UPPSALA
UNIVERSITET

Institutionen för informationsteknologi
Beräkningsvetenskap

Besöksadress:
MIC hus 2, Polacksbacken
Lägerhyddvägen 2

Postadress:
Box 337
751 05 Uppsala

Telefon:
018–471 0000 (växel)

Telefax:
018–51 19 25

Hemsida:
http://www.it.uu.se/

Department of Information Technology
Scientific Computing

Visiting address:
MIC bldg 2, Polacksbacken
Lägerhyddvägen 2

Postal address:
Box 337
SE-751 05 Uppsala
SWEDEN

Telephone:
+46 18–471 0000 (switch)

Telefax:
+46 18–51 19 25

Web page:
http://www.it.uu.se/

Assignment - An Eigenvalue Problem

Introduction

When a mechanical system is constructed special interest is often given to the loads that give rise to periodic oscillations. The amplitude of these oscillations depend on the characteristics of the spring system as well as the size and frequency of the load. At some frequencies (called *resonance frequencies* or *eigenfrequencies*) the amplitude of the oscillations can be very large and therefore give birth to unwanted strains.

A famous example is the Tacoma bridge, called *"Galloping Gertie"* in the state of Washington. The bridge collapsed in 1940 after huge oscillations. Due to the driving force of the wind the bridge started to oscillate at an eigenfrequency. After what could have been a disaster (only a dog died) the governor promised *"We are going to build the exact same bridge, exactly as before."* An engineer, von Karman, replied in a telegram, *"If you build the exact same bridge as before, it will fall into the exact same river as before."*

The same resonance phenomena has to be avoided in the spring system of a car and in the driving axis of an engine. In other systems, like an organ pipe or a radio receiver, however, resonance is the key to success.

Thus, it is of vital importance to know the eigenfrequencies of a system. In this assignment we will investigate a simple mechanical system by finding its eigenvalues.

The System

Consider now a string of unit length with one end fixed and the other one free. We want to determine at what frequencies the string will vibrate if it is disturbed from equilibrium. This can be done by solving the one-dimensional wave equation. When using the technique of separation of variables, we will get the following problem for the time-independent part of the solution.

$$\begin{cases} -u'' = \lambda u \\ u(0) = 0 \\ u'(1) = 0 \end{cases} \tag{1}$$

We are interested in solutions other than the trivial, $u \equiv 0$, and their corresponding eigenfrequencies. Finding these is an easy problem which can be solved analytically. Here, however, we will use it to demonstrate how The Finite Element Method (FEM) can be used to transform equation (1) into an algebraic eigenvalue problem.

When using FEM the approximate solution , v, is written as a linear combination of basis functions, $v = \sum_{j=1}^{n} c_j v_j$. In the problem described above, the interval $[0, 1]$ is divided into subintervals with length $h = 1/n$. We take the n basis functions to be 'hat functions', defined as

$$v_j(x) = \begin{cases} (x - x_{j-1})/h & x_{j-1} < x \leq x_j \\ (x_{j+1} - x)/h & x_j < x < x_{j+1} \\ 0 & otherwise \end{cases}$$

for $j = 1, \ldots, n$, where $x_j = jh$. With this choice of basis functions the c_j will be an approximation to the function u at $x = x_j$.

If we now derive the *weak formulation* of (1) and use the basis functions above, we will get, when evaluating the resulting integrals, the *generalized algebraic eigenvalue equation*,

$$Kc = \lambda Mc \tag{2}$$

where the matrices K and M and the vector c are defined as

$$K = \frac{1}{h} \begin{bmatrix} 2 & -1 & & 0 & \\ -1 & 2 & -1 & & \\ & -1 & \cdot & \cdot & \\ & & \cdot & 2 & -1 \\ & 0 & & -1 & 1 \end{bmatrix}_{n \times n}$$

$$M = \frac{h}{6} \begin{bmatrix} 4 & 1 & & 0 & \\ 1 & 4 & 1 & & \\ & 1 & \cdot & \cdot & \\ & & \cdot & 4 & 1 \\ & 0 & & 1 & 2 \end{bmatrix}_{n \times n}$$

$$c = (c_1, ..., c_n)^T$$

The Power Method

When solving the generalized problem (2) we can in this case use a method for solving generalized eigenvalue problems as they stand. We can also rewrite the problem as an ordinary eigenvalue problem. In our example this is easily accomplished since M is non-singular, which follows from the fact that M is diagonally dominant. Thus, we can rewrite (2), using $A = M^{-1}K$, as

$$Ac = \lambda c. \tag{3}$$

The largest eigenvalue λ_1 of a matrix A can be computed by repeated applications of A to some vector z.

$$\lim_{k \to \infty} A^k z = \lambda_1^k c v_1.$$

The vector z is normalized in each step for computational reasons, and the power method algorithm can be written

$$\begin{array}{l} z_0 = \text{initial guess} \\ \text{while } |\mu_{k+1} - \mu_k| > \text{tolerance} \\ \quad \begin{array}{lcl} y_k & = & z_k/\|z_k\|_2 \\ z_{k+1} & = & Ay_k \\ \mu_k & = & y_k^H z_{k+1} \end{array} \end{array}$$

When convergence is reached we have $\lambda_1 \approx \mu_k$.

Moreover, if λ is an approximate eigenvalue of A, we can compute the corresponding eigenvector as:

$$(A - \lambda I) z_{k+1} = \frac{z_k}{||z_k||} \quad k = 0, 1, ...$$

where z_0 is an initial vector. For each iteration we have to solve a system of linear equations. The coefficient matrix $A - \lambda I$ can be LU-decomposed and we then only need to solve two triangular systems of equations in each iteration. This is called *Inverse Iteration*.

Tasks

Task 1

Write a MATLAB function (*eig_power(A)*) that takes as input a square matrix *A* and finds its largest eigenvalue and the corresponding eigenvector using the power method. Use your function to compute the largest eigenvalue of $A = M^{-1}K$ and its eigenvector. Iterate until $|\mu_k - \mu_{k-1}| < 10^{-4}$ and compare with the result from MATLAB's command *eig(A)*. Use an initial guess with all entries equal (but not equal to zero). Plot the eigenvector. How many iterations does it take for $n = 20$, $n = 40$, $n = 80$, etc. Can you say anything about the convergence rate?

Task 2

Compute the the smallest eigenvalue and the corresponding eigenvector using the Inverse iteration method. Compare with the result from MATLAB's command *eig(A)*. How many iterations does it take for $n = 20$, $n = 40$, $n = 80$, etc. Can you say anything about the convergence rate? How does the error in the eigenvalue behave with n?

Writing a report

Write a short informal report presenting your solutions (including answers, derivations, figures and code) to the tasks above.