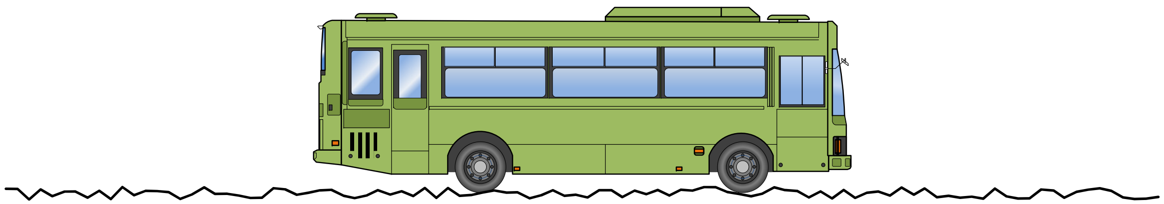
Vehicle-Bridge Interaction System

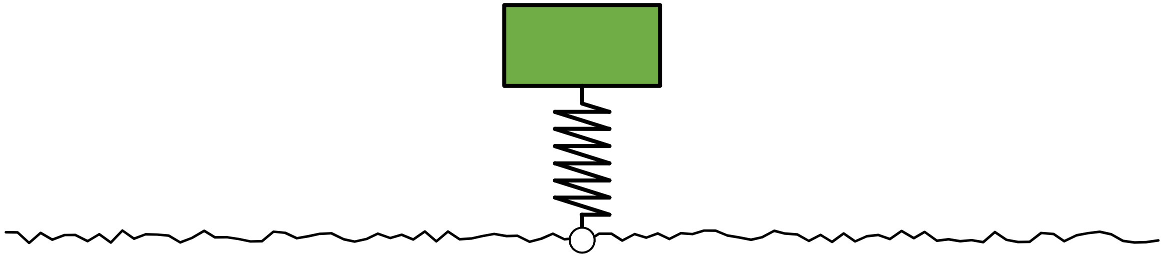
Introduction to IDE (1)

Yamamoto Kyosuke, Asst. Prof.

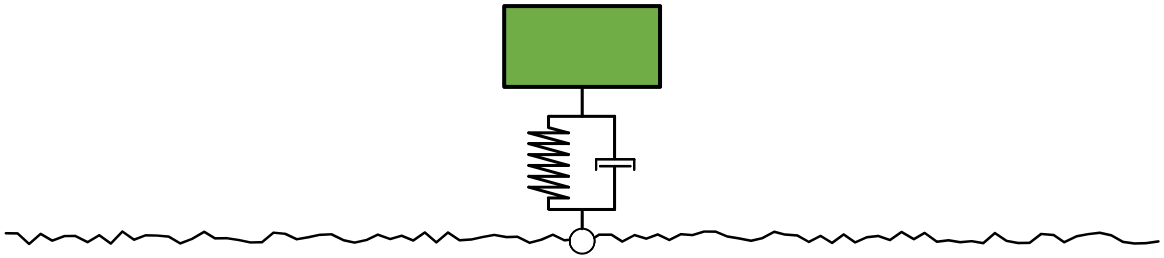
Road unevenness induces
vibrations on a travelling **vehicle**

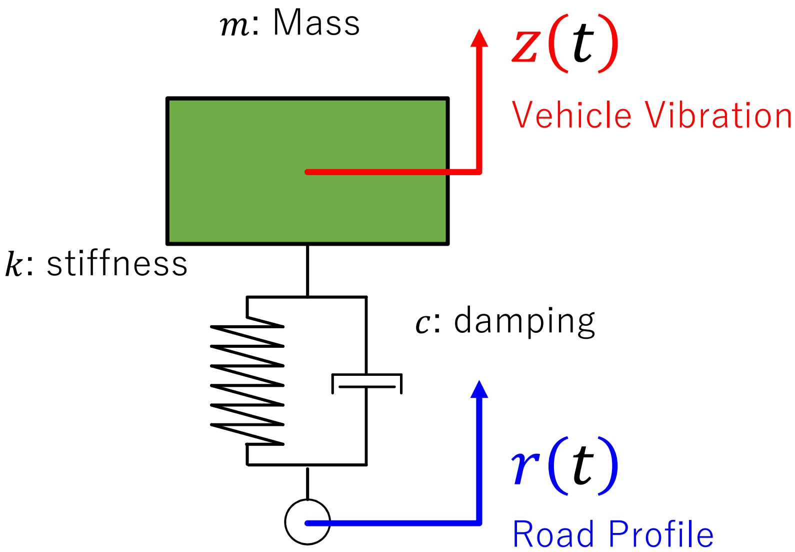


The simplest model of the travelling vehicle is
a mass-spring model



Suspensions are usually modeled by
spring and **damping**





Equation of motion of the vehicle: $(ma = F)$

$$m\ddot{z} = -c(\dot{z} - \dot{r}) - k(z - r)$$

damping force restoring force

$$(\dot{\ }) = \frac{d}{dt} \qquad (\ddot{\ }) = \frac{d^2}{dt^2}$$

$$m\ddot{z} + c\dot{z} + kz = c\dot{r} + kr$$

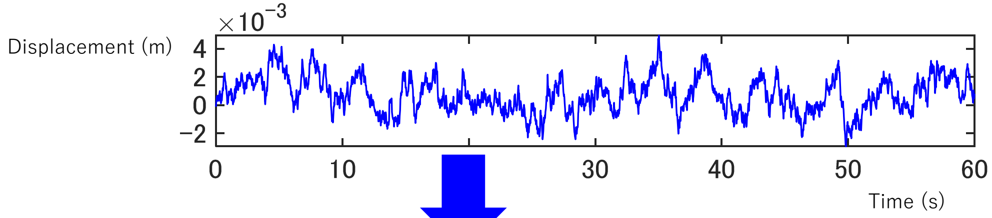
Linear Differential Equation

System Parameters: m, c, k

Output: $z(t)$

Input: $r(t)$

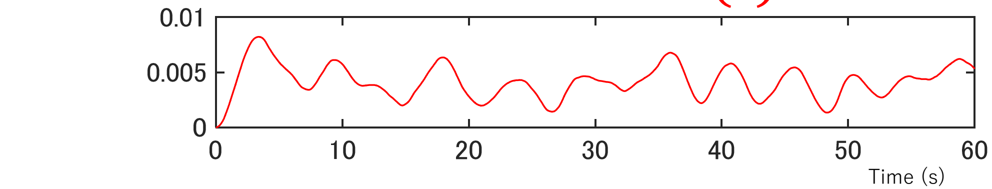
Road Profile: $r(t)$

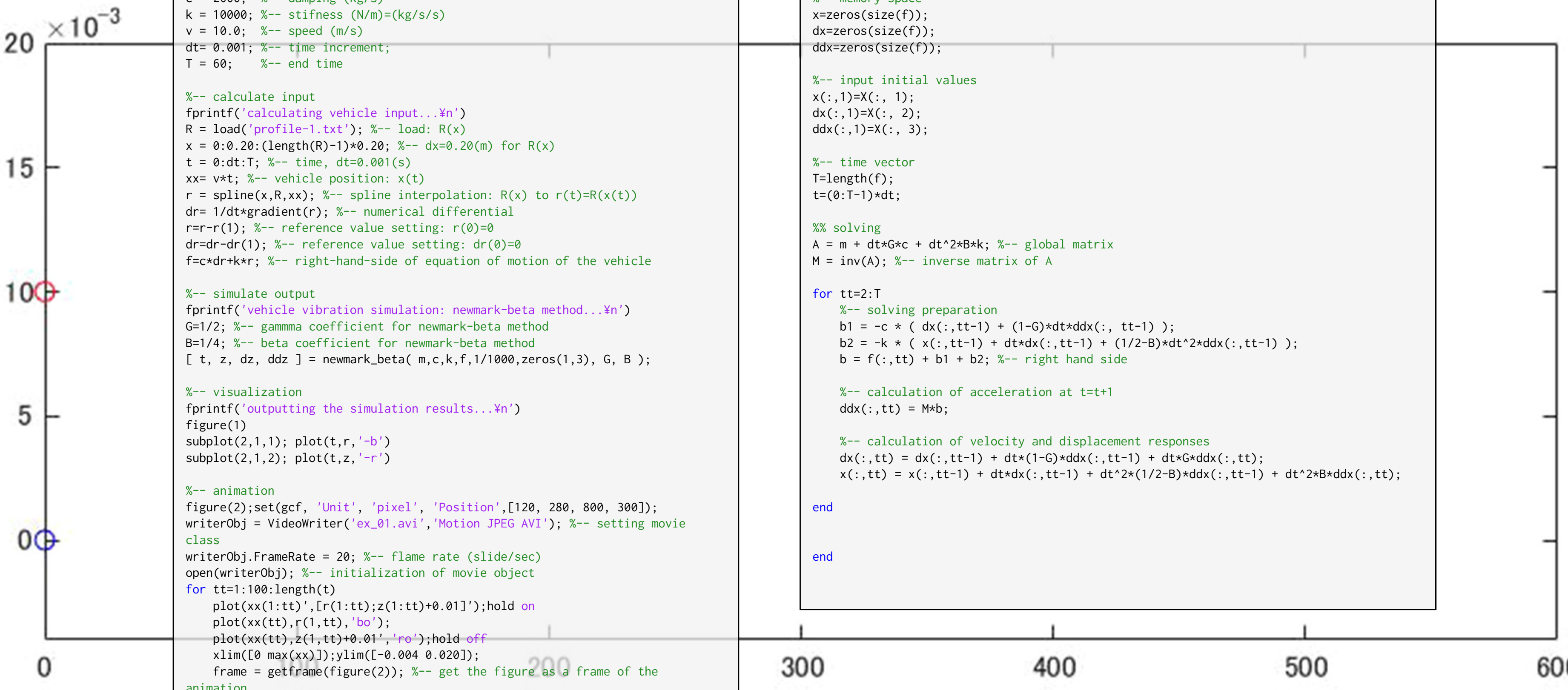


System Parameters:

