

Solving a Mixed Integer Engineering Design Problem Using the Genetic Algorithm

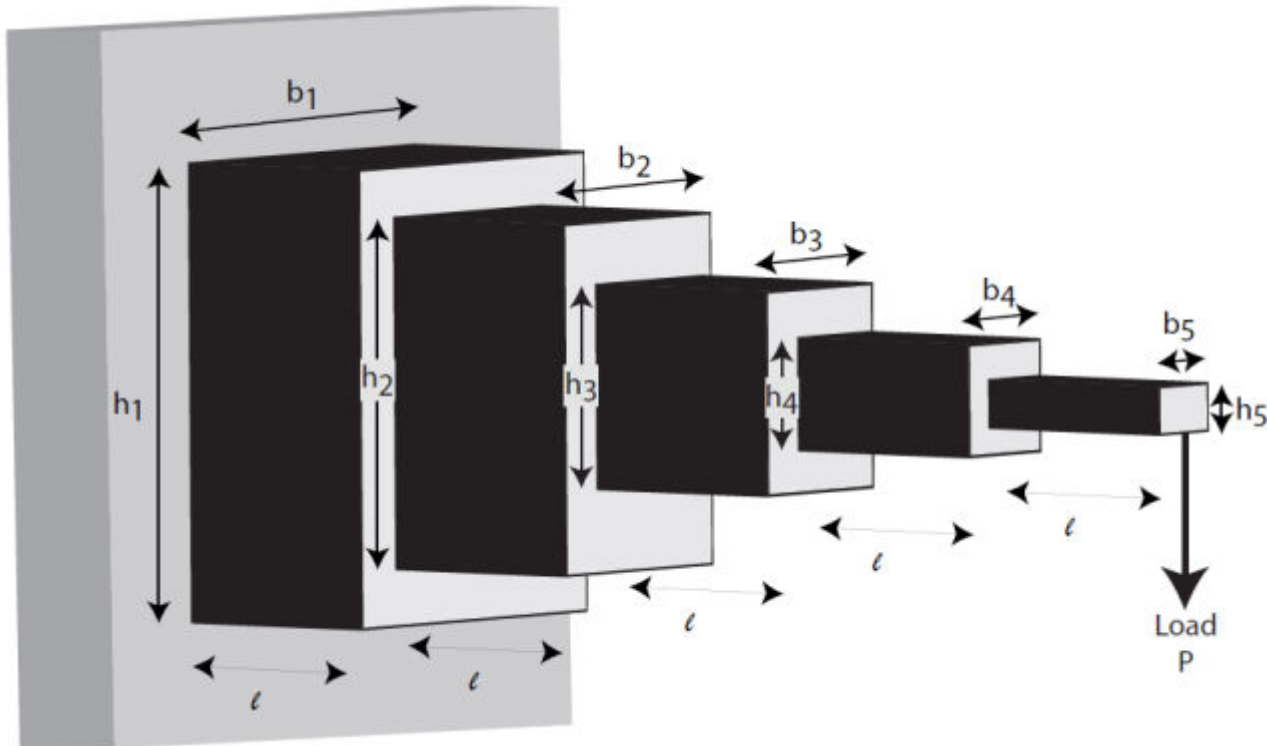
This example shows how to solve a mixed integer engineering design problem using the Genetic Algorithm (ga) solver in Global Optimization Toolbox.

The problem illustrated in this example involves the design of a stepped cantilever beam. In particular, the beam must be able to carry a prescribed end load. We will solve a problem to minimize the beam volume subject to various engineering design constraints.

In this example we will solve two bounded versions of the problem published in [1].

Stepped Cantilever Beam Design Problem

A stepped cantilever beam is supported at one end and a load is applied at the free end, as shown in the figure below. The beam must be able to support the given load, P , at a fixed distance L from the support. Designers of the beam can vary the width (b_i) and height (h_i) of each section. We will assume that each section of the cantilever has the same length, l .



Volume of the beam

The volume of the beam, V , is the sum of the volume of the individual sections

$$V = l(b_1h_1 + b_2h_2 + b_3h_3 + b_4h_4 + b_5h_5)$$

Constraints on the Design : 1 - Bending Stress

Consider a single cantilever beam, with the centre of coordinates at the centre of its cross section at the free end of the beam. The bending stress at a point (x, y, z) in the beam is given by the following equation

$$\sigma_b = M(x)y/I$$

where $M(x)$ is the bending moment at x , x is the distance from the end load and I is the area moment of inertia of the beam.

