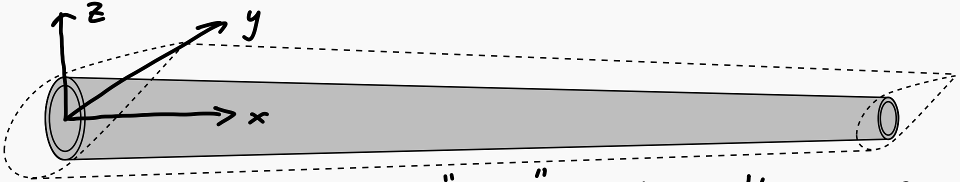
Project #2 Model

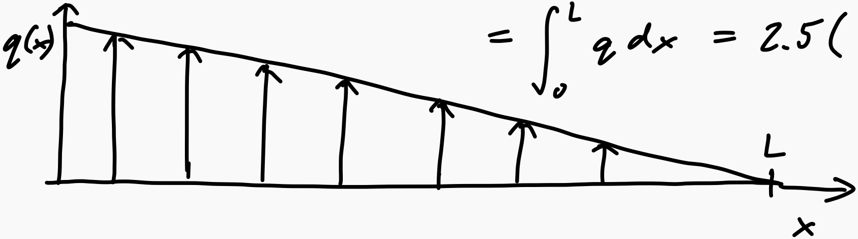
Reminder

Objective: minimize spar weight, subject to stress and manufacturing constraints



"Area" under the curve

$$= \int_0^L q \, dx = 2.5\left(\frac{\text{weight}}{2}\right)$$



Euler-Bernoulli Beam Theory

We will model the spar using Euler-Bernoulli Beam Theory.

Assumptions:

planar symmetry: longitudinal axis is straight, and cross section of beam has a longitudinal plane of symmetry

cross-section variation: cross section varies smoothly

normality: plan sections that are normal to longitudinal plane before bending remain normal after bending

strain energy: internal strain energy accounts only for bending moment deformations

Euler-Bernoulli Beam Theory (cont.)

linearization: deformations are small enough that nonlinear effects are negligible

material: the material is assumed to be elastic and isotropic

Euler-Bernoulli Beam Theory (cont.)

The displacement of the beam in the vertical direction, the direction of the load, is governed by the 4th-order PDE

$$\frac{d^2}{dx^2}\left(EI_{yy}\frac{d^2 w}{dx^2}\right) = q, \qquad \forall x \in [0, L]$$

where

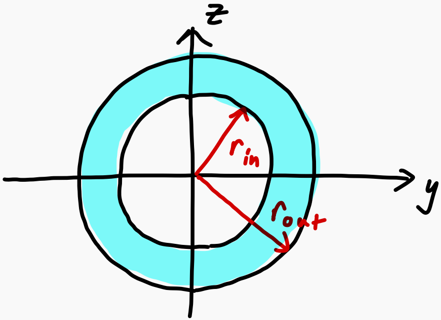
- w is the vertical displacement in the z direction; (m)
- $q(x)$ is the applied load; (N/m)
- E is the elastic, or Young's, modulus, and; (Pa)
- I_{yy} is the second-moment of area with respect to the y axis. (m^4)

Euler-Bernoulli Beam Theory (cont.)

In particular,

$$I_{yy} = \iint z^2 \, dz dy,$$

with the integral taken over the cross-sectional region. It is assumed that the centroid of the cross section is located at $(y, z) = (0, 0)$.



Tables exist for this shape.

Euler-Bernoulli Beam Theory (cont.)

We will treat the spar like a cantilever beam, for which the boundary conditions are

no vertical

$$w(x=0)=0, \qquad \frac{d^2w}{dx^2}(x=L)=0$$

no stress at the tip

or angular displacement at root

$$\frac{dw}{dx}(x=0)=0 \qquad \frac{d^3w}{dx^3}(x=L)=0.$$

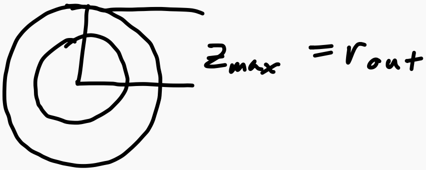
Euler-Bernoulli Beam Theory (cont.)

