Lecture 21: Higher-Order Ordinary Differential Equations

Reading:
Kreyszig Sections: 2.1, 2.2 (pages45–52, 53–58)

Higher-Order Equations: Background

For first-order ordinary differential equations (ODEs), $F(y'(x), y(x), x)$, one value $y(x_o)$ was needed to specify a particular solution. Recall the example in Lecture 19 of a first-order differencing scheme: at each iteration the function grew proportionally to its current size. In the limit of very small forward differences, the scheme converged to exponential growth.

Now consider a situation in which function's current rate of growth increases proportionally to two terms: its current rate of growth and its size.

$$\text{Change in Value's Rate of Change} + \alpha \text{ (the Value)} + \beta \text{ (Value's Rate of Change)} = 0$$

To calculate a forward differencing scheme for this case, let Δ be the forward-differencing increment.

$$\left(\frac{\frac{F_{i+2}-F_{i+1}}{\Delta} - \frac{F_{i+1}-F_i}{\Delta}}{\Delta} \right) + \alpha F_i + \beta \left(\frac{F_{i+1} - F_i}{\Delta} \right) = 0$$

and then solve for the "next increment" F_{i+2} if F_{i+1} and F_i are known.

This indicates that, for second-order equations, two independent values are needed to generate the 'solution trajectory.'

Lecture 21 MATHEMATICA® Example 1

A Second-Order Forward Differencing Example

Download notebooks, pdfs, or html from http://pruffle.mit.edu/3.016-2006.

A second order differencing formula is developed for the case of constant growth and acceleration coefficients.

```
This is the current change or approximation to velocity
1  CurrentChangePerDelta[F_, i_, Δ_] := (F[i + 1] − F[i]) / Δ
Finite difference approximation to second derivative
2  CurrentChangeinCurrentChangeperDelta[F_, i_, Δ_] :=
     Simplify[1/Δ (CurrentChangePerDelta[F, i + 1, Δ] −
       CurrentChangePerDelta[F, i, Δ])]
Let the acceleration is proportional to  size of the current function
and its velocity, let these proportions be: -α and -β
3  DifferenceRelation =
     CurrentChangeinCurrentChangeperDelta[F, i, Δ] ==
       −β CurrentChangePerDelta[F, i, Δ] − α F[i]
4  ForDiffSol = Solve[DifferenceRelation, F[i + 2]] // Flatten
5  ForDiffSolV2 = ForDiffSol /. i → j − 2
6  GrowList[ValuesList_List, Δ_, α_, β_] := Module[
       {Minus1 = ValuesList[[−1]], Minus2 = ValuesList[[−2]]},
       Append[ValuesList,
         2 + Minus1 − Minus2 +
           Δ ∗ (β ∗ (Minus2 − Minus1) − α ∗ Δ ∗ Minus2)]]
7  result = GrowList[{1, 1}, .001, 1, .1]
Generate a sequence of length 20 from initial values {1,1} for
Δ=0.001, α=1, β=0.1
8  Nest[GrowList[#, .001, 1, .1] &, {1, 1}, 20]
9  ListPlot[Nest[GrowList[#, .001, 1, .1] &, {1, 1}, 20000]]
Change parameters for Growth Function (this  shows that the
numerical solution does not converge to the accurate solution):
10 ListPlot[Nest[GrowList[#, 0.01, 0.5, 0] &, {1, 1}, 20000]]
```

1: *CurrentChangeperDelta* is an example of a first-order finite difference.

2: Applying the first-order difference operator twice, a second-order differencing operator is obtained. Notice that, as the higher the order of difference operation goes, the number of surrounding points required to evaluate the difference gets larger and larger—i.e., for the second order difference, function values are needed at three different i compared to two different i for the first-order case.

3: For a particular case of $d^2y/dx^2 = -\alpha dy/dx - \beta y$, the two difference operators replace the derivatives and a *difference relation* can be derived as a function of parameters α and β.

4: The difference operator is derived by solving the difference relation for F_{i+2}—it will depend on the immediate last value F_{i+1} and that value's antecedent F_i. Therefore, any value—including the first one calculated—requires *two values* to be specified.