Mass: m	**Damping**: c	**Stiffness**: k
10000 [kg]	2000 [kg/s]	10000 [N/m]

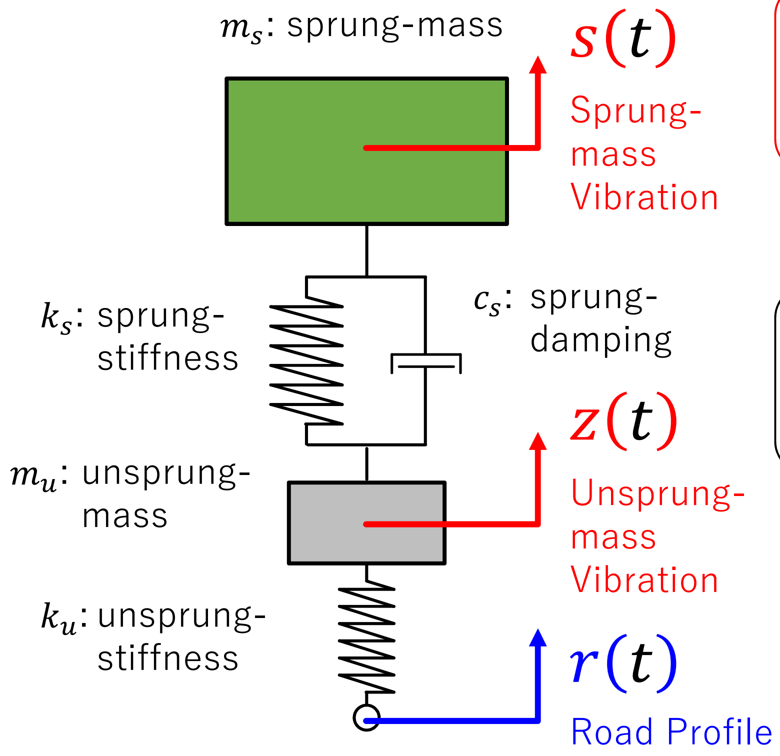
Vehicle Vibration: $z(t)$






```
%-- set vehicle parameters
fprintf('setting vehicle parameters...¥n')
m = 10000; %-- mass (kg)
c = 2000;  %-- damping (kg/s)
k = 10000; %-- stifness (N/m)=(kg/s/s)
v = 10.0;  %-- speed (m/s)
dt= 0.001; %-- time increment;
T = 60;    %-- end time

%-- calculate input
fprintf('calculating vehicle input...¥n')
R = load('profile-1.txt'); %-- load: R(x)
x = 0:0.20:(length(R)-1)*0.20; %-- dx=0.20(m) for R(x)
t = 0:dt:T; %-- time, dt=0.001(s)
xx= v*t; %-- vehicle position: x(t)
r = spline(x,R,xx); %-- spline interpolation: R(x) to r(t)=R(x(t))
dr= 1/dt*gradient(r); %-- numerical differential
r=r-r(1); %-- reference value setting: r(0)=0
dr=dr-dr(1); %-- reference value setting: dr(0)=0
f=c*dr+k*r; %-- right-hand-side of equation of motion of the vehicle

%-- simulate output
fprintf('vehicle vibration simulation: newmark-beta method...¥n')
G=1/2; %-- gammma coefficient for newmark-beta method
B=1/4; %-- beta coefficient for newmark-beta method
[ t, z, dz, ddz ] = newmark_beta( m,c,k,f,1/1000,zeros(1,3), G, B );

%-- visualization
fprintf('outputting the simulation results...¥n')
figure(1)
subplot(2,1,1); plot(t,r,'-b')
subplot(2,1,2); plot(t,z,'-r')

%-- animation
figure(2);set(gcf, 'Unit', 'pixel', 'Position',[120, 280, 800, 300]);
writerObj = VideoWriter('ex_01.avi','Motion JPEG AVI'); %-- setting movie
class
writerObj.FrameRate = 20; %-- flame rate (slide/sec)
open(writerObj); %-- initialization of movie object
for tt=1:100:length(t)
    plot(xx(1:tt)',[r(1:tt);z(1:tt)+0.01]');hold on
    plot(xx(tt),r(1,tt),'bo');
    plot(xx(tt),z(1,tt)+0.01','ro');hold off
    xlim([0 max(xx)]);ylim([-0.004 0.020]);
    frame = getframe(figure(2)); %-- get the figure as a frame of the
animation
    writeVideo(writerObj,frame); %-- add the gotten frame to the movie object
end
close(writerObj); %-- end of animation
```

```
function [ t, x, dx, ddx ] = newmark_beta( m,c,k,f,dt,X, G, B )

%% initialization
%-- memory space
x=zeros(size(f));
dx=zeros(size(f));
ddx=zeros(size(f));

%-- input initial values
x(:,1)=X(:, 1);
dx(:,1)=X(:, 2);
ddx(:,1)=X(:, 3);

%-- time vector
T=length(f);
t=(0:T-1)*dt;

%% solving
A = m + dt*G*c + dt^2*B*k; %-- global matrix
M = inv(A); %-- inverse matrix of A

for tt=2:T
    %-- solving preparation
    b1 = -c * ( dx(:,tt-1) + (1-G)*dt*ddx(:, tt-1) );
    b2 = -k * ( x(:,tt-1) + dt*dx(:,tt-1) + (1/2-B)*dt^2*ddx(:,tt-1) );
    b = f(:,tt) + b1 + b2; %-- right hand side

    %-- calculation of acceleration at t=t+1
    ddx(:,tt) = M*b;

    %-- calculation of velocity and displacement responses
    dx(:,tt) = dx(:,tt-1) + dt*(1-G)*ddx(:,tt-1) + dt*G*ddx(:,tt);
    x(:,tt) = x(:,tt-1) + dt*dx(:,tt-1) + dt^2*(1/2-B)*ddx(:,tt-1) + dt^2*B*ddx(:,tt);

end


end
```



Equation of motion of the vehicle: $(ma = F)$

$$m_s\ddot{s} = -c_s(\dot{s} - \dot{z}) - k_s(s - z)$$
$$m_u\ddot{z} = c_s(\dot{s} - \dot{z}) + k_s(s - z) - k_u(z - r)$$

$$m_s\ddot{s} + c_s\dot{s} + k_s s = c_s\dot{z} + k_s z$$
$$m_u\ddot{z} + c_s\dot{z} + (k_s + k_u)z = k_u r + c_s\dot{s} + k_s s$$

$$\begin{bmatrix} m_s & \\ & m_u \end{bmatrix}\begin{Bmatrix} \ddot{s} \\ \ddot{z} \end{Bmatrix} + \begin{bmatrix} c_s & -c_s \\ -c_s & c_s \end{bmatrix}\begin{Bmatrix} \dot{s} \\ \dot{z} \end{Bmatrix} + \begin{bmatrix} k_s & -k_s \\ -k_s & k_s + k_u \end{bmatrix}\begin{Bmatrix} s \\ z \end{Bmatrix} = \begin{Bmatrix} 0 \\ k_u r \end{Bmatrix}$$

$$\mathbf{M}\ddot{\mathbf{z}} + \mathbf{C}\dot{\mathbf{z}} + \mathbf{K}\mathbf{z} = \boldsymbol{f}$$

