Optimization—Penalty Functions

Solving constrained optimization problems in ASAP

This technical publication describes the penalty function in the Advanced Systems Analysis Program (ASAP®) from Breault Research Organization (BRO). The penalty function technique is part of the Optimization feature set in ASAP. This publication complements optimization topics in ASAP Help. For a broader description of the Optimization feature, see the Knowledge Base for the technical publication, “Optimization—General purpose optimization methods in ASAP”.

Penalty functions, a technique used in solving constrained optimization problems, are often used to restrict the solution search to designs that meet all criteria. As the name implies, a penalty is assigned to the figure of merit or merit function if a constraint is violated during optimization.

The traditional merit function has an objective function component and a residual component comprising additional constraints added to the problem. ASAP computes the Figure of Merit (FOM) as the normalized weighted sum of squares of departures of the design objectives from the corresponding objective target values, as shown in Figure 1.

\[
FOM = \frac{\sum_{i=1}^{N} w_i (o_i - t_i)^2}{\sum_{i=1}^{N} w_i}
\]

where
\(o_i\) are the design objectives;
\(t_i\) are the objective targets;
\(w_i\) are the objective weights, and
\(N\) is the number of defined design objectives,
subject to the side conditions
\(N > 0\); and
\(w_i \neq 0\) for at least one value \(i \in [1, N]\).

Figure 1 Figure of Merit for optimization
Penalty functions are related to constraints, which in general can be hard or soft. A constraint is considered “hard” if it cannot be violated during optimization, and “soft” if it can, thereby potentially leading to a better solution. Penalty functions replace constrained optimization with a series of less constrained conditions whose solutions ideally converge to the solution of the original constrained problem. The penalty function itself grows and forces the merit function to increase in value when the constraints are violated, and causes no growth when constraints are not violated. ASAP computes the penalized figure of merit (PFOM) as the FOM for design objectives, with the addition of penalty functions, as shown in Figure 2.

\[
PFOM = FOM + \sum_{i=1}^{P} \delta_i p_i
\]

where
\( \delta_i \) is unity if and only if the Trial Solution violates one of
the limits on the \( i \)th Objective Constraint and zero otherwise;
\( p_i \) are the Penalty Functions; and
\( P \) is the number of defined Penalty Functions
\( P > 0 \)

Figure 2 Penalized Figure of Merit for optimization

**Example of a penalty function problem**

To understand penalty functions or to motivate their use before illustrating them with an ASAP example, we turn to a general, common example of a penalty function: optimizing the trip of the classic traveling salesperson.

The famous traveling salesperson problem involves optimizing this person’s route when visiting several cities in a geographical region. This classic optimization problem was one of the original problems solved by simulated annealing among other techniques. The problem is to optimize or find the shortest path for a salesperson to travel between cities within a limited, geographic region to reduce travel costs. Minimum travel costs could be considered the objective function. The added constraint of this problem, and where the penalty function is applicable, is that some cities are separated by a river. The salesperson must pay additional fees to cross the river to visit cities in her sales region. For this salesperson, a river crossing would be considered a constraint; that is, she would rather not allow it to occur because of the increased cost. However, river crossings are necessary for business, so she wants a route that limits river crossings by penalizing those potential routes for excessive river crossings.

Now, in the context of an ASAP workflow, there are two initial tasks you must complete before starting an actual optimization.

1. Define at least one design variable, and confirm that it is enabled.
2. Define at least one design objective, and confirm that it is enabled.

Optionally, you can define objective constraints and penalty functions. When a penalty function is assigned to an enabled objective constraint, ASAP applies the penalty function to the merit function during any iteration of the script in which the constraint is violated as indicated in the equations in Figure 1 and Figure 2.
To illustrate this process, we will look at a simple problem designed to demonstrate the use of penalty functions, without complicating the objective function. We will design a compound parabolic concentrator (CPC) to minimize the RMS angular ray deviations out of the concentrator. A CPC is a non-imaging optical system that is normally used in solar collection or concentration applications. The CPC is a parabolic reflector segment rotated about an axis other than its axis of symmetry in a way that satisfies the edge-ray principle. The edge-ray principle forces all rays from the extreme input angle to focus at the edge of the output aperture. Figure 3 illustrates a basic CPC.

**Tip** For more information on the CPC, see a number of books by, for example, Roland Winston.

Figure 3 A basic compound parabolic concentrator (CPC) in ASAP

When a CPC is used in reverse, it becomes a luminaire, as shown in Figure 4.

