Critical Shear Force and Bending Moment calculations in Python

[Chinmay Joshi]

April 7, 2025

1 Introduction

In this report, an analysis of loading on a simple beam is demonstrated. The deliverable shear force and bending moment values are formulated and calculated in Python.

2 Problem Statement

A simply supported beam of length L is acted upon by two moving loads W_1, W_2 acting x distance apart. We need to find shear force and bending moment values for a variety of scenarios.

- **Beam Length** L: Length of the beam in m.
- **Moving Load W_1:** Magnitude of the first moving load in kN.
- **Moving Load W_2:** Magnitude of the second moving load in kN.
- **Load Spacing (x):** The fixed distance between moving loads in m.
- **x to L ratio: x_by_L*** Additional parameter for ease of calculation.
- **Unit ratio of W_1:** U_1
- **Unit ratio of W_2:** U_2

2.1 Objective

Develop a simple program to calculate the following values:

- $\mathbf{R_A^{max}}$**:** The magnitude of the maximum reaction at support A.
- $\mathbf{R_B^{max}}$**:** The magnitude of the maximum reaction at support B.
- $\mathbf{BM_{01}}$**:** The maximum bending moment in the beam when the position of W_1 is 0 (m).
- $\mathbf{SF_{01}}$**:** The maximum shear force in the beam when the position of the centroid of the moving loads is $L/2$.
- $\mathbf{SF_{max}}$**:** The maximum magnitude of shear force experienced by the beam.
- $\mathbf{BM_{max}}$**:** The maximum magnitude of bending moment experienced by the beam.

3 Methodology

3.1

The required values will be obtained using the Influence Line Diagram Method (ILD).

3.2 Derivational Strategy:

1. **Input parameters:** Organize the parameters and their values required for the derivation.
2. **ILD Construction:** Construct the ILD for the required scenario, break it down into separate cases if needed.
3. **Case Solution:** Solve for the required deliverable in one (or more) cases, as necessary.
4. **Collation:** Collate the results, substitute the values for the parameters.
5. **Create functionality:** Create a function that outputs the value of the above derivation.

4 Simply supported Beam

We shall consider a beam AB supported by a pin support at A and a roller support at B.

The free body diagram of the beam shall be as follows.

Here, the reaction $\mathbf{R^*_A}$ is 0 because of the equilibrium condition.

We shall assume henceforth that the location of the moving load $\mathbf{W_1}$ is $\mathbf{a}$ (m); which also fixes the location of $\mathbf{W_2}$ is $\mathbf{a} + \mathbf{x}$ (m).

$$\mathbf{W_1} \xrightarrow{\textbf{location}} \mathbf{a} \in (0, L - x)$$
$$\mathbf{W_2} \xrightarrow{\textbf{location}} \mathbf{b} = \mathbf{a} + \mathbf{x}$$

Let the location at which Shear Force (SF) and Bending moment (BM) is evaluated be $\mathbf{p}$ (m) from the support A.

$$\mathbf{p} \xleftarrow{\textbf{location}} \text{parameter for SF and BM ILDs.} \tag{1}$$

5 Influence Line Diagrams (ILD)

An Influence line is a graph showing a unit load response of a member as an applied load moves over the member.

For a beam of length L as described in the previous section, the ILDs for the support reactions, shear force and bending moments can be formulated as following:

1. $\mathbf{R_A}$**:** The vertical reaction in the left support.

$$r_A(c) = 1 - \frac{c}{L}$$

2. $\mathbf{R_B}$**:** The vertical reaction in the right support.

$$r_B(c) = \frac{c}{L}$$

3. $\mathbf{SF} \xleftarrow{\textbf{loc}} \mathbf{p}$**:** Shear force in the beam $+p$ (m) from the support A.

$$\mathbf{sf}(p, a) = \begin{cases} -\dfrac{c}{L} & \text{if } c < p \\ 1 - \dfrac{c}{L} & \text{if } c > p \end{cases}$$

4. $\mathbf{BM} \xleftarrow{\textbf{loc}} \mathbf{p}$**:** Bending moment in the beam $+p$ (m) from the support A.

