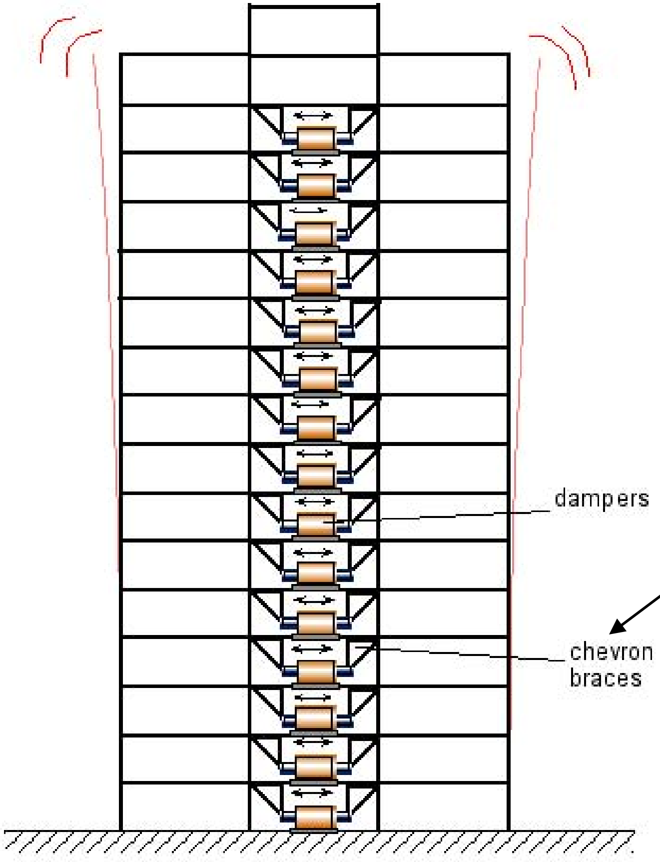
Today's goals

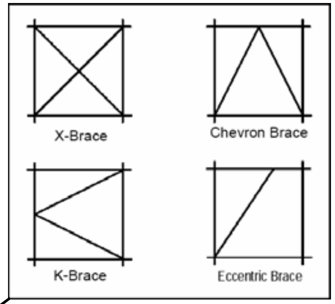
- **So far**
  - Feedback as a means for specifying the dynamic response of a system
  - Root Locus: from the open-loop poles/zeros to the closed-loop poles
  - "Moving the closed-loop poles around"
    - Proportional control: moving on the original Root Locus
    - Proportional-Derivative control: adding a zero/ speeding up the response/ maintaining constant overshoot
    - Proportional-Integral control: adding a free integrator (pole@origin) *and* a zero/ fixing steady-state error/ maintaining the speed and overshoot
- **Today**
  - The 2.004 Lab Tower plant
  - Impulse response and how it relates to the step response and the transfer function
  - State space: monitoring more than one dynamical variables at the same time

The need for sway compensation in buildings

Image from Wikimedia Commons, http://commons.wikimedia.org

Distributed active compensation system





www.atcouncil.org

Taipei 101; 101 floors; 448m tall (roof); 508m (spire)

