performed on a simple analytical model to study the behaviour of optimization algorithms using computationally inexpensive simulations. One of the reasons to study these simple cases was also to determine which approaches could be used in more realistic simulations where available computational resources put significant constraint on the number of simulations that can be run. The goal of these simple tests was to learn how genetic algorithms perform in situations similar to shielding problems to be able to use them more effectively in more realistic cases. The idea is, that through gradual increase in complexity of the problems (and in computational times required to perform suitable number of the analyses), we would learn how to effectively use them and reduce the overall computational time. Furthermore, it seems likely that simplified analyses could be used as a starting stage to figure out the initial conditions close to the optimum to reduce the number of generations needed to converge to an optimized solution.

The simple analytical model featured layers of material in a shielding approximation where each added layer of material leads to a reduction of particle transmission by a fixed factor and increase in mass defined by the position of the layer and the density of material. The geometry was assumed to be in a shape of nested spherical layers similar to the model used in the next section. In these optimization cases the algorithm could choose between a layer being empty, filled with material A or filled with material B.

Table 1: Materials and their properties used in simple analytical model.

Material	Transmission factor	Density [g/cm^3]
A	0.5	8.0
B	0.9	1.0
Void	1.0	0.0

Two cases of fitness functions were analysed: either transmission or mass of the shield were to be minimized. In both cases the optimal configuration is obvious – for the transmission case all layers should be filled with material B and for mass case all layers should be empty. Such simple cases were chosen as we wanted to study the convergence towards the optimum configuration while starting far from the optimal configuration.

Transmission factor was calculated as:

$$T = 0.5^{nr.\ of\ layers\ A} \times 0.9^{nr.\ of\ layers\ B} \quad (1)$$

thus, the position of the layer had no effect on the shielding performance and only the material counted. On the other hand, the mass of the system was calculated as a sum of masses of 20 spherical layers of equal thickness, so the position of a layer plays a role as well. All layers together describe the special shell ranging from 50 cm to 100 cm.

Mutation rates and a type of crossover

Mutation rates of 0%, 0.1%, 1% and 10% were tested for cases where there were 400 and 40 chromosomes per generation. Furthermore, for 400-chromosome cases we compared the performance of one-point crossover to a random multi-point crossover.

Results show that too high a mutation rate (in this case 10%) results in an optimization that has difficulties reaching optimum. The reason is likely the fact that optimum is represented by a single chromosome description and while crossover is a process directed towards optimum

the randomness of mutations is more likely to produce worse performing chromosomes once close to optimum. Furthermore, results show that multi-point crossover performs significantly better than single-point crossover. Perhaps in some more complex cases it makes sense to keep certain parts of the chromosome together but in this case the one-point crossover often had issues.

With these findings in mind we continued our analyses using 1% mutation rate and multi-point crossover.

Consistency of convergence, different number of chromosomes and generations

We analysed the effect of the different number of chromosomes on the number of generations and thus the total number of cases to reach optimal values. Cases with 400, 200, 100, and 40 chromosomes per generation were analysed for both transmission and mass optimization. The starting material distribution was chosen far from optimum, i.e., for transmission minimization 90% void, 5% material A and 5% material B, and for mass minimization 10% void, 45% material A and 45% material B.

To assess the consistency of the optimization we ran the same optimization 5 times for each case from randomly generated starting points based on the same starting assumptions on material composition of layers. Results are presented in Figure 3 and Table 2 including the sum of simulations required to reach the optimal value. In reality several more generations would need to be run to ensure that indeed the optimal value was found. In effect, this estimation shows results of cases where a larger number of chromosomes is used per generation in favourable light. Furthermore, due to the low number of optimizations (5) this spread represents only a rough approximation of the real spread of values.

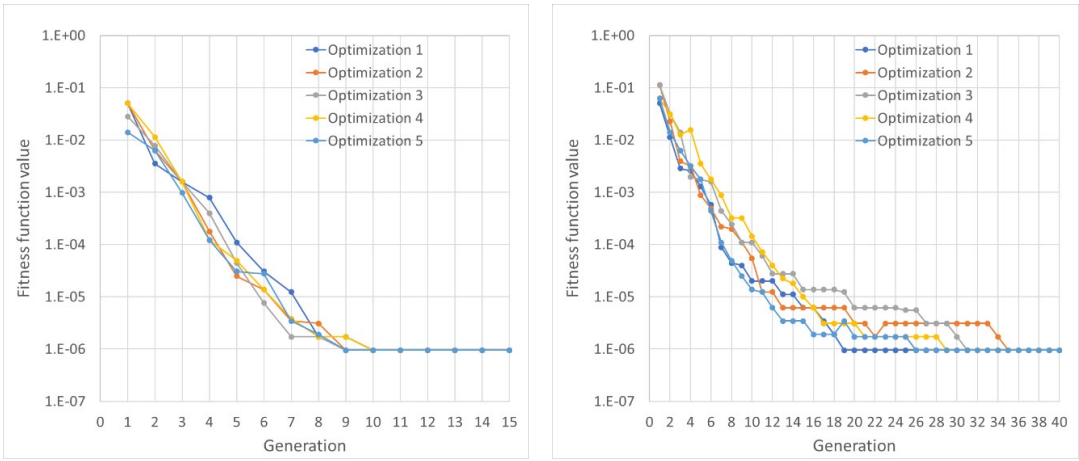

Figure 3: Testing the consistency of optimization for the case of transmission minimization. 400-chromosome case (left) and 40 chromosome case (right).

