# C1B Stress Analysis
# Lecture 1:
# Minimum Energy Principles in Mechanics

Prof Alexander M. Korsunsky

Hilary Term (January 08)

http://users.ox.ac.uk/~engs0161/4me6.html

# This course

introduces selected chapters in Stress Analysis

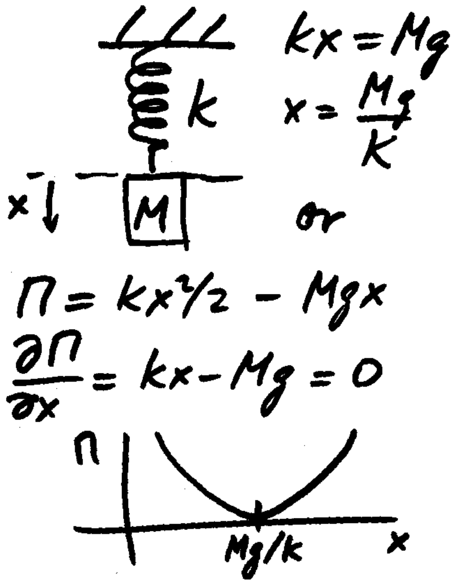
1. Energy principles in deformation theories.
   Variational calculus. Euler equation and boundary conditions.
   Application to beam bending.
   The Rayleigh-Ritz method.

2. Further Euler, Rayleigh-Ritz, and Galerkin. Generalisation to higher dimensions. Piecewise approximation, and the connection with the FEM.

3. Fundamentals of anisotropic elasticity: Stress, strain, elastic constants. The system of equations of elasticity. Analytical solution of elastic problems.
   Plane stress and plane strain.
   The wedge problem.
   Fundamental singular solution for problems involving plane contacts. Connection with the Boundary Element Method (BEM). Solution of contact problems using BEM.

4. Fundamental solutions in elasticity theory.
   The Williams wedge analysis.
   The Westergaard stress function.

# Energy minimisation



$$kx = Mg$$
$$x = \frac{Mg}{k}$$
or
$$\Pi = kx^2/2 - Mgx$$
$$\frac{\partial \Pi}{\partial x} = kx - Mg = 0$$

A simple spring model shows how problems can be solved using the energy minimisation approach, rather than the system of equilibrium equations.

The elastic energy stored in the spring is found as the work done by the internal forces over the displacements, i.e.

$$\mathrm{d}U = F\mathrm{d}x = kx\mathrm{d}x, \quad U = \int_0^{\varepsilon} Fx\mathrm{d}x = \tfrac{1}{2}kx^2$$

During uniaxial deformation of ties, columns, beams etc. the **strain energy density** (i.e. the elastic stored energy per unit volume) is given by:

$$\mathrm{d}U = \sigma\mathrm{d}\varepsilon = \sigma\mathrm{d}\varepsilon = E\varepsilon\mathrm{d}\varepsilon, \quad U = \int_0^{\varepsilon} \sigma\mathrm{d}\varepsilon = \tfrac{1}{2}E\varepsilon^2$$

**Key steps of solution by energy minimisation:**

•Identify key *parameters* that describe deformation

•Express total energy in terms of unknown parameters

•Minimise energy with respect to these parameters

To develop understanding of solving problems by energy minimisation, we will follow this strategy:

1). Discuss fundamentals of minimising **functionals** (e.g. integrals), known as the **calculus of variations**.

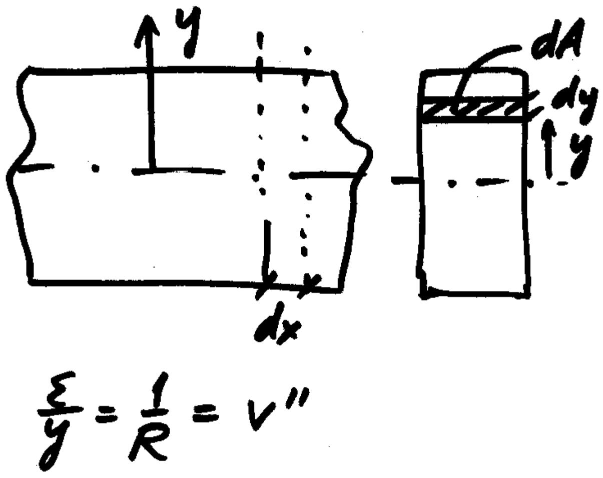
2). Learn about the derivation and use of **Euler's** equation, and the attendant boundary conditions. We will apply Euler's equation to the problem of bending a simply supported beam and a built-in cantilever.

3). Learn how fast and error-free derivations can be carried out using a symbolic algebra package, such as **Mathematica**.

3). Introduce the direct parametrisation and minimisation technique known as the **Rayleigh-Ritz** method. We will then assess its accuracy by comparison with exact solutions.

4). Introduce another method for solving minimisation problems, known as the **Galerkin** method.

# Energy Functionals



The energy to be minimised is often expressed as an integral.

