1. OPTIMIZATION TECHNIQUES FOR STRUCTURAL DESIGN OF COLD-FORMED STEEL STRUCTURES
   1. Introduction and problem formulation

Structural optimization is the search for a structural design that is optimal for a certain design criterion while satisfying other constraints. It is a specific case of general optimization problems in mathematics. A well-defined optimization problem should include design variables and usually one objective function, often with constraints formulated using the design variables. In terms of structural optimization, objective functions in general can be the minimization of compliance or weight, or the maximization of capacity. It is customary to formulate the objective as a minimization. Solutions of practical optimization problems usually have to satisfy equality and inequality constraints developed from design, manufacturability, and end-use requirements, making constrained optimization problems more difficult to solve than unconstrained problems. Another important feature of structural optimization is that it usually requires solving equilibrium problems of the structure every time the objective function needs to be evaluated. Accordingly, structural optimization problems are usually simulation-based: modeling and structural analysis tools are typically embedded in the optimization framework [1]. The objective function evaluation can be computationally costly because of the simulation involved.

Typically, structural optimization incorporates:

1. **Dimension optimization**: operates on a fixed topology by adjusting characteristic dimensions of the topology only;
2. **Shape optimization**: requires a shape generation subroutine to update the geometry of the design for better performance, but it may not be possible to change the connectivity and topology greatly;
3. **Topology optimization**: has the most flexibility for developing an optimized configuration within a given domain of material connectivity can be modified by introducing voids.

In dimension optimization, design variables are usually characteristic dimension(s) of the dictated topology and should be positive real numbers within certain bounds. For topology optimization, design variables can ideally be zero (void) or one (solid) for a given discretized domain. However, to allow the use of the gradient based method, which requires continuity and differentiability, design variables can be treated as a continuous number between zero and one in some topology optimization implementations. For shape optimization, design variables are parameters that can uniquely define the shape and are dependent on the shape generative algorithm [2].

For the scale level of structural optimization, research efforts focus on member and layout optimization of structural systems. These two levels can be combined, so the structural design can be significantly automated by generating a highly efficient layout with optimized selection of members. The history of research on optimization of cold-formed steel (CFS) framed structures started with dimension optimization of designated types of members, and has achieved some inspiring progress on shape optimization on the member level. Initial efforts on CFS framing layout optimization have recently been reported.

So far, the great majority of CFS structural optimization research has been performed on the member level mostly columns under axial compression, since the stress distribution is assumed to be uniform and no further effort of neutral axis determination is needed. Given that buckling is one important limit state for CFS members, the method used for determining nominal axial capacity (), from design specifications can have a significant influence on the formulation of optimization problems. can be an objective of maximization, or it can be a constraint that must be satisfied by the optimal design. Classical design procedure, as documented in early versions of standards specifications, adopted the long stablished effective width method ,which requires reducing a plate under nonlinear longitudinal stress into a plate with effective width under constant stress. However, the process can be cumbersome for complex cross-sections. The Direct Strength Method (DSM) [3]–[5]is much simpler, requiring only the critical load in local (), distortional (), and global buckling (), and the yield load (). Design equations in DSM are simple forms of elementary functions and can be coded easily into simulation-based optimization programs. Detailed discussion on CFS member design methods is presented in Chapter X.

Previous researchers have applied various combinations of design-code-based capacity evaluation methods and search algorithms in dimension and shape optimization of CFS members (Leng, 2015). The early work on numerical optimization by Seaburg and Salmon [6] applied the gradient-based Steepest Descent (SD) method to explore the dimensions of hat sections with the effective width method of the AISI (1968) specification [7]. Algorithms in this work belong to mathematical programming that requires first- or second-order (partial) derivative of the objective function.