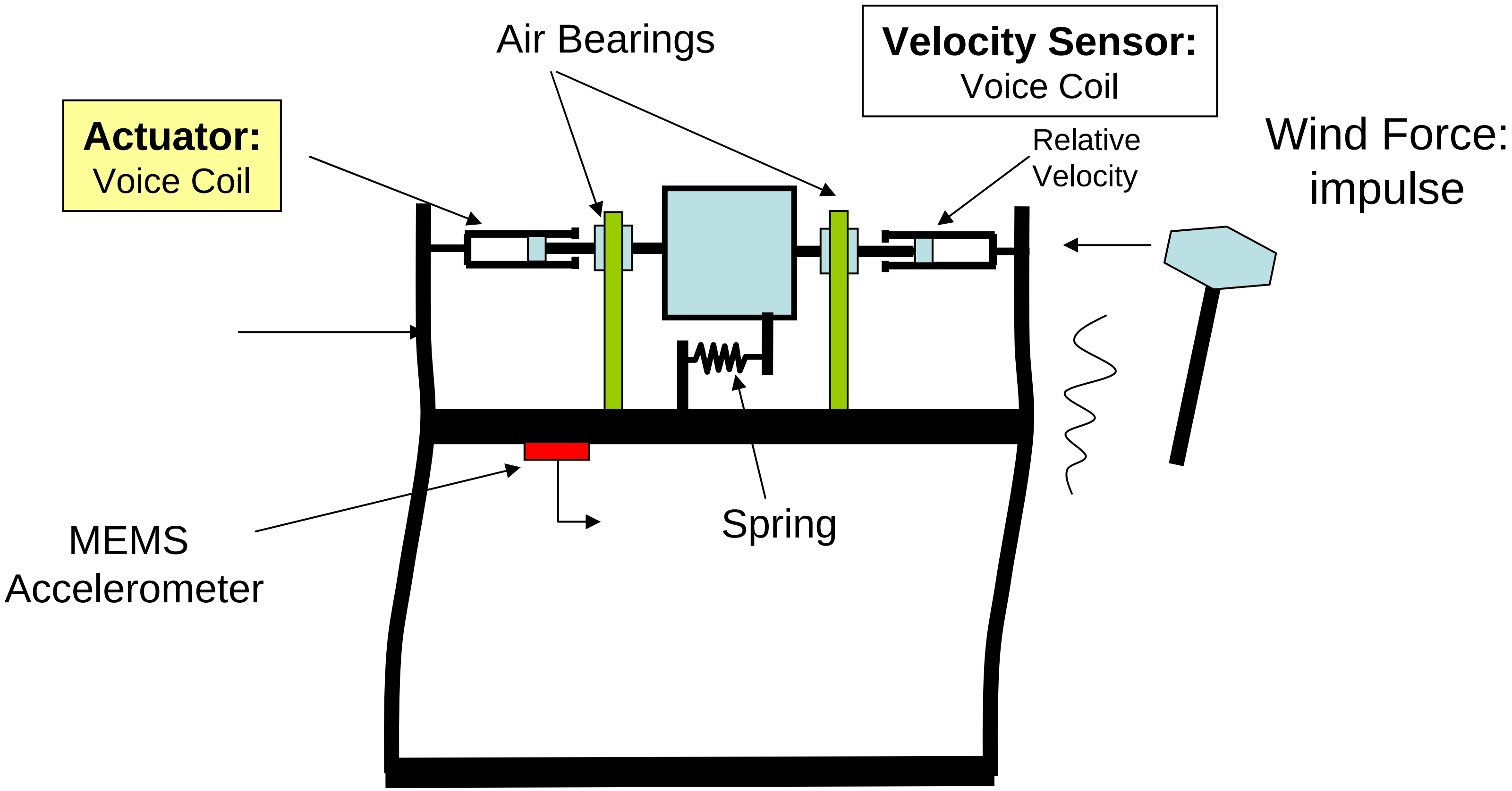
The 2.004 Tower

Goals:

