MAS252 Coursework

Dr Sam Rigby – sam.rigby@sheffield.ac.uk

The aim of this coursework is to:

- Introduce you to simple root-finding algorithms, e.g. the bisector method;
- Give you a practical example of an engineering minimisation problem;
- Explore concepts such as verification, convergence, and accuracy vs. computational time;
- Improve your programming skills and ability to communicate complex ideas in a concise way.

1 Introduction – Root finding algorithms and minimisation

"*An engineer can do for a dollar what any fool can do for two.*"

— Arthur M. Wellington

Engineers are often tasked with finding the 'best', or at least the most suitable solution. That is, one which balances cost, practicality, sustainability, aesthetics, etc. Take the roof of a stadium, for example. If you were to write a computer code to automatically design the roof, you would need to instruct it to do a number of things. First, you would need it to reject any solutions which do not meet certain criteria (are too expensive, cannot support the load, etc.), you would then need to instruct it to search for the optimal solution; in this case the roof which supports the required load and spans the required distance, for the lowest structural mass (i.e. cost).

This presents a number of challenges. How do we know whether a design passes or fails? How do we know what our optimum is, and when we've found it? And how do we get our computer code to search for it in an efficient and robust manner? These are some of the issues you will explore in this coursework.

Numerical root-finding algorithms are a useful tool in engineering optimisation/minimisation problems. Essentially, we are interested in finding input parameter(s) which give us an output that is equal to, or very close to, our desired output. For a simple mathematical operation, such as $y = 4x - 4$, if we wanted to know which value of x will give us a y value of 12, we can simply insert this into the equation, rearrange, and find that $x = 4$ satisfies this. For real-life engineering applications, however, we often do not have such a simple, well-defined relationship between input and output, and we may instead have a complex system of (time-varying) physical equations for which it is impossible or impractical to derive the input-output relationship. We need to solve these physical equations *every time* we want to examine a new set of input parameters.

To deal with this, we use root-finding algorithms such as the bisector method. An illustrative example of the bisector method is shown in Figure 1. Our search domain lies between two input parameters (here: 0 and 10), and we know our target value (here: 100) lies within this interval. We take the average of the two input values (5), evaluate its output, and adjust our search accordingly; if our guess is below the target value, we know that the root must lie on the left hand side of our domain. Thus, we halve our search domain successively until we find the single input value which gives us our desired output. This brings up another interesting point: how do we decide when to stop our search? How close is close enough? You can see in Figure 1 that if we are interested in an accuracy of 5% or so, then 3 guesses would have been enough. If we are interested in an accuracy of 0.001%, then we would have to take a few more guesses. This would clearly take much more computational time to execute, so engineers must always be aware of the trade-off between computational time and accuracy. You will be able to investigate this further as part of the coursework.

