

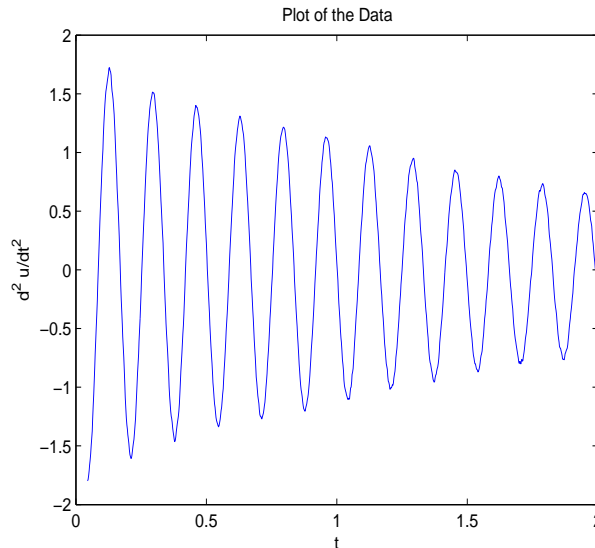
## An Ordinary Differential Equations Worksheet:

“What do a Spring/Mass System and a Vibrating Beam have in Common?”

### Worksheet Objectives:

1. To see that an ODE studied in an elementary differential equations course can provide an accurate model for data collected from the observation of a physical system.
2. To be introduced to a special kind of *inverse problem* known as a *nonlinear parameter estimation problem* – the interface between mathematics and the study of nonlinear phenomena in nature, i.e. natural phenomena modelled using differential equations.
3. To be introduced to the field of *numerical optimization*.

**The Physical System:** In June of 2003, a group of undergraduates gathered for a workshop on Inverse Problems at North Carolina State University. The students collected data on the acceleration  $u''$  of a vibrating beam. Here is a data set that one of the student groups collected:



**The Mathematical Model:** We consider the homogeneous, second-order, linear, constant coefficient ODE

$$u''(t) + c u'(t) + k u(t) = 0, \quad (1)$$

$$u(t_0) = u_0, \quad u'(t_0) = 0, \quad (2)$$

which models a spring/mass system.

**The Parameter Estimation (Inverse) Problem:** The question asked in the title of the worksheet can now be restated: Are there values for  $c$  and  $k$  in (1) such that the corresponding solution  $u''$  has a graph that closely approximates the data plotted in the above figure?

To answer this, we have to first pose the problem in nonlinear least-squares form

$$\min_{(c,k)} f(c, k) := \frac{1}{2} \|\mathbf{U}''(c, k) - \mathbf{data}\|^2, \quad (3)$$

where  $\mathbf{U}''(c, k)$  is a vector with components  $U_j''(c, k) = -c u'(t_j) - k u(t_j)$  and  $\mathbf{data} = (d_1, \dots, d_n)$  is the data vector plotted above.

**Numerical Optimization:** We solve (3) iteratively using an optimization algorithm. Such routines have the general form:

0. Choose an initial guess  $\mathbf{x}_0 = (c_0, k_0)$ .
1. For  $j = 0, 1, 2, \dots$ 
  - (a) Given  $\mathbf{x}_j = (c_j, k_j)$ , choose a step  $\mathbf{v}_j = (p_j, q_j)$  such that  $\mathbf{x}_j + \mathbf{v}_j$  provides a “sufficient decrease” in the value of  $f$ .
  - (b) Set  $\mathbf{x}_{j+1} = \mathbf{x}_j + \mathbf{v}_j$ .
2. If the stopping criteria are met, stop. Otherwise set  $j := j + 1$  and return to 1.

**Newton’s Method:** The iteration in which 1 (b) above given by

$$\mathbf{x}_{j+1} = \mathbf{x}_j - \nabla^2 f(\mathbf{x}_j)^{-1} \nabla f(\mathbf{x}_j), \quad (4)$$

$$= \begin{bmatrix} c_j \\ k_j \end{bmatrix} - \begin{bmatrix} \partial^2 f(c_j, k_j) / \partial c^2 & \partial^2 f(c_j, k_j) / \partial c \partial k \\ \partial^2 f(c_j, k_j) \partial c \partial k & \partial^2 f(c_j, k_j) / \partial k^2 \end{bmatrix}^{-1} \begin{bmatrix} \partial f(c_j, k_j) / \partial c \\ \partial f(c_j, k_j) / \partial k \end{bmatrix} \quad (5)$$

is called Newton’s Method.

**Connection to Calculus:** Recall from Calculus I, Newton’s Method for solving  $g(x) = 0$  is given by

$$x_{j+1} = x_j - \frac{g(x_j)}{g'(x_j)}. \quad (6)$$

If we extend (6) for use on equations of the type  $\mathbf{g}(\mathbf{x}) = \mathbf{0}$ , where  $\mathbf{x}, \mathbf{0} \in \mathbb{R}^n$ , we obtain the iteration

$$\mathbf{x}_{j+1} = \mathbf{x}_j - J^{-1}(\mathbf{x}_j) \mathbf{g}(\mathbf{x}_j), \quad (7)$$

where  $J(\mathbf{x})$  is the Jacobian matrix of  $\mathbf{g}$  evaluated at  $\mathbf{x}$ , and can be viewed as  $\mathbf{g}'(\mathbf{x})$ . Applying (7) to the equation

$$\nabla f(\mathbf{x}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

where  $\nabla f(\mathbf{x})$  is given on the far right in (5), we obtain the algorithm given by (4), (5).

**The Gauss-Newton Method:** Unfortunately, Newton’s Method will not solve (3) for an arbitrary initial guess  $(c_0, k_0)$ , so we use instead a slightly more sophisticated, and computational efficient, but very similar, algorithm known as the Gauss-Newton Method. This algorithm follows the general form given above. MATLAB’s version of the Gauss-Newton method is implemented in the accompanying code, and the accompanying AVI movie gives a visual demonstration of the algorithm solving (3).