5: Typically, the current j–value is expressed in terms of the $(j-1)$ and $(j-2)$–values. This form is generated by the replacement $i \rightarrow j-2$.

6: The difference operator is incorporated in *GrowList* : a function that grows a list (input as `ValuesList`) using a difference Δ and parameters α and β. The two previous values in the list become *localized variables* in a `Module` function. The `Module` returns a new list that is created using `Append` to place the current value at end of the input list.

7: Here is an example of using *GrowList* once.

8: Using `Nest` the list can be grown iteratively to N times to generate a sequence of length $N+2$ (the first two values being specified).

10: `ListPlot` visualizes the results for different growth constants α and β.

Linear Differential Equations; Superposition in the Homogeneous Case

A linear differential equation is one for which the function and its derivatives are each linear—that is they appear in distinct terms and only to the first power. In the case of a homogeneous linear differential equation, the solutions are *superposable*. In other words, sums of solutions and their multiples are also solutions.

Therefore, a linear heterogeneous ordinary differential equation can be written as a product of

general functions of the dependent variable and the derivatives for the n-order linear case:

$$\begin{aligned} 0 &= f_0(x) + f_1(x)\frac{dy}{dx} + f_2(x)\frac{d^2y}{dx^2} + \cdots + f_n(x)\frac{d^ny}{dx^n} \\ &= (f_0(x), f_1(x), f_2(x), \ldots, f_n(x)) \cdot \left(1, \frac{dy}{dx}, \frac{d^2y}{dx^2}, \ldots, \frac{d^ny}{dx^n}\right) \\ &= \vec{f}(x) \cdot \vec{D_n}y \end{aligned} \tag{21-1}$$

The homogeneous n^{th}-order linear ordinary differential equation is defined by $f_0(x) = 0$ in Eq. 21-1:

$$\begin{aligned} 0 &= f_1(x)\frac{dy}{dx} + f_2(x)\frac{d^2y}{dx^2} + \cdots + f_n(x)\frac{d^ny}{dx^n} \\ &= (0, f_1(x), f_2(x), \cdots, f_n(x)) \cdot \left(1, \frac{dy}{dx}, \frac{d^2y}{dx^2}, \ldots, \frac{d^ny}{dx^n}\right) \\ &= \vec{f_{hom}}(x) \cdot \vec{D_n}y \end{aligned} \tag{21-2}$$

Equation 21-1 can always be multiplied by $1/f_n(x)$ to generate the general form:

$$\begin{aligned} 0 &= F_0(x) + F_1(x)\frac{dy}{dx} + F_2(x)\frac{d^2y}{dx^2} + \cdots + \frac{d^ny}{dx^n} \\ &= (F_0(x), F_1(x), F_2(x), \ldots, 1)) \cdot (1, \frac{dy}{dx}, \frac{d^2y}{dx^2}, \ldots, \frac{d^ny}{dx^n}) \\ &= \vec{F}(x) \cdot \vec{D_n}y \end{aligned} \tag{21-3}$$

For the second-order linear ODE, the heterogeneous form can always be written as:

$$\frac{d^2y}{dx^2} + p(x)\frac{dy}{dx} + q(x)y = r(x) \tag{21-4}$$

and the homogeneous second-order linear ODE is:

$$\frac{d^2y}{dx^2} + p(x)\frac{dy}{dx} + q(x)y = 0 \tag{21-5}$$

Basis Solutions for the homogeneous second-order linear ODE

Because two values must be specified for each solution to a second order equation—the solution can be broken into two basic parts, each deriving from a different constant. These two independent solutions form a *basis pair* for any other solution to the homogeneous second-order linear ODE that derives from any other pair of specified values.