Figure 4 Simple Lambertian emitting circular disk source at small end of CPC in ASAP
The longer the CPC, the more collimated the light reflected from the CPC walls, so the optimization task is trivial and is meant to be so to illustrate penalty functions. To demonstrate the use of the penalty function, we will minimize the RMS ray deviations by changing the extreme ray angle of the CPC while constraining the length of the CPC by penalizing the merit function when the CPC becomes too long.

**Setting up optimization in ASAP**

First, open a script file in ASAP, select **Optimize Script** on the Optimize menu. Note that “(optimized)” is added to the Editor window title bar. Now, we define our variables, as illustrated in Figure 5 and Figure 6. The variable is the CPC entrance or extreme ray angle as defined in the ASAP **AXICONIC** command.

**DEFINING THE DESIGN VARIABLE**

In this example, we select `THETA=20` as the variable, right-click, and select **Define Design Variable** from the shortcut menu, as shown in Figure 5. This menu command displays the Optimization Setup Summary window, where the Design Variable tab shows the variable, as shown in Figure 6.

![Figure 5 Selecting the variable in the script and defining the design variable](image-url)
The Optimization Setup Summary window presents a multi-tabbed user interface for viewing and entering information that ASAP will use to optimize the system we are evaluating. Each tab is illustrated in this example.

Figure 6 Design Variable tab on Optimization Setup Summary window, with min/max values manually inserted.
DEFINING THE DESIGN OBJECTIVE

Next, we define the objective function, as illustrated in Figure 7 and Figure 8, which is composed of the RMS ray deviations out of the CPC.

![Figure 7 Selecting the variable in the script to define the design objective](https://via.placeholder.com/150)

**NOTE** The CPC length is included as a design objective but is disabled so the CPC length variable occurs in the Trial Solution table on the Optimization Results window for illustration. See Figure 14.

DEFINING THE OBJECTIVE CONSTRAINT

The constraint is the CPC length, and is constrained between 15mm and 40mm. Similarly, the penalty function is defined as 10% of the objective function. The merit function becomes the sum of the objective and penalty functions. If the constraint is not violated, the penalty function contribution is zero. If the constraint is violated, the penalty function is 10% of the current objective function or RMS ray deviation, which is added to the objective function. As the optimization forces the CPC length to exceed 40mm, the merit function normally decreases because of the longer and wider CPC, but it increases because of the penalty function. The optimization will re-
turn to a CPC length that is less than 40mm, since the merit function is smallest around this point. In fact, the function should oscillate as it converges to a minimum. We define the objective constraint, as shown in Figure 9 and Figure 10.

Figure 9 Selecting the objective constraint in the script to define

```
SRC_RAD=2.5

!! CPC PROPERTIES
THETA=20
EFL PARAX=SRC_RAD*(1+sin(THETA))
CPC RAD=SRC RAD/sin(THETA)
CPC LEN=(SRC RAD+CPC RAD)/tan(THETA)
TRUN LEN=(CPC LEN)

!! SET UP OPTICAL PROPERTIES DATA

UNIT MM
WAVELENGTHS 0.530000 UM
```

Figure 10 Objective Constraints tab on Optimization Setup Summary window with penalty function unassigned
ASSIGNING THE PENALTY FUNCTION

Figure 11 and Figure 12 illustrate the penalty function assignment. After setting the number of trials as shown in Figure 13 and using Brent's optimization method as illustrated in Figure 14, we do see a minimum reached in Figure 15 in the oscillatory nature described in “Defining the objective constraint” on page 6.
SETTING EXIT CRITERIA

The only criteria we need to set is for the number of trial solutions to evaluate in the search for an optimal design. See Figure 13.

Figure 13 Exit Criteria tab on Optimization Setup Summary window with limit set on number of trial solutions
SELECTING OPTIMIZATION METHOD

Since we are using only one variable, we will accept the default optimization method, Brent’s Method. See Figure 14.

Figure 14 Optimization Method tab on Optimization Setup Summary window with Brent’s Method selected

Now we are ready to begin optimizing by selecting Start Optimization on the Optimization Method tab. See the results illustrated in Figure 15.
VIEWING OPTIMIZATION RESULTS

Figure 15 Optimization results showing FOM versus trial solution

Optimization—Penalty Functions 11