$$\mathbf{bm}(p, c) = \begin{cases} \dfrac{c * (L - p)}{L} & \text{if } c < p \\ \dfrac{p * (L - c)}{L} & \text{if } c > p \end{cases}$$

6 Derivations

6.1 Maximum Vertical Reaction at A:

For the moving loads $W_1$and W_2, the value of R_A will be given by:

$$\begin{aligned}
\mathbf{R_A} &= W_1 * \mathbf{r_A}(a) + W_2 * \mathbf{r_A}(b) \\
\mathbf{R_A} &= W_1 * \mathbf{r_A}(a) + W_2 * \mathbf{r_A}(a+x) \\
\mathbf{R_A} &= W_1 * \left(1 - \frac{a}{L}\right) + W_2 * \left(1 - \frac{a}{L} - \frac{x}{L}\right) \\
\mathbf{R_A} &= W_1 + W_2 * \left(1 - \frac{x}{L}\right) - a * \frac{W_1 + W_2}{L} \\
\mathbf{R_A^{max*}} &= W_1 + W_2 * \left(1 - \frac{x}{L}\right) \dots (1)
\end{aligned}$$

Additionally, I shall also consider the case when $a < 0$, i.e. the load W_1 has not yet moved onto the beam.

$$\begin{aligned}
\mathbf{R_A} &= W_2 * \mathbf{r_A}(x+a) && -x < a < 0 \\
\mathbf{R_A} &= W_2 * (1 - \frac{x}{L} - \frac{a}{L}) && -x < a < 0 \\
\mathbf{R_A^{max*}} &= W_2 ... (2)
\end{aligned}$$

From (1) and (2), we can now construct the expression for $\mathbf{R_A^{max}}$.

$$\mathbf{R_A^{max}} = \max\left\{W_1 + W_2 * \left(1 - \frac{x}{L}\right),\ W_2\right\}$$

6.2 Maximum Vertical Reaction at B:

The derivation of $\mathbf{R_B^{max}}$ follows similarly. The value for $\mathbf{R_B}$ for moving loads $W_1$and W_2 shall be given by:

$$\begin{aligned}
\mathbf{R_B} &= W_1 * \mathbf{r_B}(a) + W_2 * \mathbf{r_B}(a+x) \\
\mathbf{R_B} &= W_1 * \frac{a}{L} + W_2 * \frac{a+x}{L} \qquad 0 < a < L - x \\
\implies \mathbf{R_B^{max*}} &= W_1 * \frac{L-x}{L} + W_2 * \frac{L-x+x}{L} \\
\mathbf{R_B^{max*}} &= W_2 + W_1 * \left(1 - \frac{x}{L}\right) \dots (1)
\end{aligned}$$

As with support A, let's also consider when the load W_2 moves off the beam from support B, i.e. $L - x < a < L$.

$$\begin{aligned}
\mathbf{R_B} &= W_1 * \mathbf{r_B}(a) \\
\mathbf{R_B} &= W_1 * \frac{a}{L} \\
& \qquad\qquad\qquad\qquad\qquad\qquad L - x < a < L \\
\implies \mathbf{R_B^{max*}} &= W_1 \ldots (2)
\end{aligned}$$

From (1) and (2),

$$\mathbf{R_B^{max}} = \max\left\{W_1 * \left(1 - \frac{x}{L}\right) + W_2,\ W_1\right\}$$

6.3 Bending moment BM_01:

The Bending moment BM_01 is defined as follows:

- The maximum bending moment across the beam;
- When the load W_1 is at support A; $a = 0$.