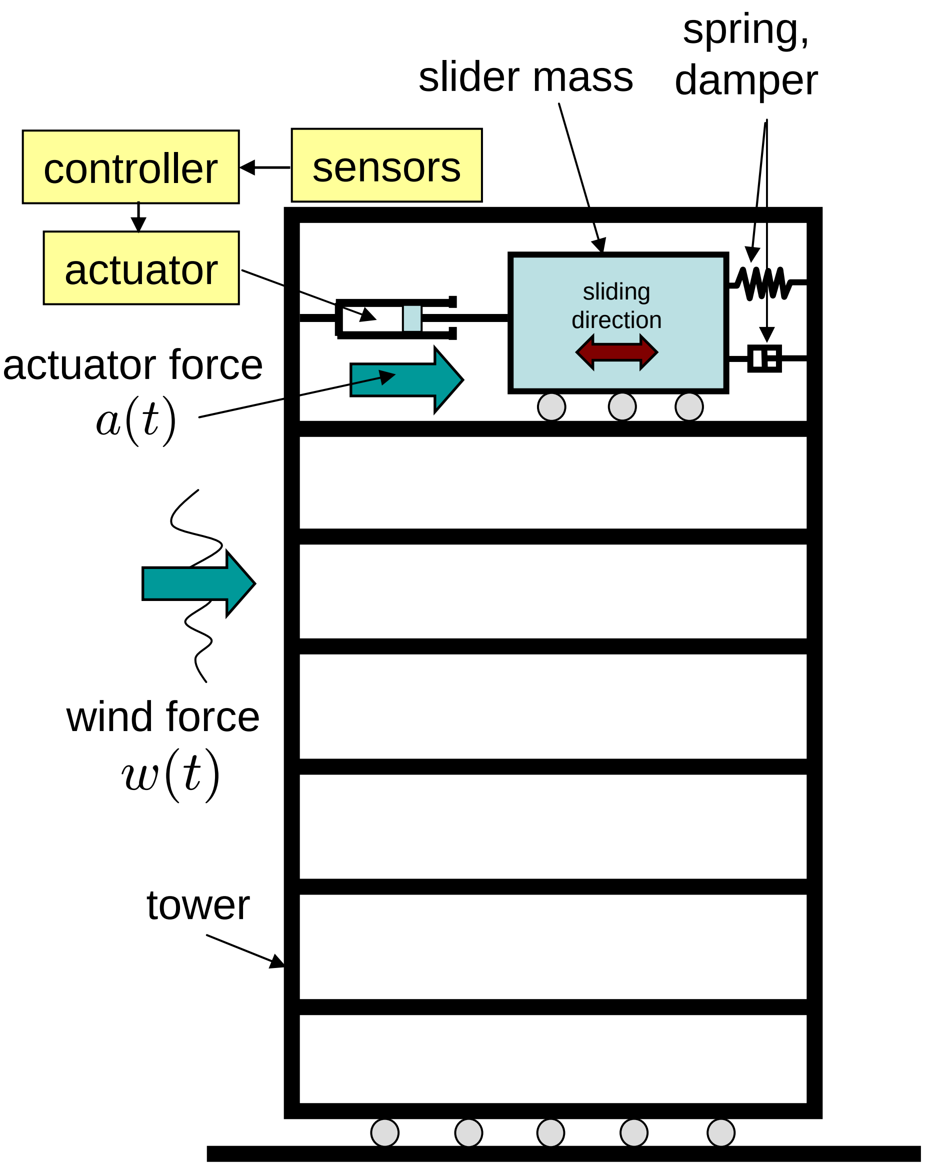
- Model
- Control

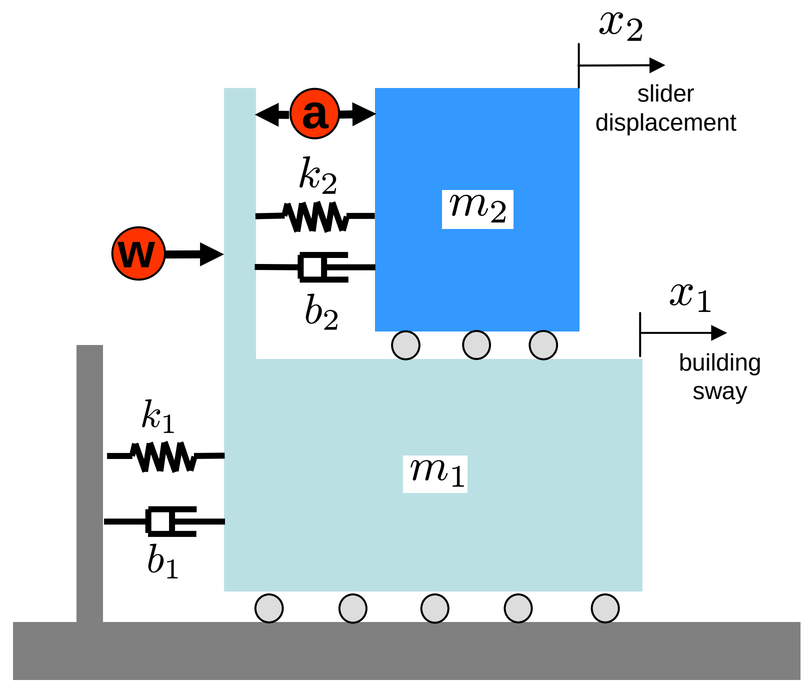
→

- Design
- Implement
- Test



Modeling the 2.004 Tower





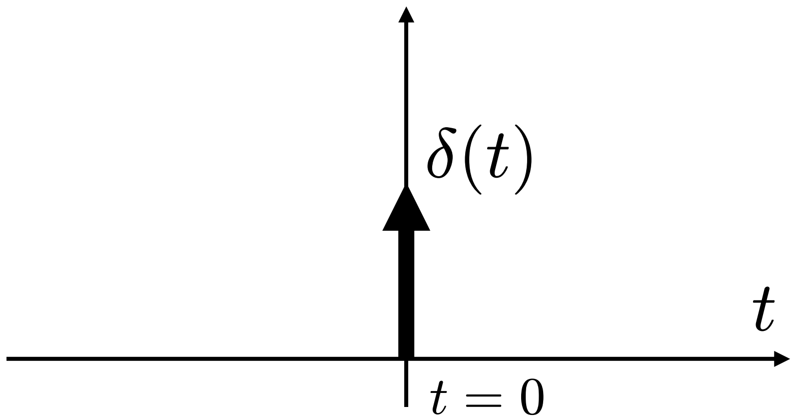
m_1	tower mass,
k_1	tower compliance,
b_1	tower damping (viscous),
m_2	slider mass,
k_2	spring on slider,
b_2	(viscous) damping on slider;
$w(t)$	wind force (impulse) on tower,
$a(t)$	actuator force on slider.

Reminder from Lecture 3: the delta (impulse) function

Impulse function (*aka* Dirac function)

$f(t)$	$F(s)$