In terms of derivative-free heuristic methods the early work is associated with Adeli et al, which developed a computational neural network model and applied it to the optimization of CFS beams with hat, I, and Z sections [8] following AISI Load Resistance Factor Design (AISI, 1991) [9] specifications. Leng et al.[10]–[13],systematically studied shape optimization of CFS columns, trough stochastic algorithms, in a search for cross-sections that can maximize the axial capacity with a given steel sheet. DSM and CUFSM were packaged to evaluate the axial capacity of columns. Several novel cross-sections were identified for intermediate-length and long-length columns with more than 100% capacity increase. In the process, they also compared the performance of gradient-based and stochastic search algorithms. Subsequently, end-use and manufacturability constraints were introduced so that optimized cross-sections are more readily fabricated and useful in practice [11].The case of a limited number of rollers was also taken into account as a reflection of fabrication cost .The modified code uses Simulated Annealing (SA) to perform stochastic search on the constrained design space. The optimized shapes offer from 50% to over 200% improvement in over reference lipped C-sections, indicating that significant enhancements can be obtained through cross-section optimization without loss of practicality and manufacturability [12]. Franco et al. [14], designed a MATLAB code with a graphical user interface using shape grammar to generate cross-sections, DSM and CUFSM to evaluate capacity, and GA to optimize design of CFS columns and beams. An earlier work of shape optimization of CFS members is credited Parastesh et al [15].They present a practical method for optimize capacity of CFS beam-column symmetric members by applying Genetic Algorithm (GA) incorporating manufactural and construction constraints under axial compression applied with different eccentricities

In general, the formulation for the optimization of a CFS framing structure, with its objective, design variables, and constraints, can be presented in the form below:

for (1)

where is the vector of design variables and is the number of constraints, expressed through constraint functions . Features of some typical optimization algorithms that have been applied in CFS structural optimization are addressed hereafter.

* 1. Gradient-based algorithms

Gradient-based algorithms have a solid mathematical background, in that Karush Kuhn Tucker (KKT) conditions are necessary for local minimal solutions. Under certain circumstances, they can also be sufficient conditions. However, solving KKT conditions directly is usually very cumbersome, in that the equations are nonlinear, so practical algorithms aim to decrease the objective function value step by step instead. This method requires the gradient of a certain function with respect to its variables. An analytical form of the gradient is not guaranteed to exist, or could be very complicated. On the other hand, finite difference approximation of the gradient can be computationally costly for simulation-based optimization due to the increased number of objective function evaluations, which is usually characteristic of structural optimizations.

SD is a simple, gradient-based optimizer that uses only first-order derivatives at the current design point to guide search. The well-established theoretical background of SD is quite intuitive: negative gradient is the SD direction in the neighborhood of the current point. SD is widely used on its own, and can also be integrated into other algorithms. The iterative scheme of SD with design variables updated at an iteration k is as follows:

(2)

where is the vector of design variables at iteration , is the vector of design variable change direction, and scalar is a step length control parameter used to ensure improvement in the objective function. For SD, the step direction is the negative of the gradient of the function to be optimized.

(3)

The gradient , formed by partial derivatives of f with respect to each component of , can be evaluated analytically or approximated by finite difference. When the vector norm of is zero, the necessary condition of a local minimum is satisfied and the algorithm has converged.

Given a step direction, a line search algorithm is used to identify the step size that produces a maximum reduction of the objective function. This is equivalent to finding the optimal value of scalar a for a function of only one variable, defined as

(4)

Trust region method is another iterative algorithm effective for unconstrained numerical optimization. The terms “trust region” relates to a quadratic model of the objective function near the design point and the radius of that neighborhood Dk, developed using the Taylor-series expansion:

(5)

(6)

where , , and is some scalar in the interval (0, 1). In the quadratic model function , an approximation of the Hessian matrix (second-order partial derivatives of the objective function) is adopted. The model function is minimized locally within the neighborhood, characterized by its radius , solving the subproblem:

f or   
 (7)

Once a step is available, the performance of the quadratic model is evaluated using the ratio as the predicted reduction of the objective function over the actual reduction of the model function:

(8)

If is close to 1, indicating a good agreement between the model mk and the objective f, it is safe to expand the trust region for the next iteration. If is positive but significantly smaller than 1, the radius of the trust region will remain the same; if is close to zero or negative, the trust region will shrink. Also, the design point will be updated as is large enough.