As examples we can consider:

- elastic bending of a beam
- axial tension or compression in ties and columns
- torsion of shafts

Let v(x) denote the deflection. Strain energy density for a bent **beam** :

$$\mathrm{d}U = \sigma \mathrm{d}\varepsilon = E\varepsilon \mathrm{d}\varepsilon, \quad U = \int_0^{\varepsilon} \sigma \mathrm{d}\varepsilon = \tfrac{1}{2} E\varepsilon^2$$

$$W = \int_V \tfrac{1}{2} E\varepsilon^2 \mathrm{d}A\mathrm{d}x = \tfrac{1}{2} E \int_A y^2 \mathrm{d}A \int_L (v'')^2 \mathrm{d}x = \tfrac{1}{2} EI \int_L (v'')^2 \mathrm{d}x$$

For a **tie/column**:

$$W = \tfrac{1}{2} EA \int_L (u')^2 \mathrm{d}x$$

For a **shaft**:

$$W = \tfrac{1}{2} GJ \int_L (\theta')^2 \mathrm{d}x$$

Note that in all cases the energy is expressed as an integral of some derivative of the deformed shape.

In order to find a solution, the *integral* must be minimised with respect to the choice of a *function*, e.g., v(x), u(x) or $\theta$(x).

Additionally, the function must satisfy some boundary conditions at the ends of the interval (or possibly within). For example, at the ends x=0, L of a bent beam free, simply supported, or built-in conditions may be prescribed.

Note that, in general, the energy integral contains a combination of the function and its derivatives of different orders.

Minimisation of a *functional* is the problem of the *calculus of variations*

# Variational calculus

**Problem:** Find a function y(x) that makes the following integral W[y] take a minimum value:

$$W[y] = \int_a^b F(x, y, y', y'')\mathrm{d}x$$

Boundary conditions on the unknown function y(x) may be prescribed:

$y(a)=y_0$, $y(b)=y_1$, $y'(a)=y'_0$, $y'(b)=y'_1$, etc.

If y(x) is the solution, how do we seek it?

Let's try to adjust the methods known from minimisation of functions.

Let y(x) be a solution. Construct **comparison** or **trial solutions**

$$y = y(x) + \alpha\eta(x)$$

When $\alpha=0$, y becomes the solution. Require $\eta(a)=0, \eta(b)=0$ , etc., so that y still satisfies all boundary conditions.

Now minimise W as a *function* of $\alpha$.

How do we determine the conditions on the solution y(x) ?
We know that at a minimum of a function f(x), its derivative $f_x$ vanishes.
Another way of saying the same thing is that the **variation** $\delta f = f_x \delta x$ vanishes.

We now develop the same approach to finding the minimum of functional W[y].

Consider an increment of the introduced parameter $\alpha$, and find the expression for the variation $\delta W$. We identify the conditions this imposes on the functions involved, which are likely to emerge as differential equations, and also some conditions at the region boundary.

# Euler equation

$$W[y] = \int_a^b F(x, y, y', y'')\mathrm{d}x \qquad (1.1)$$

We are seeking the **variation** of W[y] with y.
Consider $\delta W = W[y+\delta y] - W[y]$:

$$W[y+\delta y] - W[y] = \int_a^b F(x, y+\delta y, y'+\delta y', y''+\delta y'')\mathrm{d}x - \int_a^b F(x, y, y', y'')\mathrm{d}x$$

$$\delta W[y] = \int_a^b (F_y \delta y + F_{y'}\delta y' + F_{y''}\delta y'')\mathrm{d}x$$

By carrying out integration by parts as many times as necessary, all derivatives of y can be eliminated from under the integral sign, giving

$$\delta W[y] = \int_a^b [F_y - (F_{y'})' + (F_{y''})'']\delta y \mathrm{d}x$$

Now in order to ensure that W is at a minimum, this variation must be zero **for any** $\delta y$. This can only be guaranteed if the expression in brackets is zero.

The differential equation for the solution y(x) is **the Euler equation**:

$$\boxed{F_y - (F_{y'})' + (F_{y''})'' = 0} \qquad (1.2)$$

**Problem**: find the shape of a string of length L suspended in tension T due to distributed load w.

A line element of string dx becomes $\mathrm{d}x\sqrt{1+(y')^2} = \mathrm{d}x(1+\tfrac{1}{2}(y')^2)$ after loading. The strain is $\tfrac{1}{2}(y')^2$ and strain energy is $\tfrac{1}{2}T(y')^2$, while the work done by the load is wy per unit length. The total energy is given by (1.1), where

$$F(x, y, y', y'') = \tfrac{1}{2}T(y')^2 - wy$$

To write the Euler equation using (1.2), we need to differentiate F with respect to y and y′ as if they were two independent variables. Hence

$$Ty'' + w = 0$$

Apply boundary conditions y(0)=y(L)=0. The solution: $y(x) = \dfrac{w}{2T}x(L-x).$

# Euler: Bent Beam

**Problem:** Minimise the potential energy of a beam with deflection v under general applied force f(x) (may be a combination of distributed and point loads).

Total energy of the system:

$$W = \tfrac{1}{2} EI \int_0^l (v'')^2 \mathrm{dx} - \int_0^l f v \mathrm{dx}$$

The second integral term describes the work done by the force f over displacement v.