$\delta(t)$	1
$u(t)$	$\frac{1}{s}$
$tu(t)$	$\frac{1}{s^2}$
$t^n u(t)$	$\frac{n!}{s^n + 1}$
$e^{-at}u(t)$	$\frac{1}{s+a}$
$\sin \omega t u(t)$	$\frac{\omega}{s^2+\omega^2}$
$\cos \omega t u(t)$	$\frac{s}{s^2+\omega^2}$

Figure by MIT OpenCourseWare.



It represents a pulse of

- <u>infinitessimally small</u> duration; and
- <u>finite</u> energy.

Mathematically, it is defined by the properties

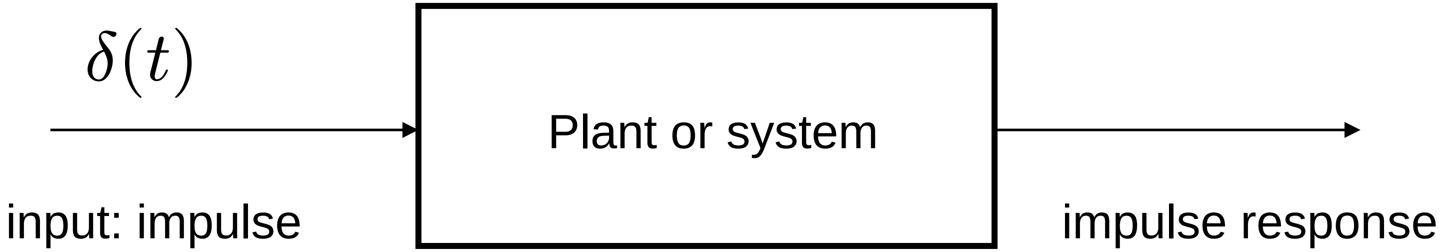
$$\int_{-\infty}^{+\infty} \delta(t) = 1; \qquad \text{(unit energy) and}$$