The idea is the following: suppose the solution to Eq. 21-5 is found the particular case of specified parameters $y(x = x_0) = A_0$ and $y(x = x_1) = A_1$, the solution $y(x; A_0, A_1)$ can be written as the sum of solutions to two *other problems*.

$$y(x; A_0, A_1) = y(x, A_0, 0) + y(x, 0, A_1) = y_1(x) + y_2(x) \tag{21-6}$$

where

$$\begin{aligned} y(x_0, A_0, 0) = A_0 \quad &\text{and} \quad y(x_1, A_0, 0) = 0 \\ y(x_0, 0, A_1) = 0 \quad &\text{and} \quad y(x_1, 0, A_1) = A_1 \end{aligned} \tag{21-7}$$

from these two solutions, any others can be generated.

The two arbitrary integration constants can be included in the definition of the general solution:

$$\begin{aligned} y(x) &= C_1 y_1(x) + C_1 y_2(x) \\ &= (C_1, C_2) \cdot (y_1, y_2) \end{aligned} \tag{21-8}$$

Second Order ODEs with Constant Coefficients

The most simple case—but one that results from models of many physical phenomena—is that functions in the homogeneous second-order linear ODE (Eq. 21-5) are constants:

$$a\frac{d^2y}{dx^2} + b\frac{dy}{dx} + cy = 0 \tag{21-9}$$

If two independent solutions can be obtained, then any solution can be formed from this basis pair.

Surmising solutions seems a sensible strategy, certainly for shrewd solution seekers. Suppose the solution is of the form $y(x) = \exp(\lambda x)$ and put it into Eq. 21-9:

$$(a\lambda^2 + b\lambda + c)e^{\lambda x} = 0 \tag{21-10}$$

which has solutions when and only when the quadratic equation $a\lambda^2 + \lambda x + c = 0$ has solutions for λ.

Because two solutions are needed and because the quadratic equation yields two solutions:

$$\begin{aligned} \lambda_+ &= \frac{-b + \sqrt{b^2 - 4ac}}{2a} \\ \lambda_- &= \frac{-b - \sqrt{b^2 - 4ac}}{2a} \end{aligned} \tag{21-11}$$

or by removing the redundant coefficient by diving through by a:

$$\lambda_+ = \frac{-\beta}{2} + \sqrt{(\frac{\beta}{2})^2 - \gamma}$$
$$\lambda_- = \frac{-\beta}{2} - \sqrt{(\frac{\beta}{2})^2 - \gamma} \tag{21-12}$$

where $\beta \equiv b/a$ and $\gamma \equiv c/a$.

Therefore, any solution to Eq. 21-9 can be written as

$$y(x) = C_+ e^{\lambda_+ x} + C_- e^{\lambda_- x} \tag{21-13}$$

This solution recreated with a slightly different method in the following MATHEMATICA® example.

Lecture 21 MATHEMATICA® Example 2

Solutions to the Homogeneous Linear Second Order ODE with Constant Coefficients

Download notebooks, pdfs, or html from http://pruffle.mit.edu/3.016-2006.

Even though MATHEMATICA® is able to determine solutions to linear second-order ODEs with constant coefficients directly, it is still instructive to use MATHEMATICA® to derive these solutions.

Analysis of basis solutions to $y'' + \beta y' + \gamma y = 0$ in terms of constant coefficients β and γ

```
1 TheODE[function_, var_] := D[function[var], {var, 2}] +
    β D[function[var], var] + γ function[var]
2 TheODE[y, x]
3 TheGuess[x_] := Exp[λ x]
4 TheODE[TheGuess, x]
5 λSolution = Solve[TheODE[TheGuess, x] == 0, λ]
```

The two roots $\lambda+$ and $\lambda+$ are:

```
6 {λMinus, λPlus} = λ /. λSolution
7 GeneralSolution[x_] :=
    C[LPlus] Exp[λPlus x] + C[LMinus] Exp[λMinus x]
8 TheODE[GeneralSolution, z]
9 Simplify[TheODE[GeneralSolution, z]]
```

1: *TheODE* represents the left-hand side of any second-order ODE with constant coefficients. It takes an argument for the name of the function (i.e., y) and the dependent variable (i.e., x in $y(x)$).

3: This will serve as a 'guess' of a solution—if we can find λ(s) that satisfy the ODE, then the solution(s) are determined.

5: Using `Solve` with the guess inserted into *TheODE* will determine solutiion conditions on λ—this will be a quadratic equation in λ.

