1. INTRODUCTION

In the recent studies of rigid body dynamics various programming languages and software are been used. MathCad is equation solver and graphing software used by scientists for solving various scientific problems [1,2]. This article presents a concise example of rigid body dynamics problems solved in MathCad and presents simple optimization problem.

D'Alembert's principle is used to create artificial equilibrium state by balancing active forces with inertial force which equals to \(-m \cdot a\). This principle enables solving dynamics problems similarly to statics and is well known in studies of vector and engineering mechanics [3,4].

Furthermore when acceleration equals 0 dynamic problem becomes static for \(-m \cdot a = 0\) and D'Alembert's force becomes 0. Analytical solution for this case is easily obtained and it is used to validate optimized dynamic solution.

2. DYNAMIC EXAMPLE PROBLEM

Problem solved for this presentation is modified textbook example [4].

2.1. Sample problem

Due to rotation frictional moment \(M_\theta = 2.5\ \text{Nm}\) appears at pivot O (see Figure 1.). Drum mass is \(m_1 = 6\ \text{kg}\) and radius of gyration \(i_1 = 225\ \text{mm}\) while weights masses are \(m_2 = 12\ \text{kg}\) and \(m_3 = 7\ \text{kg}\), drum radii are \(r_1 = 200\ \text{mm}\) and \(r_2 = 300\ \text{mm}\).

1. Determine angular acceleration of the grooved drum.
2. Determine maximum frictional moment when weights are still moving.
To obtain solution following will be done:
1) Free body diagram
2) Dynamic equilibrium equations
3) Static equilibrium equations
4) Solving equations using MathCad

2.2. Free body diagram

Identifying correct relations between the bodies is necessary to draw corresponding free body diagram. Free body diagram is shown in Figure 2.

2.3. Dynamic equilibrium equations

It is requested to determine angular acceleration and analytical solution will be obtained by the use of D’Alembert’s principle. Vector equations are following:

For weight A:
\[ \sum F_y = 0 \quad -m_1 \cdot g + m_1 \cdot a_1 + S_1 = 0 \]  
(1)

For weight B:
\[ \sum F_y = 0 \quad -m_2 \cdot g + m_2 \cdot a_2 + S_2 = 0 \]  
(2)

For drum:
\[ \sum M_O = 0 \quad S_1 \cdot r_1 - S_2 \cdot r_2 - I_O \cdot \alpha - M_o = 0 \]  
(3)

Kinematic relationships are:
\[ a_1 = \alpha \cdot r_1 \]  
(4)

\[ a_2 = \alpha \cdot r_2 \]  
(5)

2.4. Static equilibrium equations

Second problem statement requests the calculation maximum value of frictional moment \( M_o \). If frictional moment is large enough, thy system will become still and the problem becomes static. Therefore static equilibrium equations will be used to obtain solution.

For weight A:
\[ \sum F_y = 0 \quad -m_1 \cdot g + S_1 = 0 \]  
(6)

For weight B:
\[ \sum F_y = 0 \quad -m_2 \cdot g + S_2 = 0 \]  
(7)

For drum:
\[ \sum M_o = 0 \quad S_1 \cdot r_1 - S_2 \cdot r_2 - M_o = 0 \]  
(8)

3. MATHCAD SOLUTION

One way of solving system of linear equations in MathCad is using given-find block [5].

For successful calculation four steps should be made:
1) Define variables
2) Define initial values
3) Given-Find solve block
4) Store solutions

3.1. Mathcad solution of dynamic equilibrium

To enable proper calculation all constants must be defined [6, 7], local variable definitions are used for easier manipulation. Unlike global definitions, local definitions are valid from the place where are written to the right and below.

