GEOMETRIC OPTIMIZATION OF CONCENTRATING SOLAR COLLECTORS USING MONTE CARLO SIMULATION

A. J. Marston, K. J. Daun, and M. R. Collins
Department of Mechanical and Mechatronics Engineering
University of Waterloo
Waterloo, Ontario, Canada

ABSTRACT
This paper presents an optimization methodology for designing linear concentrating solar collectors. The proposed algorithm makes intelligent design updates to the collector surface geometry according to specialized numerical algorithms. The process is much more efficient than traditional “trial-and-error” methods, producing a final solution that is near-optimal. A Monte Carlo technique is used to quantify the performance of the collector design in terms of an objective function, which is then minimized using a modified Kiefer-Wolfowitz algorithm that uses sample size and step size controls. The methodology is applied to the design of a linear parabolic concentrating collector, successfully arriving at the known optimal solution.

INTRODUCTION
With the growing demand for alternative energy, solar energy is becoming an increasingly viable substitute for conventional thermal and electrical sources. As such, the development of a fully customizable, comprehensive, and straightforward procedure by which one can determine optimal collector geometry would be of great benefit to industry.

Highly-reflective specular surfaces are commonly used to redirect or concentrate thermal radiation in solar collectors. These devices are traditionally designed by “trial-and-error”, which is time-consuming and relies greatly on the experience of the designer. Furthermore, design improvements are typically stopped as soon as an acceptable, but usually suboptimal, solution is found, limiting the quality of the final design. Non-imaging optics techniques such as the edge-ray method [1] are powerful alternatives to trial-and-error, although the requirement of perfectly specular reflection and other limiting assumptions restrict their application in real-world problems. Some non-imaging optics-based techniques also require advanced mathematical skills on the part of the designer.

Numerical simulation is a powerful enhancement to these traditional design techniques. Haeberle et al. [2], for example, used the OptiCAD optical analysis software package [3] (based on ray-tracing) to perform a series of univariate parametric studies on a linear concentrating Fresnel collector. They determined the optimal collector length based on realistic weather data and material properties, including mirror reflectivity, absorber surface absorptivity, heat loss coefficient, and mirror manufacturing error. This approach, however, neglects nonlinear interactions between the design variables.

Muschaweck et al. [4] optimized the reflector geometry for a non-tracking linear (trough) solar collector, based on calculated maximum yearly average utilisable power per absorber area. Their focus was to investigate how real-world effects may make the optimal reflector geometry differ from the solution predicted by nonimaging optics. Computational demand was reduced by limiting the design space to a rectangular domain around the ideal edge-ray solution. They then mapped the topography of the two-dimensional design space and determined the optimal design by inspection. Although this treatment includes nonlinear interaction between the variables, it can only examine the relationship between two at a time, and is also computationally inefficient because simulations must be performed to construct the objective function over the entire design space.

More efficient approaches utilize an optimization algorithm to automate the search for the optimal solution over the design space. Ashdown et al. [5] used ray-tracing coupled with genetic algorithms to design the reflector geometry of various illuminating devices. Genetic algorithms [6] are a class of optimization technique that mimic natural selection: design parameters are each assigned to a “gene”; and future generations of designs come from random “mating” of “parent” designs. Such algorithms have the added advantage of being able to generate novel designs due to randomization in the form of “mutations”.

Copyright © 2009 by ASME
Daun et al. [7] incorporated Monte Carlo ray-tracing into a gradient-based optimization algorithm to design the geometry of a nonimaging process heat transfer furnace with the goal of uniformly irradiating the product being heat-treated. Optimization was carried out using the Kiefer-Wolfowitz method, an adaptation of the steepest-descent algorithm for objective functions containing stochastic uncertainty.