The subproblem (Eq. (7)) is a constrained optimization problem, but it has some unique features that make it easier to solve. A straightforward method that is related to the idea of SD is to find along the direction of , called the Cauchy point.

The trust region method has shown the power of using the quadratic model function in the optimization of general nonlinear objective functions. For general nonlinear programming problems with nonlinear objective functions and nonlinear equality and inequality constraints.

* 1. Stochastic search algorithms

Stochastic search algorithms are designed for problems with inherent random noise or deterministic problems solved by injected randomness. In structural optimization, these are problems with uncertainties of design variables or those where adding random perturbation to deterministic design variables is the method to perform the search. The search favors designs with better performance. An important feature of stochastic search algorithms is that they can carry out broad search of the design space and thus avoid local optima. Also, stochastic search algorithms do not require gradients to guide the search, making them a good fit for discrete problems. However, there is no necessary condition for an optimum solution and the algorithm must run multiple times to make sure the attained solutions are robust. To handle constraints, penalties can also be applied on designs that violate constraints. For constraints that are difficult to be formulated explicitly, a true/false check is straightforward to implement. Randomly perturbed designs are checked against constraints, and only those passing the check will enter the stage of performance evaluation. Stochastic search can be applied on one design or on a population of them, using for example SA or GA, respectively.

SA is a mimic of the natural process of annealing in metallurgy [16]. The algorithm performs iteratively; the code generates a new candidate design by randomly perturbing the variables of the current elite design that performs best. A unique feature of SA is that its “hill-climbing” property allows inferior designs to be accepted in place of elite ones to expand the search space and prevent the algorithm from becoming trapped in a low-quality local minimum. The probability that a suboptimal design is accepted is a function of the magnitude of performance loss and a user-selected parameter. This parameter is tightened as the optimization progresses, reducing the probability of accepting suboptimal designs. Two influential parameters of the algorithm are initial “temperature” and the rate at which this temperature is reduced, referred to as the “cooling rate” .The reduction of occurs when a certain number () of qualified designs has been evaluated. SA terminates after the temperature has been reduced times. The product of and is the maximum number of objective function evaluations, commonly used as an indicator of algorithm efficiency. Convergence is said to occur if the elite design does not change over a large number of iterations ().

Leng et al. [17], presents an example of using SA in CFS member optimization. Their objective was to maximize the axial capacity of a CFS column by updating its cross-section shape. The only feasibility constraint is the openness of the cross-section. As shown in the figure, if the new design is feasible and offers a higher , it becomes the elite design. The constraint of open cross-sections is realized by the true/false check on perturbed designs (ie, perturbed designs with overlapping will be rejected).

GAs are popular stochastic search algorithms based on the idea of Darwin’s evolution theory [18]. Rather than operating on a single design and its perturbation, as in SA, GA operates on a population of designs. The designs are then analyzed and ranked according to their objective function performance (fitness). The generation of a new design population includes random selection of two designs (parents) and random exchange of a portion of properties of them (reproduction). Occasionally, a design is also randomly perturbed (mutation). This process is repeated until an entire new population (children) is formed. Designs with higher fitness have a higher probability of being selected as parents, and thus the performance of the population as a whole should improve as the optimization progresses. Similar to other stochastic search algorithms, GA terminates if either a maximum number of iterations is achieved (), or convergence is detected. Convergence is said to occur if the elite design does not change over a large number of iterations ().

Leng et al. [10], presents an example of using GA in CFS member optimization. Similar to SA example [17], objective again is to maximize the column’s axial capacity. In the formulation, the vector x of a given design is rounded to a user-specified precision and converted into a binary string. This is a straightforward process and facilitates the exchange of information between designs. Parent selection is based on the roulette wheel algorithm, and single-point cross-over is used to exchange information between two parents. To handle the constraint on overlapping, the penalty method is used. If element crossing is detected in a new design, the computed strength is penalized by subtracting a large number. The procedure in the flow chart can be readily applied to other design objectives

The Ant Colony Optimization (ACO) is another type of metaheuristic algorithm, inspired by the behavior of ants seeking a path between their colony and a source of food. An ant randomly looks for food and returns to the colony once a food source is located, leaving a chemical called pheromone on its trail so that other ants are likely to follow this path and reach the food. However, the pheromone trail evaporates over time. As a result, a shorter pheromone trail will be more frequently visited before it evaporates and thus the pheromone density becomes higher. The idea of ACO mimics this process, so the optimal solution will be found like the identification of the shortest pheromone trail.