A diagrammatic representation is as follows:

To evaluate the maximum, the domain of $p \in [0, L]$ can be split as $p < x$ and $p > x$.

$$\begin{aligned}
\mathbf{BM_01}^* &= \max_p \left(W_2 * \mathbf{bm}(p, x)\right) \\
\mathbf{BM_01}^* &= \max_p \left(W_2 * p * \frac{L - x}{L}\right) \\
& \qquad\qquad\qquad\qquad\qquad\qquad 0 < p \leq x \\
\implies \mathbf{BM_01}^* &= W_2 * x * \left(1 - \frac{x}{L}\right) \ldots (1)
\end{aligned}$$

The other case also trivially gives the same expression, which can also be qualitatively confirmed from the figure.

$$\mathbf{BM\_01} = W_2 * x * \left(1 - \frac{x}{L}\right)$$

6.4 Shear Force SF_01:

The shear force **SF_01** is defined as follows:

- The maximum magnitude of shear force in the beam;

- When the centroid of the moving loads is at the center of the beam;

$$a = \frac{L - x}{2}$$

The scenario is visualized in the following figure.

To evaluate the maximum, the domain of $p \in [0, L]$ can be split into $p < \frac{L-x}{2}$, $p \in \left[\frac{L-x}{2}, \frac{L+x}{2}\right]$, and $p > \frac{L+x}{2}$.

The shear force contributions in the second split oppose each other in direction, so it can be disregarded for maximization.

The shear force **SF_01** from the other splits can be evaluated as:

$$\begin{aligned} \mathbf{SF\_01}* &= \max_p \left(W_1 * \mathbf{sf}\left(p, \frac{L-x}{2}\right) + W_2 * \mathbf{sf}\left(p, \frac{L+x}{2}\right)\right) \\ \implies \mathbf{SF\_01} &= \frac{1}{2} * \left(W_1 + W_2 + \frac{x}{L} * |W_1 - W_2|\right) \end{aligned}$$

6.5 Maximum Shear Force $\mathbf{SF_{max}}$:

The maximum magnitude of Shear Force will be experienced:

- When the shear components of both the loads have the same direction.
- At the end supports as that is the position where the shear force Influence Line peaks.

The cases satisfying the above are visualized in the following figure.

The maximum Shear Force $\mathbf{SF_{max}}$ shall then be:

$$\mathbf{SF_{max}} = \max \begin{cases} W_1 + W_2 * \left(1 - \frac{x}{L}\right) & @p = 0 \\ W_2 + W_1 * \left(1 - \frac{x}{L}\right) & @p = L \end{cases}$$

6.6 Maximum Bending Moment $\mathbf{BM_{max}}$:

The maximum Bending moment at any point **p** in the length of the beam shall occur at one of three moments:

1. When W_1 falls directly on p, i.e. $a = p$.
2. When W_2 falls directly on p, i.e. $a + x = p$.
3. At some critical point during the transition from case 2 to case 1.

The region represented by these conditions is visualized in the following figure.

The total Bending Moment function can be formulated as follows:

$$\mathbf{BM}(p, a) = W_1 * \mathbf{bm}(p, a) + W_2 * \mathbf{bm}(p, a + x)$$

This is a multivariate function defined on the $a - p$ plane.

The maximum of this function on the predefined region in the figure shall be the maximum over the corner points, boundaries and the interior.

The full derivation of the critical points has been omitted for brevity.

$$\left(\frac{1}{2} * (L + U_1 * x), \frac{1}{2} * (L + U_1 * x) - x\right),$$
$$\left(\frac{1}{2} * (L - U_2 * x), \frac{1}{2} * (L - U_2 * x)\right),$$
$$(L - x, L - x),$$
$$(x, 0)$$