This paper extends the methodology developed by Daun et al. [7] to treat solar concentrating collectors. The paper starts with an overview of design optimization, followed by a description of the Monte Carlo method as it applies to the analysis of a concentrating collector. The Kiefer-Wolfowitz method is then discussed, including modifications that improve its performance and reliability. Finally, the technique is demonstrated by using it to design the ideal reflecting-surface geometry for a linear parabolic solar concentrating collector.

**NOMENCLATURE**

- $a, b, c$ = points in quadratic line search
- $c_i(\Phi)$ = design constraint
- $C$ = concentrator surface
- $d$ = direction vector for ray
- $f(x_j)$ = outcome of $j^{th}$ ray-trace event
- $F(\Phi)$ = objective function
- $\nabla F(\Phi)$ = gradient vector
- $g(a)$ = objective function for line search
- $h$ = difference parameter
- $K$ = scalar value for sampling control condition
- $n$ = problem dimensionality
- $N_b$ = bundle (sample) size
- $N_{trials}$ = number of sub-trials
- $P_i$ = step direction
- $P_{i, P_1, P_2}$ = control points of concentrator surface spline
- $r$ = ray vector
- $u$ = parameter for concentrator surface
- $w$ = parameter for ray
- $x_j$ = random value for Monte Carlo ray-trace
- $\alpha$ = step size for design update
- $\gamma$ = expansion parameter for bracketing algorithm
- $\Delta$ = initial search step for bracketing algorithm
- $\phi_i$ = design parameter for $i^{th}$ dimension
- $\Phi$ = vector of design parameters
- $\sigma$ = standard deviation

**Subscripts and Superscripts**

- $a, b$ = exponents for diminishing sequence
- $i$ = design dimension
- $k$ = design iteration
- $MC$ = stochastic (Monte Carlo)
- $TS$ = Taylor series
- $0$ = initial
- $*$ = optimal design

**DESIGN OPTIMIZATION TECHNIQUE**

In order to implement an optimization technique, one must first define an appropriate objective function, $F(\Phi)$, whose minimum corresponds to the ideal design performance. This objective function depends on a set of chosen design parameters stored in the vector $\Phi$. Next, an optimization routine is used to minimize the objective function by making a series of intelligent updates to the design parameters that reduce the value of $F(\Phi)$. The vector of design parameters

$$\Phi^* = \arg \min \left[ F(\Phi) \right], \Phi \in \mathbb{R}^n \tag{1}$$

thus corresponds to the optimal design outcome. In many optimization problems, it is also necessary to impose design constraints on $\Phi$ of the form,

$$c_i(\Phi) = 0, i = 1 \ldots m$$

$$c_i(\Phi) \geq 0, i = m + 1 \ldots m$$

to ensure that the design remains within a specified design space, known as a feasible region.

In applying an optimization technique to design a concentrating collector, the objective function could, for example, represent the fraction of incoming radiation that is not absorbed by the receiver, while the design parameters could be a set of coordinates of points defining the collector surface geometry as well as the shape and location of the absorber. An appropriate minimization algorithm could then be applied, arriving at the set of design parameters $\Phi^*$ corresponding to the overall geometry that results in the greatest fraction of collected radiation, subject to specified design constraints.

Gradient-based optimization techniques are commonly used for problems for which the feasible region is convex and the defined objective function and constraints are continuously differentiable. Such algorithms progressively reduce the objective function by updating the set of design parameters at each iteration by

$$\Phi_{k+1} = \Phi_k + \alpha_k \cdot P_k \tag{3}$$

where $P_k$ is a descent direction vector and $\alpha_k$ is a scalar step size, both at the $k^{th}$ iteration. The search direction is usually a function of the first- and sometimes second-order curvature of $F(\Phi)$; in this work, a modified steepest-descent algorithm is used with a typical design update as above, where $P_k$ is the steepest descent direction defined by