Table 2: Results of consistency test and number of simulations needed to reach minimum.

Nr. of chromosomes per generation	Transmission factor minimization		Mass minimization	
	Nr. of generations	Sum of simulations	Nr. of generations	Sum of simulations
400	9 – 10	3600 - 4000	12 – 15	4800 – 6000
200	9 – 13	1800 - 2600	13 – 23	2600 – 4600
100	12 – 32	1200 - 3200	15 – 28	1500 – 2800
40	19 - 35	760 - 1400	28 – 63	1120 – 2520

3.3 Simple spherical model for Monte Carlo simulations

The next step towards more realistic models were optimizations of neutronic performance using a simple MCNP model consisting of nested layers as shown in Figure 4. We used a 2 MeV gamma ray source and 20 layers of equal thickness between the inner and outer radii of 50 cm and 100 cm, i.e., each layer is 2.5 cm thick. Chromosomes determined the material composition of each layer, and the choice was between an empty layer or a single material (void or Eurofer) or two materials (void or Eurofer or water) and we used minimization of gamma transmission as a fitness function. Such a simple optimization problem was chosen as a way to start testing the procedure with the most simple case where the optimal solution is obvious.

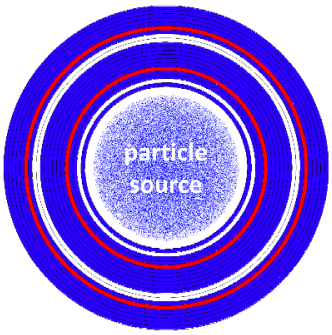

Figure 4: Simple spherical MCNP model of the geometry. The colour defines the material composition of the layer.

Based on results from previous sections we used multi-point crossover, 1% mutation rate, and 40 chromosomes per generation. Results for the two analysed cases are in Figure 5 and the number of generations needed to reach optimum is within interval for 40-chromosome generations in Table 2.

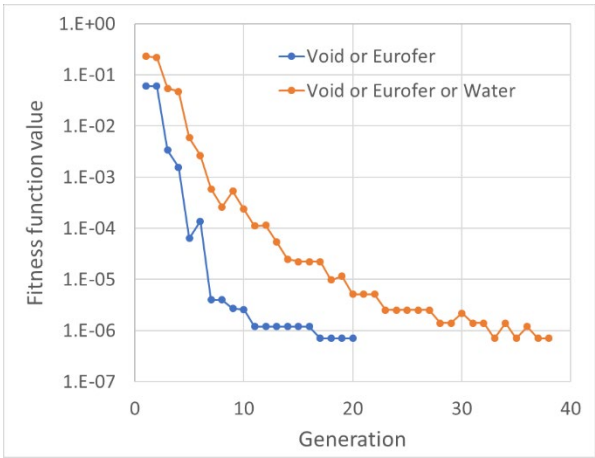

Figure 5: Optimization process for two cases with either two choices of material (void or Eurofer) or three (void or Eurofer or water).

One way we could use the results from previous analytical simulations to speed up the optimization process is by using the solution from that analysis as a starting point in the initialization phase. However, in this simple case this would directly lead to a confirmation that the approximation of the best solution is already the best solution.

3.4 Next steps

Next step from analyses with simple models is to apply these algorithms to increasingly complex reactor models. However, models need to be reliably produced in automated fashion. With this in mind, STOK [9] was developed and for future work Paramak [10] is being assessed. Both tools are developed around Python and CadQuery and aim to simplify the production of tokamak models with sufficiently realistic features for many of the analyses in fusion neutronics. Future studies will thus include increasingly complex studies with layered spherical model and later more reactor-relevant models generated with STOK or Paramak.

STOK is an in-house developed tool that produces reactor models with rectangular cross-section in fully parametric fashion. While such models are far from realistic, they are often useful for analyses where a small number of parameters contributes to understanding of fundamental aspects of a problem or design. As such we see it as a steppingstone to get valuable experience with optimization in reactor-like models that can be produced with a relatively small number of parameters before applying the tools to more realistic models.

Paramak is a more general tool that produces more realistic tokamak models. It can produce a large set of significantly different tokamak models and we plan to use it when we get to the point when this level of modelling is beneficial.

However, production of each model using these tools takes significantly more computational time than varying of parameters in typical CSG based parametric model. Typical time needed to prepare a single model using STOK once the parameters were selected is around 20 seconds (and another 2 or more minutes, depending on mesh fineness, for exporting the model into .stl data format) while the time for Paramak is comparable or even faster at times due to its higher maturity at this time. Producing tens or hundreds of models in such a way can thus be time consuming but in many cases still not as time consuming as manually modifying the models. However, there are multiple ways to speed up the process, e.g., by producing multiple models in parallel and through bookkeeping and reusing the models of reactor components whenever possible.