$$\int_{-\infty}^{+\infty} \delta(t) f(t) = f(0) \qquad \text{(sifting.)}$$

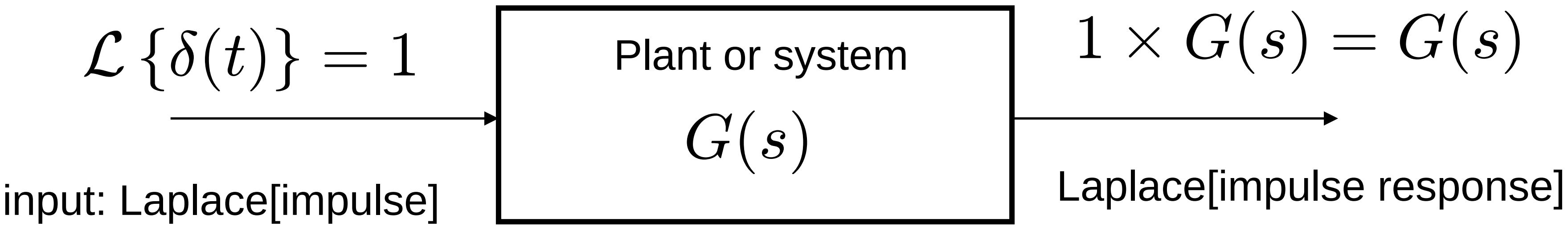
Nise Table 2.1

Impulse response

<u>Time domain</u>



<u>Laplace domain</u>



The Laplace transform of the impulse response
is the Laplace transform of the transfer function

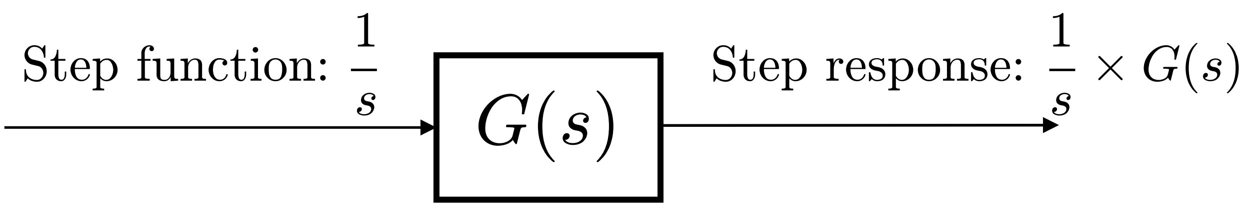
Impulse response and step response

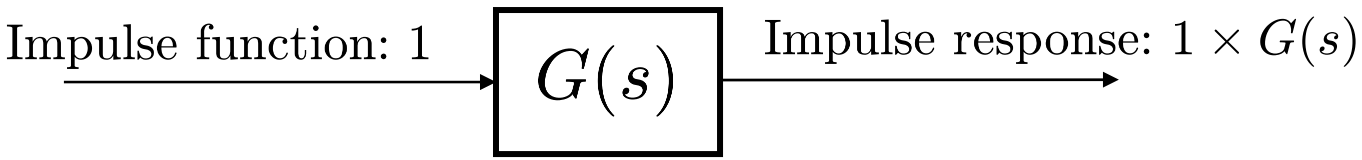
$f(t)$	$F(s)$
$\delta(t)$	1
$u(t)$	$\frac{1}{s}$

Figure by MIT OpenCourseWare.

