Optimization of fin arrays with a genetic algorithm

Lino Mediavilla[1], Javier Miranda[2]

[1,2] Universidad San Francisco de Quito, Department of Mechanical Engineering

Abstract

The use of fins extending from walls to enhance heat transfer from a body to a surrounding fluid is a widespread practice. The design of these devices has been approached empirically, but theoretical analyses have been carried out for a variety of fin shapes and arrays as well. In this paper, a software prototype with built-in theoretical expressions for arrays of straight fins is introduced, with the novelty that it not only automates the thermal analysis task, but also finds the optimal design parameters for an array of straight fins under arbitrary circumstances. This is accomplished by maximizing the heat transfer rate from the array while subjected to geometric and economic constraints. The optimization software introduced here takes advantage of a powerful, meta-heuristic search method known as genetic algorithm.

Keywords: *Extended Surface Design, Engineering Optimization, Genetic Algorithms, Meta-heuristic methods.*

Introduction

There are practical cases in which it is desired to increase the heat transfer rate from a body to an adjoining fluid, in order to keep the body below a certain temperature. Among increasing the fluid convection coefficient, reducing its temperature or increasing the contact area of the body with the fluid, the latter is typically preferred, for its feasibility. Hence, *fins* extending from the wall into the fluid are used. It is desired that the temperatures from the base to the tip of a fin vary as little as possible. Actually, “in the limit of infinite thermal conductivity, the entire fin would be at the temperature of the base surface.” [1]

The design of *fins* has been approached both empirically and theoretically. The former is useful when there are no exact or numerical solutions available, so expressions have to be derived experimentally. Two examples of this are the empirical correlations included in Webb (1994) for the design of finned tubes, and Manglik & Bergles (1995) for plate-fin extended surfaces [4].

In their book *Fundamentals of Heat and Mass Transfer*, Bergman, Dewit, Lavine & Incropera approach the problem theoretically and derive a general form of the energy equation for any extended surface. They also present solutions for different fin shapes and boundary conditions. Simplifying assumptions were made.

This first software prototype is built upon the equations resulting from the latter approach, specifically those corresponding to heat transfer rates and efficiency of arrays of *straight fins*.

The thermal performance of a fin array is not only determined by its geometry, but also by the material it is made of. At first glance, it seems like the materials with the highest conductivities should be preferred without hesitation, but there's another key factor involved in engineering design: the cost.

In order to find the design parameters that would maximize heat transfer from the fin array to the fluid, while taking into account economic and geometric constraints, a *genetic algorithm* (GA) was implemented.

The GA and the expressions resulting from the theoretical analysis of heat transfer from arrays of *straight fins* were packed in a software prototype, which includes a graphical user interface (GUI). Currently, the prototype runs in Matlab (version R2013b or later).

In the GUI, users can specify the geometry of the wall that the fin array will extend from, a minimum pitch between the fins, boundary conditions, a file with material properties, a minimum heat rate to accomplish, and a budget constraint.

Heat transfer from extended surfaces

An extended surface attached to a plane wall is considered a *straight fin*. Its cross sectional area may be non-uniform, but it cannot be circular, since that would make it a *pin fin* [1].

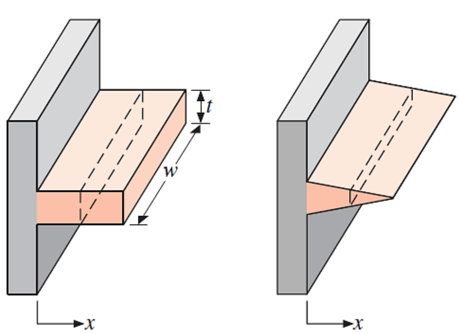

Figure 1. Straight fins: uniform cross-section (left) & non-uniform cross-section (right) [1].

There is a third type: *annular fins,* which are "circumferentially attached to cylinders" [1].

Bergman et. al, performed a general conduction analysis on extended surfaces under the following assumptions [1]:

- The temperature only changes in the x-direction (see figure 1) and at any point in x, it is uniform across the fin thickness.
- Negligible radiation from the surface.
- Uniform convection coefficient over the surface.
- Steady state conditions.
- Constant thermal conductivity.

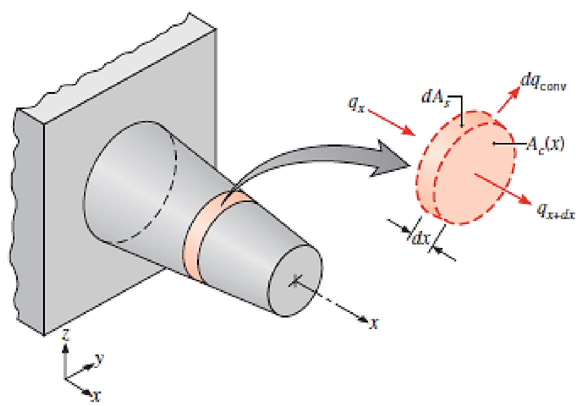

Figure 2. Energy balance in a differential element of an extended surface [1].