The first variation of W is found by considering increments δv and δv''

$$\delta W = \tfrac{1}{2} EI \int_0^l 2v''\delta v'' \mathrm{dx} - \int_0^l f \delta v \mathrm{dx} = EI[v''\delta v']_0^l - EI \int_0^l v''' \delta v' \mathrm{dx} - \int_0^l f\delta v \mathrm{dx}$$

and using integration by parts

$$\delta W = EI[v''\delta v']_0^l - EI[v'''\delta v]_0^l + \int_0^l [EIv'''' - f]\delta v \mathrm{dx}$$

The Euler equation for a bent beam is found as:

$$\boxed{EIv'''' - f = 0} \qquad (1.3)$$

We now discuss the **boundary conditions**.

Some boundary conditions arise from the variation of δW (see above). They are not a consequence of any additional constraints imposed on the system, and for this reason are called **natural** boundary conditions.

Other boundary conditions appear because of additional constraints.

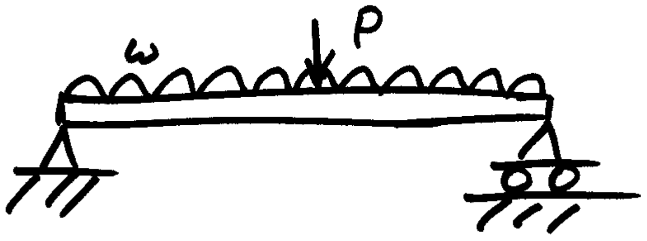
For a bent beam, the following options can be encountered:

(a)If beam ends are free, then δv and δv′ are arbitrary, and their coefficients in the expression for δW must vanish (these are the **natural** b.c.). Hence at a **free end** v″=0 and v‴=0.

(b)If an end is **simply supported**, then the deflection and the bending moment M=-Eiv″ must vanish, leading to the requirements v=0 and v″=0.

(c)If an end is **built-in**, then the deflection and slope must vanish, leading to the requirements v=0 and v′=0.

# Euler: Bent Beam$_2$



$$\boxed{EIv''''-f=0}$$

**Example problem**: a simply supported beam of length L and bending stiffness EI is loaded by a combination of UDL ω and a point load P at mid-span.

Use Euler equation to solve the problem.
The load is given by $f(x)=w+P\delta(x-l/2)$

Here δ(x) is Dirac's delta-function. It is often used in physics and engineering to represent point loads, concentrated charges, etc. Its most important property is expressed in an integral sense:

when the delta-function is integrated over an interval which contains the zero of its argument, the result is unity; otherwise it is zero.

Note that this is precisely what we mean by a unit point load!

To find v(x), we must integrate (four times) Euler equation in the form

$$v''''=[w+P\delta(x-l/2)]/EI$$

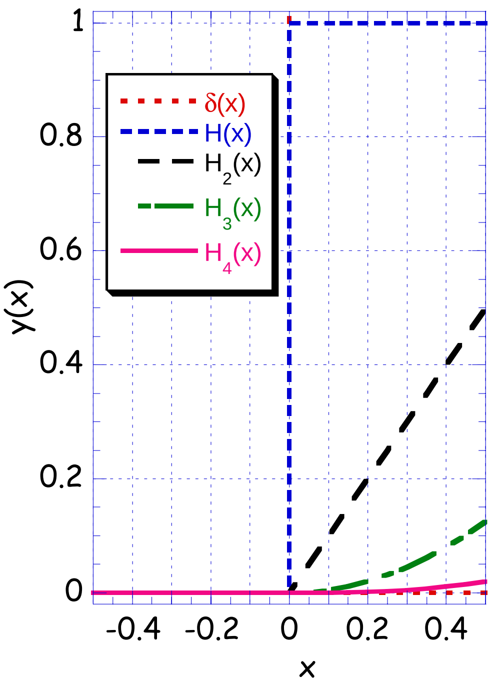
Integrating the delta-function is easy: the first integral produces a unit-step function, which is changes from zero to unity as we go through zero of the argument.

If we denote this first integral of the delta-function by H(x), then subsequent integrations produce functions of the form

$$H_n(x)=H(x)\,x^{n-1}/(n-1)!$$

You must note here the similarity between the delta-function and its integrals, and the Macaulay brackets. It is also important to note that constants of integration must be introduced when solving differential equations.

**Integrating δ-function**



# Using Mathematica

In order to find the solution of the example problem, integration must be performed four times. Four constants will be introduced in the process, which will need to be found from the four boundary conditions. You may want to practice doing that for the exam. For our purposes we need an error-free and fast way to perform these analytical manipulations. We can use one of the **symbolic algebra** packages, such as **Mathematica**.