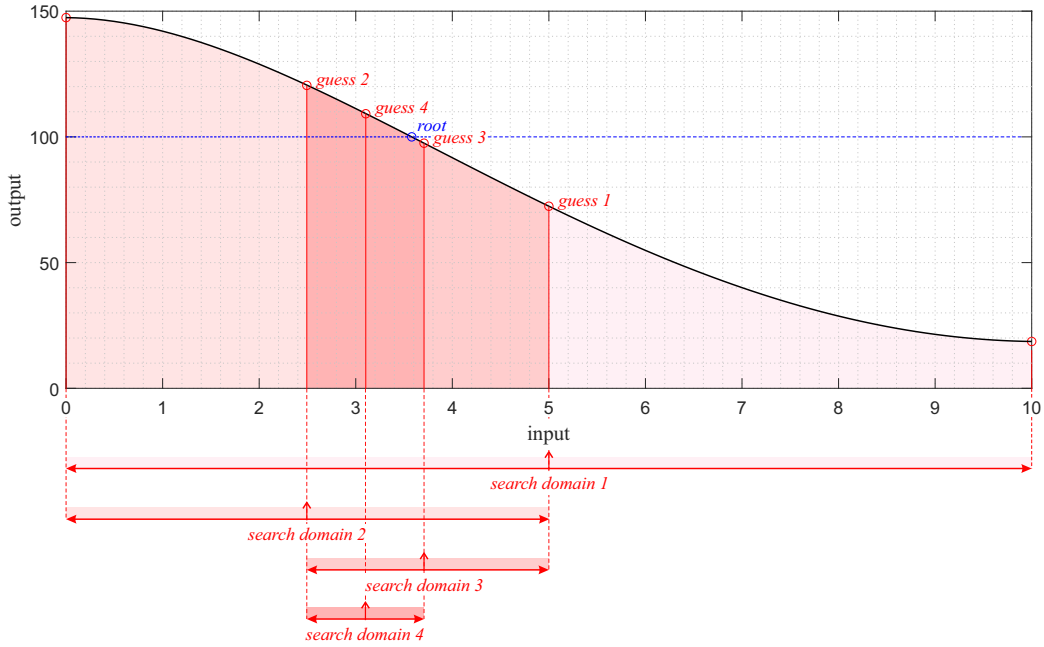

Figure 1: An indicative example of the bisector method in operation. The root (in this case, 100) is converged on with successive iterations, with the search domain halving each time

2 The coursework – Design of a cantilever blast wall

2.1 Overview

You are tasked with finding the lowest-cost design of a cantilevered blast wall. The blast wall is shown in Figure 2. The client, who owns a high-profile office block whose entrance is behind the blast wall, initially specified the wall to have a mass, $m = 200$ kg and stiffness, $k = 1$ MN/m. The wall is subjected to the blast load from a certain sized device located a certain distance away, although the exact details are withheld for security reasons. However, you are told that the design blast load is linearly-decaying and has a peak force, $F_{peak} = 1$ MN, and a duration, $t_d = 20$ ms. This is applied uniformly across the face of the blast wall, as shown in Figure 2

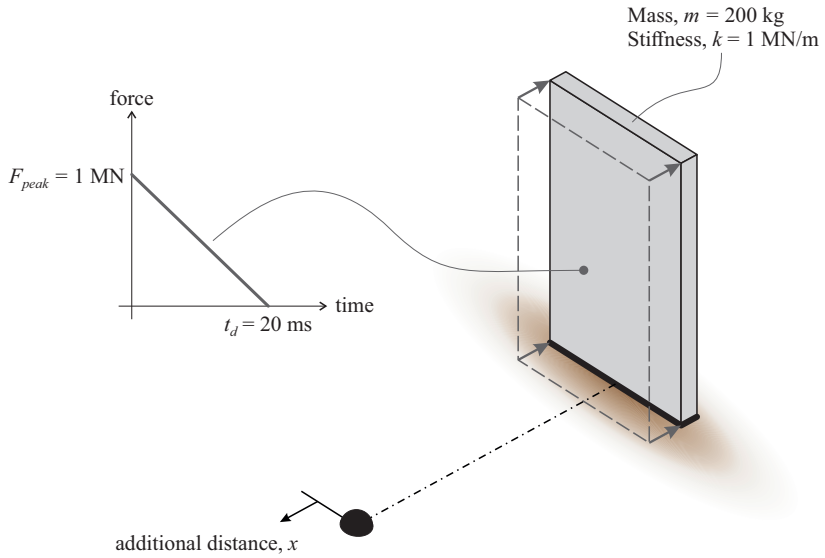

Figure 2: Schematic of the blast wall, with initial properties and initial design blast load shown

The un-damped time-varying deflection of the tip of the cantilever, $z(t)$, after application of a uniform linearly-decaying blast load is computed in two stages: response during load application ($t \leq t_d$) and response after load application ($t > t_d$). The relationship for $z(t)$ is given below[1]

$$z(t) = \begin{cases} \dfrac{F_{peak}}{k}[1 - \cos(\omega t)] + \dfrac{F_{peak}}{k t_d}\left(\dfrac{\sin(\omega t)}{\omega} - t\right), & t \leq t_d \\ \dfrac{F_{peak}}{k\omega t_d}\left[\sin(\omega t) - \sin(\omega\,[t - t_d])\right] - \dfrac{F_{peak}}{k}\cos(\omega t), & t > t_d \end{cases} \tag{1}$$

where F_{peak} (in **N**) is peak force, k (in **N/m**) is stiffness, t_d is loading duration (in **s**) as above, and $z(t)$ is expressed in **m**. The natural circular frequency, ω is given as $\omega = \sqrt{k/m}$, where k is as above and m is mass (in **kg**). This gives the blast wall's natural period, T (in **s**), as $2\pi/\omega$.

If the deflection exceeds 100 mm, the blast wall will fail

An appraisal of the initial design of the blast wall reveals that it will **fail** under the design blast load, reaching a peak displacement of 669 mm (see Figure 3).