The energy balance described in figure 2 yields:

$$q_x = q_{x+dx} + dq_{conv} \qquad (1)$$

The amount of heat leaving the element is:

$$q_{x+dx} = q_x + \frac{dq_x}{dx}dx \qquad (2)$$

And from Fourier's law:

$$q_x = -kA_c\frac{dT}{dx} \qquad (3)$$

Where A_c is the cross-sectional area (not necessarily constant) .

Then eqn. 2 becomes:

$$q_{x+dx} = -kA_C\frac{dT}{dx} - k\frac{d}{dx}\left(A_C\frac{dT}{dx}\right)dx \qquad (4)$$

The convection heat transfer rate can be expressed by:

$$dq_{conv} = hdAs(T - T_\infty) \qquad (5)$$

Substituting (4) and (5) in (1) leads to:

$$\frac{d^2T}{dx^2} + \left(\frac{1}{A_c}\frac{dA_c}{dx}\right)\frac{dT}{dx} - \left(\frac{1}{A_c}\frac{h}{k}\frac{dA_s}{dx}\right)(T - T_\infty) = 0 \qquad (6)$$

Equation (6) is the general energy equation for an extended surface [1]. Note that the cross-sectional area was at no point assumed constant.

The temperature distribution $T(x)$ for an arbitrary fin shape results from solving equation (6) with the appropriate boundary conditions and an expression for the cross-sectional area $A_c(x)$.

It is common to work with an *excess temperature* defined as $\theta\,(x) = T(x) - T_\infty$ [1].

Once the temperature distribution is calculated, the heat transfer rate from a fin can be known by applying Fourier's law at the fin base, which yields [1]:

$$q_f = q_b = -kA_c \left.\frac{d\theta}{dx}\right|_{x=0} \qquad (7)$$

The *n-efficiency* η_f provides a measure of the thermal performance of a fin. It can be expressed as a ratio of the actual heat transfer rate from the fin and the heat transfer rate it could accomplish if the temperature across the entire fin were the same as in its base [1]:

$$\eta_f = \frac{q_f}{hA_f\theta_b} \qquad (8)$$

Where $\theta_b = T_b - T_\infty$. In this first prototype, only *straight fins* were considered for the design search space. Table 1 contains a summary of the equations that yield the efficiencies for three types of straight fin profiles.

Arrays of fins

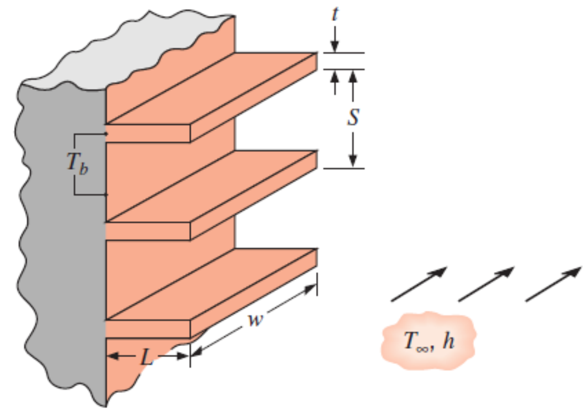

Figure 3. An array of straight fins [1].

When analyzing arrays of fins, the measure of thermal performance is provided by the overall surface efficiency, which is expressed as a ratio between the actual heat transfer rate from the array and the heat transfer rate it could achieve if all of it were at the same temperature as the base [1]:

$$\eta_o = \frac{q_t}{hA_t\theta_b} \qquad (9)$$

Where A_t is a total surface area involving the fins and the sum of the exposed portions of the base [1]:

$$A_t = NA_f + A_b \qquad (10)$$

The total heat rate from an array of fins is expressed as [1]:

$$q_t = hA_t\left[1 - \frac{NA_f}{A_t}(1 - \eta_f)\right]\theta_b \qquad (11)$$

It is clear that equation (11) depends on the type of profile chosen for the array of fins, since the surface area and fin efficiency vary from one profile type to another.

Fin arrays may be attached to a wall by means of machining them as *integral* parts of it, or by a metallurgical or adhesive joint [1]. Only if the array is *integral* with the wall there will be no contact resistance at the base. This is the approach chosen for the software prototype presented here

Table 1. Area and efficiency equations for straight fins [1].