6: By inspecting the solution, assignments can be made to the two possible λ.

7: This is the form of the general solution in terms of two arbitrary constants.

9: This should show that the general solution always satisfies the ODE.

Lecture 21 MATHEMATICA® Example 3

Characterizing the Solution Behavior for the Second-Order ODE with Constant Coefficients

Download notebooks, pdfs, or html from http://pruffle.mit.edu/3.016-2006.

Because the fundamental solution depend on only two parameters β and γ, the behavior (i.e., whether $\Re\lambda \overset{>}{<} 0$ and $\Im\lambda \overset{?}{=} 0$) of all solutions can be visualized in the γ-β plane.

```
1  Reduce[λPlus ∈ Reals && λMinus ∈ Reals, {β, γ}, Reals]
2  CplexReal = Plot[β^2/4, {β, -1, 1}, AxesLabel → {"β", "γ"}]
3  CplexRealAnnote = Show[CplexReal, Graphics[Text[
       "Complex\nCongugate\nRoots", {0.25, 0.25}, {-1, 1}]],
     Graphics[Text["Real Roots", {0.75, 0.05}]]]
4  CplexPosNeg = ParametricPlot[{0, t},
     {t, 0, .25}, PlotStyle → {Thickness[0.015], Hue[0]},
     DisplayFunction → Identity]
5  CplexPosNegAnnote = Show[CplexPosNeg,
     Graphics[Text[StyleForm["Positive\nReal\nPart",
       FontColor → Hue[0]], {-.5, 0.15}, {-1, 1}]],
     Graphics[Text[StyleForm["Negative\nReal\nPart",
       FontColor → Hue[0]], {.5, 0.15}, {1, 1}]],
     DisplayFunction → $DisplayFunction]
6  CplexPlot = Show[CplexRealAnnote, CplexPosNegAnnote]
7  Reduce[{λPlus > 0, λMinus > 0}]
8  AnnotePosRealRoots =
     Graphics[Text[StyleForm["Positive Roots",
       FontColor → Hue[.6]], {-1.0, 0.025), {-1, 0}]]
9  Reduce[{λPlus > 0, λMinus < 0}]
10 MixedRealRoots =
     Plot[0, {t, -1, 1}, PlotStyle → {Hue[0.6], Thickness[0.015]},
       DisplayFunction → Identity]
   AnnoteMixedRealRoots = Show[MixedRealRoots,
     Graphics[Text[StyleForm["Mixed Real Roots",
       FontColor → Hue[.6]], {0.2, -0.1), {-1, 0}]],
     DisplayFunction → $DisplayFunction]
11 Reduce[{λPlus < 0, λMinus < 0}]
12 AnnoteNegRealRoots =
     Graphics[Text[StyleForm["Negative Roots",
       FontColor → Hue[.6]], {1.0, 0.025), {1, 0}]]
13 Show[CplexPlot, AnnotePosRealRoots,
     AnnoteMixedRealRoots, AnnoteNegRealRoots]
```

1: `Reduce` is a function for determing the conditions on parameters (here β and γ assumed to be real numbers) such that an expression satisfies particular constraints. The result will create the following graphic.

2: This will create a plot that distinguishes two regions in the γ-β plane: above $\gamma = \beta^2/4$, the λ are real; below, the λ are complex and oscillatory solutions appear (because $\exp(r + \imath\theta) = \exp(r)(\cos(\theta) + \imath\sin(\theta))$).

3: `Graphics` and `Text` create annotation; `Show` combines annotation and the plot.

4: This will create a plot for the conditions that the $\Re(\lambda)$ are real and either positive or negative. The sign of the real part of λ in $\exp(\lambda x)$ determines whether the solution grows without bound ($\Re(\lambda) > 0$) or shrinks asympotically towards 0 ($\Re\lambda < 0$).

5: This creates graphics to annotate and display together with the plot. The extra function `StyleForm` allows the passage of options (such as `FontColor`, `FontFamily`, etc.) to be passed simply.