Now, in the stepped cantilever beam shown in the figure, the maximum moment of each section of the beam is PD_i , where D_i is the maximum distance from the end load, P , for each section of the beam. Therefore, the maximum stress for the i -th section of the beam, σ_i , is given by

$$\sigma_i = PD_i(h_i/2)/I_i$$

where the maximum stress occurs at the edge of the beam, $y = h_i/2$. The area moment of inertia of the i -th section of the beam is given by

$$I_i = b_i h_i^3 / 12$$

Substituting this into the equation for σ_i gives

$$\sigma_i = 6PD_i / b_i h_i^2$$

The bending stress in each part of the cantilever should not exceed the maximum allowable stress, σ_{max} . Consequently, we can finally state the five bending stress constraints (one for each step of the cantilever)

$$\frac{6Pl}{b_5 h_5^2} \leq \sigma_{max}$$

$$\frac{6P(2l)}{b_4 h_4^2} \leq \sigma_{max}$$

$$\frac{6P(3l)}{b_3 h_3^2} \leq \sigma_{max}$$

$$\frac{6P(4l)}{b_2 h_2^2} \leq \sigma_{max}$$

$$\frac{6P(5l)}{b_1 h_1^2} \leq \sigma_{max}$$

Constraints on the Design : 2 - End deflection

The end deflection of the cantilever can be calculated using Castigliano's second theorem, which states that

$$\delta = \frac{\partial U}{\partial P}$$

where δ is the deflection of the beam, U is the energy stored in the beam due to the applied force, P . The energy stored in a cantilever beam is given by

$$U = \int_0^L M^2/2EI \, dx$$

where M is the moment of the applied force at x .

Given that $M = Px$ for a cantilever beam, we can write the above equation as

$$U = P^2/2E \int_0^l [(x+4l)^2/I_1 + (x+3l)^2/I_2 + (x+2l)^2/I_3 + (x+l)^2/I_4 + x^2/I_5] \, dx$$

where I_n is the area moment of inertia of the n -th part of the cantilever. Evaluating the integral gives the following expression for U .

$$U = (P^2/2)(l^3/3E)(61/I_1 + 37/I_2 + 19/I_3 + 7/I_4 + 1/I_5)$$

Applying Castigliano's theorem, the end deflection of the beam is given by

$$\delta = Pl^3/3E(61/I_1 + 37/I_2 + 19/I_3 + 7/I_4 + 1/I_5)$$

Now, the end deflection of the cantilever, δ , should be less than the maximum allowable deflection, δ_{max} , which gives us the following constraint.

$$Pl^3/3E(61/I_1 + 37/I_2 + 19/I_3 + 7/I_4 + 1/I_5) \leq \delta_{max}$$

Constraints on the Design : 3 - Aspect ratio

For each step of the cantilever, the aspect ratio must not exceed a maximum allowable aspect ratio, a_{max} . That is,

$$h_i/b_i \leq a_{max} \text{ for } i = 1, \dots, 5$$

State the Optimization Problem

We are now able to state the problem to find the optimal parameters for the stepped cantilever beam given the stated constraints.

Let $x_1 = b_1$, $x_2 = h_1$, $x_3 = b_2$, $x_4 = h_2$, $x_5 = b_3$, $x_6 = h_3$, $x_7 = b_4$, $x_8 = h_4$, $x_9 = b_5$ and $x_{10} = h_5$

Minimize:

$$V = l(x_1x_2 + x_3x_4 + x_5x_6 + x_7x_8 + x_9x_{10})$$

Subject to:

$$\frac{6Pl}{x_9x_{10}^2} \leq \sigma_{max}$$

$$\frac{6P(2l)}{x_7x_8^2} \leq \sigma_{max}$$

$$\frac{6P(3l)}{x_5x_6^2} \leq \sigma_{max}$$

$$\frac{6P(4l)}{x_3x_4^2} \leq \sigma_{max}$$

$$\frac{6P(5l)}{x_1x_2^2} \leq \sigma_{max}$$

$$\frac{Pl^3}{E} \left(\frac{244}{x_1x_2^3} + \frac{148}{x_3x_4^3} + \frac{76}{x_5x_6^3} + \frac{28}{x_7x_8^3} + \frac{4}{x_9x_{10}^3} \right) \leq \delta_{max}$$

$$x_2/x_1 \leq 20, x_4/x_3 \leq 20, x_6/x_5 \leq 20, x_8/x_7 \leq 20 \text{ and } x_{10}/x_9 \leq 20$$

The first step of the beam can only be machined to the nearest centimetre. That is, x_1 and x_2 must be integer. The remaining variables are continuous. The bounds on the variables are given below:-

$$1 \leq x_1 \leq 5$$

$$30 \leq x_2 \leq 65$$

$$2.4 \leq x_3, x_5 \leq 3.1$$

$$45 \leq x_4, x_6 \leq 60$$

$$1 \leq x_7, x_9 \leq 5$$

$$30 \leq x_8, x_{10} \leq 65$$

Design Parameters for this Problem

For the problem we will solve in this example, the end load that the beam must support is $P = 50000N$.