Fin Profile	A_f	A_p	η_f
Rectangular	$2wL_c$	tL	$\frac{\tanh(mL_c)}{mL_c}$
Triangular	$2w[L^2 + (t/2)^2]^{1/2}$	$(t/2)L$	$\frac{1}{mL}\frac{I_1(2mL)}{I_o(2mL)}$
Parabolic	$w\{C_1 L + (L^2/t)\ln[(t/L) + C_1]\}$	$(t/3)L$	$\frac{2}{[4(mL)^2 + 1]^{1/2} + 1}$

$L_c = L + (t/2)$; $C_1 = [1 + (t/L)^2]^{\frac{1}{2}}$; $m = (2h/kt)^{1/2}$; I_1, I_0: modified Bessel equations of the 1^{st} kind

Genetic Algorithms

Genetic algorithms (GAs) are search procedures based on the mechanism of natural evolution [2]. What mostly drives such algorithms is the idea that Darwin introduced in 1859 and was then formalized by Mendel in 1865, which is that through genetic inheritance, individuals could experience an increase in fitness (a measure of adaption degree with respect to the environment) [2].

GAs are considered robust methods because they search from a population of points instead of just one. Through probabilistic transition rules, crossover, survival of the best individuals and mutations from one generation to another, the initial GA population evolves until a convergence criterion is reached [2]. (See flowchart diagram in figure 5).

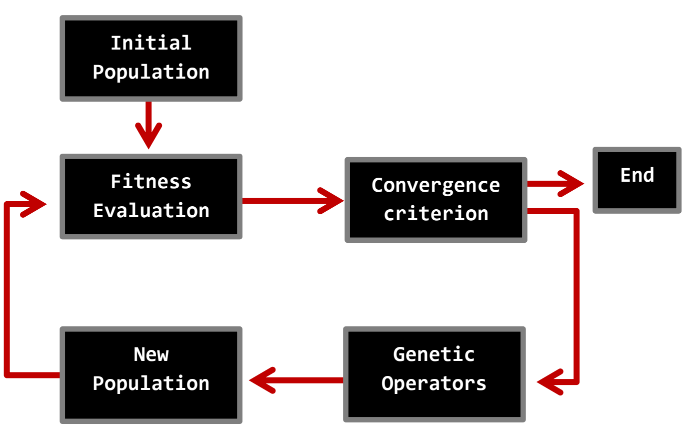

Figure 4. Genetic Algorithm flowchart diagram [2].

The downside of genetic algorithms is that fitness evaluations could represent a high computational cost. In this case, the computational cost was not a major factor.

To operate correctly, the GA needs a defined search space. This differs from the initial guess or 'seed' used in gradient-based optimization methods, because it merely implies placing bounds on the variables to keep the GA from spending time evaluating solutions that would make no sense in the context of the problem; e.g. a negative number of fins.

Additional constraints can be placed according to the problem and the user's needs. The constraint – handling method used in the prototype is discussed in this section. Individuals that do not violate any constraint are said to be inside the *feasible* region.

Population

Each individual in this software's underlying genetic algorithm is represented by a *chromosome*, which in turn is composed of 6 *genes*, each corresponding to a design variable. The 6 components of a *chromosome* are structured in the following way:

1. Number of fins: an integer between 1 and a maximum number calculated according to the

base body height and minimum pitch between the fins of the array.

2. Width: double-precision floating point number that represents the width of the fins in the array
3. Thickness: double-precision floating point number that represents the thickness of the fins in the array
4. Length: double-precision floating point number that represents the length of the fins in the array. Limited to $2.65/m$, where $m = \sqrt{2h/kt}$ for all three profile types because it makes no sense for the fins to extend beyond that length [3].
5. Profile type: an integer between 1 and 3 corresponding to rectangular, triangular or parabolic profile.
6. Material: an integer between 1 and the maximum number of materials in the file. The number in this position is used only as an index. Then the thermal conductivity and density of the corresponding material are extracted.

The first generation consists of individuals distributed evenly over the n-dimensional search space. In a 2-dimensional search space it is easy to visualize the initial population in a grid-like configuration (see figure 6). This is a way to improve the initial search space exploration. Once the ‘evenly-spaced’ chromosomes have been generated, the GA is ready to proceed onto the *fitness evaluation* stage.

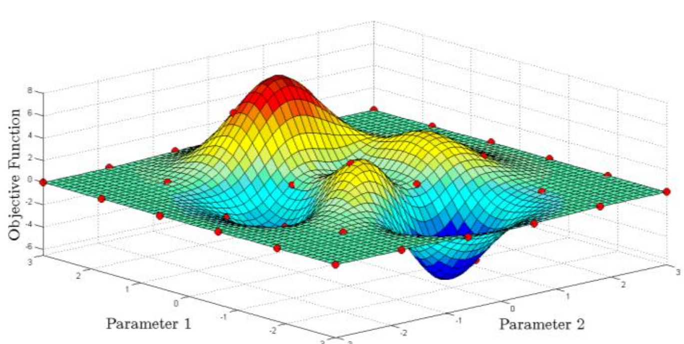

Figure 5. GA initial population on a surface with multiple local maxima.

Fitness evaluation

The concept of fitness is key to a genetic algorithm, in that it is the only way the algorithm can distinguish good individuals from bad individuals. The former are the ones supposed to drive the population evolution towards the global maximum or minimum of the fitness function over a defined search space. Seeking for convergence to a maximum or to a minimum depends on the problem.