7: `Reduce` determines the conditions on β and γ so that both λ are positive and real. These will be unbounded and non-oscillating solutions.

8: The conditions will be annotated by greating a graphical object.

10: Here the curves and annotation are created for the case of mixed real roots (i.e., $\lambda_+ > 0$ and $\lambda_- < 0$—one growing and one decaying non-oscillatory solutions)

12: The final region to be determined and annotated is the one with the monotonically decaying solutions $\lambda_- < \lambda_- < 0$.

13: Collecting all the graphical objects together into one image that was used to construct Fig. 21-21. The behavior of all solutions can be collected into a simple picture:

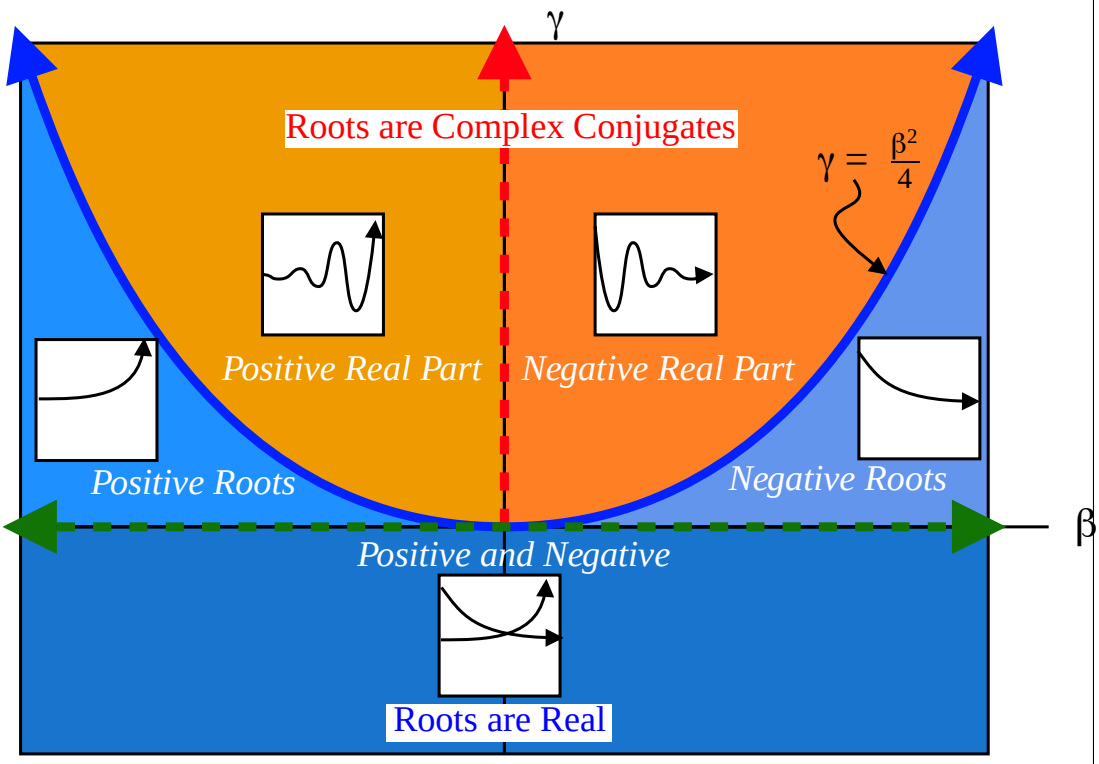

Figure 21-21: The behaviors of the linear homogeneous second-order ordinary differential equation $\frac{d^2y}{dx^2} + \beta\frac{dy}{dx} + \gamma y = 0$ plotted according the behavior of the solutions for all β and γ.

The case that separates the complex solutions from the real solutions, $\gamma = (\beta/2)^2$ must be treated separately, for the case $\gamma = (\beta/2)^2$ it can be shown that $y(x) = \exp(\beta x/2)$ and $y(x) = x\exp(\beta x/2)$ form an independent basis pair (see Kreyszig *AEM*, p. 74).