3.1.1. Defining variables

Green wavy line indicates that some previous definitions are overridden. MathCad recognizes dozens physical constants and units. If variable name is underlined then is recommended to rename it (Figure 3.).
3.1.2. Initial values

For all unknowns initial values and units should be defined (Figure 4.). Similar as defining constants local variable definition is used. For nonlinear systems it is more effective to define 1 rather than 0, in some cases 0 fails performance of Given-Find block.

\[
g := \frac{9.81}{s^2}, \quad M_O := 2.5N \cdot m
\]
\[
r_1 := 200mm, \quad r_2 := 300mm, \quad \dot{\theta} := 225mm
\]
\[
m_1 := 12kg, \quad m_2 := 7kg, \quad m_d := 6kg
\]
\[
I_O := m_d \cdot \frac{2}{3} = 0.304m^2 \cdot kg
\]

Figure 3. MathCad code defining variables

3.1.3. Solving system

Given-Find solve block is numerical solve block where the number of unknown variables must be same as number of equations in the block. Solution of given system is shown in Figure 5.

\[
S_1 := 0kN, \quad S_2 := 0kN
\]
\[
a_1 := \frac{0}{s^2}, \quad a_2 := \frac{0}{s^2}, \quad \alpha := \frac{0}{s^2}
\]

Figure 4. Defining initial values

3.1.4. Storing solution

Solve function Find is used to obtain solution. In this example vector is defined with same number of elements as given equations, containing variable names in same order as in Find function and thus solutions are stored. This is optional, but usually done for further calculation needs.

If solutions have various physical quantities, numerical values are separately recalled (Figure 6.).

\[
S_1 = 116.968N, \quad S_2 = 69.328N, \quad \alpha = \frac{0.3131}{s^2}
\]

Figure 6. Recalling solutions

3.2. Mathcad solution of static equilibrium

For static equilibrium solution similar procedure is done. Constants are same as in dynamic equilibrium and this step is unnecessary.

3.2.1. Initial values

Although mostly same constants are used, number of equations is changed and different initial values are defined using appropriate physical units, Mathcad syntax for this procedure is shown in Figure 7.

\[
S_1 := 0kN, \quad S_2 := 0kN, \quad M_O := 1N \cdot m
\]

Figure 7. Defining initial values for second solution

3.2.2. Solving system and storing solutions

In this step solve block Given-Find is used. Solutions are stored and numerical values are recalled separately because different physical units are used. Different units can be recalled inside same matrix only when symbolical evaluation is used (arrow).

\[
S_1 \cdot r_1 - S_2 \cdot r_2 - M_O = 0
\]
\[
-m_1 \cdot g + S_1 = 0
\]
\[
-m_2 \cdot g + S_2 = 0
\]

\[
\begin{pmatrix}
S_1 \\
S_2 \\
a_1 \\
a_2 \\
\alpha
\end{pmatrix} := \text{Find}(S_1, S_2, a_1, a_2, \alpha)
\]

\[
\begin{pmatrix}
S_1 \\
S_2 \\
M_O
\end{pmatrix} := \text{Find}(S_1, S_2, M_O) \rightarrow
\begin{pmatrix}
\frac{117.72kg \cdot m}{s^2} \\
\frac{68.67kg \cdot m}{s^2} \\
\frac{2943.0kg \cdot m \cdot mm}{s^2}
\end{pmatrix}
\]

\[
M_O = 2.943J
\]

Figure 8. Second solution block
4. OPTIMIZATION OF DYNAMIC SOLUTION

Another approach is possible to obtain solution for maximum frictional moment $M_0$. When frictional moment reaches high value weights decelerate, subsequent angular acceleration reaches minimum value. Solution of problem is finding frictional moment for minimum angular acceleration. Although many known optimization procedures are known [8, 9], for this purpose built in MathCad functions will be used.

4.1. Defining solution function

In this step function for obtaining range of solutions is defined. System of equations solved in chapter 3.2.2 is optimized using symbolical calculation in MathCad.