A fitness function is of the form:

$$f\colon \mathbb{R}^n \to \mathbb{R}$$

Which means it is a scalar function that takes an n-dimensional input, where n is the number of variables, and returns a scalar. When an individual’s *chromosome* is passed to the *fitness function*, it returns the *fitness* of the corresponding individual [3].

In this software, the fitness function is simply the heat rate resulting from a set of straight fin array parameters, i.e. profile type, dimensions, material and number of fins. Each individual’s *chromosome* represents a fin array parameter set.

Since the quantity measured by the fitness function is a heat rate, this is a maximization problem and the best individual’s *chromosome* at the end of a run will represent the fin array parameters that yielded the maximum heat rate among all the evaluated parameter sets in the *feasible* region.

As the definition of the genetic algorithm suggests; the top scoring individuals are rewarded with higher probabilities of mating and surviving on to the next generation with their ‘genetic’ material unaltered.

Constraints

One major advantage of the GA is that it allows handling linear and non-linear constraints with ease. The lower and upper bounds on the dimensions of the fin array are already one type of constraint, but the most challenging ones are the minimum heat rate and the maximum total cost of a fin array. In this algorithm they are handled by penalizing the fitness value of an individual that violates any of these constraints. Note that this is not equivalent to simply discarding the individual; after applying penalization, a non-zero probability of being picked for selection and crossover still exists.

This strategy is very common, because an individual that violates a constraint may still provide some useful information. More often than not, global maxima in a constrained problem are located in the borders of the feasible region, so it could be worthy to have the GA explore areas near to the borders defined by the constraints [2].

The total cost of a fin array of a certain material is determined by the product of the cost per unit mass of the material and the total mass used. Another factor that heavily influences costs is the complexity of manufacturing. Currently, the effect of this is implemented through cost amplifying factors. For instance, rectangular profiles were assigned an amplifying factor of 1, the triangular profiles were assigned a 1.5, and the parabolic profiles were assigned a 2.5.

Selection

In the GA implemented, a fraction of the population survives on to the next generation with its genes unaltered. Tournament mode is used by default.

It consists of picking individuals from the population at random, to form subgroups in which the individuals with the highest fitness are declared winners. These winners are automatically allowed to survive on to the next generation with their genetic material intact, and are inserted into a mating pool to form couples and perform crossover [3].

Crossover

The crossover method used in this algorithm creates new *offspring* from a couple of *parents* by means of linear combinations between the parents' *chromosomes.*

Let $\overline{\boldsymbol{m}}$, $\overline{\boldsymbol{d}}$ be the parents' chromosomes. Then,

$\overline{\boldsymbol{m}}$, $\overline{\boldsymbol{d}}$ $\in \mathbb{R}^n$, where n is the number of variables.

The resulting offspring is calculated as follows:

$$\overline{\boldsymbol{offspring_1}} = \overline{\boldsymbol{m}} - \langle \beta_1(m_1 - d_1), \dots, \beta_n(m_n - d_n) \rangle$$

$$\overline{\boldsymbol{offspring_2}} = \overline{\boldsymbol{d}} + \langle \beta_1(m_1 - d_1), \dots, \beta_n(m_n - d_n) \rangle$$

Where $\beta_1, \dots, \beta_n$ are independent uniform random numbers between 0 and 1 [3].

Mutation

Mutation on an individual is performed by replacing one of it *genes* with a random uniformly distributed number ranging from the lower to the upper bounds of the corresponding variable.

At each generation, a set of individuals are picked randomly from the population (excluding the best individual) according to a mutation probability parameter. For every individual in this set, a random integer is picked from a uniform distribution between 1 and n (since Matlab arrays start at index 1) and the gene at the corresponding position in its chromosome is mutated as described above.

New population

The new population at the end of each generation contains the surviving individuals, the crossover offspring, and the mutated individuals.

Convergence Criterion

Currently, a maximum of 150 generations is set as the convergence criterion.

