Methodology Documentation: Generic Beam Analysis Tool

Technical Documentation

November 27, 2025

Abstract

This document details the numerical methodology and engineering mechanics principles implemented in the "Generic Beam Analysis Tool." The tool utilizes the **Direct Stiffness Method** (Finite Element Method) to analyze 1D beam structures. It is capable of solving statically determinate and indeterminate systems, handling various support conditions, point constraints, and arbitrary loading configurations through matrix structural analysis.

Contents

1 Introduction

The provided code implements a Finite Element Analysis (FEA) solver specifically tailored for Euler-Bernoulli beam theory. Unlike analytical integration methods (such as Macaulay's Method), this tool discretizes the beam into finite elements, assembles a global stiffness matrix, and solves the resulting system of linear equations.

The core physics engine operates on the equation:

$$[K]\{d\} = \{F\} \tag{1}$$

Where:

- $[K]$ is the Global Stiffness Matrix.
- $\{d\}$ is the Global Displacement Vector (unknowns).
- $\{F\}$ is the Global Load Vector (external forces and moments).

2 Discretization and Mesh Generation

To perform the analysis, the continuous beam is discretized into discrete nodes and elements.

2.1 Node Identification

The algorithm first identifies critical locations along the beam length L. Nodes are generated at:

- The start ($x = 0$) and end ($x = L$) of the beam.
- Locations of all point loads and moments.
- Start and end locations of distributed loads.
- Locations of point constraints (rollers/internal supports).

2.2 Mesh Refinement

To ensure accurate visualization of internal force diagrams, the code performs mesh refinement. If the distance between two critical nodes exceeds a threshold (defined as $L/50$), the segment is subdivided into smaller elements. This ensures that the deflection curve is rendered smoothly and internal force variations are captured accurately.

3 Finite Element Formulation

3.1 Element Stiffness Matrix

The tool uses the standard 4-Degree-of-Freedom (DOF) Euler-Bernoulli beam element. Each node has two DOFs:

1. Vertical translation (v)
2. Rotational displacement (θ)

For an element of length l, Young's Modulus E, and Moment of Inertia I, the local stiffness matrix k_e is:

$$k_e = \frac{EI}{l^3}\begin{bmatrix} 12 & 6l & -12 & 6l \\ 6l & 4l^2 & -6l & 2l^2 \\ -12 & -6l & 12 & -6l \\ 6l & 2l^2 & -6l & 2l^2 \end{bmatrix} \tag{2}$$

The code assumes constant material properties (EI) for calculation purposes, as the tool solves for equilibrium distribution which is independent of stiffness magnitude in statically determinate structures, and dependent on relative stiffness in indeterminate ones.

3.2 Global Assembly

The global stiffness matrix $[K]$ is assembled by mapping the local element matrices to their corresponding global DOFs. For a system with N nodes, the global matrix size is $2N \times 2N$.

4 Load Vector Formulation

The global load vector $\{F\}$ is constructed using the principle of **Equivalent Nodal Forces**.

4.1 Point Loads and Moments

Point loads and moments applied directly at nodal coordinates are added directly to the corresponding index in the global load vector $\{F\}$.

4.2 Distributed Loads (Fixed End Actions)

Distributed loads w (force/length) are not applied to the element lengths directly in the matrix solver. Instead, they are converted into equivalent Fixed End Actions (forces and moments) at the nodes:

For a Uniformly Distributed Load (UDL) q acting over an element of length l:

$$F_{y,start} = \frac{ql}{2} \qquad M_{start} = \frac{ql^2}{12} \tag{3}$$

$$F_{y,end} = \frac{ql}{2} \qquad M_{end} = -\frac{ql^2}{12} \tag{4}$$

These values are added to the global load vector $\{F\}$.

5 Boundary Conditions and Constraints

The tool handles boundary conditions using two distinct methods depending on the type of constraint.

5.1 Support Conditions (Partitioning)

For the main supports at the left and right ends (Pinned, Fixed), the code identifies the constrained Degrees of Freedom.

- **Pinned:** Vertical displacement (v) is constrained.
- **Fixed:** Vertical displacement (v) and Rotation (θ) are constrained.

The solver removes these DOFs from the system of equations (effectively partitioning the matrix) and solves only for the free DOFs.

5.2 Internal Constraints (Penalty Method)

For point constraints (rollers) located arbitrarily along the beam, the tool utilizes the **Penalty Method**.

Instead of resizing the matrix, a very large stiffness value ($k_{penalty}$) is added to the diagonal element of the global stiffness matrix corresponding to the constrained DOF:

$$K_{ii}^{new} = K_{ii}^{old} + k_{penalty} \tag{5}$$

where $k_{penalty} \approx 10^{14} \times EI$.

If a non-zero displacement δ is required, a force of $F = k_{penalty} \times \delta$ is added to the load vector. This forces the displacement at that node to approximate the constrained value.

6 Solution and Post-Processing

6.1 Linear Solver

The system of linear equations is solved using **Gaussian Elimination** with partial pivoting to determine the nodal displacements $\{d\}$.

6.2 Interpolation (Shape Functions)

Once nodal displacements (u) and rotations (θ) are known, the deflection $v(x)$ within an element is interpolated using cubic **Hermite Shape Functions**:

$$v(\xi) = N_1(\xi)u_1 + N_2(\xi)\theta_1 + N_3(\xi)u_2 + N_4(\xi)\theta_2 \tag{6}$$

6.3 Internal Force Calculation

The internal forces are derived from the derivatives of the displacement function:

- **Bending Moment (M):** Proportional to curvature.

$$M(x) = EI\frac{d^2v}{dx^2} \tag{7}$$

- **Shear Force (V):** Proportional to the change in moment.

$$V(x) = -EI\frac{d^3v}{dx^3} \tag{8}$$

6.4 Superposition for Distributed Loads

Since cubic shape functions only model linear shear and linear moment variation (constant shear, linear moment) between nodes, they cannot natively represent the parabolic moment caused by distributed loads within a single element.

To correct this, the code applies the **Principle of Superposition** during the visualization phase. It calculates the local Simply Supported beam solution for the distributed load ($V_{local} = q(L/2 - x)$, $M_{local} = \frac{qx}{2}(L - x)$) and adds these values to the FEM results to generate the accurate parabolic diagrams.