Sharafi et al. [19] formed the ant colony by placing “simulated ants” at nodes of the grid of feasible paths. The route of an ant depicts a cross-section shape, and the pheromone density and its update are tied to the performance of the member with the generated cross-section. ACO belongs the general category of swarm intelligence algorithms, among others like Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC) algorithm, bat algorithm, etc. In Chapter X, depth discussion about PSO and swarm intelligence, are developed.

* 1. Examples of shape optimization of CFS members

As discussed, dimension optimization of CFS members requires a designated cross-section shape so that the physical quantities characterizing the shape become design variables. The selection of design variables is dependent on the specific shape, and minimum weight is usually seen as the objective. Optimal designs are identified for several span lengths or load cases, and the sensitivity of the objective with respect to design variables can be predicted. Since the designated cross-section shapes, span lengths, and load cases vary from one publication to another, direct comparison of past results is not feasible.

The results of shape optimization of CFS members are more illustrative, since optimized shapes are identified. Although a rigorous benchmark problem is not yet available, a typical problem investigated by several researchers since Liu et al. [20] is the search for an optimal cross-section shape with a fixed length of steel sheet so the CFS column’s axial capacity is maximized. Leng et al. .[10]–[13] tackled this by first using unconstrained shape optimization for more expansive exploration of the design space, and then implementing a series of manufacturability and end-use constraints to identify constrained optimal designs with much improved practicality. This research influenced other scholars whose results are presented in later publications, so it is introduced here as an archetype.

The initial design is a lipped channel section Figure 1 for unconstrained and constrained shape optimization. However, the discretization of cross-sections and formulation of design variables differ: a cross-section is meshed by equal-width finite strip elements in CUFSM so that the turn angles between them form the design vector of unconstrained optimization Figure 1but both strip widths and turn angles are design variables that account for the limit on the number of rollers in constrained optimization Figure 2.