$$P_k = \frac{\nabla F(\Phi_k)}{\| \nabla F(\Phi_k) \|} \tag{4}$$

In this way, the algorithm performs design updates, each time making its way toward the optimal design found at the minimum, at which point $\nabla F(\Phi) \approx 0$. Although the steepest descent direction is usually not the best choice for deterministic optimization, it is appropriate in this case for reasons discussed shortly.
There are different strategies for choosing effective step sizes, some of which involve one-dimensional line searches of the objective function along the step direction [9]. In an exact line search, for example, the value of \(\alpha_k\) is found that minimizes \(F(\Phi + \alpha_k \mathbf{p}_k)\). Because such line searches can be computationally expensive, however, the step size is often specified as a predetermined series or set of conditions. In such cases, a general rule of thumb is to decrease the step size as the optimal point is approached, and to ensure that “step size control conditions” [10] are satisfied to prevent the algorithm from getting “stuck” at a non-optimal point. A non-vanishing step size is a standard choice that satisfies these requirements. Both of the abovementioned strategies will be discussed and compared in detail later on in this paper.

**RADIANT ENCLOSURE ANALYSIS THROUGH MONTE CARLO TECHNIQUE**

A prerequisite for carrying out design optimization is a method for evaluating the performance of a given concentrator geometry. Although analytical approaches such as geometric optics are sometimes tractable, they are often mathematically complex and require restrictive assumptions, such as collimated incident radiation and perfectly-reflecting surfaces, departing from real-world conditions. The Monte Carlo technique [11] is an appealing alternative because it is easy to implement and capable of modeling realistic surface properties.

Although the term Monte Carlo encompasses a broad range of integration tools based on random sampling, in the case of radiant enclosure analysis there is a convenient physical analogy: “bundles” of photons are ray-traced as they enter the collector from randomly-sampled locations over the aperture, and then undergo multiple randomized surface-interaction events until they are either absorbed at a surface or reemerge from the collector aperture. The overall collector performance is inferred by aggregating results for a large sample of simulated trajectories. The concept is illustrated using a simple linear parabolic concentrator with collimated incoming radiation, as shown in Fig. 1. Because the concentrator is linear, it can be treated as a two-dimensional problem without loss in generality.

Mathematically, the above procedure is equivalent to estimating the integral

\[
F(\Phi) = \int_{x=0}^{x=1} f(x) \, dx = \bar{F}(\Phi) = \frac{1}{N_b} \sum_{j=1}^{N_b} f(x_j)
\]

where \(N_b\) is the number of bundles traced and \(f(x_j)\) is the outcome for the ray originating at the random value \(x_j\). The tilde indicates a statistical estimate subject to unbiased uncertainty. In our case, we want to minimize the number of rays that are not collected, so \(f(x_j) = 0\) if the ray is collected and otherwise \(f(x_j) = 1\). Performing a large sample of such ray-traces provides us with an unbiased estimate of our objective function, \(F(\Phi)\).

**STOCHASTIC OPTIMIZATION**

In many practical optimization problems like this one, neither the objective function nor the gradient can be evaluated deterministically. In such cases, one typically estimates the objective function stochastically and then approximates the gradient through finite-difference estimations. In a multi-dimensional problem, this is done one dimension at a time, i.e. the gradient term in the \(j^{th}\) dimension is approximated as

\[
\frac{\partial F}{\partial \Phi_{k,j}} \approx \frac{\partial \bar{F}}{\partial \Phi_{k,j}} = \frac{\bar{F}(\Phi_{k,j} + h) - \bar{F}(\Phi_{k,j})}{h}
\]

where \(h\) is the difference parameter. A forwards-difference approximation is used so that each additional dimension requires only one further function evaluation, reducing overall computational time when compared to a divided-difference approach.