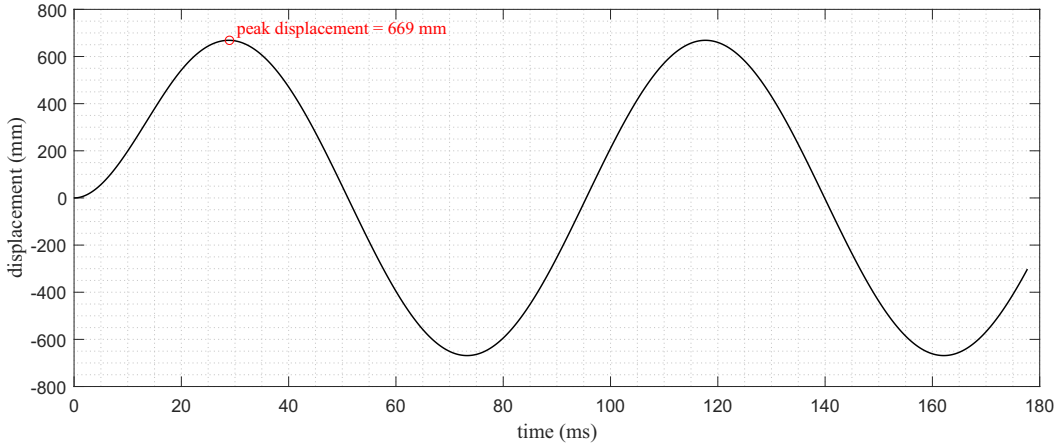

Figure 3: Response-time history of the initial design of the blast wall

2.2 The task

The client has approached you and asked you to find the lowest-cost design of a cantilevered blast wall **which will not exceed 100 mm**. There are three ways to reduce the displacement of the blast wall, and it is likely that the optimal solution will require some combination of these:

1. You can pay to increase the stiffness of the structure, up to a maximum of $k = 7$ MN/m.
2. You can pay to increase the mass of the structure, up to a maximum of $m = 1200$ kg.
3. You can purchase additional land. This will allow you to move the explosive further away, where x is the additional distance you can move the explosive away, up to a maximum of $x = 10$ m.

Increasing the distance from the explosive to the blast wall results in a significant *decrease* in the peak force, F_{peak}, but a slight *increase* in the loading duration, t_d. The equations for how force and duration change with increasing distance are given below and are shown in Figure 4

$$\begin{aligned} F_{peak}(x) &= 1000 + 9(x^2) - 183x \quad \text{(in \textbf{kN})} \\ t_d(x) &= 20 - 0.12(x^2) + 4.2x \quad \text{(in \textbf{ms})} \end{aligned} \tag{2}$$

[1] JM Biggs (1964), *Introduction to Structural Dynamics*, McGraw-Hill, USA

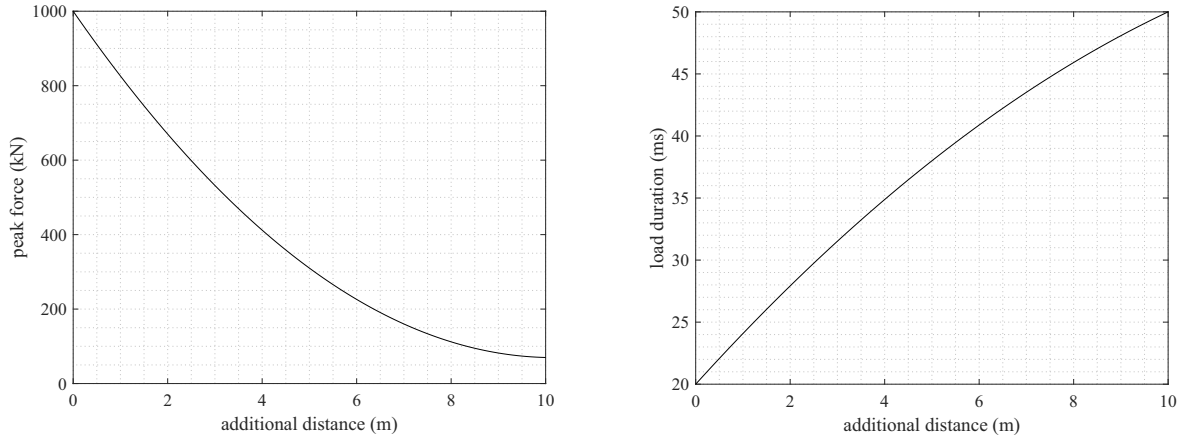

Figure 4: Relationships of peak force, F_{peak}, and duration, t_d, vs. additional distance, x

2.3 Associated costs

- The costs associated with increasing stiffness are:
$$C_k \text{ (in £)} = 900 + 825(k^2) - 1725k \tag{3}$$
where k is expressed in **MN/m**.

