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:

![Plot of the Data](image)

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

\[
 u''(t) + c u'(t) + k u(t) = 0, \\
 u(t_0) = u_0, \quad u'(t_0) = 0,
\]

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

\[
 \min_{(c,k)} f(c, k) := \frac{1}{2} ||U''(c, k) - \text{data}||^2,
\]

where $U''(c, k)$ is a vector with components $U_j''(c, k) = -c u'(t_j) - k u(t_j)$ and $\text{data} = (d_1, \ldots, d_n)$ 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 $x_0 = (c_0, k_0)$.

1. For $j = 0, 1, 2, \ldots$
   
   (a) Given $x_j = (c_j, k_j)$, choose a step $v_j = (p_j, q_j)$ such that $x_j + v_j$ provides a “sufficient decrease” in the value of $f$.

   (b) Set $x_{j+1} = x_j + v_j$.

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

$$x_{j+1} = x_j - \nabla^2 f(x_j)^{-1} \nabla f(x_j),$$

is called Newton’s Method.

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

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

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

$$x_{j+1} = x_j - J^{-1}(x_j)g(x_j),$$  \(7\)

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

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

where $\nabla f(x)$ 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).