---

Fig. 1: Simple parabolic collector design example

Due to the statistical nature of this approximation, there is always a certain amount of stochastic uncertainty present, denoted \(\varepsilon_{MC}\). This uncertainty is estimated by sampling the results of a set of sub-trials [12]

\[
\varepsilon_{MC} = \frac{\sigma}{\sqrt{N_{\text{trials}}}}
\]

where \(\sigma\) is the standard deviation of the set of \(N_{\text{trials}}\) sub-trials. (Since the bundle trajectories are mutually-independent, the grouping of the sub-trials is arbitrary.) For all objective function evaluations used throughout this work, this uncertainty is estimated based on the outcome of twenty trials, each with \(N_b/20\) bundles. The magnitude of \(\varepsilon_{MC}\) decreases as \(N_b\) increases, as one may expect, with the uncertainty approaching zero as the bundle size tends to infinity. Note, however, that \(N_b\) is proportional to the total amount of computation time required by the simulation, since the majority of CPU time is consumed by ray-tracing, especially for more complex geometries [13].
Because the finite gradient approximation is simply a Taylor series expansion with truncated higher order terms, each approximation introduces an additional uncertainty, $\varepsilon_{TS}$. Without previous knowledge of the exact solution, the magnitude of this uncertainty is unknown, though it approaches zero asymptotically as $h$ tends to zero. The gradient-vector approximation for an $n$-dimensional problem is formed by summing the results for each dimension

$$\nabla \tilde{F}(\Phi_k) = \sum_{i=1}^{n} \frac{\partial F}{\partial \Phi_{k,i}} \hat{e}_i$$ (8)

where $\hat{e}_i$ is the unit direction vector for the $i^{th}$ dimension.

Other optimization techniques such as Newton’s method or the quasi-Newton method achieve superior convergence in a deterministic setting by calculating the search direction using higher-order curvature information. In this application, however, higher-order curvature is obscured by stochastic uncertainty, rendering such schemes impractical. In the optimization algorithm used in this paper, each design iteration requires an objective function evaluation through Monte Carlo analysis, followed by a steepest-descent update based on a forwards-difference gradient approximation in each dimension. Although the concept is straightforward, difficulties arise with respect to choosing appropriate values for bundle size, $N_b$, and difference parameter, $h$, due to their effects on the magnitudes of the previously mentioned uncertainties. For the algorithm to reach optimality the combined uncertainty must be controlled, and thus the parameters $N_b$ and $h$ must be managed properly.

For each gradient approximation, as with any calculation based on estimated or measured values, the combined uncertainty is a function of the uncertainties of the component estimations. Uncertainty propagation is estimated using the Kline-McClintock method, which states that for function $R$ of $n$ measured variables,

$$R = R(X_1, X_2, ..., X_n)$$ (9)

the overall uncertainty can be estimated by

$$\varepsilon_R = \pm \sqrt{ \left( \left( \frac{\partial R}{\partial X_1} \varepsilon_{X_1} \right)^2 + \left( \frac{\partial R}{\partial X_2} \varepsilon_{X_2} \right)^2 + ... + \left( \frac{\partial R}{\partial X_n} \varepsilon_{X_n} \right)^2 \right)}$$ (10)

where $\varepsilon_{X_i}$ are the uncertainties of the component estimations. For a gradient approximation in one dimension, then, the stochastic uncertainties combine as follows

$$\frac{\partial F}{\partial \Phi_{k,i}} = \frac{F(\Phi_{k,i} + h) - F(\Phi_k)}{h} \pm \sqrt{\varepsilon_{MC_1}^2 + \varepsilon_{MC_2}^2}$$ (11)

where $\varepsilon_{MC_1}$ and $\varepsilon_{MC_2}$ are the Monte Carlo and Taylor series uncertainties, respectively. The resulting uncertainties in the gradient approximations are quantified by comparing the estimations with results found using the known analytical solution

$$F_{1-2} = \frac{1}{2} \left( 1 + y - \sqrt{1 + y^2} \right)$$ (12)

and corresponding gradient

$$\frac{\partial F_{1-2}}{\partial y} = \frac{1}{2} \left( 1 - \frac{y}{\sqrt{1 + y^2}} \right)$$ (13)

