# Introduction to Robotics

## Manipulation and Programming

## Unit 2: Kinematics

PYTHON EXAMPLE: MATRIX INVERSE, FUNCTION DERIVATIVES AND INTEGRAL

DR. ERIC CHOU

IEEE SENIOR MEMBER

# Objectives

- Inverse of Matrix
- Parametric Equations
- Derivatives
- Integrals

# Inverse of Matrix

SECTION 1

# Inverse of Matrix

```
import numpy as np

E = 2

x = np.array([[1,3],[2,1]])
y = np.linalg.inv(x)
y = np.round(y, E)
```

```
X:
[[1 3]
 [2 1]]

Y:
[[-0.2  0.6]
 [ 0.4 -0.2]]

I=XY:
[[ 1. -0.]
 [ 0.  1.]]
```

# Parametric Equations

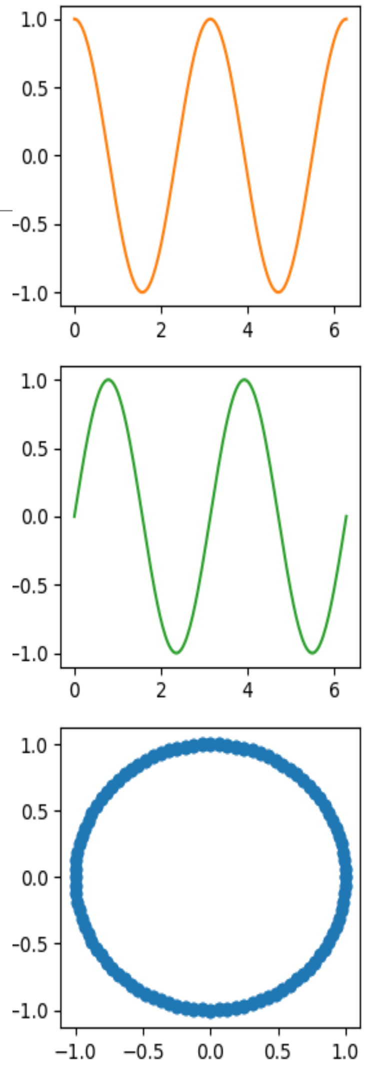
SECTION 2

# Parametric Equations

```python
import numpy as np
from pylab import *

t = linspace(0, 2.0*np.pi, 201)
w = 2    # radians / sec
x = [np.cos(w * t[i]) for i in range(len(t))]
y = [np.sin(w * t[i]) for i in range(len(t))]

fig, (plot1, plot2, plot3) = subplots(3)
plot1.plot(t, x, 'tab:orange')
plot2.plot(t, y, 'tab:green')
plot3.scatter(x, y)
show()
```



# Derivatives

SECTION 3

# Taking Derivative

```
def derivative(f, w, t, h):
    return (f(w*(t+h))-f(w*t))/h
```

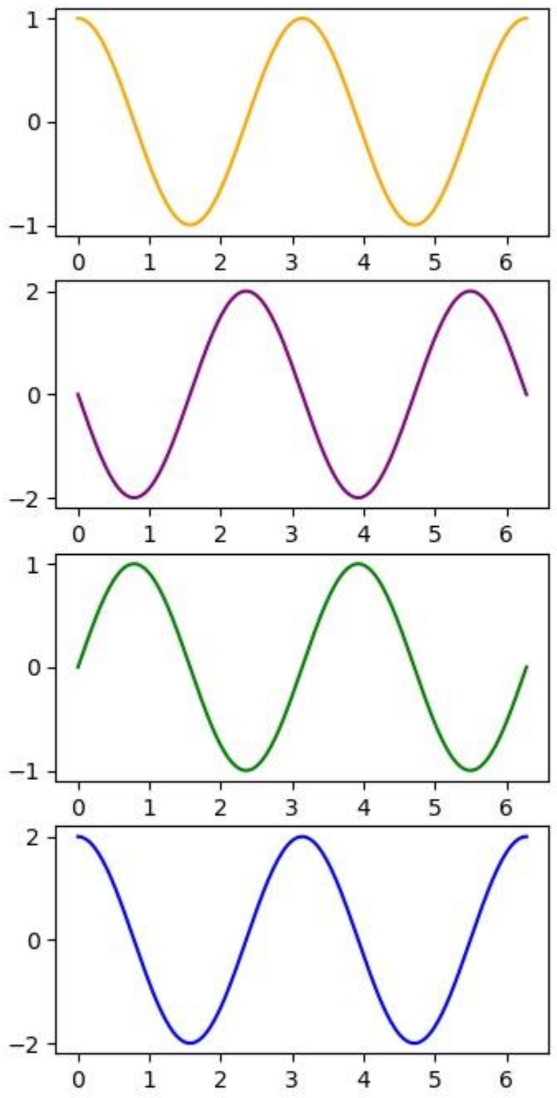
```python
import numpy as np
from pylab import *
E = 10

def derivative(f, w, t, h):
    return (f(w*(t+h))-f(w*t))/h

t = linspace(0, 2.0*np.pi, 201)
w = 2    # radians / sec
h = 10**(-E)

x = [np.cos(w * i) for i in t]
dxdt = [derivative(np.cos, w, t[i], h) for i in range(len(t))]
y = [np.sin(w * i) for i in t]
dydt = [derivative(np.sin, w, t[i], h) for i in range(len(t))]

fig, (plot1, plot2, plot3, plot4) = subplots(4)
plot1.plot(t, x, 'orange')
plot2.plot(t, dxdt, 'purple')
plot3.plot(t, y, 'green')
plot4.plot(t, dydt, 'blue')
show()
```



# Integrals

SECTION 4

# Integral (definite)

```
def integral(f, w, c, a, b, h):
    y = [c]                                     # initial value
    lenx = int((b-a)/h)
    intf = c
    t = a
    for i in range(lenx-1):                     # integral engine
        intf0 = intf
        intf  = intf0 + f(w*t)*h
        t += h
        y.append(intf)
    return y
```

```python
import numpy as np
from pylab import *
E = 3
h = 10**(-E)

def integral(f, w, c, a, b, h):
    y = [c]             # initial value
    lenx = int((b-a)/h)
    intf = c
    t = a
    for i in range(lenx-1):
        # integral engine
        intf0 = intf
        intf  = intf0 + f(w*t)*h
        t += h
        y.append(intf)
    return y
```

```python
a = 0
b = np.pi*2
t = np.linspace(a, b, int((b-a)/h))
w = 2   # radians / sec
x = [np.cos(w*t[i]) for i in range(len(t))]
intx = integral(np.cos, w, 0, a, b, h)
y = [np.sin(w*t[i]) for i in range(len(t))]
inty = integral(np.sin, w, -0.5, a, b, h)

fig, (plot1, plot2, plot3, plot4) =
subplots(4)
plot1.plot(t, x, 'orange')
plot2.plot(t, intx, 'purple')
plot3.plot(t, y, 'green')
plot4.plot(t, inty, 'blue')
show()
```