Matrix and **Vector**

System of Equations:

$$a_{11}x_1 + a_{12}x_2 = b_1$$
$$a_{21}x_1 + a_{22}x_2 = b_2$$

Using Matrix: $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

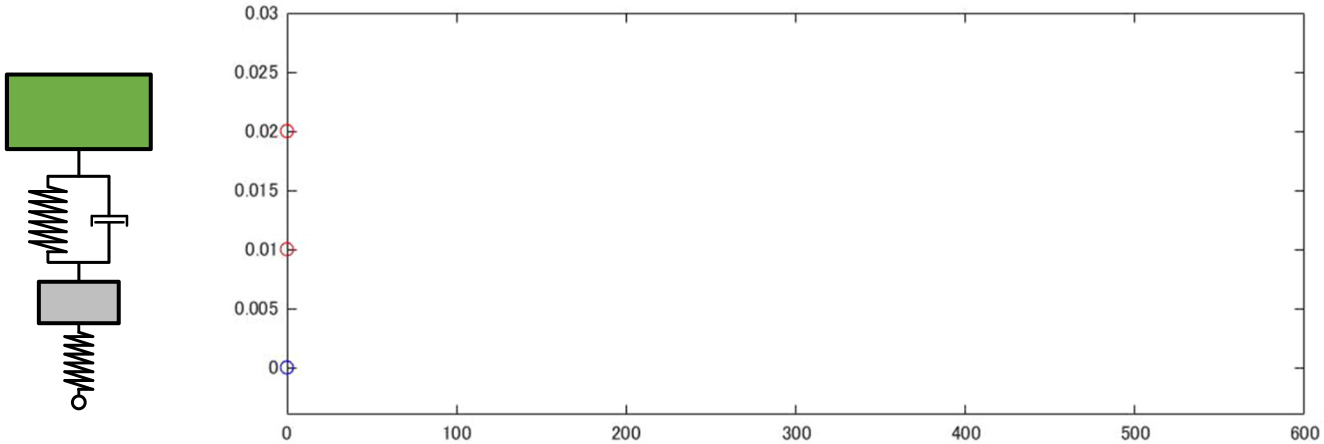
$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \end{Bmatrix}$$

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1$$
$$a_{21}x_1 + a_{22}x_2 + a_{13}x_3 = b_2$$
$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix}$$

Solution: $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$

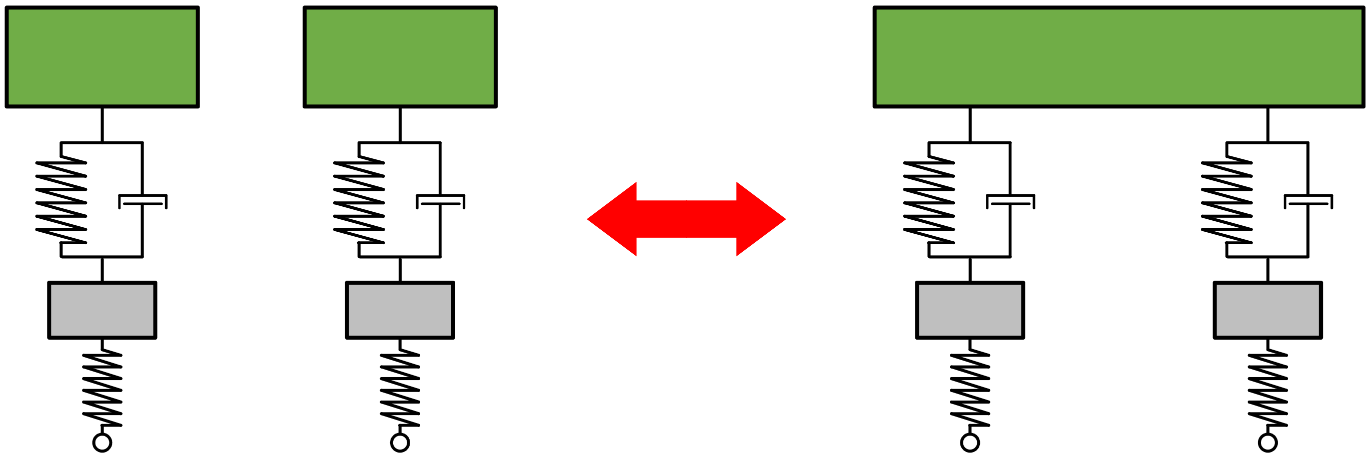
Inverse matrix

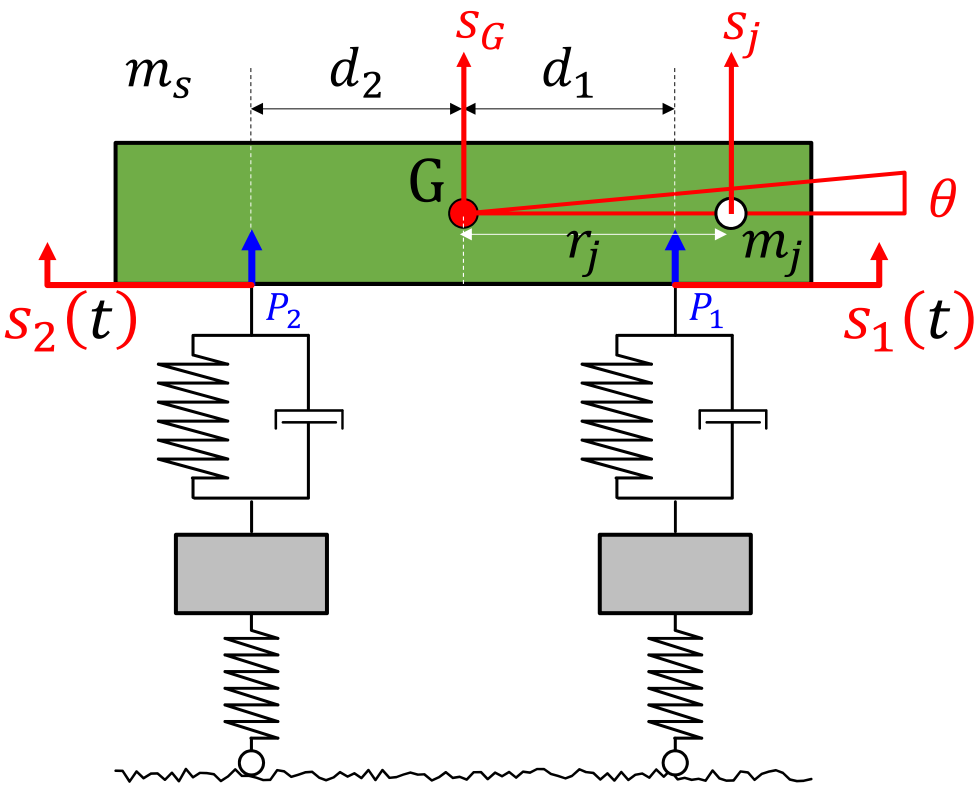


1) Download: http://www.kz.tsukuba.ac.jp/~yamamoto_k/material/matlab_for_ide.zip
2) Decompress the downloaded **zip** file
3) Launch **Matlab** and set the decompressed directory as **current** directory
4) Run the following script:

```
>> sample_02
```

Two quarter-cars can be easily extended to a **half-car model**





Definition:

$s_j = r_j\theta$

$\dot{s}_j = r_j\dot{\theta}$

$\ddot{s}_j = r_j\ddot{\theta}$

$$\sum_j m_j = m_s$$

$$\sum_j m_j s_j = \frac{d_2 m_s}{d_1 + d_2} s_1 + \frac{d_1 m_s}{d_1 + d_2} s_2 = m_s s_G$$

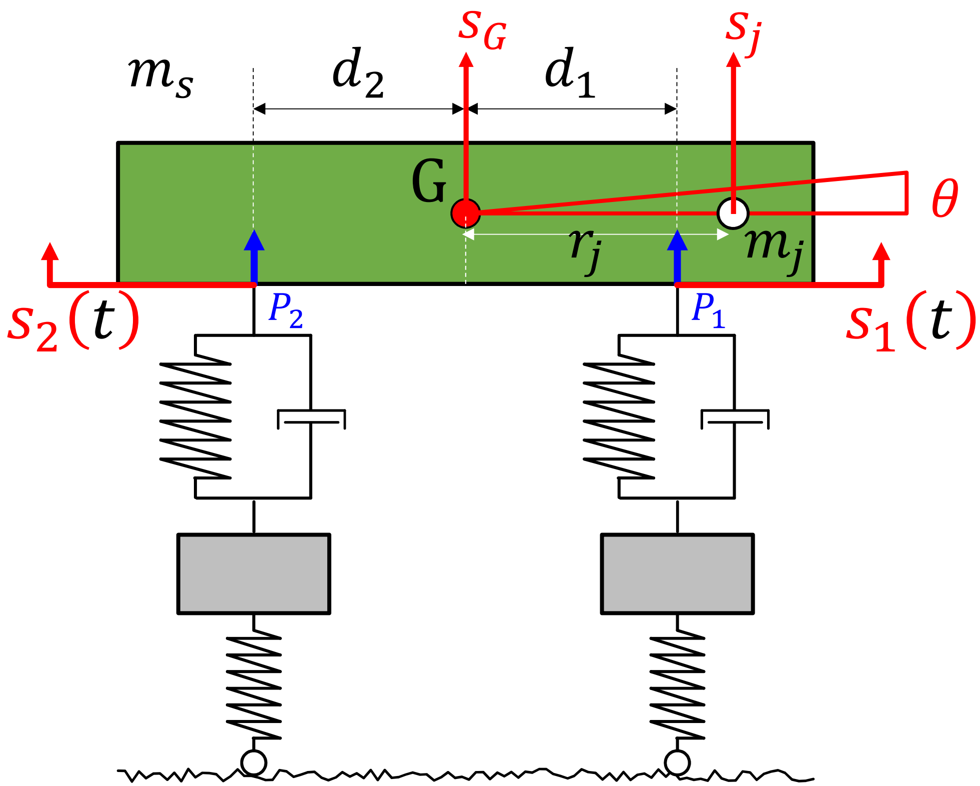
$$\theta = \frac{s_1 - s_2}{d_1 + d_2}$$