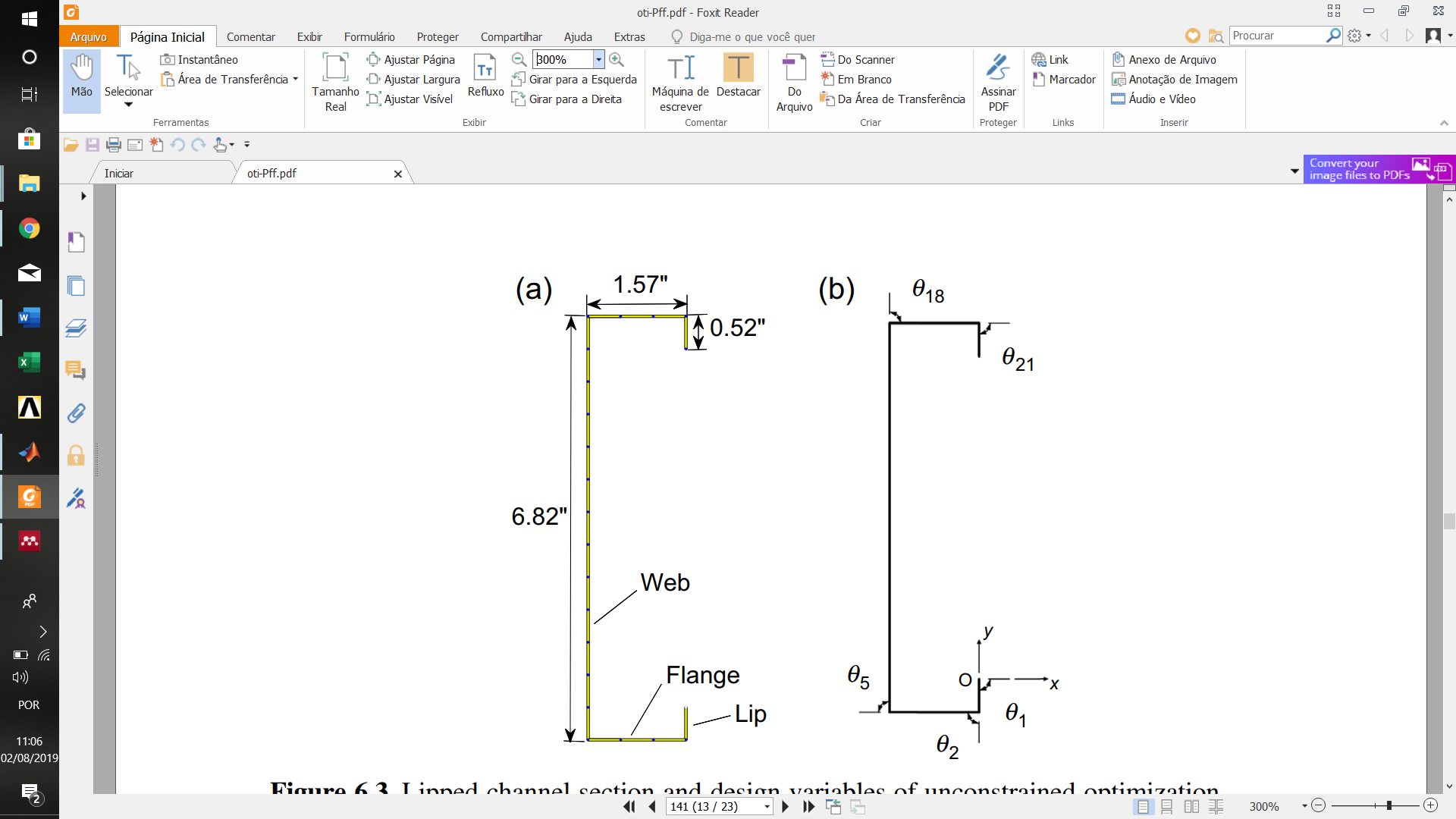


Figure 1: Lipped channel section and design variables of unconstrained optimization.(a) Dimensions and mesh of lipped channel section; (b) nonzero turn angles of lipped channel section