eulerbeam.nb

```
Clear@v, v1, v2, v3, c1, c2, c3, c4D
H  Euler equation: EI v''''=w + P DiracDelta@x-dD  L
v3 = Integrate@Hw + P DiracDelta@x - dDL ê EI, xD + c1
v2 = Integrate@v3, xD + c2
v1 = Integrate@v2, xD + c3
v = Integrate@v1, xD + c4
```

$$c1 + \frac{w\,x + P\,\text{UnitStep}Hx - dL}{\text{EI}}$$

$$\frac{w\,x^2}{2\,\text{EI}} + c1\,x + c2 + \frac{P\,Hx - dL\,\text{UnitStep}Hx - dL}{\text{EI}}$$

$$\frac{w\,x^3}{6\,\text{EI}} + \frac{c1\,x^2}{2} + c2\,x + c3 - \frac{d\,P\,Hx - dL\,\text{UnitStep}Hx - dL}{\text{EI}} + \frac{P\,Hx^2 - d^2L\,\text{UnitStep}Hx - dL}{2\,\text{EI}}$$

$$\frac{w\,x^4}{24\,\text{EI}} + \frac{c1\,x^3}{6} + \frac{c2\,x^2}{2} + c3\,x + c4 + \frac{d^2\,P\,Hx - dL\,\text{UnitStep}Hx - dL}{2\,\text{EI}} - \frac{d\,P\,Hx^2 - d^2L\,\text{UnitStep}Hx - dL}{2\,\text{EI}} + \frac{P\,Hx^3 - d^3L\,\text{UnitStep}Hx - dL}{6\,\text{EI}}$$

```
v2 ê. x → 0 ê. UnitStep@L - dD → 1 ê. UnitStep@-dD → 0
v ê. x → 0 ê. UnitStep@L - dD → 1 ê. UnitStep@-dD → 0
```

c2

c4

```
v = v ê. c2 → 0 ê. c4 → 0;
v2 = v2 ê. c2 → 0 ê. c4 → 0;
eq2 = v2 ê. x → L ê. UnitStep@L - dD → 1 ê. UnitStep@-dD → 0
v = v ê. Solve@eq2 « 0, c1D@@1DD
v2 = v2 ê. Solve@eq2 « 0, c1D@@1DD
```

$$\frac{w\,L^2}{2\,\text{EI}} + c1\,L + \frac{HL - dL\,P}{\text{EI}}$$

$$\frac{w\,x^4}{24\,\text{EI}} - \frac{HwL^2 + 2\,P\,L - 2\,d\,PL\,x^3}{12\,\text{EI}\,L} + c3\,x + \frac{d^2\,P\,Hx - dL\,\text{UnitStep}Hx - dL}{2\,\text{EI}} - \frac{d\,P\,Hx^2 - d^2L\,\text{UnitStep}Hx - dL}{2\,\text{EI}} + \frac{P\,Hx^3 - d^3L\,\text{UnitStep}Hx - dL}{6\,\text{EI}}$$

$$\frac{w\,x^2}{2\,\text{EI}} - \frac{HwL^2 + 2\,P\,L - 2\,d\,PL\,x}{2\,\text{EI}\,L} + \frac{P\,Hx - dL\,\text{UnitStep}Hx - dL}{\text{EI}}$$

```
eq = Simplify@v ê. x → L ê. UnitStep@L - dD → 1 ê. UnitStep@-dD → 0 ê. c2 → 0 ê. c4 → 0D
v = v ê. Solve@eq « 0, c3D@@1DD
v2 = v2 ê. Solve@eq « 0, c3D@@1DD
```

$$-\frac{w\,L^4 + 8\,d\,P\,L^2 - 24\,c3\,\text{EI}\,L - 12\,d^2\,P\,L + 4\,d^3\,P}{24\,\text{EI}}$$

$$\frac{w\,x^4}{24\,\text{EI}} - \frac{HwL^2 + 2\,P\,L - 2\,d\,PL\,x^3}{12\,\text{EI}\,L} - \frac{H-w\,L^4 - 8\,d\,P\,L^2 + 12\,d^2\,P\,L - 4\,d^3\,PL\,x}{24\,\text{EI}\,L} + \frac{d^2\,P\,Hx - dL\,\text{UnitStep}Hx - dL}{2\,\text{EI}} - \frac{d\,P\,Hx^2 - d^2L\,\text{UnitStep}Hx - dL}{2\,\text{EI}} + \frac{P\,Hx^3 - d^3L\,\text{UnitStep}Hx - dL}{6\,\text{EI}}$$

$$\frac{w\,x^2}{2\,\text{EI}} - \frac{HwL^2 + 2\,P\,L - 2\,d\,PL\,x}{2\,\text{EI}\,L} + \frac{P\,Hx - dL\,\text{UnitStep}Hx - dL}{\text{EI}}$$

