Computer Analysis of Structures

ASEN 2001-Lab Section 11

Sage Herrin and Courtney Gilliam

10/2/2017

Theory Manual

Newton's first law states that "an object at rest will stay at rest unless acted upon by an unbalanced force" (The Physics Classroom, 2017). This principle is the foundation of equilibrium. Equilibrium is a state in which all forces are balanced, therefore producing no net force. Without a net force, the object is moving at a constant speed or remaining at rest. Static equilibrium refers to the specific situation of a system at rest. The system may contain and be impacted by many forces, but each force cancels out and yields a net force of zero. The same concept is applicable to moments. A moment demonstrates a force's ability to create turning or twisting around an axis in the system (Luebkeman and Peting, 1998). The net value of moments must be zero in order to maintain equilibrium. In a complex system, the calculations can become lengthy and problematic. A Matlab code has been created so that the process is less challenging and more efficient.

To begin an analysis of a structure or system in equilibrium, one must define the axes. For 3-D structures, there will be an x-axis, y-axis, and z-axis. Two of these axes must be perpendicular to each other, and the other is in the direction of their cross product. Placement of each axis is crucial for accurate force measurements. Once the axis has been chosen, one must interpret the direction of each force based on the axis location. Each of the three axes will have a separate equation that contains every force acting parallel to its direction. The total summation of the forces for each equation must be zero. Moments also have three equations that add up to zero. However, moments are calculated with the distance and the force in the proper direction. These six equations and an overall net force of zero prove that a system is in static equilibrium.

The code allows a user to do these calculations in a more precise manner within a shorter time span.

Although the code creates a more efficient process, it does not carry out calculations for systems that have more or less than six supports. The code supports any combination of moments and forces, but the total amount of them remains the same. The code requires that the text file is somewhat specific and resembles the format of the original given text file.

Developer/User Manual

The code is created to read an input file, complete calculations, and write them to an output file. For clarity, the first priority is to clear out the workspace of variables. Next, the file is opened. The input file must follow a relatively specific organizational pattern to be correctly read and interpreted by the program. It is split into categories of number of external forces and moments, the coordinates at which they are applied, the respective magnitudes and directions, location of supports of the structure, and type and direction of reaction forces and moments. Each category of information has its own matrix. Each matrix is pre-allocated as a zeros matrix so that the calculation is done faster. The text file is read into Matlab as a cell, and a for-loop then puts data from the text file into the matrices, row by row. The for-loops are broken when the last included row of data has been input. This break in the text file is delimited by a pound sign (#). However, in the script, the start and end points of the loop are determined by the amount of previous lines read. In order to figure out which line is the last line, an algebraic expression has been composed for each portion of the text document. The number of external forces and moments can vary, therefore lengthening or shortening the code. In order for this code to not be specific to one situation, the last line of data cannot be referred to with a specific amount of lines. An additional variable in the parameter of the loops accounts for the extra lines of words and descriptions between the sets of data. Only the numbers in the file are needed. For the last two matrices, an if-else loop is used to determine if the data on a row belongs in the moments matrix or forces matrix. It keeps count of the number of force and moment reactions by reassigning the ‘F’ or ‘M’ in the text file to a ‘1’ if it is a force and a ‘0’ if it is a moment. These values are later used in a logical statement. A cartesian matrix of zeros is created, and a for-loop inputs in the

values from the force magnitude matrix. Another cartesian matrix is constructed of zeros for the moments caused by external forces. These moments are calculated by crossing the matrix that contains locations of application for the external forces and the matrix of force magnitudes. This calculation illustrates the definition of a moment. Next, a couples moment matrix is made and the values from the text file are entered into it, depending on the number of external moments. It is crucial that after each matrix has been filled with the appropriate values, and it is normalized by dividing each cartesian vector by its respective magnitude to get unit vectors in the x, y, and z directions. These unit vectors are then multiplied by their respective magnitudes, giving the magnitude of that quantity in the x, y, or z direction. This ensures that cross products are taken properly and that correct results are calculated. It also allows us to sum the magnitudes of forces and moments together in the coordinate axes directions. To solve the matrices, the sum of all force and moments acting on the structure must add up to zero. The sums of the forces and moments in the coordinate axes directions taken from each moment and force reaction are placed into the six by six matrix, commonly referred to as the "A" matrix. This matrix is used to solve for the magnitudes of the reaction forces and moments in the typical "Ax = B" format. A for-loop formulates the unit vectors of the reaction forces and places them in a matrix. A new zeros matrix is created, and a conditional statement inside a loop places the reaction force and reaction moment coefficients into the pre allocated six by six matrix. This is used to solve for the unknown magnitudes of the force and moment reactions.

All of this is done by referring back to the ones and zeros that were assigned to external forces and moments, respectively. The conditional statement determines if the values being placed in the 'ith' row of the "A" matrix are going to be the coefficients of the external forces or

external moments. The for-loop iterates from one to the length of the pre allocated zeros matrix. The loop makes sure this iterated conditional statement happens the appropriate amount of times. In this case it is six. It is better to leave the for-loop in the form of '1:length(matrix)' opposed to explicitly going from one to six so that the application is kept as general as possible.

The development of the "B" matrix is slightly simpler than that of the "A" matrix. The "B" matrix in the "Ax = B" format is the matrix of known values of each equilibrium equation that are all put on one side. Each equilibrium equation is created simply by summing the x, y, and z components of the given external forces and couple moments. Since the "B" matrix is effectively the numerical values of the equilibrium equations being placed on the other side, it is multiplied by negative one to account for this. This format allows the 'x' matrix of unknown reaction magnitudes to be solved for with correct signs.

In the process of using matrices to represent and simultaneously solve a linear system of equations, the inverse of the "A" matrix of the "Ax=B' format is solved for. This in turn solves for the 'x' matrix of values, which is what we are looking for. Seeing as how taking the inverse of a six by six matrix is hardly trivial, we use Matlab to do it for us. Since the typical "Ax = B" format is being used in this lab, taking the inverse of the "A" matrix and multiplying it by the negative of the "B" matrix will give us our 'x' matrix of values needed. In our script, the "B" matrix of values is represented by a v_1, and the "A" matrix is represented by the variable matrix.

As specified, once the "x" matrix of variables is written to a text file using the 'dlmwrite' function. The most logical delimiter in this case is a new line delimited file, represented as \n in our script.

References

Luebkeman, C. and Peting, D. (1998). *What is a Moment?*. [online] Web.mit.edu. Available at: http://web.mit.edu/4.441/1_lectures/1_lecture5/1_lecture5.html [Accessed 26 Sep. 2017].

The Physics Classroom. (2017). *Newton's First Law*. [online] Available at: http://www.physicsclassroom.com/class/newtlaws/Lesson-1/Newton-s-First-Law [Accessed 28 Sep. 2017].

University of Colorado Boulder (2017). *Harmann Ax=b*. [video] Available at: https://learn.colorado.edu/d2l/le/content/214498/viewContent/3210657/View?ou=214498 [Accessed 26 Sep. 2017].