The beam lengths and maximum end deflection are:

- Total beam length, $L = 500cm$
- Individual section of beam, $l = 100cm$
- Maximum beam end deflection, $\delta_{max} = 2.7cm$

The maximum allowed stress in each step of the beam, $\sigma_{max} = 14000N/cm^2$

Young's modulus of each step of the beam, $E = 2 \times 10^7 N/cm^2$

Solve the Mixed Integer Optimization Problem

We now solve the problem described in *State the Optimization Problem*.

Define the Fitness and Constraint Functions

Examine the MATLAB files `cantileverVolume.m` and `cantileverConstraints.m` to see how the fitness and constraint functions are implemented.

A note on the linear constraints: When linear constraints are specified to `ga`, you normally specify them via the `A`, `b`, `Aeq` and `beq` inputs. In this case we have specified them via the nonlinear constraint function. This is because later in this example, some of the variables will become discrete. When there are discrete variables in the problem it is far easier to specify linear constraints in the nonlinear constraint function. The alternative is to modify the linear constraint matrices to work in the transformed variable space, which is not trivial and maybe not possible. Also, in the mixed integer `ga` solver, the linear constraints are not treated any differently to the nonlinear constraints regardless of how they are specified.

Set the Bounds

Create vectors containing the lower bound (`lb`) and upper bound constraints (`ub`).

```
lb = [1 30 2.4 45 2.4 45 1 30 1 30];
ub = [5 65 3.1 60 3.1 60 5 65 5 65];
```

Set the Options

To obtain a more accurate solution, we increase the `PopulationSize`, and `MaxGenerations` options from their default values, and decrease the `EliteCount` and `FunctionTolerance` options. These settings cause `ga` to use a larger population (increased `PopulationSize`), to increase the search of the design space (reduced `EliteCount`), and to keep going until its best member changes by very little (small `FunctionTolerance`). We also specify a plot function to monitor the penalty function value as `ga` progresses.

Note that there are a restricted set of `ga` options available when solving mixed integer problems - see Global Optimization Toolbox User's Guide for more details.

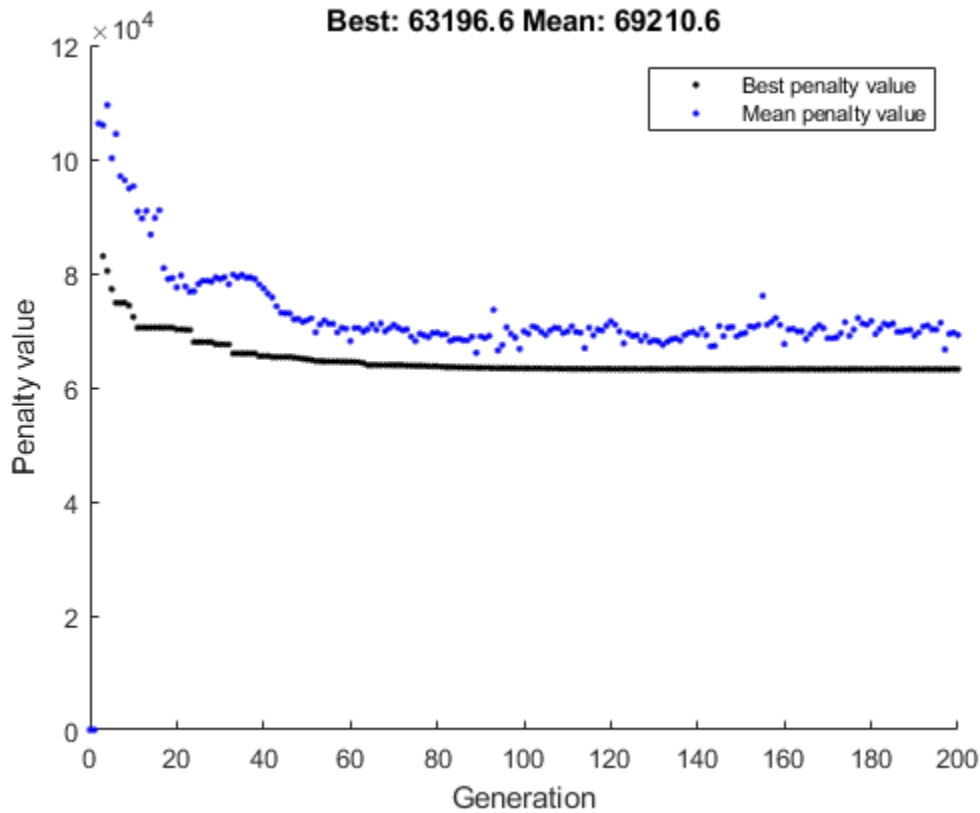
```
opts = optimoptions(@ga, ...
    'PopulationSize', 150, ...
    'MaxGenerations', 200, ...
    'EliteCount', 10, ...
    'FunctionTolerance', 1e-8, ...
    'PlotFcn', @gaplotbestf);
```