# Using Mathematica$_2$

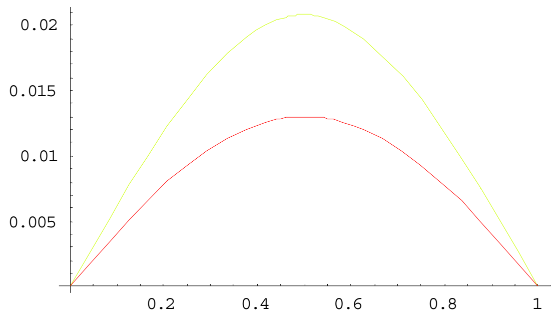
```
In[26]:= valuep = 8EI → 1, L → 1, d → 0.5, P → 1, w → 0<;
         valuew = 8EI → 1, L → 1, d → 0.5, P → 0, w → 1<;
         nvp = v ê. valuep
         nv2p = v2 ê. valuep
         nvw = v ê. valuew
         nv2w = v2 ê. valuew
         Plot@8nvp, nvw<, 8x, 0, 1<, PlotStyle → 8Hue@0.2D, Hue@1D<D
         8nvp ê. x → 0.5, nvw ê. x → 0.5, nvp ê. x → 0.25, nvw ê. x → 0.25<
         Plot@8-nv2p, -nv2w<, 8x, 0, 1<, PlotStyle → 8Hue@0.2D, Hue@1D<D
         8-nv2p ê. x → 0.5, -nv2w ê. x → 0.5, -nv2p ê. x → 0.25, -nv2w ê. x → 0.25<
```

Out[28]= $-0.0833333 x^3 + 0.0625 x + 0.125$ Hx - 0.5L UnitStepHx - 0.5L - 0.25 I$x^2$ - 0.25M UnitStepHx - 0.5L + $\frac{1}{6}$ I$x^3$ - 0.125M UnitStepHx - 0.5L

Out[29]= Hx - 0.5L UnitStepHx - 0.5L - 0.5 $x$

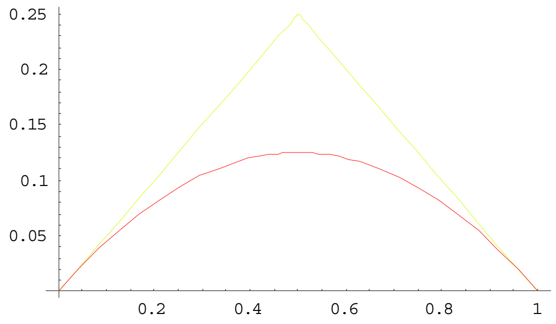
Out[30]= $\frac{x^4}{24} - \frac{x^3}{12} + \frac{x}{24}$

Out[31]= $\frac{x^2}{2} - \frac{x}{2}$



Out[32]= Ü Graphics Ü

Out[33]= 80.0208333, 0.0130208, 0.0143229, 0.00927734<



Out[34]= Ü Graphics Ü

Out[35]= 80.25, 0.125, 0.125, 0.09375<

| Summary | EIv(L/2) | EIv(L/4) | -EIv''(L/2) | -EIv''(L/4) |
|---|---|---|---|---|
| P at L/2 | $0.0208PL^3$ | $0.0143PL^3$ | 0.25PL | 0.125PL |
| UDL ω | $0.0130\omega L^4$ | $0.0093\omega L^4$ | $0.125\omega L^2$ | $0.094\omega L^2$ |

# Rayleigh-Ritz Method

This approach is a powerful alternative to finding and solving Euler equation. It is a **direct** method, in the sense that the solution is assumed to be unknown but of a certain form containing unknowns, which are then found by minimisation.

**Key steps:**

(a) Express the unknown function (e.g. deflected shape) as a linear combination of known functions with unknown coefficients:

$$v(x) = A_1 v_1(x) + A_2 v_2(x) + A_3 v_3(x) + \dots \tag{1.4}$$

(b) Express the total energy of the system W as a function of constants $A_i$.

(c) Find the minimum by requiring

$$\mathrm{grad}_{A_i} W = 0 \quad \text{i.e.} \quad \partial W / \partial A_i = 0 \tag{1.5}$$

The Rayleigh-Ritz method transforms a problem with an infinite number of degrees of freedom into one with a finite number (as many as $A_i$'s used).

## Rayleigh-Ritz: Bent Beam

We seek a solution to the bent beam problem by seeking the deflected shape as a series of the type (1.4), but containing only one term:

$$v(x) = A \sin\frac{\pi x}{L}$$

We find right away that: $v''(x) = -A\left(\frac{\pi}{L}\right)^2 \sin\frac{\pi x}{L}$

The function v(x) is chosen to satisfy the deflection boundary conditions v(0)=v(L)=0 at simply supported ends. Note that it also satisfies the moment boundary conditions, since for it v"(0)=v"(L)=0.

Total energy of the system can be evaluated (e.g. using Mathematica)

$$W = \tfrac{1}{2} EI \int_0^L (v'')^2 \mathrm{d}x - \int_0^L wv \mathrm{d}x - Pv(L/2) = \frac{\pi^4 EIA^2}{4L^3} - PA - \frac{2wLA}{\pi}$$

Since in this case (1.5) reduces to a single equation, the immediate result is

$$\frac{\partial W}{\partial A} = \frac{2\pi^4 EIA}{4L^3} - P - \frac{2wL}{\pi} = 0, \quad A = \frac{2L^3}{\pi^4 EI}\left(P + \frac{2wL}{\pi}\right)$$