Boundary Value Problems

It has been shown that all solutions to $\frac{d^2y}{dx^2} + \beta\frac{dy}{dx} + \gamma y = 0$ can be determined from a linear combination of the basis solution. Disregard for a moment whether the solution is complex or real, and ignoring the special case $\gamma = (\beta/2)^2$. The solution to any problem is given by

$$y(x) = C_+ e^{\lambda_+ x} + C_- e^{\lambda_- x} \tag{21-14}$$

How is a solution found for a particular problem? Recall that *two values* must be specified to get a solution—these two values are just enough so that the two constants C_+ and C_- can be obtained.

In many physical problems, these two conditions appear at the boundary of the domain. A typical problem is posed like this:

Solve

$$m\frac{d^2y(x)}{dx^2} + \nu\frac{dy(x)}{dx} + ky(x) = 0 \qquad \text{on } 0 < x < L \tag{21-15}$$

subject to the boundary conditions

$$y(x = 0) = 0 \qquad \text{and} \qquad y(x = L) = 1$$

or, solve

$$m\frac{d^2y(x)}{dx^2} + \nu\frac{dy(x)}{dx} + ky(x) = 0 \qquad \text{on } 0 < x < \infty \tag{21-16}$$

subject to the boundary conditions

$$y(x=0) = 1 \qquad \text{and} \qquad y'(x=L) = 0$$

When the value of the function is specified at a point, these are called *Dirichlet* conditions; when the derivative is specified, the boundary condition is called a *Neumann* condition. It is possible have boundary conditions that are mixtures of Dirichlet and Neumann.

Lecture 21 MATHEMATICA® Example 4

Determining Solution Constants from Boundary Values

Download notebooks, pdfs, or html from http://pruffle.mit.edu/3.016-2006.

Here is an example of taking the general solution with undetermined constants and using boundary conditions to determine a specific solution.

1: *GeneralSolution* is the solution to $y'' + \beta y' + \gamma y = 0$ with undetermined constants `Cplus` and `Cminus`.

2: To find the constants for a particular solution the boundary conditions, $y(0) = 0$ and $y(L) = 1$ where $y(x)$ is the general solution, are used with `Solve` to determine the constants.

3: The form of the particular solution is obtained by back-substituting the solution for the constants into the general solution.

4: For application of a Neumann condition, the symbolic form of the derivative is required.

6: The particular solution for boundary conditions $y'(0) = y(0) = 0$ is obtained by inserting these equations into `Solve` and subsequent replacement into the general solution.

```
1  GeneralSolution[x_] :=
     CPlus Exp[λPlus x] + CMinus Exp[λMinus x]

Second order ODEs require that two conditions be specified to
generate a particular solution. For y(0) = 0 and y(L)=1

2  SolutionOne =
     Solve[{GeneralSolution[0] == 0, GeneralSolution[L] == 1},
       {CPlus, CMinus}]

3  SpecificSolutionOne =
     Simplify[GeneralSolution[x] /. SolutionOne]

Second example with different form of boundary condition:
y(0) = 1 and y'(0)=0

4  DGen = D[GeneralSolution[x], x]

5  SolutionTwo =
     Solve[{GeneralSolution[0] == 1, (DGen /. x → 0) == 0},
       {CPlus, CMinus}]

6  SpecificSolutionTwo =
     Simplify[GeneralSolution[x] /. SolutionTwo]
```

Fourth Order ODEs, Elastic Beams

Another linear ODE that has important applications in materials science is that for the deflection of a beam. The beam deflection $y(x)$ is a linear fourth-order ODE:

$$\frac{d^2}{dx^2}\left(EI\frac{d^2y(x)}{dx^2}\right) = w(x) \tag{21-17}$$

where $w(x)$ is the load density (force per unit length of beam), E is Young's modulus of elasticity for the beam, and I is the moment of inertia of the cross section of the beam:

$$I = \int_{\mathrm{A}_{\times-sect}} y^2 dA \tag{21-18}$$

is the second-moment of the distribution of heights across the area.