Call `ga` to Solve the Problem

We can now call `ga` to solve the problem. In the problem statement x_1 and x_2 are integer variables. We specify this by passing the index vector `[1 2]` to `ga` after the nonlinear constraint input and before the options input. We also seed and set the random number generator here for reproducibility.

```
rng(0, 'twister');
[xbest, fbest, exitflag] = ga(@cantileverVolume, 10, [], [], [], [], ...
    lb, ub, @cantileverConstraints, [1 2], opts);
```

Optimization terminated: maximum number of generations exceeded.



Analyze the Results

If a problem has integer constraints, `ga` reformulates it internally. In particular, the fitness function in the problem is replaced by a penalty function which handles the constraints. For feasible population members, the penalty function is the same as the fitness function.

The solution returned from `ga` is displayed below. Note that the section nearest the support is constrained to have a width (x_1) and height (x_2) which is an integer value and this constraint has been honored by GA.

```
display(xbest);
```

```
xbest =
```

```
Columns 1 through 7
```

```
3.0000    60.0000    2.8326    56.6516    2.5725    51.4445    2.2126
```

```
Columns 8 through 10
```

```
44.2423    1.7512    34.9805
```

We can also ask `ga` to return the optimal volume of the beam.

```
fprintf('\nCost function returned by ga = %g\n', fbest);
```

Cost function returned by ga = 63196.6

Add Discrete Non-Integer Variable Constraints

The engineers are now informed that the second and third steps of the cantilever can only have widths and heights that are chosen from a standard set. In this section, we show how to add this constraint to the optimization problem. Note that with the addition of this constraint, this problem is identical to that solved in [1].

First, we state the extra constraints that will be added to the above optimization

- The width of the second and third steps of the beam must be chosen from the following set:- [2.4, 2.6, 2.8, 3.1] cm
- The height of the second and third steps of the beam must be chosen from the following set:- [45, 50, 55, 60] cm

To solve this problem, we need to be able to specify the variables x_3 , x_4 , x_5 and x_6 as discrete variables. To specify a component x_j as taking discrete values from the set $S = v_1, \dots, v_k$, optimize with x_j an integer variable taking values from 1 to k , and use $S(x_j)$ as the discrete value. To specify the range (1 to k), set 1 as the lower bound and k as the upper bound.

So, first we transform the bounds on the discrete variables. Each set has 4 members and we will map the discrete variables to an integer in the range [1, 4]. So, to map these variables to be integer, we set the lower bound to 1 and the upper bound to 4 for each of the variables.