Equation of motion of mass point m_j

$$m_j \ddot{s}_j = f_j$$ internal force

Law of Action and Reaction

$$\begin{cases} \sum_j f_j = P_1 + P_2 \\ \sum_j r_j \times f_j = d_1 \times P_1 + d_2 \times P_2 \end{cases}$$

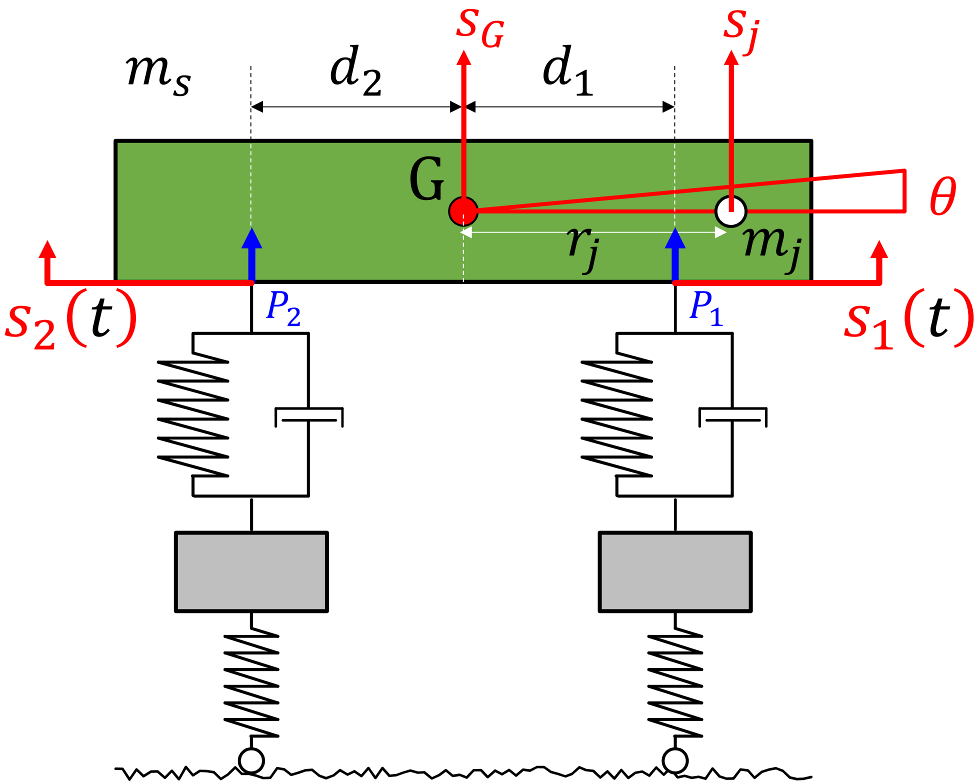


Summation of **Equation of motion**

$$\sum_j m_j \ddot{s}_j = \sum_j f_j$$

Equation of motion of translation: $ma = F$

$$m_s \ddot{s}_G = P_1 + P_2$$



Summation of **Equation of motion** × **arm length**

$$\sum_j m_j r_j \ddot{s}_j = \sum_j r_j f_j$$

Equation of motion of rotation: $I\ddot{\theta} = T$

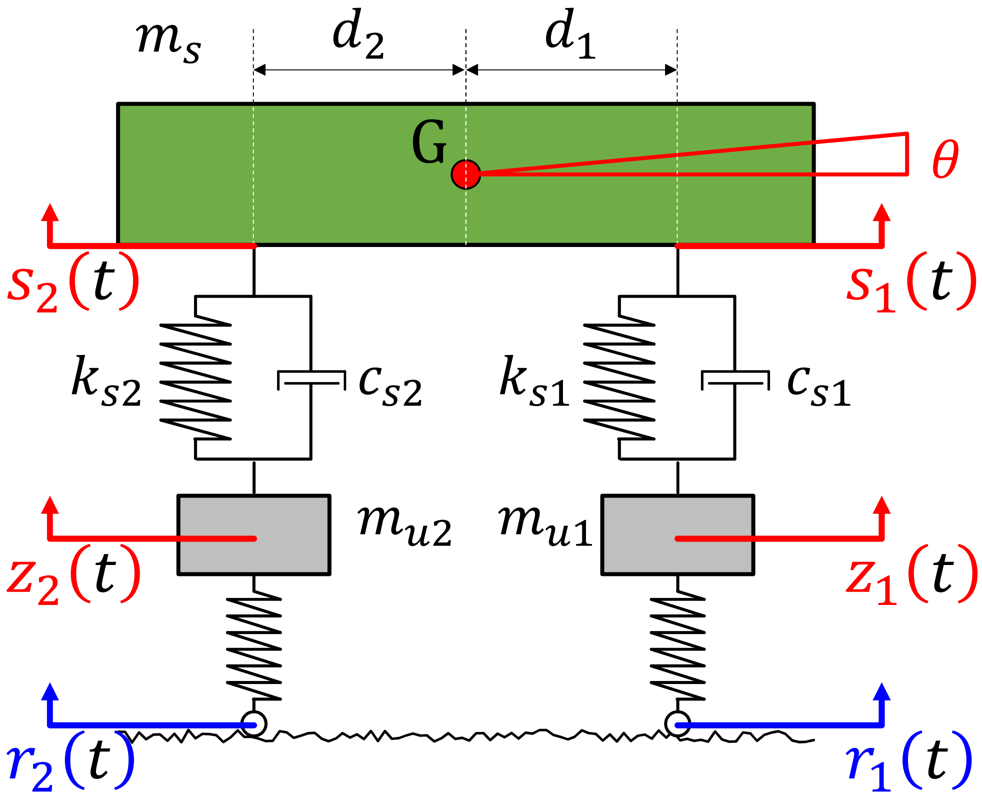
$$\left(\sum_j m_j r_j^2\right)\ddot{\theta} = d_1 P_1 + d_2 P_2$$

$$I\ddot{\theta} = T$$

Torque

Inertia Moment

Angular Acceleration



Equation of Motion of **Translation**

$$\frac{d_2 m_s}{d_1 + d_2}\ddot{s}_1 + \frac{d_1 m_s}{d_1 + d_2}\ddot{s}_2 = -c_{s1}(\dot{s}_1 - \dot{z}_1) - c_{s2}(\dot{s}_2 - \dot{z}_2) - k_{s1}(s_1 - z_1) - k_{s2}(s_2 - z_2)$$

Equation of Motion of **Rotation**

$$\frac{I}{d_1 + d_2}s_1 - \frac{I}{d_1 + d_2}s_2 = -d_1 c_{s1}(\dot{s}_1 - \dot{z}_1) + d_2 c_{s2}(\dot{s}_2 - \dot{z}_2) - d_1 k_{s1}(s_1 - z_1) + d_2 k_{s2}(s_2 - z_2)$$

$$m_{u1}\ddot{z}_1 = c_{s1}(\dot{s}_1 - \dot{z}_1) + k_{s1}(s_1 - z_1) - k_{u1}(z_1 - r_1)$$

$$m_{u2}\ddot{z}_2 = c_{s2}(\dot{s}_2 - \dot{z}_2) + k_{s2}(s_2 - z_2) - k_{u2}(z_2 - r_2)$$

$$\begin{bmatrix} \frac{d_2 m_s}{d_1+d_2} & \frac{d_1 m_s}{d_1+d_2} & & \\ \frac{I}{d_1+d_2} & \frac{-I}{d_1+d_2} & & \\ & & m_{u1} & \\ & & & m_{u2} \end{bmatrix} \begin{Bmatrix} \ddot{s}_1 \\ s_2 \\ z_1 \\ z_2 \end{Bmatrix} + \begin{bmatrix} c_{s1} & c_{s2} & -c_{s1} & -c_{s2} \\ d_1 c_{s1} & -d_2 c_{s2} & -d_1 c_{s1} & d_2 c_{s2} \\ -c_{s1} & & c_{s1} & \\ & -c_{s2} & & c_{s2} \end{bmatrix} \begin{Bmatrix} \dot{s}_1 \\ s_2 \\ z_1 \\ z_2 \end{Bmatrix} + \begin{bmatrix} k_{s1} & k_{s2} & -k_{s1} & -k_{s2} \\ d_1 k_{s1} & -d_2 k_{s2} & -d_1 k_{s1} & d_2 k_{s2} \\ -k_{s1} & & k_{s1}+k_{u1} & \\ & -k_{s2} & & k_{s2}+k_{u2} \end{bmatrix} \begin{Bmatrix} s_1 \\ s_2 \\ z_1 \\ z_2 \end{Bmatrix} = \begin{Bmatrix} \\ \\ k_{u1} r_1 \\ k_{u2} r_2 \end{Bmatrix}$$

$$\mathbf{M}\ddot{\mathbf{z}}(t) + \mathbf{C}\dot{\mathbf{z}}(t) + \mathbf{K}\mathbf{z}(t) = \boldsymbol{f}(t)$$