The resulting combined Monte Carlo and Taylor series uncertainties, $\varepsilon_{MC}$ and $\varepsilon_{TS}$, as well as their combined uncertainty $\varepsilon_{combined}$, are shown in Fig. 3. Note that the Kline-McClintock method is used to determine both $\varepsilon_{MC}$ and $\varepsilon_{combined}$.
shows that $\epsilon_{MC}$ and $\epsilon_{TS}$ follow expected and opposite trends with $h$, and $\epsilon_{combined}$ reaches a clear minimum and grows rapidly to either side. Accordingly, choosing an inappropriate value for $h$ results in a large combined uncertainty, which in some cases can render the gradient approximation invalid. This relationship further depends on the bundle size through Monte Carlo uncertainties.

The Kiefer-Wolfowitz method [8] is a standard technique for optimizing stochastic systems for which unbiased estimates of the objective function are available, but not for the gradient. The method uses the steepest-descent update described previously, with the following sequences recommended for step size and difference parameter

$$
\alpha_{k+1} = \frac{\alpha_0}{k^a}, \quad h_{k+1} = \frac{h_0}{k^b}
$$

Pflug [15] recommends $a = 1$ and $b = 1/3$ for the above denominator exponents. Such diminishing sequences are an attempt to control the overall uncertainty throughout the optimization process. The designer chooses an initial step size $\alpha_0$ and difference parameter $h_0$, usually based on some heuristic knowledge or an initial error study; as the design approaches $\Phi^*$, step size is decreased to allow convergence, and the difference parameter is decreased to obtain increasingly accurate gradient approximations, necessary due to the decreasing gradient magnitude, $||\nabla F(\Phi_k)||$.

The uncertainty management strategy described above is an example of “step size control.” As mentioned previously, however, the success of the algorithm depends also on the number of bundles, $N_b$. An insufficiently large bundle size leads to poor gradient approximations, which can result in additional design iterations and, in extreme cases, prevent the algorithm from converging to the minimum. Using an excessive number of bundles, on the other hand, keeps the Monte Carlo uncertainty small, but leads to unnecessarily long computational times. Newer and more sophisticated “sampling control” algorithms avoid these drawbacks by continuously adapting bundle size. Dupuis and Simha [16] show that sampling control is theoretically better at removing the effects of estimator bias and transient behaviour compared to step size control, and that it requires less computational time.

Simha [17] presents a sampling-controlled Kiefer-Wolfowitz algorithm in which the step size is fixed, while the difference parameter diminishes according to the previous power-law recommendation. To reduce computational time, the design is initially updated using less-accurate approximations based on a smaller sample size, and the sample size is then increased with subsequent iterations as $\Phi_k$ approaches $\Phi^*$. A logarithmic sequence is used to control the sample size, similar to the bundle size control used by Daun et al. [7]

$$
N_b = A \log k + B
$$

where $A$ and $B$ are heuristics chosen based on an initial bundle refinement study.

The performance of the above-described strategies depends on choosing appropriate initial parameters, as will be demonstrated shortly. In an attempt to develop an optimization strategy that is less-dependent on such variables, a combination of step size and sample size control is implemented in this paper, with both variables based on the current state of the design, instead of on any pre-assigned heuristics. Before the developed algorithm is presented further, however, the optimization problem to which it is applied is described.

**DEMONSTRATION OF METHOD**

**Description of Test Problem**

The design optimization procedure is demonstrated by using it to optimize the geometry of the linear parabolic (trough) collector seen in Fig. 1; although this is a simple problem, it serves as a good validation exercise because it has an analytical solution, i.e. the geometry which locates the foci of the parabola completely on the receiver.