$$\delta(t) = \frac{\mathrm{d}u(t)}{\mathrm{d}t}.$$

$$u(t) = \int_{0-}^{t} \delta(t')\mathrm{d}t'.$$

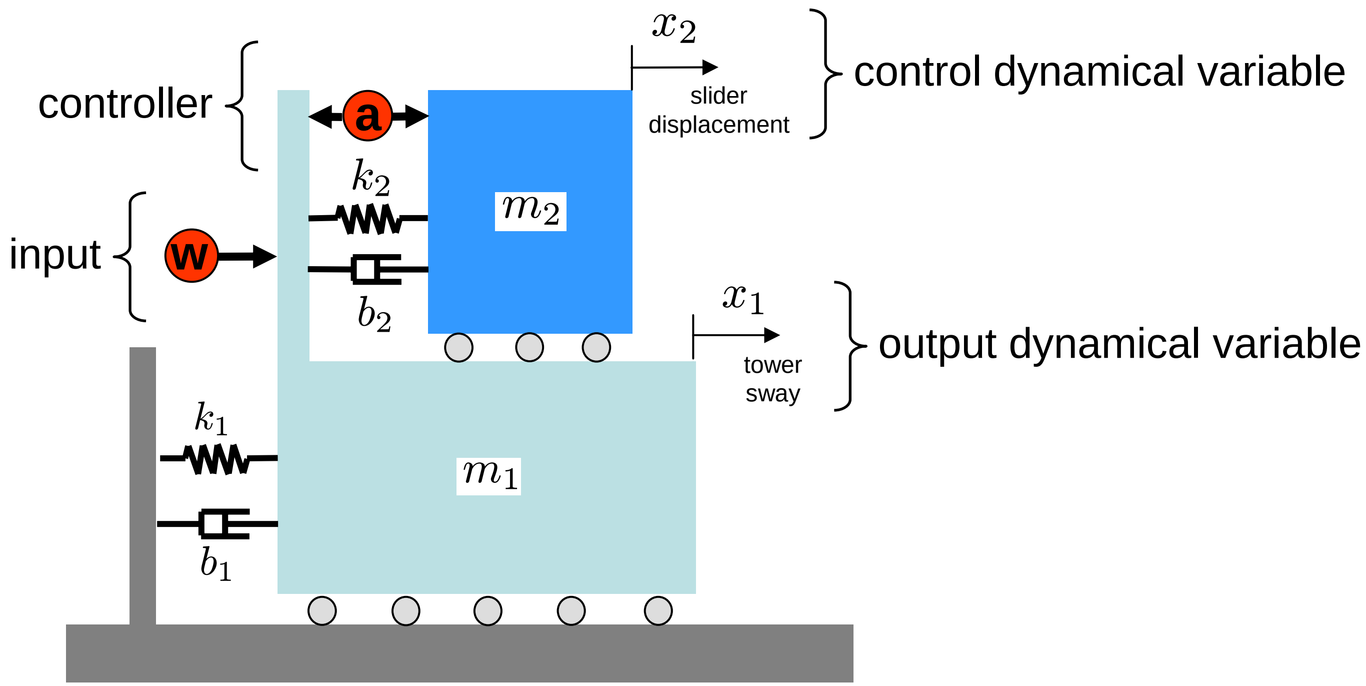




$$\text{Impulse response} = \frac{\mathrm{d}}{\mathrm{d}t}\,(\text{Step response})$$

$$\text{Step response} = \int_{0-}^{t} (\text{Impulse response})\,(t')\mathrm{d}t'.$$

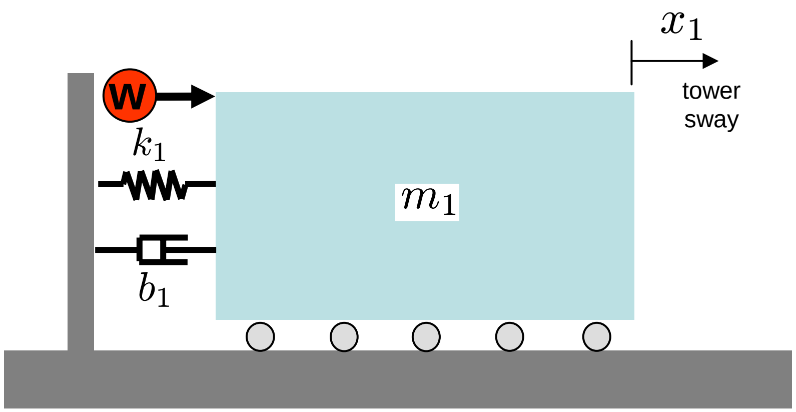
Controling the tower



The objective of control is to ***minimize the output*** (tower sway) subject to an ***impulse input*** (wind force)

The controller's actuation (force) is applied on an ***intermediate*** dynamical variable (slider displacement.) Moreover, there is a choice of feedback variables (e.g., tower displacement x_1, velocity $\dot{x}_1$, acceleration $\ddot{x}_1$)

Let’s start with a simpler system …



We begin by considering the tower by itself,
i.e., without the slider-spring-damper compensation system.