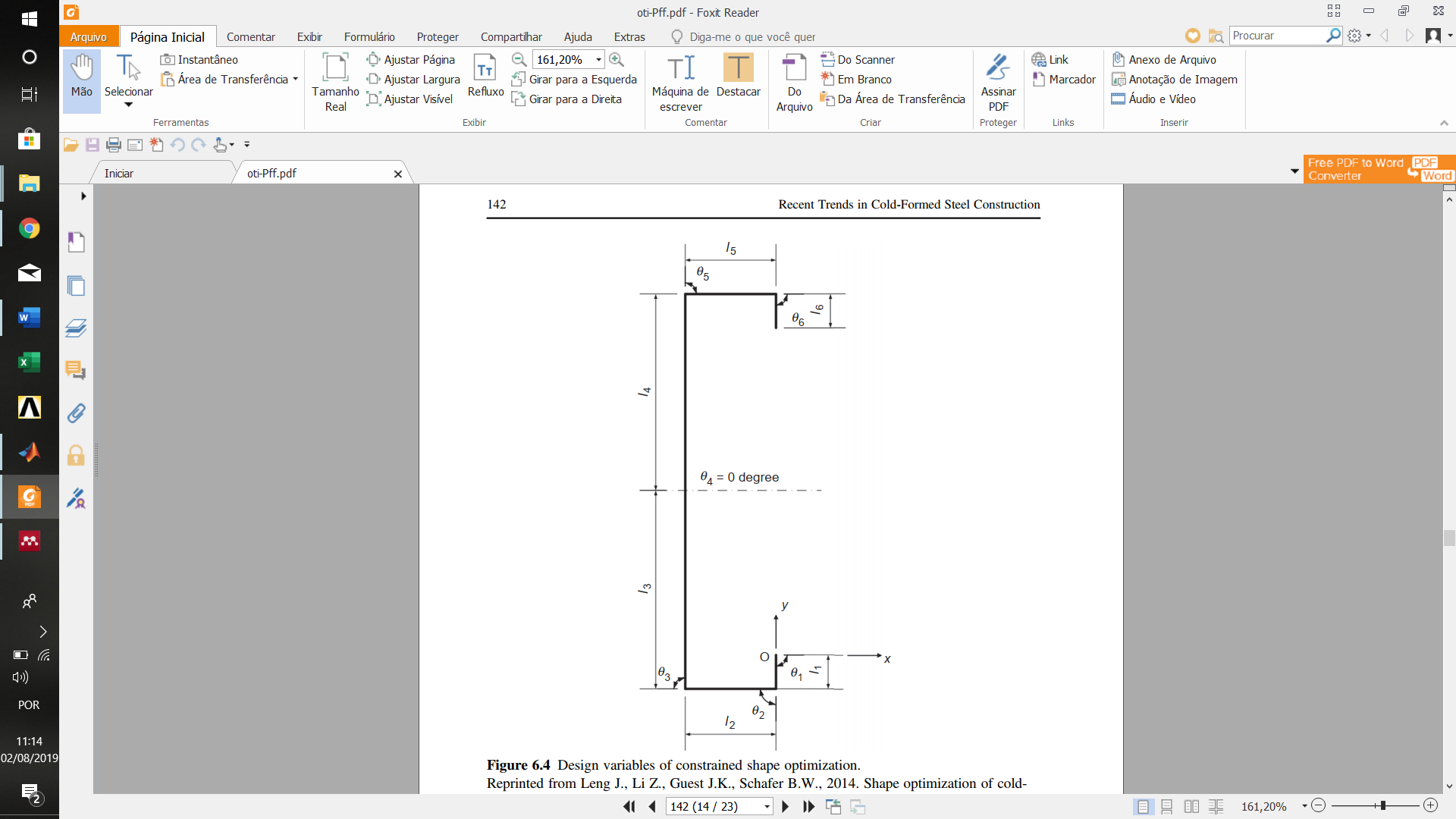


Figure 2: Design variables of constrained shape optimization.

The algorithms for unconstrained shape optimization include gradient-based (SD) and stochastic search algorithms (SA and GA). The intention was to make a thorough search for optimized designs close to global optima, and to compare the performance of algorithms. For the purpose of comparison, SD was also applied at initial cross-sections other than the lipped channel, like hat and sigma sections. SA and GA were performed with 10 replications to check the robustness of solutions. The manufacturability and end-use constraints include symmetry and antisymmetry, parallel flanges and dimension constraints, utility pass-through allowance, open section, and limit on the number of folds (rollers). Most of these constraints cannot be formulated analytically, so the true/false check approach was adopted and coded under the framework of SA for solving constrained optimization. To study the influence of controlling buckling mode on optimization results, unconstrained optimization considered 4 ft (1220 mm) and 16 ft (4880 mm) long columns, and a 2 ft (610 mm) column length is further included in constrained optimization.