Future work thus includes:

- Optimizations of STOK-generated models.
- Multiparametric optimizations, definition of fitness functions.
- Analyses of other important effects
    - For Monte Carlo analyses – the balance between lower statistical errors vs larger number of simulations.
    - Defining algorithm exit conditions depending on fitness function and available computational resources.
    - Introduction of variance reduction in Monte Carlo simulations (on-the-fly variance reduction might be suitable).

Furthermore, we plan to further pursue and develop the methodology where we use lower fidelity models to determine more suitable starting distributions for higher fidelity models and through this save computational resources. The hierarchy of models for this use is envisaged as analytical → simple CSG → STOK → Paramak → realistic reactor model.

3.5 Lessons learned

Random sampling/initial conditions

- Starting conditions closer to the optimum lead to faster convergence.
- Care needs to be taken to ensure that initial conditions are sufficiently diverse.

Optimization algorithms

- Mutation rate (1% worked well in our cases)
    - Low mutation rate can lead to results being stuck in suboptimal configurations.
    - High mutation rate can preclude the solutions to reliably reach optimum as mutation of optimal chromosome always performs sub-optimally.
    - Our hypothesis is that at least for some cases starting with higher mutation rate and then reducing it through generations might be beneficial (similar to simulated annealing).
- Crossover
    - Multipoint crossover performed better than single point crossover.

Model preparation

- Quick and reliable – any human intervention must be limited otherwise the use in automated optimization scheme is not feasible.
- Short computation time is crucial. This is where Monte Carlo methods and realistic models are often still too computationally demanding.

Analyses and computational resources

- Bookkeeping to ensure that simulations using already analysed model configurations are not simulated again.
- Variance reduction will be crucial. On-the-fly variance reduction methods could be used to reduce the need for analyst intervention.
- It would be beneficial to limit the Monte Carlo simulations by a target value for statistical error instead of by the number of particles simulated or by CPU time used. This way too-high statistical uncertainties would not threaten to reduce the effectiveness of sorting due to statistical noise and on the other hand the lower than necessary statistical errors would not lead to unnecessary long computational times.

4 CONCLUSIONS

Automation or partial automation of optimization processes is a common tool in many fields of research and industry. It is likely that with advances in computational resources it will increasingly find uses in both fusion and fission reactor design.

In this paper we investigated the performance of simple genetic algorithms for use in optimization of neutral particle transport problems. As first steps into this field we investigate which parameters of this algorithm work best in such cases and suggest relation between simpler and more sophisticated models which could be used as a form of algorithm speed-up. This speed-up could be achieved using faster but less sophisticated models to produce starting

distributions for more sophisticated models. The work will continue with the in-house developed tool for production of fusion reactor models in parametric fashion – STOK. However, as the production of these models and simulations with them will be more computationally expensive we see the need for development and use of different much simpler models in parallel to make sure the optimizations are performed in reasonable computer times.

ACKNOWLEDGMENTS

The authors acknowledge the financial support from the Slovenian Research Agency (research project Z2-3201, research program No. P2-0073).

REFERENCES

[1] SuperMC: Y. Wu, FDS Team, CAD-based interface programs for fusion neutron transport simulation, Fusion Engineering and Design 84 (2009), 1987-1992.

[2] A. Čufar et al., Shielding concept and neutronic assessment of the DEMO lower remote handling and pumping ports, Fusion Engineering and Design 157 (2020), 111615.

[3] A. Kolšek et al., Shutdown dose rate mitigation in the ITER upper ports, Fusion Engineering and Design 136 (2018), 228-232.

[4] L. Snoj et al., Calculations to support JET neutron yield calibration: Modelling of the JET remote handling system, Nuclear Engineering and Design 261 (2013), 244-250.

[5] MCNP5 1.60: F. Brown et al, Verification of MCNP5-1.60, LA-UR-10-05611, Los Alamos National Laboratory (2010).

[6] Serpent: J. Leppänen et al., The serpent Monte Carlo code: status, development and applications in 2013, Ann. Nucl. Energy 82 (2015), 142-150.

[7] P.P.H. Wilson, et al. Acceleration techniques for the direct use of CAD-based geometry in fusion neutronics analysis, Fusion Engineering and Design 85 (2010), 1759-1765.

[8] P.K. Romano et al., OpenMC: a state-of-the-art Monte Carlo code for research and development, Ann. Nucl. Energy, 82 (2015), 90-97.

[9] A. Gabrijel et al., STOK – A Tool For Parametric Modelling Of Simple Tokamaks, Nuclear Energy for New Europe 2022, Portorož, 2022.

[10] J. Shimwell et al., The Paramak: automated parametric geometry construction for fusion reactor designs, F1000Research 2021 (last updated: 31 Aug. 2022).