The parabolic concentrator surface, $C$, is defined using a third-order Bezier spline,

$$
C(u) = (1-u)^2P_0 + 2u(1-u)P_1 + u^2P_2, \quad u \in [0,1]
$$

where $P_0$, $P_1$, and $P_2$ are the control points and $u$ is the parameter defining current position along the curve. Defining the surface parametrically facilitates shape adaptation by modifying the coordinates of the control points. One drawback of this representation, however, is that ray-surface intersections cannot be determined analytically. Instead, the ray is defined parametrically as follows,

$$
r(w) = r_0 + wd, \quad w > 0
$$

where $r$ is the ray, $d$ is the direction vector, and $w$ is a parametric distance along the ray, and points of intersection are then found using the Newton-Raphson scheme with updates based on the current inverse Jacobian, as recommended by Toth [18]. Once the intersection location is found, the reflected ray is easily computed using the vector algebra shown in Fig. 4. Although there are more straightforward methods to raytrace a parabola, the approach used here is generic and easily extended to problems with complex geometries.

To further simplify the Monte Carlo analysis, the following assumptions are made: (i) incoming radiation is perfectly collimated; (ii) concentrator surfaces are perfectly specular; and (iii) the receiver is a blackbody. The diameter and location of the receiver and the size of the opening aperture are fixed. To allow visualization of the objective function topography, the problem is limited to two design parameters: the x- and y-coordinates of the second control point, although it is important to appreciate that realistic problems involve much larger sets of design parameters. By making updates to $\Phi_2 = [\Phi_1, \Phi_2]$, the depth and curvature of the parabolic concentrator surface can be modified. To provide a physical appreciation of this problem, Fig. 5 shows a design at a typical initial and optimized state. Note that the optimal design outcome occurs when the design parameters are set at $\Phi_1 = 10$.
where $K$ is a scalar chosen to maintain a desired level of improvement and $\varepsilon_{MC}$ is the uncertainty in the gradient normal calculation

$$\|\nabla F(\Phi_k)\| = \sqrt{\left(\frac{\partial F}{\partial \Phi_{k,1}}\right)^2 + \left(\frac{\partial F}{\partial \Phi_{k,2}}\right)^2 + \varepsilon_{MC}}. \tag{20}$$

as estimated by Eq. (10).

To determine an appropriate value for $K$, a series of trials are carried out on the simple design optimization problem, starting from the initial parameters $\Phi_0 = [8; -3]$ and $N_0 = 20,000$. The standard diminishing sequence recommendation for $h$ and $\alpha$ is used, with $h_0 = \frac{1}{4}$ and $\alpha_0 = 2$. These parameters are found to result in reliable convergence of the algorithm. Fig. 6 below shows the results of the trials, based on average CPU time to convergence for fifteen repeated trials. As expected, larger values of $K$ lead to slower convergence, as they cause the bundle size to increase more often. Although lower $K$ values allow less accurate gradient approximations, this does not seem to impede convergence; throughout the rest of this paper, a value of $K = 2$ is used.

**Step Size Control**

As mentioned previously, the performance of the algorithm when using a diminishing step size sequence depends strongly on properly selecting the initial step size, $\alpha_0$. Since the step size sequence is nonvanishing, the algorithm should, in theory, eventually make its way to the optimum; in practice, however, it may take unreasonable CPU time to do so, since if $\alpha_0$ is too small the step size can diminish such that progress cannot realistically be made (as in Fig. 7). For $\alpha_0$ too large, on the other hand, the design may pass over the optimum many times before reaching it.