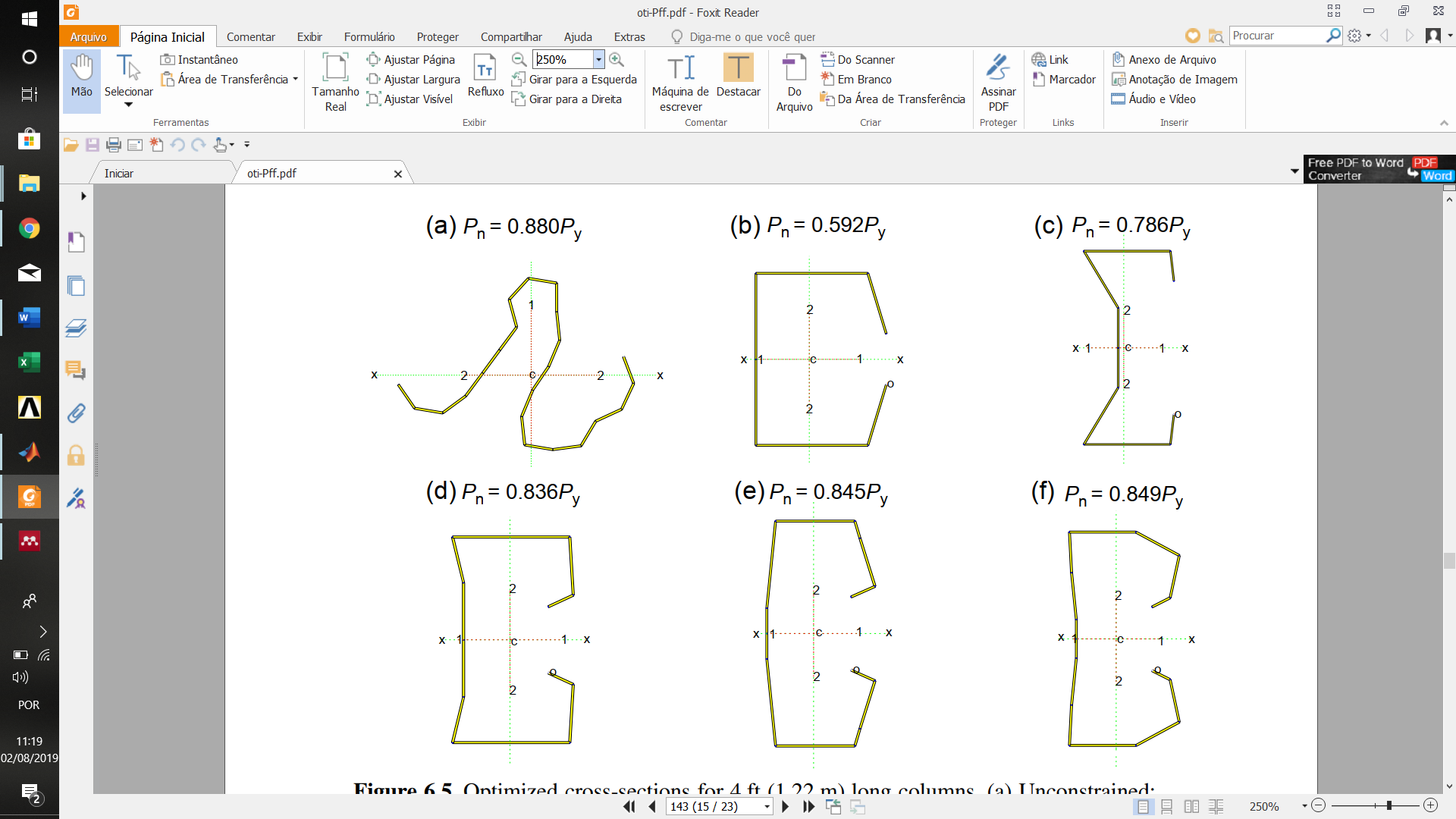


Figure 3: Optimized cross-sections for 4 ft (1220 mm) long columns. (a) Unconstrained; (b) constrained, four rollers; (c) constrained, six rollers; (d) constrained, eight rollers; (e) constrained, 10 rollers; (f) constrained, 12 rollers.

The results and conclusions from this work are comprehensive. Firstly, a number of novel cross sections were identified by unconstrained optimization. The “bobby pin” and squashed “S” sections in Figure 3(a) and is example. Secondly, the implementation of manufacturability and end-use constraints successfully drives the search toward much more practical designs (that are still able to retain a significant increase in axial capacity. Thirdly, constrained optimization results suggest that practical optimal designs can be produced with only marginal increase of manufacturing cost. Constrained optimized sections can still achieve 50% capacity growth from the initial lipped channel (C) section even with only four rollers. Last but not least, comparison of algorithm performance indicates SD is a local optimizer that is dependent on the initial design; while SA and GA are both capable of searching the design space globally, but SA is preferred for this problem since it runs only on one design instead of a population and saves computational cost.