Our goal is to see how can access ***intermediate dynamical variables*** (such as the tower’s velocity) from ***a single dynamical model***

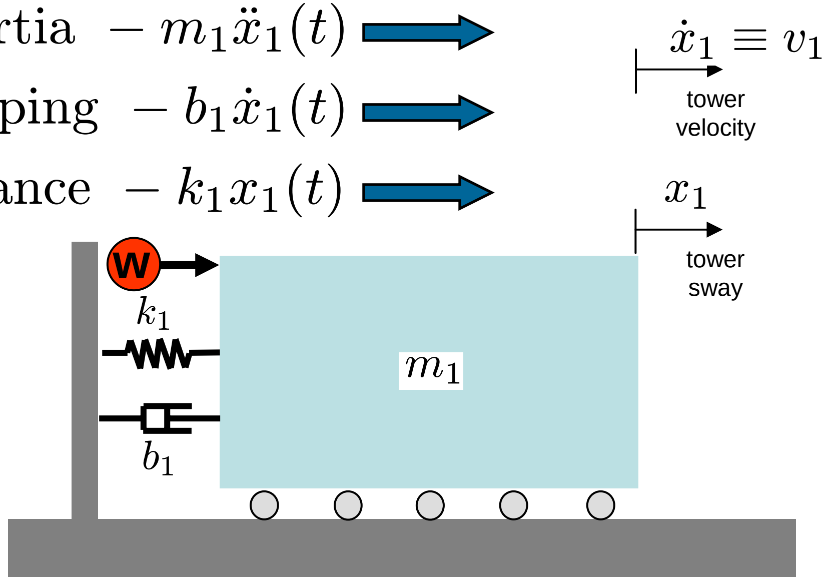
Force balance on the uncompensated tower

Wind force $w(t)$ →

Tower inertia $-m_1\ddot{x}_1(t)$ →

Tower damping $-b_1\dot{x}_1(t)$ →

Tower compliance $-k_1 x_1(t)$ →



$$\text{Force balance: } w(t) - m_1\ddot{x}_1(t) - b_1\dot{x}_1(t) - k_1 x_1(t) = 0 \Rightarrow$$

$$m_1\ddot{x}_1(t) + b_1\dot{x}_1(t) + k_1 x_1(t) = w(t)$$

(equation of motion)

From the equation of motion to the state-space representation

Recall: we would like to model two dynamical variables **simultaneously:** the tower position $x_1(t)$ and the tower velocity $\dot{x}_1(t) \equiv v_1(t)$. To see if we can achieve this goal, let us define a **state vector**

$$\mathbf{q}(t) = \begin{pmatrix} q_1(t) \\ q_2(t) \end{pmatrix} = \begin{pmatrix} x_1(t) \\ v_1(t) \end{pmatrix}.$$

The state vector components $q_1(t)$, $q_2(t)$ are called **state variables.** Now let's try to write a differential equation for this vector that is equivalent to the original system's equation of motion. That is, we need to compute the derivatives $\dot{q}_1(t)$, $\dot{q}_2(t)$ as function of $q_1(t)$, $q_2(t)$.

The differential equation that we target should be 1^{st}–order (*i.e.*, involving only the first derivatives of the state variables) and it should involve **linearly independent** variables. For example, if it turned out that $q_2(t) = (\text{constant}) \times q_1(t)$, then our attempt would not have worked. Fortunately, that is not the case with our choice of tower displacement and velocity as state variables. There are formal rules for selecting state variables while avoiding linear dependence between them; we are not yet ready to cover these rules in detail.

From the equation of motion to the state-space representation

Taking linear independence as given, we begin from the obvious place, the definition of velocity:

$$\dot{x}_1(t) = v_1(t) \Rightarrow \dot{q}_1(t) = q_2(t).$$

We can also re–write the equation of motion in terms of the state variables:

$$m_1\ddot{x}_1(t) + b_1\dot{x}_1(t) + k_1x_1(t) = w(t) \Rightarrow m_1\dot{q}_2(t) + b_1q_2(t) + k_1q_1(t) = w(t).$$

We can solve the above equation for $\dot{q}_2(t)$:

$$\dot{q}_2(t) = -\frac{k_1}{m_1}q_1(t) - \frac{b_1}{m_1}q_2(t) + \frac{1}{m_1}w(t).$$