Once the Euler-Bernoulli equation is solved for w, these displacements can be used to solve for the normal stress as a function of x:

$$\sigma_{xx}(x) = -z_{\max} E \frac{d^2 w}{dx^2}$$

where $z_{\max}$ is the maximum height of the cross section (in this case, the outer radius).

- Since we are interested only in the magnitude of σ_{xx}, the negative sign can be ignored.



Finite-Element Discretization

We will discretize the Euler-Bernoulli beam equation using the finite-element method.

- solution is represented using Hermite-cubic shape functions
- finite-element equations result from the minimization of the potential energy functional

Matlab Implementation

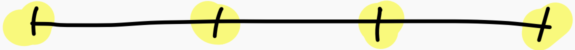
This finite-element discretization of the beam equation is implemented by the (top-level) Matlab function

$w, \frac{dw}{dx}$ in previous slides

```matlab
function [u] = CalcBeamDisplacement(L, E, Iyy, force, Nelem)
% Estimate beam displacements using Euler-Bernoulli
% Inputs:
%   L - length of the beam
%   E - longitudinal elastic modulus
%   Iyy - moment of area with respect to the y axis
%   force - force per unit length along the axis x
%   Nelem - number of finite elements to use
% Outputs:
%   u - displacements at each node along the beam
%-------------------------------------------------
```

Nelem + 1 arrays

2(Nelem + 1) array



Matlab Implementation (cont.)

Once the `u` displacements are known, they can be passed to `CalcBeamStress` to obtain the stress:

```
function [sigma] = CalcBeamStress(L, E, zmax, u, Nelem)
% Compute stress in beam using Euler-Bernoulli
% Inputs:
%   L - length of the beam
%   E - longitudinal elastic modulus
%   zmax - maximum height of the beam at each node
%   u - displacements at each node along the beam
%   Nelem - number of finite elements to use
% Outputs:
%   sigma - stress at each node in the beam
%-----------------------------------------------
```

Nelem + 1 array

from previous function

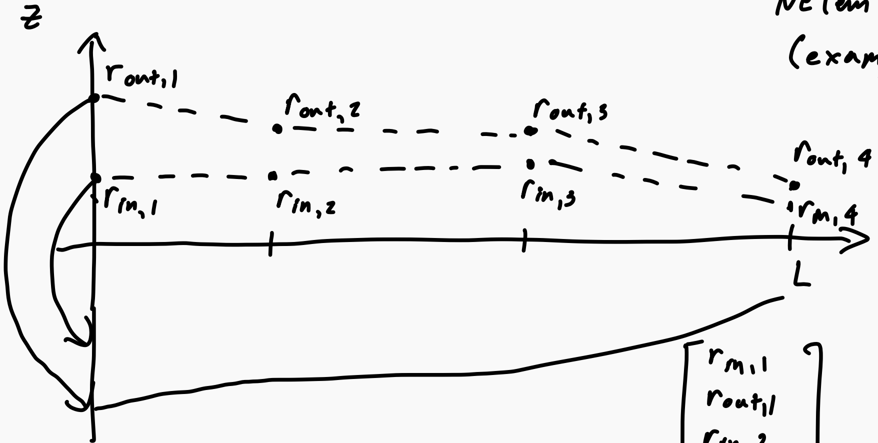
Nelem + 1 array

High-level Steps For Constraint Function

How do you want to represent the design?

$N_{elem} = 3$

(example!)



$$\begin{bmatrix} r_{in,1} \\ r_{out,1} \\ r_{in,2} \\ r_{out,2} \\ \vdots \end{bmatrix} \qquad \begin{bmatrix} r_{in,1} \\ r_{in,2} \\ \vdots \\ r_{out,1} \\ r_{out,2} \end{bmatrix}$$