# Rayleigh-Ritz Method$_2$

In Mathematica, the solution is found by following the three steps we have indentified (`ritzbeam.nb`):

```
H  Bent beam by Rayleigh-Ritz, one term  L Off@General::spell, General::spell1D
H  Step 1: Define shape in terms of A  L
v = A Sin@Pi x ê LD; v2 = D@v, 8x, 2<D
H  Step 2: Calculate W in terms of A  L
W = Integrate@EI v2^2 ê 2 - w v, 8x, 0, L<D - P H v ê. x → L ê 2L
H  Step 3: Minimise dW ê dA to find solution  L
vr = v ê. Solve@D@W, AD « 0, AD@@1DD
```

Out[6]= $-\frac{A p^2 \sin J \frac{p\,x}{L} N}{L^2}$

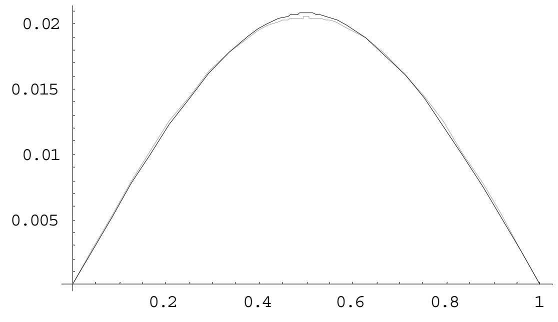
Out[7]= $\frac{A^2\,\mathrm{EI}\,p^5 - 8\,A L^4\,w}{4 L^3 p} - A P$

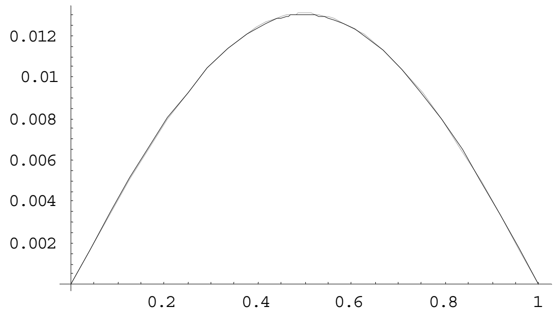
Out[8]= $\frac{2 L^3 H p P + 2 L\, w L \sin J \frac{p\,x}{L} N}{\mathrm{EI}\,p^5}$

It is now possible to compare the results with the exact solutions (by Euler equation), by giving the following commands:

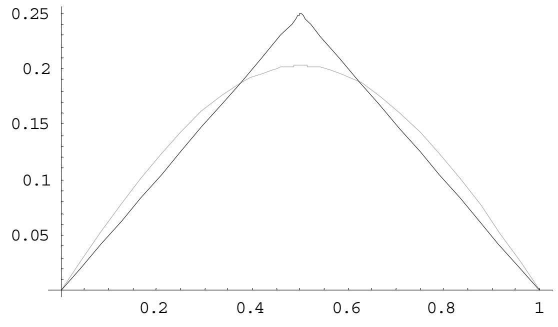
```
In[112]:= vr2 = D@vr, 8x, 2<D
          nvrp = vr ê. valuep
          nvr2p = vr2 ê. valuep
          nvrw = vr ê. valuew
          nvr2w = vr2 ê. valuew
          Plot@8nvrp, nvp<, 8x, 0, 1<, PlotStyle → 8GrayLevel@0.6D, GrayLevel@0D<D
          Plot@8nvrw, nvw<, 8x, 0, 1<, PlotStyle → 8GrayLevel@0.6D, GrayLevel@0D<D
          Plot@8-nvr2p, -nv2p<, 8x, 0, 1<, PlotStyle → 8GrayLevel@0.6D, GrayLevel@0D<D
          Plot@8-nvr2w, -nv2w<, 8x, 0, 1<, PlotStyle → 8GrayLevel@0.6D, GrayLevel@0D<D
```

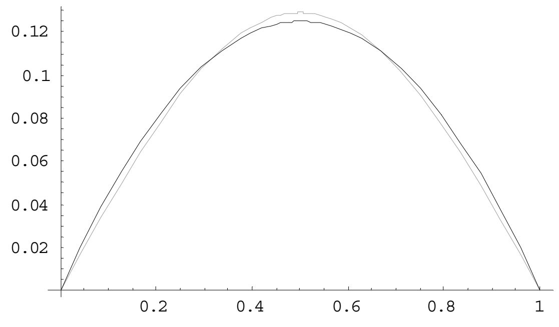
We can now compare the solutions for deflected shapes





and bending moment profiles.





# Rayleigh-Ritz Method$_3$

We compare the exact solution with the R-R approximation:

```
In[59]:= H  Comparison for deflection  L
         H  Exact  L8nvp ê. x → 0.5, nvw ê. x → 0.5, nvp ê. x → 0.25, nvw ê. x → 0.25<
         H  R-R  L8nvrp ê. x → 0.5, nvrw ê. x → 0.5, nvrp ê. x → 0.25, nvrw ê. x → 0.25<

Out[59]= 80.0208333, 0.0130208, 0.0143229, 0.00927734<

Out[60]= 80.020532, 0.0130711, 0.0145183, 0.00924263<

         H  Comparison for bending moment  L
         H  Exact  L8-nv2p ê. x → 0.5, -nv2w ê. x → 0.5, -nv2p ê. x → 0.25, -nv2w ê. x → 0.25<
         H  R-R  L8-nvr2p ê. x → 0.5, -nvr2w ê. x → 0.5, -nvr2p ê. x → 0.25, -nvr2w ê. x → 0.25<

Out[57]= 80.25, 0.125, 0.125, 0.09375<

Out[58]= 80.202642, 0.129006, 0.14329, 0.0912211<
```

R-R captures the deflected shape better than the bending moment profile (i.e. stress). This is because stresses and strains depend on the derivatives of deflection, and require higher order approximation for good agreement.