Newmark- β method

$$\mathbf{A}\ddot{\mathbf{z}}(t) = \boldsymbol{b}(t)$$

$$\ddot{\mathbf{z}}(t) = \mathbf{A}^{-1}\boldsymbol{b}(t)$$

Solving the system of equations

$$\mathbf{M}\ddot{\boldsymbol{z}}(t) + \mathbf{C}\dot{\boldsymbol{z}}(t) + \mathbf{K}\boldsymbol{z}(t) = \boldsymbol{f}(t)$$

Consider the next time step

$$\mathbf{M}\ddot{\boldsymbol{z}}(t+\Delta t) + \mathbf{C}\dot{\boldsymbol{z}}(t+\Delta t) + \mathbf{K}\boldsymbol{z}(t+\Delta t) = \boldsymbol{f}(t+\Delta t)$$

Newmark-β method

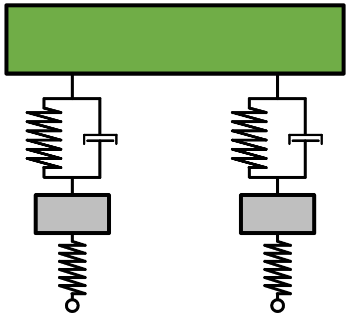
$$\dot{\boldsymbol{z}}(t+\Delta t) = \dot{\boldsymbol{z}}(t) + \Delta t(1-\gamma)\ddot{\boldsymbol{z}}(t) + \Delta t\gamma\ddot{\boldsymbol{z}}(t+\Delta t)$$

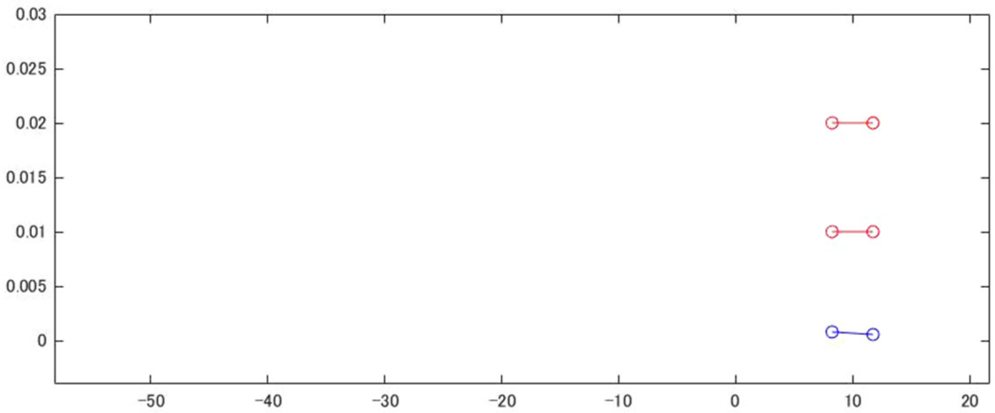
$$\boldsymbol{z}(t+\Delta t) = \boldsymbol{z}(t) + \Delta t\dot{\boldsymbol{z}}(t) + \Delta t^2\left(\frac{1}{2}-\beta\right)\ddot{\boldsymbol{z}}(t) + \Delta t^2\beta\ddot{\boldsymbol{z}}(t+\Delta t)$$

$$\left[\mathbf{M} + \Delta t\gamma\mathbf{C} + \Delta t^2\beta\mathbf{K}\right]\left\{\ddot{\boldsymbol{z}}(t+\Delta t)\right\} = \left\{\begin{matrix}\boldsymbol{f}(t+\Delta t) - \mathbf{C}\{\dot{\boldsymbol{z}}(t) + \Delta t(1-\gamma)\ddot{\boldsymbol{z}}(t)\} \\ -\mathbf{K}\left\{\boldsymbol{z}(t) + \Delta t\dot{\boldsymbol{z}}(t) + \Delta t^2\left(\frac{1}{2}-\beta\right)\ddot{\boldsymbol{z}}(t)\right\}\end{matrix}\right\}$$

$$\mathbf{A}\ddot{\boldsymbol{z}}(t+\Delta t) = \boldsymbol{b}$$

Once $\boldsymbol{z}(t)$, $\dot{\boldsymbol{z}}(t)$, $\ddot{\boldsymbol{z}}(t)$and $\boldsymbol{f}(t+\Delta t)$ are obtained, we can solve $\boldsymbol{z}(t+\Delta t)$, $\dot{\boldsymbol{z}}(t+\Delta t)$ and $\ddot{\boldsymbol{z}}(t+\Delta t)$.

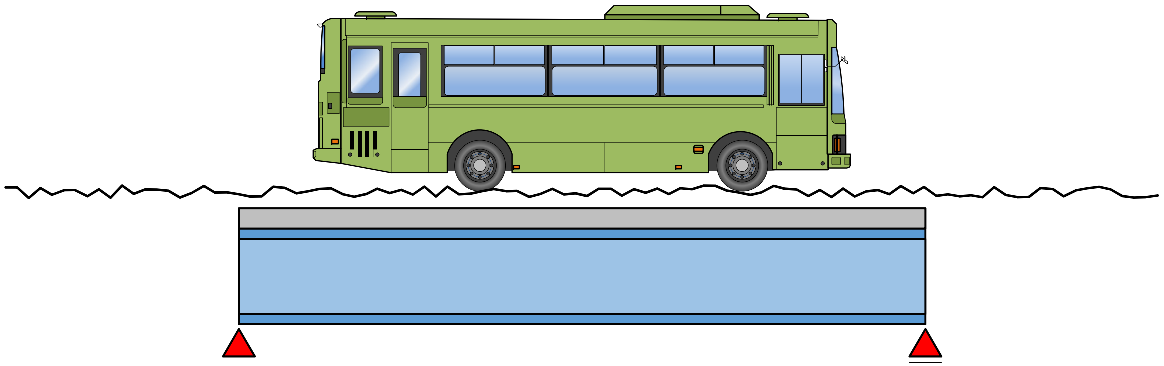




1) Download: http://www.kz.tsukuba.ac.jp/~yamamoto_k/material/matlab_for_ide.zip
2) Decompress the downloaded **zip** file
3) Launch **Matlab** and set the decompressed directory as **current** directory
4) Run the following script:

```
>> sample_03
```

Consider the effect of
Bridge Vibrations

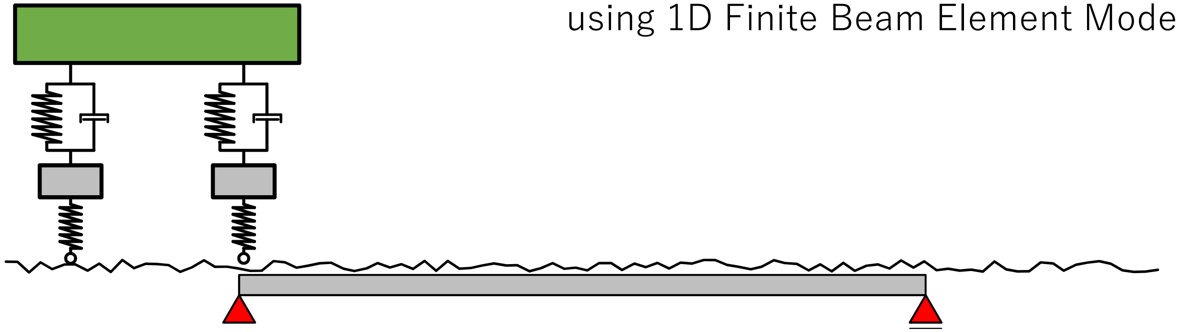


Vehicle and Bridge can be modeled by

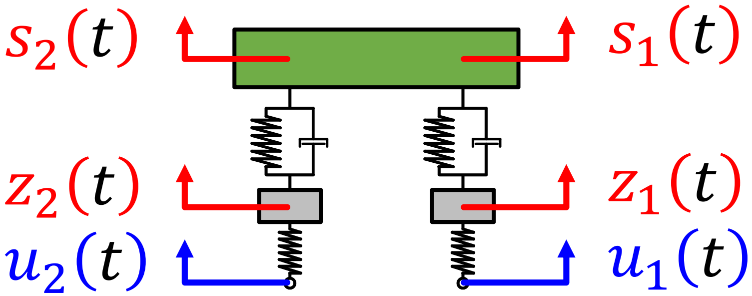
Half-car model and **Beam**

(**R**igid **B**ody **S**pring **M**odel)