References

[1] J. Leng, *Optimization techniques for structural design of cold-formed steel structures*. Elsevier, 2016.

[2] J. F. A. Madeira, J. Dias, e N. Silvestre, “Multiobjective optimization of cold-formed steel columns”, *Thin-Walled Struct.*, vol. 96, p. 29–38, 2015.

[3] AISI, “S100-07/S2-10, North American Specification for the Design of Cold-Formed Steel Structural Members, Supplement No.2”, *American Iron and Steel Institute*, Washington, DC, 2010.

[4] ABNT, “NBR 14762 -Dimensionamento de estruturas de aço constituídas por perfis formados a frio”, *Associação Brasileira de Normas Técnicas*, Rio de Janeiro,RJ, p. 237, 2010.

[5] European Committee for Standardization, “Eurocode 3: Design of steel structures, Part1-3: General rules for cold formed thin gauge members and sheeting”, *European Convention for Constructional Steelwork*, Brussels, Belgium, 1996.

[6] C. G. Seaburg, P.A., Salmon, “Minimum weight design of light gage steel members”, *J. Struct. Div.*, vol. 97, p. 203–222, 1971.

[7] AISI, “Specification for the Design of Cold-formed Steel Structural Members”, *American Iron and Steel Institute*, Washington, DC, 1968.

[8] H. Adeli e A. Karim, “Neural Network Model for Optimization of Cold-Formed Steel Beams”, *J. Struct. Eng.*, 2002.

[9] AISI, “Load and Resistance Factor Design Specification for Cold-formed Steel Structural Members”, *American Iron and Steel Institute*, Washington, DC., 1991.

[10] J. Leng, J. K. Guest, e B. W. Schafer, “Thin-Walled Structures Shape optimization of cold-formed steel columns”, *Thin Walled Struct.*, vol. 49, no 12, p. 1492–1503, 2011.

[11] J. Leng, Z. Li, J. K. Guest, e B. W. Schafer, “Constrained shape optimization of cold-formed steel columns”, in *21st International Specialty Conference on Cold-Formed Steel Structures - Recent Research and Developments in Cold-Formed Steel Design and Construction*, 2012.

[12] J. Leng, Z. Li, J. K. Guest, e B. W. Schafer, “Shape optimization of cold-formed steel columns with fabrication and geometric end-use constraints”, *Thin-Walled Struct.*, 2014.

[13] Z. Li, J. Leng, J. K. Guest, e B. W. Schafer, “Thin-Walled Structures Two-level optimization for a new family of cold-formed steel lipped channel sections against local and distortional buckling”, *Thin Walled Struct.*, vol. 108, p. 64–74, 2016.

[14] J. Moara, S. Franco, J. Pinto, E. De Miranda, e A. Landesmann, “Thin-Walled Structures Shape Grammar of steel cold-formed sections based on manufacturing rules”, *Thin Walled Struct.*, vol. 79, p. 218–232, 2014.

[15] H. Parastesh, I. Hajirasouliha, H. Taji, e A. Bagheri Sabbagh, “Shape optimization of cold-formed steel beam-columns with practical and manufacturing constraints”, *J. Constr. Steel Res.*, vol. 155, p. 249–259, 2019.

[16] S. Kirkpatrick, C. D. Gelatt, e M. P. Vecchi, “Optimization by simulated annealing”, *Science (80-. ).*, 1983.

[17] J. Leng, J. K. Guest, e B. W. Schafer, “Shape optimization of cold-formed steel columns”, *Thin-Walled Struct.*, vol. 49, no 12, p. 1492–1503, 2011.

[18] J. H. Holland, *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence*. 1975.

[19] P. Sharafi, L. H. Teh, e M. N. S. Hadi, “Shape optimization of thin-walled steel sections using graph theory and ACO algorithm”, *J. Constr. Steel Res.*, 2014.

[20] H. Liu, T. Igusa, e B. W. Schafer, “Knowledge-based global optimization of cold-formed steel columns”, *Thin-Walled Struct.*, 2004.