The quality of approximation **can only improve** with extra terms in (1.4).

Assume $v(x) = \sum_{i=1..N} A_i \sin\frac{i\pi x}{L}$

An efficient solution is obtained if, after writing down the integral expression for W, differentiation with respect to A$_i$ is carried out under the integral sign:

$$\frac{\partial W}{\partial A_i} = EI\int_0^L v''\frac{\partial v''}{\partial A_i}\mathrm{d}x - P\frac{\partial v}{\partial A_i}(L/2) - \int_0^L w\frac{\partial v}{\partial A_i}\mathrm{d}x$$

The resulting integrals are familiar from Fourier analysis. In particular from

$$\int_0^L \sin\left(\frac{i\pi x}{L}\right)\sin\left(\frac{j\pi x}{L}\right)\mathrm{d}x = \delta_{ij}\frac{L}{2}$$

it follows that

$$0 = EI\left(\frac{i\pi}{L}\right)^4\frac{L}{2}A_i - P\sin\left(\frac{i\pi}{2}\right) - \frac{wL}{i\pi}(\cos i\pi - 1)$$

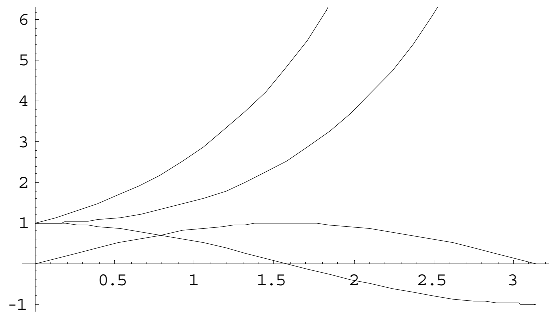
, and

$$A_i = \frac{2L^3}{(i\pi)^4 EI}\left[P\sin\left(\frac{i\pi}{2}\right) + \frac{wL}{i\pi}(\cos i\pi - 1)\right]$$

You can inspect a possible Mathematica implementation of this sine series solution in the file `ritzbeamn.nb`, or construct your own.

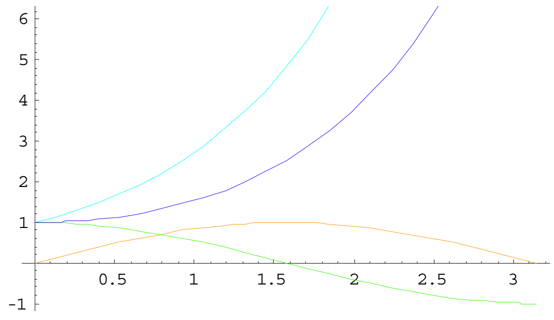
# Mathematica tutorial

```
H  Introduction to using Mathematica for 4 ME6 Stress Analysis  L
H  In the derivations I use a very small part of what Mathematica can do.  L
H  You will have guessed that parenthesis-asterisk is used for comments  L
H  Mathematica understands about most functions.
   To run a CELL, you need to press Shitf+Enter - top status bar should flash
    ' RUNNING', and come up with a plot below  L
Plot@8Sin@xD, Cos@xD, Exp@xD, Cosh@xD<, 8x, 0, Pi<D
```



ÛGraphicsÛ

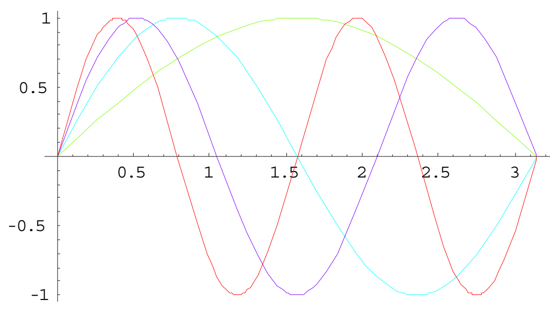
```
H  You may have noticed that the functions are grouped together between curly brackets,
 and separated by commas - they form a LIST. The plotting argument and range also form a list -
  8x, 0, Pi<.
    The plot above looks difficult to read,
 since no colour was used. Instruction can be given to the plot command,
 using PlotStyle→8Hue...< to choose a HUE for each function respectively. HUE argument
   can go from 0 to 1. To find what's what, change the numbers below, and re-run  L
Plot@8Sin@xD, Cos@xD, Exp@xD, Cosh@xD<, 8x, 0, Pi<, PlotStyle → 8Hue@0.1D, Hue@0.3D, Hue@0.5D, Hue@0.7D<D
```