Finite **E**lement **M**ethod
using 1D Finite Beam Element Model



Vehicle Inputs include **Bridge vibrations** as well as **Road unevenness**

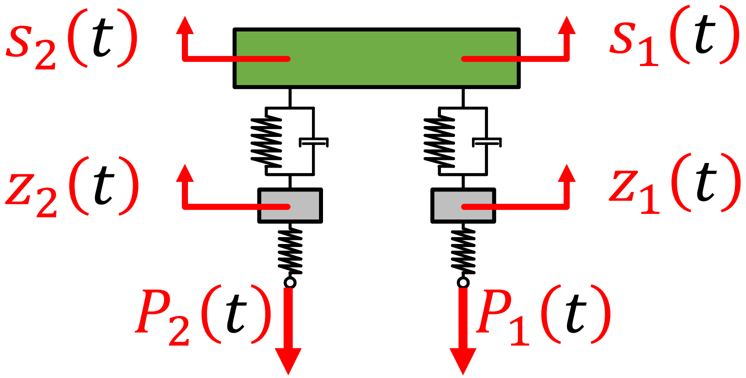


$$u_1(t) = r_1(t) + y_1(t)$$

$$u_2(t) = r_2(t) + y_2(t)$$

road unevenness　　bridge vibration

The **Vehicle Vibrations** affect on the **Contact Forces** acting on the Bridge



$$P_1(t) = \frac{d_2 m_s}{d_1 + d_2}\left(g - \ddot{s}_1(t)\right) + m_{u1}\left(g - \ddot{z}_1(t)\right)$$

$$= \frac{d_2 m_s}{d_1 + d_2} g + m_{u1} g + k_{u1}(z_1 - u_1)$$

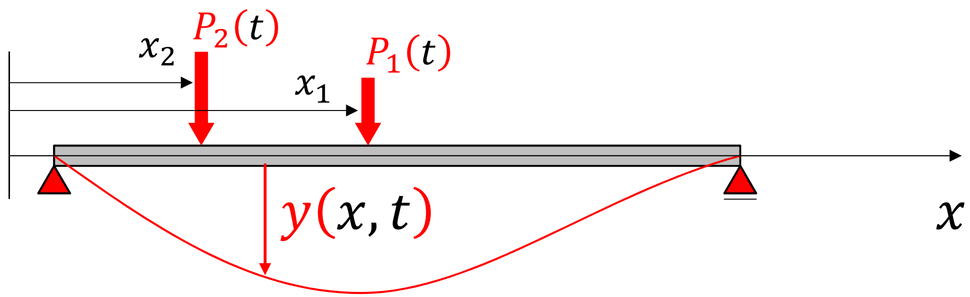
$$P_2(t) = \frac{d_1 m_s}{d_1 + d_2}\left(g - \ddot{s}_2(t)\right) + m_{u2}\left(g - \ddot{z}_2(t)\right)$$

$$= \underbrace{\frac{d_1 m_s}{d_1 + d_2} g + m_{u2} g}_{\text{weight}} + +k_{u2}(z_2 - u_2)$$

weight

restoring force
of unsprung-stiffness

Bridge Deflection Vibrations can be calculated by **Structure Mechanics**



$$\rho A \ddot{y}(x,t) + \frac{\partial^2}{\partial x^2} EI \left(\frac{\partial^2}{\partial x^2} y(x,t) \right) = \delta\big(x - x_1(t)\big) P_1(t) + \delta\big(x - x_2(t)\big) P_2(t)$$

Mass per unit length

Flexural Rigidity

Delta Function

$$\begin{cases} \delta(x) = 0 & \text{, when } x \neq 0 \\ \delta(x) = \infty & \text{, when } x = 0 \\ \int_{-\infty}^{+\infty} \delta(x)\, dx = 1 \end{cases}$$

The equation of motion of Vehicle System:

$$\frac{d_2 m_s}{d_1+d_2}\ddot{z}_{s1}(t)+\frac{d_1 m_s}{d_1+d_2}\ddot{z}_{s2}(t)=-k_{s1}\big(z_{s1}(t)-z_{u1}(t)\big)-k_{s2}\big(z_{s2}(t)-z_{u2}(t)\big)-c_{s2}\big(\dot{z}_{s2}(t)-\dot{z}_{u2}(t)\big)-c_{s2}\big(\dot{z}_{s2}(t)-\dot{z}_{u2}(t)\big)$$

$$\frac{I_s}{d_1+d_2}\ddot{z}_{s1}(t)-\frac{I_s}{d_1+d_2}\ddot{z}_{s2}(t)=-d_1\times k_{s1}\big(z_{s1}(t)-z_{u1}(t)\big)+d_2\times k_{s2}\big(z_{s2}(t)-z_{u2}(t)\big)-d_1\times c_{s2}\big(\dot{z}_{s2}(t)-\dot{z}_{u2}(t)\big)-d_2\times c_{s2}\big(\dot{z}_{s2}(t)-\dot{z}_{u2}(t)\big)$$

$$m_{u1}\,\ddot{z}_{u1}(t)=k_{s1}\big(z_{s1}(t)-z_{u1}(t)\big)+c_{s1}\big(\dot{z}_{s1}(t)-\dot{z}_{u1}(t)\big)-k_{u1}\big(z_{u1}(t)-u_1(t)\big)$$

$$m_{u2}\,\ddot{z}_{u2}(t)=k_{s2}\big(z_{s2}(t)-z_{u2}(t)\big)+c_{s2}\big(\dot{z}_{s2}(t)-\dot{z}_{u2}(t)\big)-k_{u2}\big(z_{u2}(t)-u_2(t)\big)$$

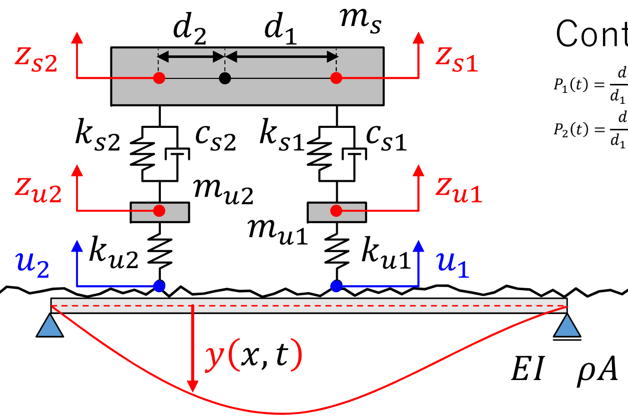
Contact Forces:

$$P_1(t)=\frac{d_2 m_s}{d_1+d_2}\big(g-\ddot{z}_{s1}(t)\big)+m_{u1}\big(g-\ddot{z}_{u1}(t)\big)$$

$$P_2(t)=\frac{d_1 m_s}{d_1+d_2}\big(g-\ddot{z}_{s2}(t)\big)+m_{u2}\big(g-\ddot{z}_{u2}(t)\big)$$

Input Profile:

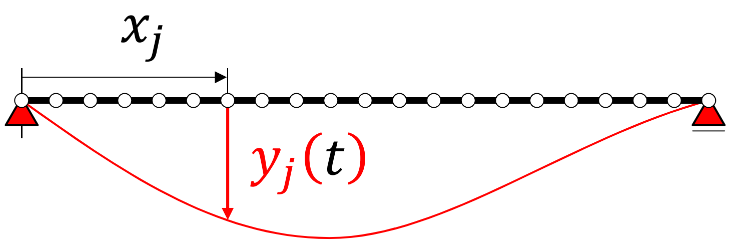
$$u_1(t)=R\big(x_1(t)\big)+y\big(x_1(t),t\big)$$
$$u_2(t)=R\big(x_2(t)\big)+y\big(x_2(t),t\big)$$



The equation of motion of Bridge System:

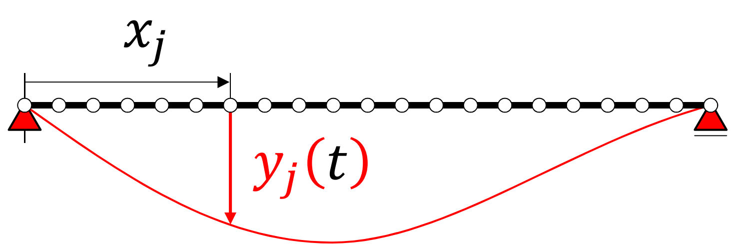
$$\rho A\ddot{y}(x,t)+\frac{\partial^2}{\partial x^2}EI\left(\frac{\partial^2}{\partial x^2}y(x,t)\right)=\delta\big(x-x_1(t)\big)P_1(t)+\delta\big(x-x_2(t)\big)P_2(t)$$