We have reached our goal, and we can do even better by <u>combining</u> the **two** 1^{st}–order **scalar** differential equations <u>into</u> a **single** 1^{st}–order **vector** differential

equation: $$\begin{pmatrix} \dot{q}_1(t) \\ \dot{q}_2(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -k_1/m_1 & -b_1/m_1 \end{pmatrix}\begin{pmatrix} q_1(t) \\ q_2(t) \end{pmatrix} + \begin{pmatrix} 0 \\ 1/m_1 \end{pmatrix} w(t).$$

From the equation of motion to the state-space representation

The equation we have just derived is the **state equation of motion** which expresses the dynamics of our system in vector–matrix notation. More formally, it is written as

$$\dot{\mathbf{q}}(t) = \mathbf{A}\mathbf{q}(t) + \mathbf{b}w(t),$$

where the matrix $\mathbf{A}$ and vector $\mathbf{b}$ are

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -k_1/m_1 & -b_1/m_1 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} 0 \\ 1/m_1 \end{pmatrix},$$

and are called the **system matrix** and **input vector**, respectively. (The input vector is also referred to as "excitation vector.")

Note that in our uncompensated tower system, there is a **single** input, which in fact happens to be a disturbance that we intend to cancel in the compensated system. However, the state–space formulation allows us to handle multiple inputs as well, by replacing the scalar $w(t)$ by a vector of inputs and the input vector $\mathbf{b}$ by an input matrix $\mathbf{B}$. In this class, we will deal with single–input single–output systems only. [In the compensated tower, the actuation force $a(t)$ will be the input and $w(t)$ will be treated as a disturbance.]

From the equation of motion to the state-space representation

Another benefit of the state–space approach is that we need not be constrained to a single output. We can define the **system output** $y(t)$ as a scalar that might be either the tower's position or its velocity, as follows. Let

$$y(t) = \mathbf{c}\mathbf{q}(t), \quad \text{where } \mathbf{c} = \begin{pmatrix} c_1 & c_2 \end{pmatrix}.$$

We refer to **c** as the **output vector.** For example, by choosing

$$\mathbf{c} = \begin{pmatrix} 1 & 0 \end{pmatrix} \Rightarrow y(t) = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} q_1(t) \\ q_2(t) \end{pmatrix} = q_1(t) = x_1(t)$$

we have selected the tower's displacement $x_1(t)$ to be the output. Or, choosing

$$\mathbf{c} = \begin{pmatrix} 0 & 1 \end{pmatrix} \Rightarrow y(t) = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} q_1(t) \\ q_2(t) \end{pmatrix} = q_2(t) = v_1(t)$$

so now the output is the tower velocity $v_1(t)$. We may choose $y(t)$ to be **any linear combination** of the state variables, e.g. $\mathbf{c} = \begin{pmatrix} 0.1 & 0.9 \end{pmatrix}$. We might also opt for more than one variables, in which case $y(t)$ and **c** would become a vector and matrix, respectively.

From the equation of motion to the state-space representation

The combination of equations

$$\begin{aligned}\dot{\mathbf{q}}(t) &= \mathbf{A}\mathbf{q}(t) + \mathbf{b}w(t), \qquad \text{[dynamics–equation of motion]} \\ y(t) &= \mathbf{c}\mathbf{q}(t) \qquad \text{[output or observation equation]}\end{aligned}$$

are the **state equations** or **state–space representation** of our system. If you look this up in Nise's textbook or in the literature, you will probably see a slightly more general form, where the vectors **b** and **c** are replaced by matrices (this is to handle multiple–input multiple–output systems, as we've pointed out); and the output contains an additional term that is a linear combination of the inputs. These representations are used for full generality in more advanced contexts; for our tower compensation problem and the scope of material that we cover in this class, the reduced single–input single–output state–space representation given here will suffice.

In Problem Set 8, we will walk you through the derivation of a state–space representation for the compensated 2.004 tower (pages 4 and 8 of these notes) where $w(t)$ is treated as a disturbance and $a(t)$ is the input. While you develop the model in the lab, in the lectures we will learn how to add state–space to our existing arsenal of control techniques (root locus, P/PI/PD compensators, etc.)