ÛGraphicsÛ

```
H  Mathematica can also build lists using Table:  L
Table@Sin@n Pi ê 6D, 8n, 0, 6<D
Plot@8Sin@xD, Sin@2 xD, Sin@3 xD, Sin@4 xD<, 8x, 0, Pi<, PlotStyle → Table@Hue@0.25 jD, 8j, 1, 4<DD
```

:0, $\frac{1}{2}$, $\frac{è\overline{3}}{2}$, 1, $\frac{è\overline{3}}{2}$, $\frac{1}{2}$, 0>



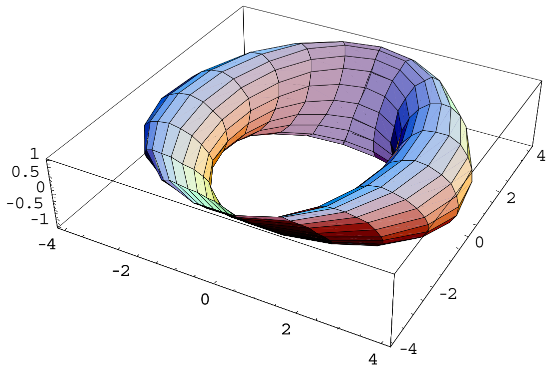
ÛGraphicsÛ

# Mathematica tutorial$_2$

```
H  It can also build arrays:  L
consts = Array@c, 5D
```

8cH1L, cH2L, cH3L, cH4L, cH5L<

```
H  It' s very good at producing pretty 3 D plots  L
u = Sin@rhoD H3 + Cos@thDL;
v = Cos@rhoD H3 + Sin@thDL;
z = Sin@thD;
ParametricPlot3D@8u, v, z<, 8rho, 0, 2 Pi<, 8th, 0, 2 Pi<D
```



ÜGraphics3DÜ

```
H  Mathematica can differentiate using D@D Heven several timesL and find indefinite
  and definite integrals using Integrate@D   L
D@Sin@xD, xD
D@Sin@xD, 8x, 2<D
Integrate@Sin@xD, xD
Integrate@Sin@xD, 8x, 0, Pi<D
```

cosHxL

- sinHxL

- cosHxL

2

```
H  It can do series!  L
Series@Tan@xD, 8x, 0, 15<D
```

$$x+\frac{x^3}{3}+\frac{2x^5}{15}+\frac{17x^7}{315}+\frac{62x^9}{2835}+\frac{1382x^{11}}{155925}+\frac{21844x^{13}}{6081075}+\frac{929569x^{15}}{638512875}+O\text{I}x^{16}\text{M}$$

```
H  Mathematica can solve equations - note the use of ==  L
Solve@x^2 + 5 x - 6 == 0, xD
```

88$x$ Ø - 6<, 8$x$ Ø 1<<

```
H  It can also solve differential equations!  L
DSolve@u''@xD - 9 u@xD « 0, u@xD, xD
```

99$u$HxL Ø ‰$^{-3x}$ $c_1$ + ‰$^{3x}$ $c_2$==

```
DSolve@u''@xD - 4 u'@xD + 4 u@xD « 0, u@xD, xD
```

99$u$HxL Ø ‰$^{2x}$ $c_1$ + ‰$^{2x}$ $x\,c_2$==

# Mathematica tutorial$_3$

```
H  Mathematica is very good at solving systems of equations. Command SOLVE needs a
  list of equations, followed by a list of unknowns to find.  L
SOLUTION = Solve@8x + 3 y + z == 2,
   -x - 2 y + z == 5,
   3 x + 7 y + z == -3<, 8x, y, z<D
```

$::x Ø -\frac{13}{2}, y Ø 2, z Ø \frac{5}{2}>>$

```
H  We call the result 'SOLUTION'. It contains RULES. Each RULE describes a
  substitution: it shows, using an arrow →,
 which values should be given to each
  unknown.
   Our SOLUTION contains the rules enclosed in double curly brackets. To extract the rules,
 we needto use double square brackets, and specify we need the first element  L
SOLUTION@@1DD
```

$:x Ø -\frac{13}{2}, y Ø 2, z Ø \frac{5}{2}>$

```
H  To use the solution, we need to perform the substitution. This is done by using the command ê.
  The result is a list of values of x,y,z that we found  L
8x, y, z< ê. SOLUTION@@1DD
```

$:-\frac{13}{2}, 2, \frac{5}{2}>$

```
H  Finally, procedures can be defined using MODULE.
    This one calculates the sum of odd integers up to n.
    It uses OddQ - a query function, which is true of the integer is odd,
 and false otherwise. Note the C-style calculation of the total  L
OddSum@n_D :=
 Module@8i, total = 0<,
  Do@If@OddQ@iD, total += i, 0D,
   8i, n<D;
  total
 D

H  We can now use OddSum as a function  L
OddSum@100D
```

```
2500
```

In order to appreciate the capabilities of Mathematica, you must try running it yourself.

You can use the program to study the examples from
http://users.ox.ac.uk/~engs0161/4me6.html
 in the ECS lab (the Solarium) on any of the machines booth36.ecs to booth50.ecs (inclusive).
To run Mathematica, type `start_mathematica`. This will start a window titled "`Mathematica4.0 Execution Window`".
Run Mathematica in this window by typing `mathematica`.