7 Solution in Python

```python
from numpy import max, argmax


class problem:
    def __init__(self, L, W1, W2, x):
        """
        L: The length of the beam in metres.
        W1: The first load in kN
        W2: The second load in kN
        x: The spacing between the loads in metres.
        """

        self.L = L
        self.W1, self.W2 = W1 * 1000, W2 * 1000  # converted kN to N
        self.x = x
        self.valid_problem = (
            x < L
        )  # Spacing between the loads shouldn't exceed the beam length
        self.x_by_L = x / L
        self.U1 = self.W1 / (self.W1 + self.W2)
        self.U2 = self.W2 / (self.W1 + self.W2)
```

```python

    def BM_infline(self, p, a):
        """
        p: Distance from support A where BM is being evaluated
        a: Distance of the load from support A

        Returns:
        The value of the influence line for the bending moment.
        """
        if a < 0 or a > self.L:
            return 0
        elif a <= p:
            return a * (1 - p / self.L)
        else:
            return (1 - a / self.L) * p

    def SF_infline(self, p, a):
        """
        p: Distance from support A where BM is being evaluated
        a: Distance of the load from support A

        Returns:
        The value of the influence line for the Shear Force.
        """
        if a < 0 or a > self.L:
            return 0
        elif a < p:
            return -(a / self.L)
        else:
            return 1 - a / self.L

    def SF_calc(self, p, a):
        a1, a2 = a, a + self.x  # distance of both loads from A

        # Total SF = sum (load * unit contribution of load)
        return self.W1 * self.SF_infline(p, a1) + self.W2 * self.SF_infline(p, a2)

    def BM_calc(self, p, a):
        a1, a2 = a, a + self.x

        # Total BM = sum (load * unit contribution of load)
        return self.W1 * self.BM_infline(p, a1) + self.W2 * self.BM_infline(p, a2)

```

```
    def get_max_A(self):  # calculate maximum reaction at A
        return max([self.W2, self.W1 + self.W2 * (1 - self.x_by_L)])

    def get_max_B(self):  # calculate maximum reaction at B
        return max([self.W1, self.W2 + self.W1 * (1 - self.x_by_L)])

    def get_BM_01(self):
        return self.W2 * self.x * (1 - self.x_by_L)

    def get_SF_01(self):
        return 0.5 * (self.W1 + self.W2 + self.x_by_L * abs(self.W1 - self.W2))

    def get_max_SF(self):
        if self.W1 > self.W2:
            return 0, self.W1 + self.W2 * (1 - self.x_by_L)
        else:
            return self.L, self.W2 + self.W1 * (1 - self.x_by_L)

    def get_max_BM(self):
        # Critical points enumerated in 6.6
        crit_p = [
            [
                0.5 * (self.L + self.U1 * self.x),
                0.5 * (self.L + self.U1 * self.x) - self.x,
            ],
            [0.5 * (self.L - self.U2 * self.x), 0.5 * (self.L - self.U2 * self.x)],
            [self.L - self.x, self.L - self.x],
            [self.x, 0],
        ]

        # Calculate the BM values by passing to BM_calc
        BM_values = [self.BM_calc(x[0], x[1]) for x in crit_p]

        # Maximize
        max_BM = max(BM_values)
        max_BM_pos = crit_p[argmax(BM_values)][0]

        # Return the max value and where it occurs.
        return max_BM_pos, max_BM


if __name__ == "__main__":
    # L = input("Enter the length of the Beam (m): ")
```

```python
    # W1 = input("Enter the first load (kN): ")
    # W2 = input("Enter the second load (kN): ")
    # x = input("Enter the spacing x (m): ")

    # Toy problem
    osdag_task = problem(10, 1, 1, 4)

    # Maximum Reaction at A
    max_A = osdag_task.get_max_A()
    print(f"Maximum reaction at support A: {max_A} N")

    # Maximum Reaction at B
    max_B = osdag_task.get_max_B()
    print(f"Maximum reaction at support B: {max_B} N")

    # Get BM_01
    BM_01 = osdag_task.get_BM_01()
    print(f"Value of BM_01: {BM_01} N.m")

    # Get SF_01
    SF_01 = osdag_task.get_SF_01()
    print(f"Value of SF_01: {SF_01} N")

    # Maximum value of Shear Force
    pos_SF, max_SF = osdag_task.get_max_SF()
    print(f"Max Shear force {max_SF} N occurs at {pos_SF} m
   from support A.")

    # Maximum value of Bending Moment
    pos_BM, max_BM = osdag_task.get_max_BM()
    print(f"Max Bending moment {max_BM} occurs at {pos_BM} m
   from support A.")
```

8 Conclusion

This report summarizes the computation used to solve for specified values and deliverables required.