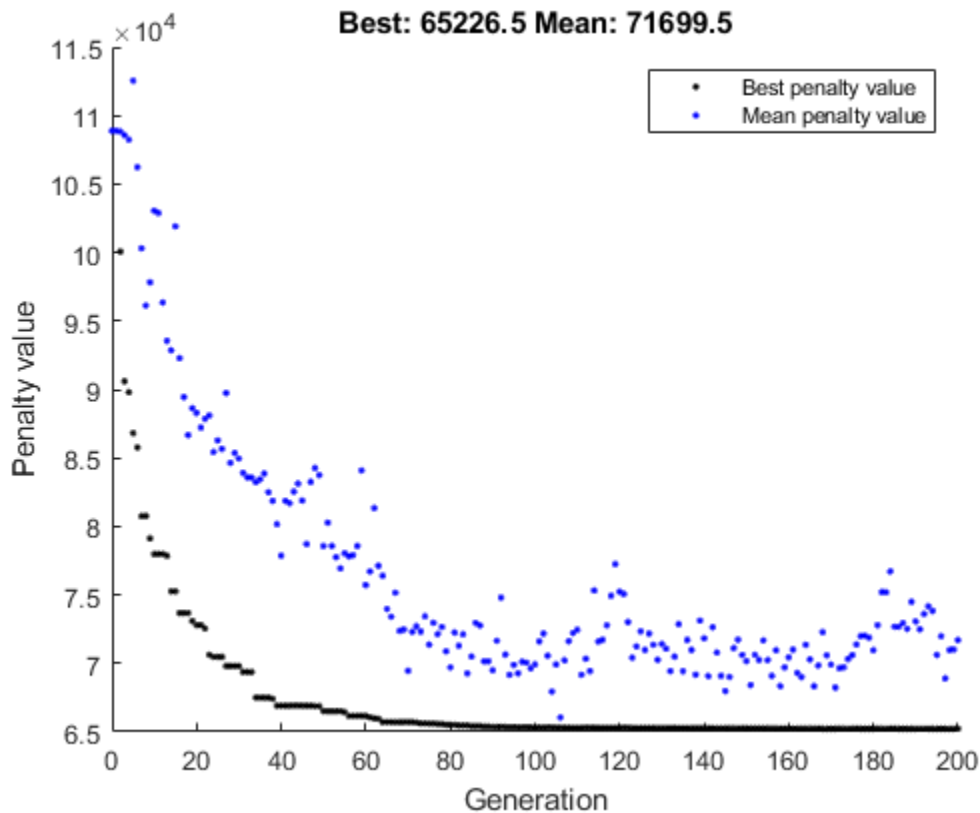
```
lb = [1 30 1 1 1 1 1 30 1 30];
ub = [5 65 4 4 4 4 5 65 5 65];
```

Transformed (integer) versions of x_3 , x_4 , x_5 and x_6 will now be passed to the fitness and constraint functions when the ga solver is called. To evaluate these functions correctly, x_3 , x_4 , x_5 and x_6 need to be transformed to a member of the given discrete set in these functions. To see how this is done, examine the MATLAB files `cantileverVolumeWithDisc.m`, `cantileverConstraintsWithDisc.m` and `cantileverMapVariables.m`.

Now we can call ga to solve the problem with discrete variables. In this case $x_1, \dots, x_6$ are integers. This means that we pass the index vector 1:6 to ga to define the integer variables.

```
rng(0, 'twister');
[xbestDisc, fbestDisc, exitflagDisc] = ga(@cantileverVolumeWithDisc, ...
    10, [], [], [], [], lb, ub, @cantileverConstraintsWithDisc, 1:6, opts);
```

Optimization terminated: maximum number of generations exceeded.



Analyze the Results

`xbestDisc(3:6)` are returned from `ga` as integers (i.e. in their transformed state). We need to reverse the transform to retrieve the value in their engineering units.

```
xbestDisc = cantileverMapVariables(xbestDisc);
display(xbestDisc);
```

```
xbestDisc =
```

```
Columns 1 through 7
```

```
3.0000    60.0000    3.1000    55.0000    2.8000    50.0000    2.3036
```

```
Columns 8 through 10
```

```
43.6153    1.7509    35.0071
```

As before, the solution returned from `ga` honors the constraint that x_1 and x_2 are integers. We can also see that x_3 , x_5 are chosen from the set [2.4, 2.6, 2.8, 3.1] cm and x_4 , x_6 are chosen from the set [45, 50, 55, 60] cm.

Recall that we have added additional constraints on the variables $x(3)$, $x(4)$, $x(5)$ and $x(6)$. As expected, when there are additional discrete constraints on these variables, the optimal solution has

a higher minimum volume. Note further that the solution reported in [1] has a minimum volume of 64558cm^3 and that we find a solution which is approximately the same as that reported in [1].

```
fprintf('\nCost function returned by ga = %g\n', fbestDisc);
```

```
Cost function returned by ga = 65226.5
```

Summary

This example illustrates how to use the genetic algorithm solver, `ga`, to solve a constrained nonlinear optimization problem which has integer constraints. The example also shows how to handle problems that have discrete variables in the problem formulation.

References

[1] Survey of discrete variable optimization for structural design, P.B. Thanedar, G.N. Vanderplaats, J. Struct. Eng., 121 (3), 301-306 (1995)

See Also

More About

- “Mixed Integer `ga` Optimization” on page 5-35