FEM is a method for discretizing a Continuum Body



$$\boldsymbol{y}(t) = \begin{Bmatrix} \vdots \\ y_j(t) \\ \vdots \end{Bmatrix}$$

The equation of motion of Bridge can be also expressed by matrices and vectors



$$\mathbf{M}_{\mathrm{B}}\ddot{\boldsymbol{y}}(t) + \mathbf{K}_{\mathrm{B}}\boldsymbol{y}(t) = \boldsymbol{L}(t)\boldsymbol{P}(t)$$

Mass Matrix

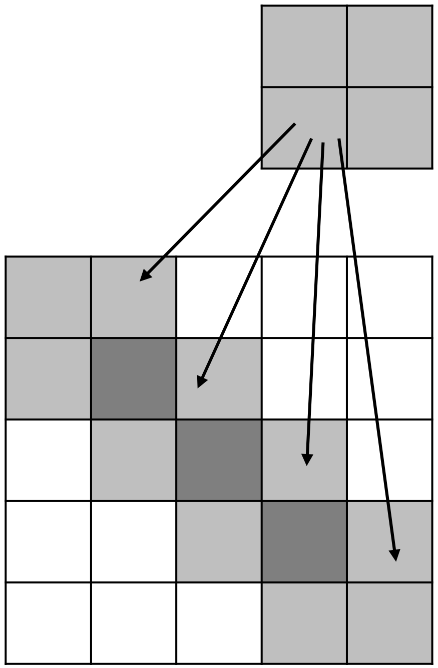
Stiffness Matrix

Equivalent Nodal Force Distribution Matrix

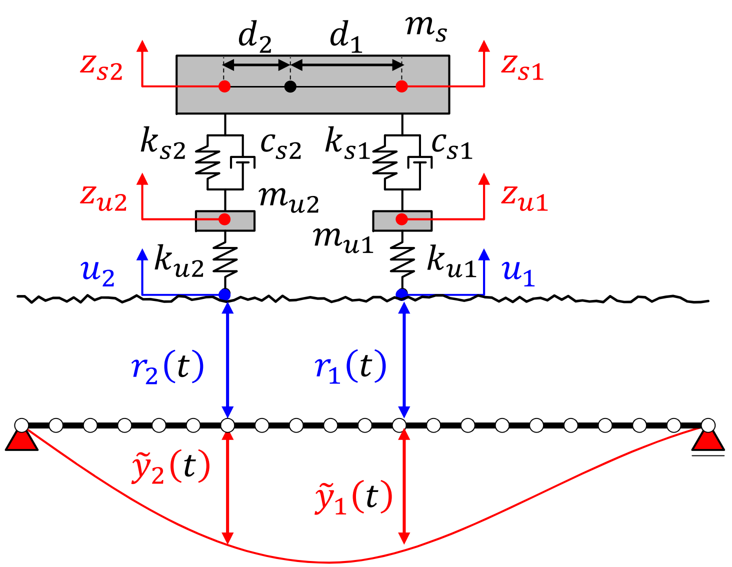
Contact Force

$$\mathbf{K}_{\mathrm{B}}^{(e)} = \frac{EI}{l^3}\begin{bmatrix} 12 & 6l & -12 & 6l \\ 6l & 4l^2 & -6l & 2l^2 \\ -12 & -6l & 12 & -6l \\ 6l & 2l^2 & -6l & 4l^2 \end{bmatrix}$$

$\mathbf{K}_{\mathrm{B}} =$



$$\mathbf{M}_{\mathrm{V}}\ddot{\boldsymbol{z}}(t) + \mathbf{C}_{\mathrm{V}}\dot{\boldsymbol{z}}(t) + \mathbf{K}_{\mathrm{V}}\boldsymbol{z}(t) = \mathbf{K}_{\mathrm{U}}\big(\boldsymbol{r}(t) + \boldsymbol{L}(t)\boldsymbol{y}(t)\big)$$



$$\mathbf{M}_{\mathrm{B}}\ddot{\boldsymbol{y}}(t) + \mathbf{K}_{\mathrm{B}}\boldsymbol{y}(t) = \boldsymbol{L}(t)\mathbf{M}_{\mathrm{V}}\big(\boldsymbol{g} - \ddot{\boldsymbol{z}}(t)\big)$$

$$\begin{bmatrix} \mathbf{M}_{\mathrm{V}} & \\ \boldsymbol{L}(t)\mathbf{M}_{\mathrm{V}} & \mathbf{M}_{\mathrm{B}} \end{bmatrix} \begin{Bmatrix} \ddot{\boldsymbol{z}} \\ \ddot{\boldsymbol{y}} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}_{\mathrm{V}} & \\ & \mathbf{C}_{\mathrm{B}} \end{bmatrix} \begin{Bmatrix} \dot{\boldsymbol{z}} \\ \dot{\boldsymbol{y}} \end{Bmatrix} + \begin{bmatrix} \mathbf{K}_{\mathrm{V}} & \mathbf{K}_{\mathrm{U}}\boldsymbol{L}(t) \\ & \mathbf{K}_{\mathrm{B}} \end{bmatrix} \begin{Bmatrix} \boldsymbol{z} \\ \boldsymbol{y} \end{Bmatrix} = \begin{Bmatrix} \mathbf{K}_{\mathrm{U}}\boldsymbol{r}(t) \\ \boldsymbol{L}(t)\mathbf{M}_{\mathrm{V}}\boldsymbol{g} \end{Bmatrix}$$

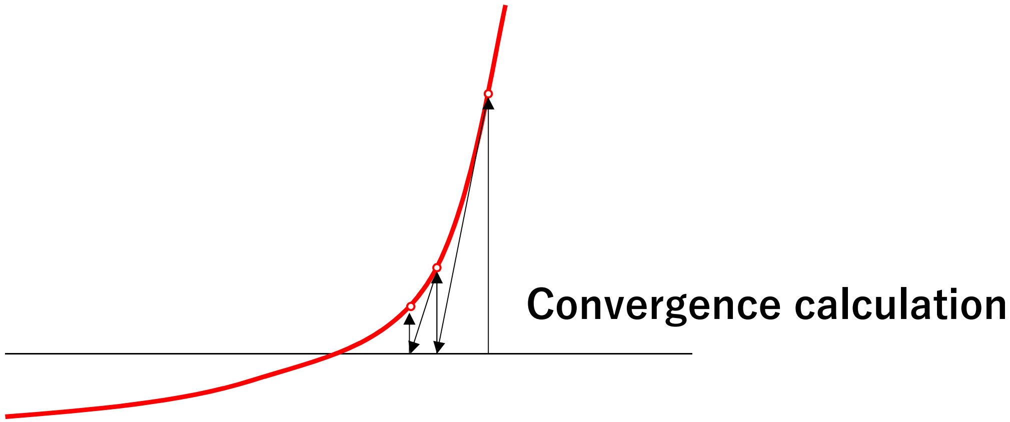
Equation of Motion of VBI system is NOT a linear differential equation

$$\begin{bmatrix} \mathbf{M}_\mathrm{V} & \\ \boldsymbol{L}(t)\mathbf{M}_\mathrm{V} & \mathbf{M}_\mathrm{B} \end{bmatrix} \begin{Bmatrix} \ddot{\boldsymbol{z}} \\ \ddot{\boldsymbol{y}} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}_\mathrm{V} & \\ & \mathbf{C}_\mathrm{B} \end{bmatrix} \begin{Bmatrix} \dot{\boldsymbol{z}} \\ \dot{\boldsymbol{y}} \end{Bmatrix} + \begin{bmatrix} \mathbf{K}_\mathrm{V} & \mathbf{K}_\mathrm{U}\boldsymbol{L}(t) \\ & \mathbf{K}_\mathrm{B} \end{bmatrix} \begin{Bmatrix} \boldsymbol{z} \\ \boldsymbol{y} \end{Bmatrix} = \begin{Bmatrix} \mathbf{K}_\mathrm{U}\boldsymbol{r}(t) \\ \boldsymbol{L}(t)\mathbf{M}_\mathrm{V}\boldsymbol{g} \end{Bmatrix}$$

$$\mathbf{M}(t)\ddot{\boldsymbol{x}}(t) + \mathbf{C}\dot{\boldsymbol{x}}(t) + \mathbf{K}(t)\boldsymbol{x}(t) = \boldsymbol{f}(t)$$