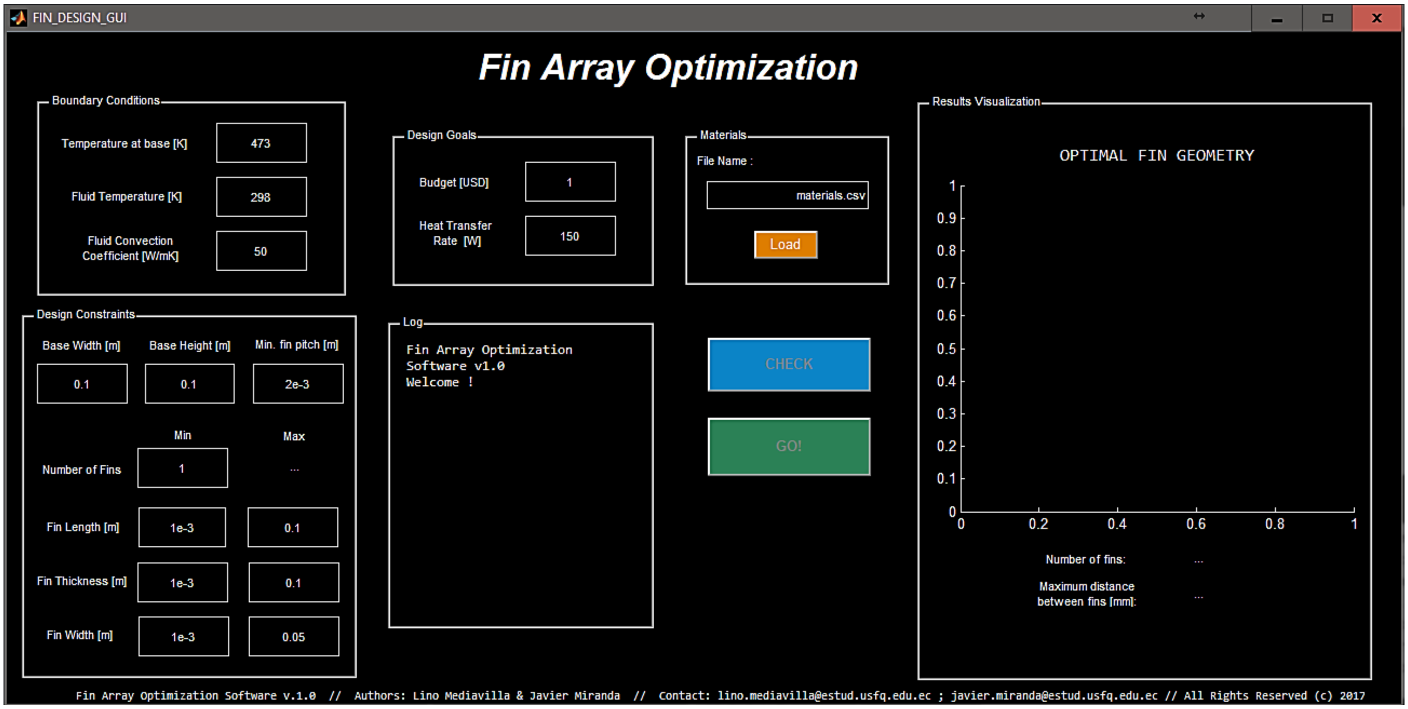

Figure 6. Fin Array Optimization GUI implemented in Matlab

Implementation & Results

The GUI

The graphical user interface in figure 7 was developed so that this computational tool could be used with ease. The program interprets all inputs in SI units, and all calculations are performed in SI units as well.

The user can input the following data:

- Boundary Conditions: the temperature at the base, the fluid temperature and the fluid convection coefficient (assumed constant).
- Design Goals: a maximum budget for the design, and the minimum heat transfer rate required from the fin array.
- Design Constraints: the dimensions of the base, the minimum pitch between fins, a minimum number of fins, lower and upper bounds for fin length, thickness and width.
- Materials File: a file with material names, conductivities, densities (both assumed constant in the working temperature range), and the cost per unit mass. The possibility of adding a customized materials file makes the results more meaningful because the algorithm will be working with the user's design options and constraints more closely. The recommended input file format is .csv .

When the program launches, all the boxes for input show preset values, but these can be overwritten by the user. The CHECK button must be activated before execution. This contains internal methods to check for constraints coherence, missing data, and failure to read the input file. If no issue arises, the GO button becomes enabled so the user can proceed to the optimization stage. Every time a change in the input data is made, the "GO" button becomes disabled until a new check with the "CHECK" button is performed.

Performance Tests

The materials and properties used in all the tests are listed in table 2. The prices per unit mass were arbitrarily assigned for the sake of the tests.

Name	K [W/mK]	ρ [kg/m^3]	Cost [USD/kg]
Al-2024-T6	177	2770	10
Al-195-Cast	168	2790	10
Pure Cu	401	8933	30
Cartridge brass	110	8530	20
Gold	317	19300	1000

Table 2. Contents of the materials file for the tests presented here. K and ρ were taken from [1].

In all the following cases, the base temperature was fixed at 473 [K], the fluid temperature at 298 [K] and the fluid convection coefficient at 50 [W/mK].

- Test # 1: An extreme case without budget constrain and a minimum heat transfer rate requirement of 350 [W]. (See annex 1.1 for details).
  The most optimal array turned out to be 14 fins of rectangular profile, made of Pure Cu. The heat transfer rate achieved was **1138.65 [W]**. Triangular and Parabolic profiles of the same dimensions and material would have yielded **1060.15 [W]** and **951.22 [W]** respectively.

- Test # 2: An extreme case with a very low budget (**.005 USD**) and a minimum heat transfer rate requirement of **80 [W]**. (See annex 1.2 for details).
  In this case, the most optimal array turned out to be simply 1 fin of rectangular profile, made of Al-2024-T6 alloy. The heat transfer rate achieved was **88.13 [W]** with a total cost of **0.00499 USD.** Triangular and Parabolic profiles of the same dimensions and material would have yielded **87.62 [W]** and **86.35 [W]** respectively.