- The costs associated with increasing mass are:
$$C_m \text{ (in £)} = 10(m) - 2000 \tag{4}$$
where m is expressed in **kg**.

- The costs associated with purchasing land are:
$$C_x \text{ (in £)} = 2400([x^2]/4) \tag{5}$$
where x is expressed in **m**.

The total cost to improve the design is simply given as $C = C_k + C_m + C_x$. For the initial design, ($k = 1$ MN/m, $m = 200$ kg, $x = 0$ m), all costs are zero but the wall fails so this solution is rejected.

2.4 Suggested order of operations

It is your choice how you solve the coursework, but consider structuring your code around the following:

1. A function which calculates the peak force and duration for a given x distance (Equation 2)
2. A function which calculates the peak displacement of the cantilevered blast wall, z_{max}, for given input values of m, k, and F_{peak} and t_d (or just x if you decide to combine Step 1&2)

   Note that Equation 1 is continuous, i.e. it is true for *all instances* of time. In practice, we don't need to know what the cantilever is doing at all times, only at *select* times. This is known as *temporal discretisation*, and is defined by a time-step value, Δt, such that your equation is solved at $t = 0$, $t = \Delta t$, $2\Delta t$, etc. You should consider having Δt as an input parameter, and you should also consider investigating how sensitive your output (z_{max}) is to this parameter

   You need to consider how long you solve Equation 1 for. You might expect that peak displacement is reached within 1 natural period cycle, T, but it might be worth solving for the first $2T$ just to be safe.

   You should consider including a user-defined input which, when it is set to a certain value (i.e. 1), your code produces a plot of displacement-time, like Figure 3. You can use plotting to sense-check and *verify* that your code is behaving as expected, but you should switch this off if your code has to access this function multiple times (as below).

3. A root-finding algorithm that uses the bisector method to find a value of x (between 0 and 10 m) for any combination of k and m, which gives a peak displacement of exactly (or very close to!) 100 mm. This gives the minimum additional distance required for the blast wall to 'pass'.

4. A piece of code which considers *all* reasonable combinations of k and m, calculates the required distance, x, assigns a cost to these three values (Equations 3–5), and subsequently finds the lowest-cost stiffness and mass pair.

   You may want to consider how your choice of k and m inputs affects your total cost. For example, is it sensible to express the optimal mass to the nearest milligram?

3 The submission

The client has asked you to write a short report (8 pages maximum) which details how your code works, and provides the clients with the optimal value(s) of mass and stiffness to design the blast wall.

Whilst there may be a single optimum, consider whether it is better to provide the client with this exact optimum, or whether it is better to provide a range of permissible options.

You do not need to submit your code. You may want to include small snippets of code to help explain certain concepts, but instead you may want to use flow charts, bullet-pointed lists, and sketches and diagrams to provide a 'walk-through' of the interesting and relevant aspects of your code. Remember what your client wants: they want to know that they are spending their money wisely, so you need to show that your code is accurate, that you are solving the appropriate equations to a suitable accuracy with due consideration of some of the wider engineering issues introduced in this brief.

They do not need to know the exact detail of every line of code you have written, but they will want to know your general approach, an explanation of your logic, and how you instructed the computer to perform certain commands (i.e. your root finding algorithm).

You do not need to include a coversheet, contents page, or appendix. The 8 page limit is a strict limit, and anything beyond the 8th page will not be marked. There is no formal mark scheme, as I want to be flexible and encourage creativity. Your mark will be awarded based on the following:

- Presentation style and clarity of structure
- Written quality, language use, and conciseness
- Accuracy of optimal solution(s)
- Use of sketches, diagrams and flow charts to explain concepts
- Exploration of the results, including use of visuals (2D and 3D plots)
- Consideration and appreciation of wider engineering context and issues related to convergence and numerical accuracy
- Your programming ability, and how well you've explained these concepts to a non-specialist

The coursework forms the entirety of your assessment for the programming aspect of MAS252, and hence accounts for 20% of your total module mark.

The deadline is 9:00 am on Friday of Week 12 (17th Dec 2021).
The coursework forms 20% of your mark for MAS252