Coefficients temporary changes

Newton-Raphson Method for solving a non-linear problem



Road Profile

Vehicle

	Sprung-mass: m_s (kg)	Unsprung-mass: m_{ui} (kg)	Sprung-damping: c_{si} (kg/s)	Sprung-stiffness: k_{si} (N/m)	Unsprung-stiffness: k_{ui} (N/m)	Distance from G: d_i (m)	Travel speed: v (m)
Front: i=1	9000	500	2000	4500	60000	1.75	10.0
Rear: i=2		500	2000	4500	60000	1.75	

Vehicle Vibration

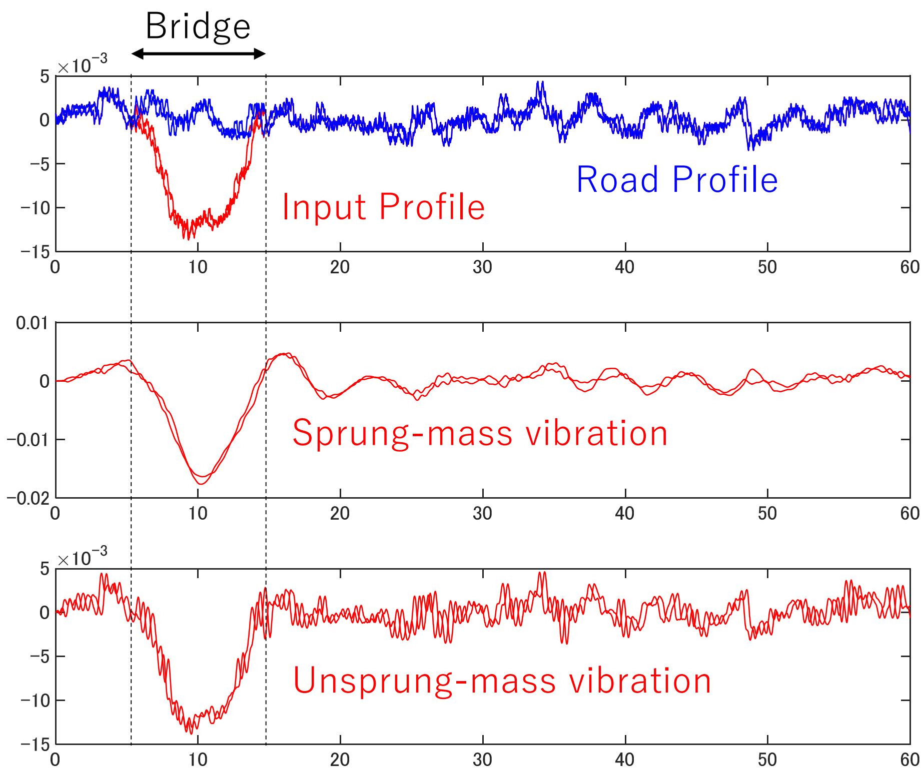
Contact Force

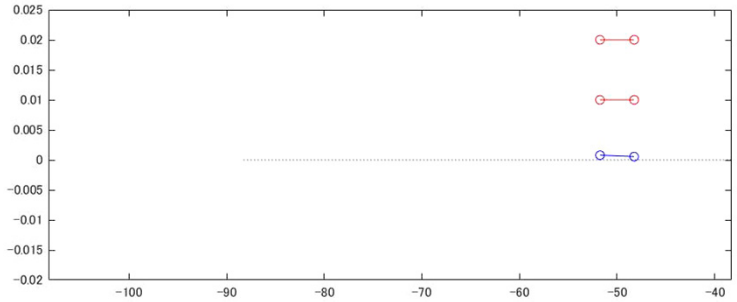
Input Profile

Bridge Vibration

Bridge

Flexural Rigidity: EI (GNm^2)	Mass per unit Length: ρA (kg/m)	Span Length: L (m)	Element Number
156	3000	100	20





1) Download: http://www.kz.tsukuba.ac.jp/~yamamoto_k/material/matlab_for_ide.zip
2) Decompress the downloaded **zip** file
3) Launch **Matlab** and set the decompressed directory as **current** directory
4) Run the following script:

```
>> sample_04
>> sample_04_application
```

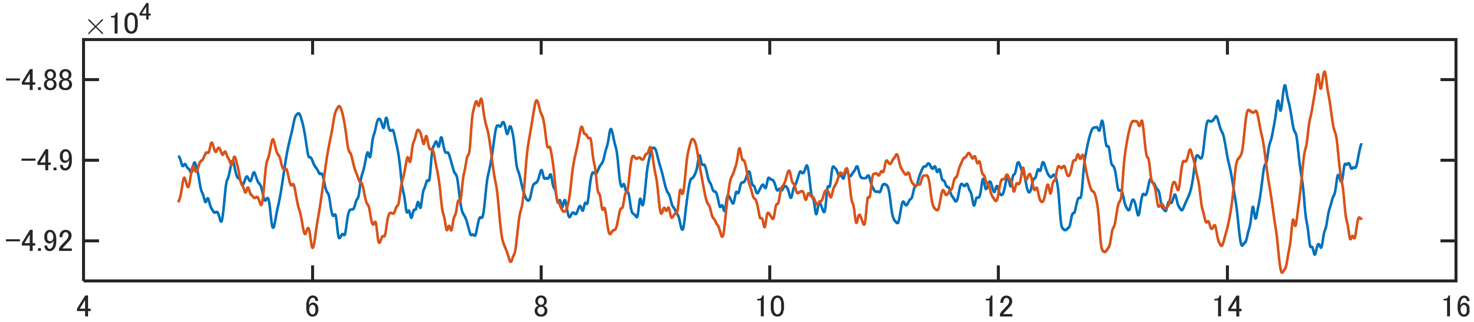
VBI simulation is applied to Bridge Design and Maintenance

Traffic Load for Bridge Design

Vibration-based **S**tructure **H**ealth **M**onitoring

Future Technology (**Watching Logistics**)

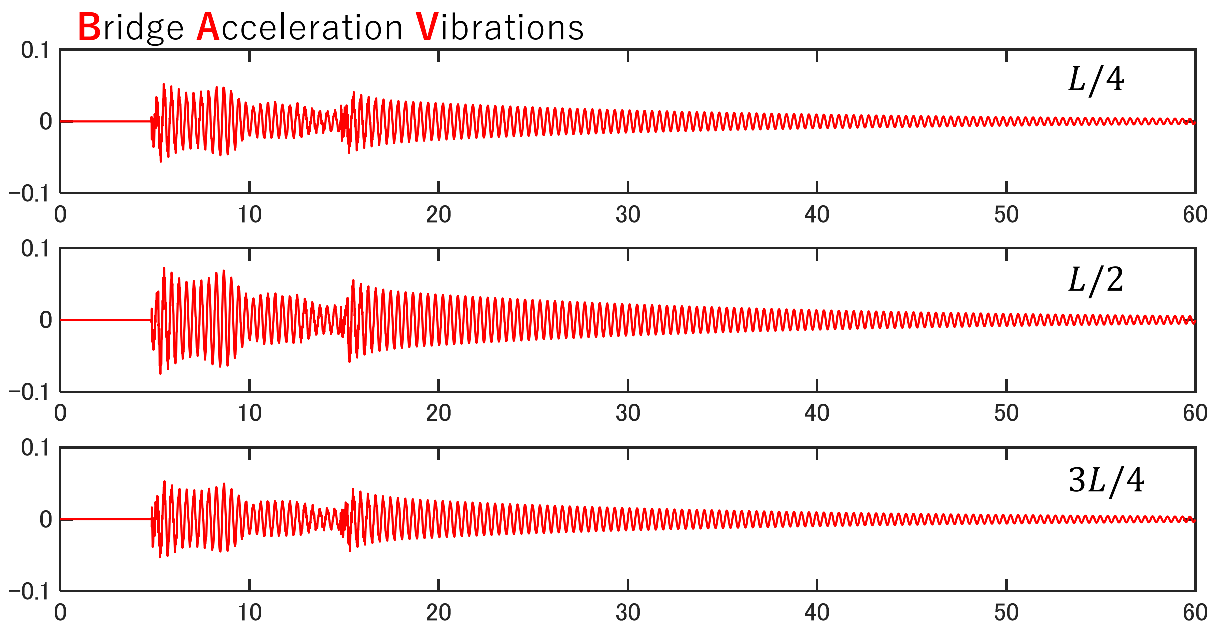
Traffic Load for Bridge Design



Maximum Value of P_i of **10t** (100kN) vehicle is about **50** (kN)

(**NOTICE**: Only considering **very smooth** road profile)

Vibration-based
Structure Health Monitoring



Watching Logistics

(On-going Identification of Vehicle-Bridge-Road)

Only **Measuring**
Vehicle Acceleration **Vibrations** and **Position**

→

Identifying
Vehicle Parameters, **Bridge** Parameters
and **Road** Profile

筑波大学
University of Tsukuba

IMAGINE
THE
FUTURE.