- Test # 3: Features a more relaxed budget constraint (**1 USD**) and a minimum heat transfer rate requirement of **150 [W].** (See annex 1.3 for details).
  This test brought up very interesting results: the optimal array was found to be 12 fins of parabolic profile made of AL-20214-T6 alloy. The heat transfer rate achieved was **237.83 [W]**, with a total cost of **0.9997 USD**, whereas fins of triangular profile of the same dimensions and material would have yielded **252.34 [W]** but a total cost of **USD 1.125**. Similarly, fins of rectangular profile would have yielded **267.80 [W]** but a cost of **USD 1.4996**.

Discussion

From tests # 1 and # 2, it is clear that under the same circumstances, i.e. same boundary conditions, material and dimensions, rectangular fins are more efficient in terms of heat transfer rate.

In test # 1, in which the budget was not a constraint, the most optimal profile was rectangular because the volume of the fin was not even a factor. In test case # 2 however, the budget was a very critical factor, and the main reason why the optimal fin profile turned out to be rectangular was simply the cost amplifying factor associated with the complexity of manufacturing each geometry.

Test # 3 showcases the effect of the constraint handling method implemented in this algorithm. Even though the goal of the GA is to find the fin array parameters that maximize the heat transfer rate, individuals that violate the budget constrain are heavily penalized. The solution found is satisfactory because it satisfies both the thermal performance requirement and the budget constraint.

Conclusions

The software prototype presented allows to find the best possible array of straight fins under arbitrary design constraints.

The possibility of adding a customized material property file makes the results more meaningful, especially with respect to the economic constraints.

Remarks

Although the results from the thermal analysis may be encouraging, the mechanical properties should also be assessed to ensure that the potential designs are safe. Future versions of this work should focus on coupling the thermal analysis with a mechanical analysis.

The resulting mechanical response of the fin arrays could be included as another constraint for the GA to penalize geometries that would be optimal in terms of thermal performance, but would fail mechanically.

An enhancement to the underlying thermal model could be the implementation of contact resistance equations, to add the possibility of separately manufactured fins to the design search space.

Another aspect that could be improved is the way in which the algorithm takes into account the additional costs of manufacturing the more complicated geometries. Currently, this is implemented through cost amplifying factors.

There could be a set of constraints such that an excessively reduced feasible region is generated. This might conflict with the design goals, namely, the budget or the minimum heat rate, making it impossible to entirely satisfy one or all of them. In that case, the user should consider relaxing the constraints.

References

[1] Bergman, T., Dewit, D., Lavine, A. and Incropera, F. (2011). Fundamentals of heat and mass transfer. New Jersey: John Wiley & Sons.

[2] De Santos, C. (2015) Backanalysis Methodology Based on Multiple Optimization techniques for Geotechnical Problems. PhD Thesis. Universitat Politécnica de Catalunya - BarcelonaTECH. Deparment of Geotechnical Engineering and Geo-Sciences. Barcelona, 2015.

[3] Haupt, R. L., & Haupt, S. E. (2004). Practical genetic algorithms. Hoboken, NJ: Wiley-Interscience.

[4] Kreith, F., & Goswami, D. Y. (2005). *The CRC handbook of mechanical engineering*. Boca Raton: CRC Press.