If the moment of inertia and the Young's modulus do not depend on the position in the beam (the case for a uniform beam of homogeneous material), then the beam equation becomes:

$$EI\frac{d^4y(x)}{dx^4} = w(x) \tag{21-19}$$

The homogeneous solution can be obtained by inspection—it is a general cubic equation $y_{homog}(x) = C_0 + C_1x + C_2x^2 + C_3x^3$ which has the four constants that are expected from a fourth-order ODE.

The particular solution can be obtained by integrating $w(x)$ four times—if the constants of integration are included then the particular solution naturally contains the homogeneous solution.

The load density can be discontinuous or it can contain Dirac-delta functions $F_o\delta(x - x_o)$ representing a point load F_o applied at $x = x_o$.

It remains to determine the constants from boundary conditions. The boundary conditions can be determined because each derivative of $y(x)$ has a specific meaning as illustrated in Fig. 21-22.

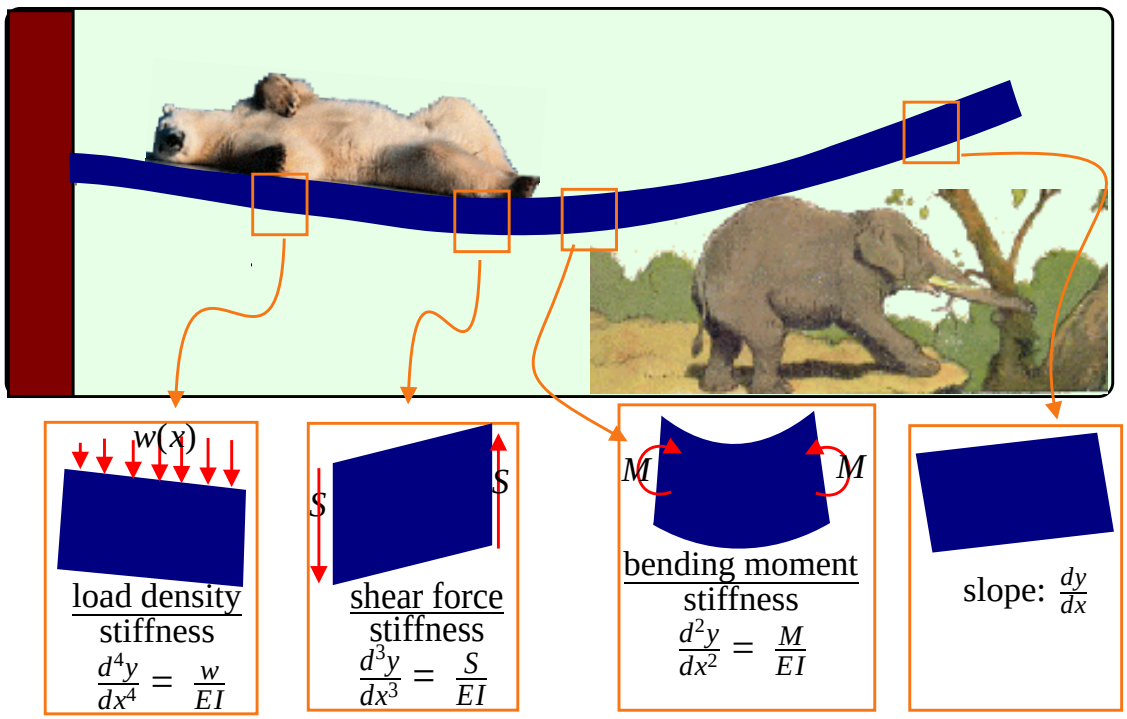

Figure 21-22: The shape of a loaded beam is determined by the loads applied over its length and its boundary conditions. The beam curvature is related to the local moment (imagine two handles rotated in opposite directions on a free beam) divided by the effective beam stiffness. Shear forces are related to the rate of change of moment along the beam.
(Polar Bear Photo Art Wolfe The Zone Network
http://classic.mountainzone.com/climbing/greenland/graphics/polar-bear.html)

There are common loading conditions that determine boundary conditions:

Free No applied moments or applied shearing force:

$$M = \left.\frac{d^2y}{dx^2}\right|_{boundary} = 0$$

$$S = \left.\frac{d^3y}{dx^3}\right|_{boundary} = 0$$

Point Loaded local applied moment, displacement specified.

$$M = \left.\frac{d^2y}{dx^2}\right|_{boundary} = M_o$$

$$y(x)|_{boundary} = y_o$$

Clamped Displacement specified, slope specified

$$\left.\frac{dy}{dx}\right|_{boundary} = s_o$$

$$y(x)|_{boundary} = y_o$$

Lecture 21 MATHEMATICA® Example 5

Visualizing Beam Deflections

Download notebooks, pdfs, or html from http://pruffle.mit.edu/3.016-2006.