4.1.1. Defining variables

When running symbolic calculations use of units requires some additional calculation steps and in this calculation physical units are omitted for faster calculation (Figure 9.).

\[
\begin{align*}
\alpha & := 0.200 \\
\alpha & := 0.300 \\
\alpha & := 0.225 \\
m_1 & := 6 \\
m_2 & := 12 \\
m_3 & := 7 \\
I_0 & := m_D \cdot \frac{g}{1000} = 0.304 \\
g & := 9.81
\end{align*}
\]

Figure 9. Defined variables with omitted units

4.1.2. Defining initial values

For effective use of symbolic calculation initial values for unknown variables are defined to override any stored numerical value (Figure 10.).

\[
\begin{align*}
M_0 & := M_0 \\
a_1 & := a_1 \\
a_2 & := a_2 \\
\alpha & := \alpha \\
S_1 & := S_1 \\
S_2 & := S_2
\end{align*}
\]

Figure 10. Initial values for symbolic calculation

When full symbolic solution is wanted, all used physical quantities are to be defined this way, regardless if they are already defined as constants.

4.1.3. Defining solution function

Following is function formed of symbolic solution of dynamic equilibrium equations and function argument is frictional moment.

For obtaining solution solve function is used combined with simplify and float keywords for optimizing result. Solution of system is five elements function vector (Figure 11.).

\[
\begin{align*}
\alpha(M_0) := & \begin{pmatrix} S_1 \\ S_2 \\ a_1 \\ a_2 \\ \alpha \end{pmatrix} \\
solve, & \\
\rightarrow & \begin{pmatrix} 1.698 \cdot M_0 + 112.724 & -1.485 \cdot M_0 & 73.042 & -0.141 \cdot M_0 \\ 1.416 & -0.212 \cdot M_0 & 0.625 & -0.707 \cdot M_0 + 2.082 \end{pmatrix}
\end{align*}
\]

Figure 11. Defining whole solution function

4.2. Optimizing solution function

Angular acceleration as stated in solve function is fifth element. Default starting index (MathCad built in variable ORIGIN) is 0 so the function $\alpha(M)$ representing angular acceleration dependent on momentum is defined. This function is used to extract single solution value from solution vector (Figure 12.).

\[
\alpha(M) := \alpha(M_0, 4)
\]

Figure 12. Defining single solution function

4.2.1. Solution plot

Range variable is defined for plotting solution an linear plot of solution is made.

\[
\begin{align*}
mome & := 0, 0.1, 0.3 \\
\text{angular acceleration} & := \phi \\
\text{moment} & := M
\end{align*}
\]

Figure 13. Solution plot
Mathematical solution of problem shows linear (Figure 13.) dependence between moment and angular acceleration, thus solution plot is represented with line and minimum of such function in this case would be $-\infty$.

4.2.2. Minimizing function

For obtaining physically correct solution, minimum value of angular acceleration equals zero and maximum frictional moment equals root of function $\alpha(M)$. Root function in this form requires defining initial value for solution and syntax is as follows in Figure 14.

$$M_0 \Leftarrow 3, \quad M_0 = \text{root}(\alpha(M_0), M_0) = 2.943$$

Figure 14. Root function solution

5. CONCLUSION

Short theoretical problem is used to show one approach to solving optimization problems by use of symbolic calculations in MathCad engineering software. Physics laws demand that real conditions are considered while solutions are validated. Use of mathematical software shortens the process of solution verification and enables devising more theoretical problems and obtaining reliable solutions in shorter time even so without the use of graphical software.

6. REFERENCES


Kontakt autora:
Katarina Pisačić, dipl. ing.
Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3
42000 Varaždin
e-mail: katarina.pisacic@unin.hr

Marko Horvat, dipl. ing.
Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3
42000 Varaždin
e-mail: marko.horvat@unin.hr

Izv. prof. dr. sc. Vinko Višnjić.
Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3
42000 Varaždin
e-mail: vinko.visnjic@unin.hr