Vibrations - Lecture 11
2021-03-04

What is q ?

→ it's a time-dependent function, called the modal coordinate, which multiplies the eigenvector to give you the displacement

- q_1 is the time-dependent amplitude of $\vec{X}^{(1)}$

What is Q?

→ the modal force, it's what forces the ODE for q and it's related to the physical forcing on the masses by a linear xformation

... these are abstract analogues to x and F

... q_1 associated w/ ω_1 associated w/ $\vec{X}^{(1)}$

↑ comes from ODE w/ ω_1 in it

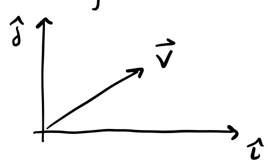
$$[X]^T[m][X]\ddot{\vec{q}} + [X]^T[k][X]\vec{q} = [X]^T\vec{F}$$

$$[X]^T[m][X] = [I] \quad \text{and} \quad [X]^T[k][X] = diag(\omega_1^2, \omega_2^2, \ldots, \omega_n^2)$$

$$\vec{Q}(t) = [X]^T\vec{F}(t)$$

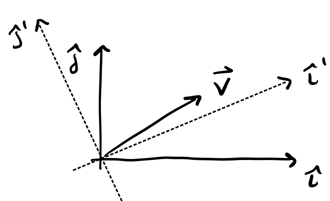
$$\ddot{q}_i(t) + \omega_i^2 q_i(t) = Q_i(t) \text{ for } i = 1, 2, \ldots, n$$

An analogy to think of q vs. x :



$$\vec{v} = v_x\hat{i} + v_y\hat{j}$$

now suppose someone else came along and said they wanted $\hat{i}'$ and $\hat{j}'$



$$\vec{y} = [A]\vec{x}$$

↑ transformation

but then what if have info of $\vec{y}$ in new coord. system? why need the q's

GIVEN:

$$\ddot{q}_i + \omega_i^2 q_i = Q_i \quad i = 1, 2, \ldots, n$$

$q_i(0)$ $\quad \dot{q}_i(0)$ $\quad Q_i(t)$

TO FIND:

$q_i(t)$ $\quad i = 1, 2, \ldots, n$

State Space

standard notation: $\dot{\vec{y}}_{(t)} = [A]\vec{y}_{(t)} + [B]\vec{u}_{(t)}$; $y(0) = y_0$ } 1^{st} order system

→ how do we go from 2^{nd} order system & scalar ($\ddot{q}_i + \dots$) to this 1^{st} order system & vector?

→ given $[A], [B], \vec{u}(t)$, and $\vec{y}(t)$, MATLAB finds $\vec{y}(t)$ by ODE45

Let's define $\vec{y}(t) = \begin{Bmatrix} \overbrace{q_i(t)}^{\text{modal displacement}} \\ \underbrace{\dot{q}_i(t)}_{\text{modal velocity}} \end{Bmatrix}$

then $\dot{\vec{y}}(t) = \begin{Bmatrix} \dot{q}_i(t) \\ \ddot{q}_{ii}(t) \end{Bmatrix}$

$$\underbrace{\begin{bmatrix} & \\ & \end{bmatrix}}_{S} \underbrace{\begin{Bmatrix} \dot{q}_i(t) \\ \ddot{q}_i(t) \end{Bmatrix}}_{\dot{\vec{y}}} - \underbrace{\begin{bmatrix} & \\ & \end{bmatrix}}_{R} \underbrace{\begin{Bmatrix} q_i(t) \\ \dot{q}_i(t) \end{Bmatrix}}_{\vec{y}} = \underbrace{\begin{Bmatrix} \\ \end{Bmatrix}}_{\vec{G}}$$

... nevermind ...

Let's go straight into state space form ...

$$\ddot{q}_i + \omega_i^2 q_i = Q_i$$

$$\underbrace{\begin{Bmatrix} \dot{q}_i \\ \ddot{q}_i \end{Bmatrix}}_{\dot{\vec{y}}} = \begin{bmatrix} 0 & 1 \\ -\omega_i^2 & 0 \end{bmatrix} \underbrace{\begin{Bmatrix} q_i \\ \dot{q}_i \end{Bmatrix}}_{\vec{y}} + \begin{Bmatrix} 0 \\ Q_i \end{Bmatrix}$$

→ $\dot{q}_i = \dot{q}_i$ ✓

→ the equation of motion (always this row)

$\ddot{q}_i = -\omega_i^2 q_i + Q_i$

So what is our state space form?

$$\dot{\vec{y}} = [A]\vec{y} + [B]\begin{Bmatrix} 0 \\ Q_i \end{Bmatrix}$$

$[B] = [I]$

$\vec{u}(t)$

↑ u gets tricky because has time-dependence and comes from the forcing

$[A] = \begin{bmatrix} 0 & 1 \\ -\omega_i^2 & 0 \end{bmatrix}$

The way ODE45 goes is:

you write a function $\dot{\vec{y}}$ = FUNCTION (t, $\vec{y}$, any other parameters)

"MATLAB, if you give me a time and a y vector, I'll give you back a $\dot{y}$ vector"

↳ it's a time stepping algorithm

when MATLAB gives me this y vector at a bunch of times, they are not equally spaced

↑ $\vec{y}(t)$

Let's sketch out the algorithm (in order that we do in codes):

Equation of motion: $[m]\ddot{\vec{x}} + [k]\vec{x} = \vec{F}(t)$

Given $[m]$ + $[k]$, x_0, $\dot{x}_0$, $F(t)$

$\downarrow$ solve eigenvalue problem

$([k] - \omega_i^2[m])\vec{X}^{(i)} = 0$

$\downarrow$ take eigenvector matrix to compute Q

$\vec{Q}(t) = [X]^T \vec{F}(t)$

$\downarrow$ solve each differential equation

$\ddot{q}_i + \omega_i^2 q_i = Q_i(t)$

we'll need to know $q_i(0)$ and $\dot{q}_i(0)$
via $\vec{x}(t) = [X]\vec{q}(t)$ and evaluate @ zero

$\vec{x}(0) = [X]\vec{q}(0)$

$\vec{q}(0) = [X]^T[m]\vec{x}(0)$

$\dot{\vec{q}}(0) = [X]^T[m]\dot{\vec{x}}(0)$

$\downarrow$ final step

$\vec{x}(t) = [X]\vec{q}(t)$ ... the x at any particular time

(*) mdof_undamped_forced(M, K, x_0, x_dot_0, force, t)

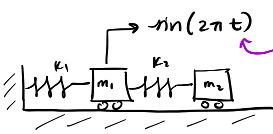
force ↑ the name of a MATLAB function as a character variable such that if I call that function and send in time as a scalar it returns the force vector on my physical system

t ↑ vector

When looping through modal coordinates, going thru the above derivation and algorithm

→ line 35 does the work!

- it completely does the time integration for the n^{th} modal coordinate using the state space formulation
- does numerical integration to find q but it does this under state-space terms
  ... so it's finding q and $\dot{q}$ which is our y-vector



period T = 1 sec, $\omega = 2\pi$

$f = 1$ Hz, $\omega = 2\pi f$

=> [v, D] = eig(k, M)
omega_n = sqrt(diag(D))
T_n = 2*pi./omega_n

$m_1 = m_2 = k_1 = k_2 = 1$