Annex

1. Performance Test Results

1.1 Test # 1

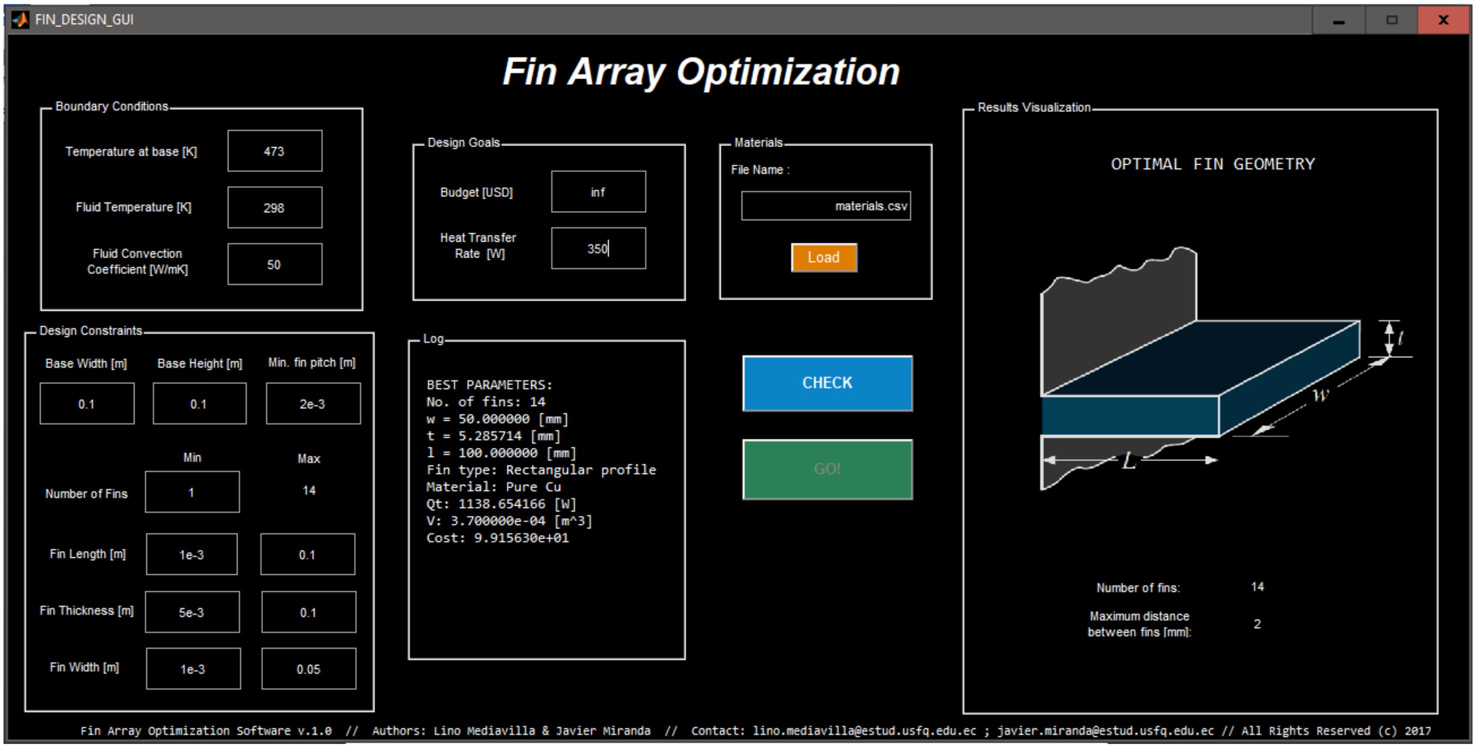

Figure A.1.1 Results of an extreme case without budget constraint.