A method for solving and visualizing the deflection of a uniform beam is developed for typical boundary conditions and load distributions

1: *BeamEquation* takes arguments for the (unknown) deflection y and its dependent argument x, a loading density $w(x)$, and boundary condition lists `BC1` and `BC2`, and uses `DSolve` to return replacement rules for a particular solution to the beam deflection equation (i.e., $d^4y/dx^4 = w(x)$).

2: *Clamp* , *PointLoad* , and *FreeEnd* are functions that specify the typical boundary conditions: *Clamp* takes an x-value where the clamp is applied, the displacement (`position`) of the clamp and the clamps angle. *PointLoad* takes a position, deflection, and applied torque; *FreeEnd* specifies a point that is unloaded and untorqued.

3: *noload* is an example loading distribution where there is no applied load.

4: As an example of application of *BeamEquation* , here the solution for an unloaded beam is calculated with a fixed horizontal clamp at the origin and a fixed torque-free displacent at the end.

5: To plot the beam deflection, the solution condition is applied to $y(x)$.

6: The function *BeamViz* collects the solution with the visualization for beams of unit normalized length, and uniform normalized stiffness EI.

7: etc. Several different loading conditions and boundary conditions are visualized as examples of *BeamViz*

```
1  BeamEquation[y_, x_, w_, BC1_, BC2_] := DSolve[
       Flatten[{y''''[x] == w[x], BC1, BC2}], y[x], x] // Flatten

2  Clamp[y_, x_, position_, slope_] :=
     {y[x] == position, y'[x] == slope}
   PointLoad[y_, x_, position_, moment_] :=
     {y[x] == position, y''[x] == moment}
   FreeEnd[y_, x_] := {y''[x] == 0, y'''[x] == 0}

3  noload[x_] := 0

4  BeamEquation[y, x, noload,
     Clamp[y, 0, 0, 0], PointLoad[y, 1, -.1, 0]]

5  Plot[Evaluate[
       y[x] /. BeamEquation[y, x, noload,
         Clamp[y, 0, 0, 0], PointLoad[y, 1, -.25, 0]]
     ], {x, 0, 1}, PlotRange → {-0.5, 0.5}, AspectRatio → 1]

6  BeamViz[DistLoadx_, BC1_, BC2_] :=
     Plot[Evaluate[
         y[x] /. BeamEquation[y, x, DistLoadx, BC1, BC2]]
       , {x, 0, 1}, PlotRange → {-0.5, 0.5}, AspectRatio → 1,
       PlotStyle -> {Thickness[0.03], Hue[0]}]

7  unitload[x_] := 1

8  BeamViz[unitload, Clamp[y, 0, 0, 0], FreeEnd[y, 1]]

9  midload[x_] := -10 DiracDelta[x - 1/2]

10 BeamViz[midload, Clamp[y, 0, 0, 0], PointLoad[y, 1, 0, 0]]

11 BeamViz[midload, PointLoad[y, 0, 0, 0], PointLoad[y, 1, 0, 0]]

12 testload[x_] := 500 * (1/2 - x)

13 BeamViz[testload, PointLoad[y, 0, 0, 0], PointLoad[y, 1, 0, 0]]

14 boxload[x_] := -500 *
     (UnitStep[x - (3/4 - 1/8)] - UnitStep[x - (3/4 + 1/8)])

15 Plot[boxload[x], {x, 0, 1}]

16 BeamViz[boxload, Clamp[y, 0, 0, 0], Clamp[y, 1, 0, 0]]
```