Differentiable force-density method: an easily extensible framework for the constrained design of funicular shapes

Pierre CUVILLIERS*, Caitlin MUELLER[a]

*, [a] Building Technology program, Massachusetts Institute of Technology
77 Massachusetts Avenue, Room 5-414, Cambridge, MA, USA
pcuvil@mit.edu

Abstract

Automatic differentiation (AD), a computational tool that automatically finds an exact yet efficient gradient of general procedures, has recently seen renewed interest due to its usefulness in machine learning algorithms [1]. The tools developed there are useful for many other optimization problems, such as those encountered in form-finding problems with constraints or objectives of best-fit to target. In this paper, we will demonstrate this by building a differentiable framework for the form-finding of funicular surfaces, with additional constraints. The word differentiability here is taken to mean that the system will be able to output gradients for all its numerical procedures, without the programmer explicitly encoding them. Adding a constraint or objective will only require a function giving its value, the gradient is automatically computed from this function.

This is related to other design systems for funicular structures, such as [2], that uses thrust-network analysis (TNA) to generate funicular forms, and optimizes these to match a best-fit-to-target objective. Here, we use the force-density method as a generator, tack on additional objectives and constraints on our designs, and integrate this in an optimization program, with the force densities as variables. In previous systems, there was a high cost associated to adding a new objective, because an explicit procedure computing its gradient is required to get an efficient optimization program. For example, see the original force density paper [3], which details the calculations of gradients for some constraints.

We overcome this issue by implementing the force density method in a package for AD of vector and matrix computations (specifically, we use Autograd). In this package, the derivatives of all operators are defined, as well as ways of combining them. Gradients are then an immediate by-product of any numerical function written in this package. Note that the procedures generated for the gradients are exact, unlike a finite-difference approximation of the gradient. These procedures are computationally efficient, unlike the results of a full non-simplified symbolic differentiation. Thus, they are good candidates for use in optimization, generally as good as what can be achieved by explicitly coding the gradient.

The resulting framework can generate funicular shapes with arbitrary constraints and objectives as defined by the designer. Because no gradients are required, it is easy to modify the output by changing the constraints or objectives. For example, we have experimented with objectives that make all elements lengths similar, or chosen from a list of possible lengths, as well as flat panels, similar angles, and best-fit-to-target. The optimization programs are efficiently solved, returning in a time reasonable for iterative design even for problems with as many as 10,000 elements. This shows the relevance of automatic differentiation technique in form-finding problems, and generally writing computational design as differentiable programs. It would be very interesting to apply these techniques to other form-finding problems, or to any other design method based on optimization.

References

[1] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic Differentiation in Machine Learning: a Survey," *J. Mach. Learn. Res.*, vol. 18, no. 153, pp. 1–43, 2018.

[2] T. Van Mele, D. Panozzo, O. Sorkine-Hornung, and P. Block, "Best-fit thrust network analysis: rationalization of freeform meshes," in *Shell Structures for Architecture: Form Finding and Optimization*, New York, NY, USA: Routledge, 2014, pp. 157–168.

[3] H.-J. Schek, "The force density method for form finding and computation of general networks," *Comput. Methods Appl. Mech. Eng.*, vol. 3, pp. 115–134, 1974.