1.2 Test # 2

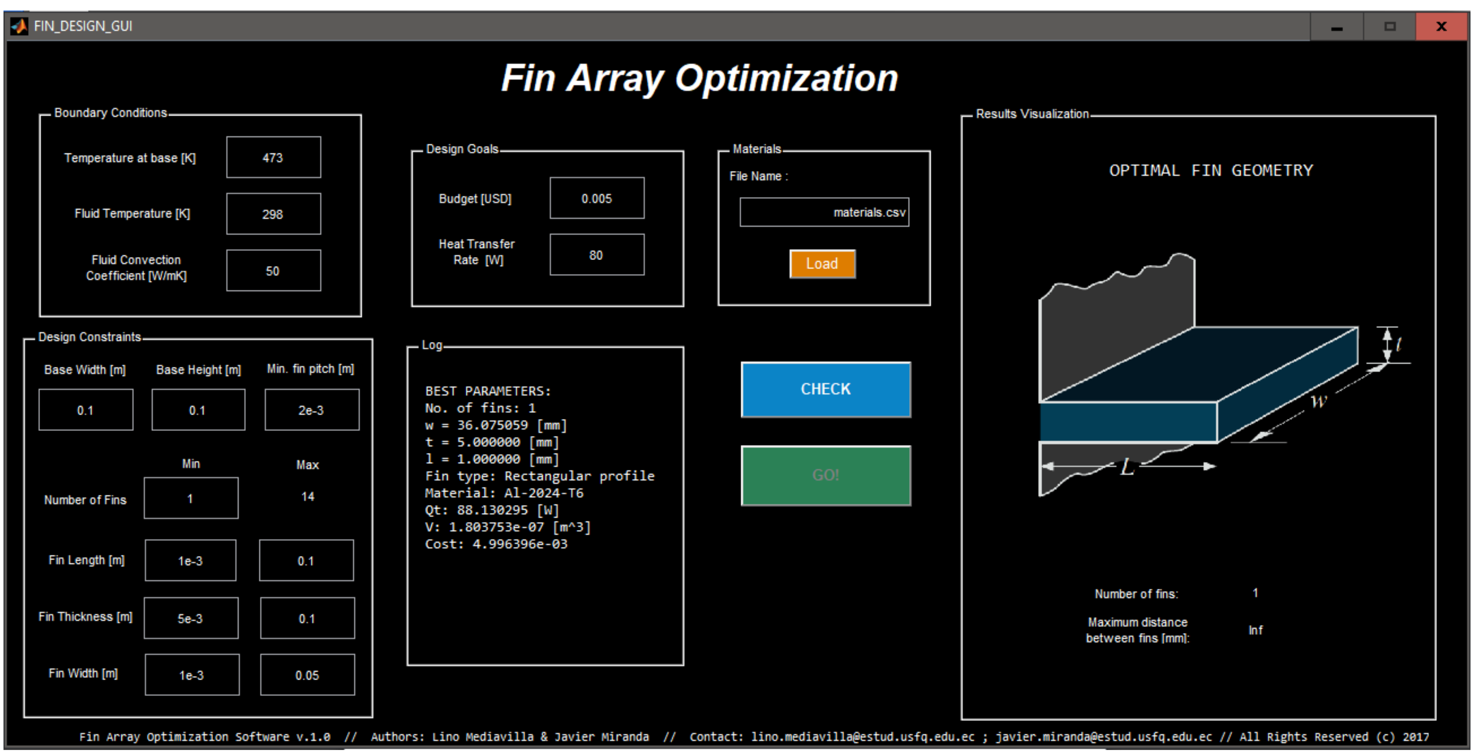

Figure A.1.2 Results of an extreme case with a very low budget.

1.3 Test # 3

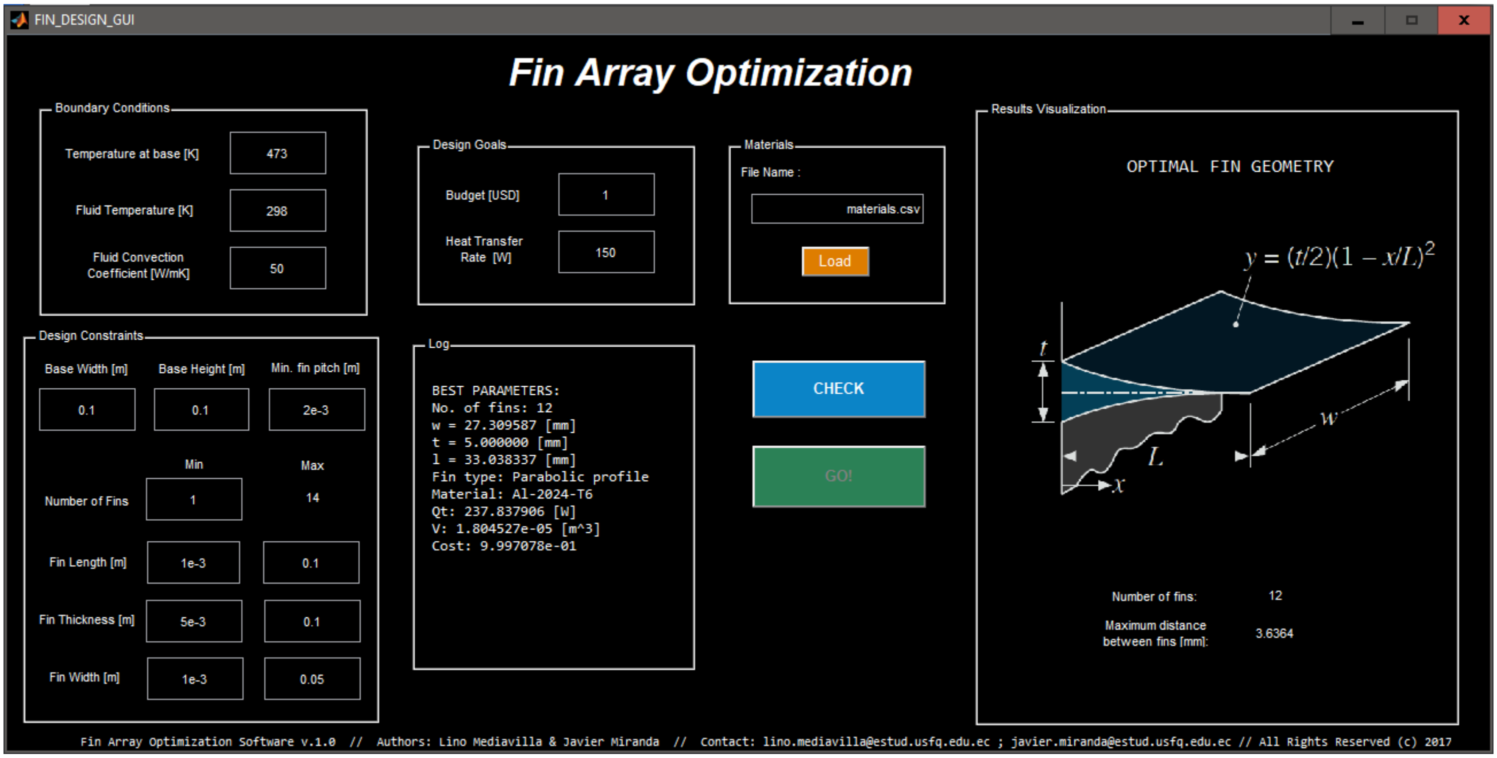

Figure A.1.3 Results of a case with a more relaxed budget constraint (with respect to Test # 2).