To avoid these problems, a quadratic interpolation line search was applied to determine an appropriate step size at each iteration. In this procedure a set of points, $a$, $b$, and $c$, must first be identified which bracket the minimum along the step direction so that

$$a < b < c \quad \text{and} \quad g(a) > g(b) < g(c) \tag{21}$$

where $g(a)$ is the value of the objective function after a step of $a$ in the previously identified step direction. Simple “bracketing algorithms” for identifying such a set of points exist in the literature, such as that presented by Belegundu and Chandrupatla [19], consisting of consecutively evaluating points at distances of $\Delta$, $\gamma \Delta$, $\gamma^2 \Delta$, ... from the current point until such a pattern is found. In this strategy, $\Delta$ is the initial search step, and $\gamma$ is the “expansion parameter”. Once a three-point pattern is found, the minimizing step size $\bar{\alpha}$ is approximated based on the interpolation [9]

$$\bar{\alpha} = \frac{1}{2} \frac{g(a)(c^2 - b^2) + g(b)(a^2 - c^2) + g(c)(b^2 - a^2)}{g(a)(c-b) + g(b)(a-c) + g(c)(b-a)} \tag{22}$$
By determining the step size at each design iteration in this way the algorithm typically converges in fewer design iterations when compared to the diminishing step size approach. Figures 8 and 9 below show the successful solution paths of the same problem, with one solved using quadratic interpolation and the other using the diminishing step size approach. Although fewer design iterations are experienced when using quadratic interpolation, each iteration takes significantly longer CPU time due to the three-point pattern search. Depending on the particular initial design parameters, as well as the choice of initial search step $\Delta$ and expansion parameter $\gamma$, these line searches can cause the overall CPU time to be larger than when using the nonvanishing power series with heuristically-chosen parameters.

Table 1 below provides the results of numerous trials performed using both types of step size control and different combinations of initial parameters. All simulations use an initial bundle size $N_b = 20,000$ which is doubled whenever the sample size control condition in Eq.(19) is satisfied. The recommended diminishing sequence for $h$ is also used, with $h_0 = 1/4$. Quadratic interpolation trials are performed with an initial line search step size $\Delta = 1/2$, as this is found to work well. Each CPU time provided is an average result of five repeated trials, and simulation times are based on performance of a 1.83 GHz dual core laptop PC (only using one core) with 2.49 GB of RAM. To prevent the algorithm from getting stuck at any point, any trial that takes over 300 seconds is taken to be a failure and is subsequently terminated.

Although the quadratic scheme tends to converge in fewer design updates, both schemes exhibit similar average convergence times. In general, convergence times depend strongly on the set of initial design parameters, $\Phi_0$, due to local differences in objective function curvature. The performance of both approaches also depends on the parameter values, with no single choice that consistently performs best. The only factor which clearly separates the two control methods is the number of “failures” experienced: the algorithm failed to reach the optimum seven out of the ninety trials for the diminishing sequence approach, but not once for the quadratic method. Although both schemes should, given reasonable parameter choices, converge successfully, the quadratic approach is much more reliable.
CONCLUSIONS AND FUTURE WORK

The concentrating collector design optimization methodology presented in this paper is a substantial improvement over the current design practice. It consistently and efficiently identifies the optimal solution, in contrast to design by trial-and-error, and it can be applied to a much wider range of problems compared to the limited set that can be solved through nonimaging optics.

In response to drawbacks of the standard Kiefer-Wolfowitz method for stochastic optimization, a condition-based sampling control approach is developed and applied to the simple parabolic concentrator problem, with promising results. A more problem-independent method for step size control, based on quadratic interpolation, is also applied to the problem, which enhances the reliability of the method.

We will shortly apply the algorithm to more challenging problems, including those with more than two design variables and more complicated geometric constraints. We will also develop an objective function with appropriate multi-variable dependence to develop optimal designs for more advanced applications. One such possibility, for example, is to maximize the energy flux per receiver area in order to improve the overall thermal performance of solar thermal collectors. An additional possibility is the inclusion of real-world effects, such as the buildup of dust or corrosion on reflecting surfaces, in order to optimize for specific harsh conditions. Further improvements to the optimization algorithm performance could be made through the use of low-discrepancy methods including quasi-Monte Carlo and randomized quasi-Monte Carlo for stochastic error reduction. As one can see, the introduction of this methodology to solar energy opens up a variety of possibilities.

REFERENCES