Generating 3D Topologies with Multiple Constraints on the GPU

Krishnan Suresh

Associate Professor, Mechanical Engineering
UW-Madison, Madison, Wisconsin 53706, USA
suresh@engr.wisc.edu

ABSTRACT

The objective of this paper is to demonstrate a topology optimization method that can handle multiple displacement and stress constraints. Such problems are known to be harder to solve than pure displacement, or pure stress-constrained problems, but arguably more important.

The proposed method relies on the concept of topological sensitivity. The latter captures the sensitivity of a quantity of interest to a topological change. In this paper, well known adjoint methods are used to compute the topological sensitivity field for each of the constraints. Then, a topological level-set is constructed through a weighted combination of these topological fields. The weights are determined dynamically, and various weighting techniques are explored for their effectiveness.

At the start of the optimization process, an optimal topology of a slightly reduced volume fraction (say 0.95) is extracted using the topological level set. This involves a fixed-point iteration, and the converged topology is 'optimal' with respect to all the constraints. Next, the volume fraction is decremented to 0.90, and the iteration is repeated until the optimization cannot proceed further.

Satisfying multiple constraints entails numerous finite element analysis (FEA), each of which can be computationally expensive in 3D. Hence, an efficient FEA implementation is critical. Classic density-based methods can lead to ill-conditioned matrices that can pose serious computational challenges. Therefore, a salient feature of the proposed method is that partial elements are completely avoided. Consequently, not only are the stresses are well-defined at all points within the evolving topology, the stiffness matrices are well conditioned. Therefore, simple iterative solvers, such as Jacobi-preconditioned conjugate gradient, that are easy to parallelize, can be used.

In such iterative solvers, the primary computational cost is sparse matrix-vector multiplication (SpMV). To accelerate SpMV, we propose a matrix-free implementation of the finite element method. This exploits modern multi-core architectures to efficiently solve finite element problems involving millions of degrees of freedom. Specifically, in multicore CPU implementation, parallelization is attained through OpenMP commands. In the GPU implementation using CUDA on NVidia graphics cards, SpMV is implemented by assigning each node of the voxel-grid to a scalar processor. When a block of threads are launched, all threads share identical element matrices, and therefore a single memory fetch of the stiffness is sufficient. Finally, many of the low-level operations such as vector-product were implemented using CUBLAS library.

The proposed methodology is illustrated through numerical experiments in 2D and 3D; comparisons are made against previously published results. Further, a series of topologies of decreasing volume fractions are generated in a single optimization run.