MATH 152 Lab 1

Ram Paramathmuni 501

```
import sympy as sp
from sympy.plotting import (plot,plot_parametric)

x,u = sp.symbols('x u', real=True)
```

Instructions: Complete the lab assignment in your assigned groups. Unless stated otherwise, your answers should be obtained using Python code.

Do not modify the cell above, as it contains all the packages you will need. It is highly recommended to not use any additional packages.

Question 1

1a

```
# Enter your code here.
f = x**3 + 3*x**2 + 3*x + 2
fprime = sp.diff(f, x)
x0 = 1

derivative1 = fprime.subs(x, x0)
# derivative from slope
derivative2 = sp.limit((f - f.subs(x, x0)) / (x - x0), x, x0)
print(fprime)
print(derivative1)
print(derivative2)
```

```
3*x**2 + 6*x + 3
12
12
```

1b

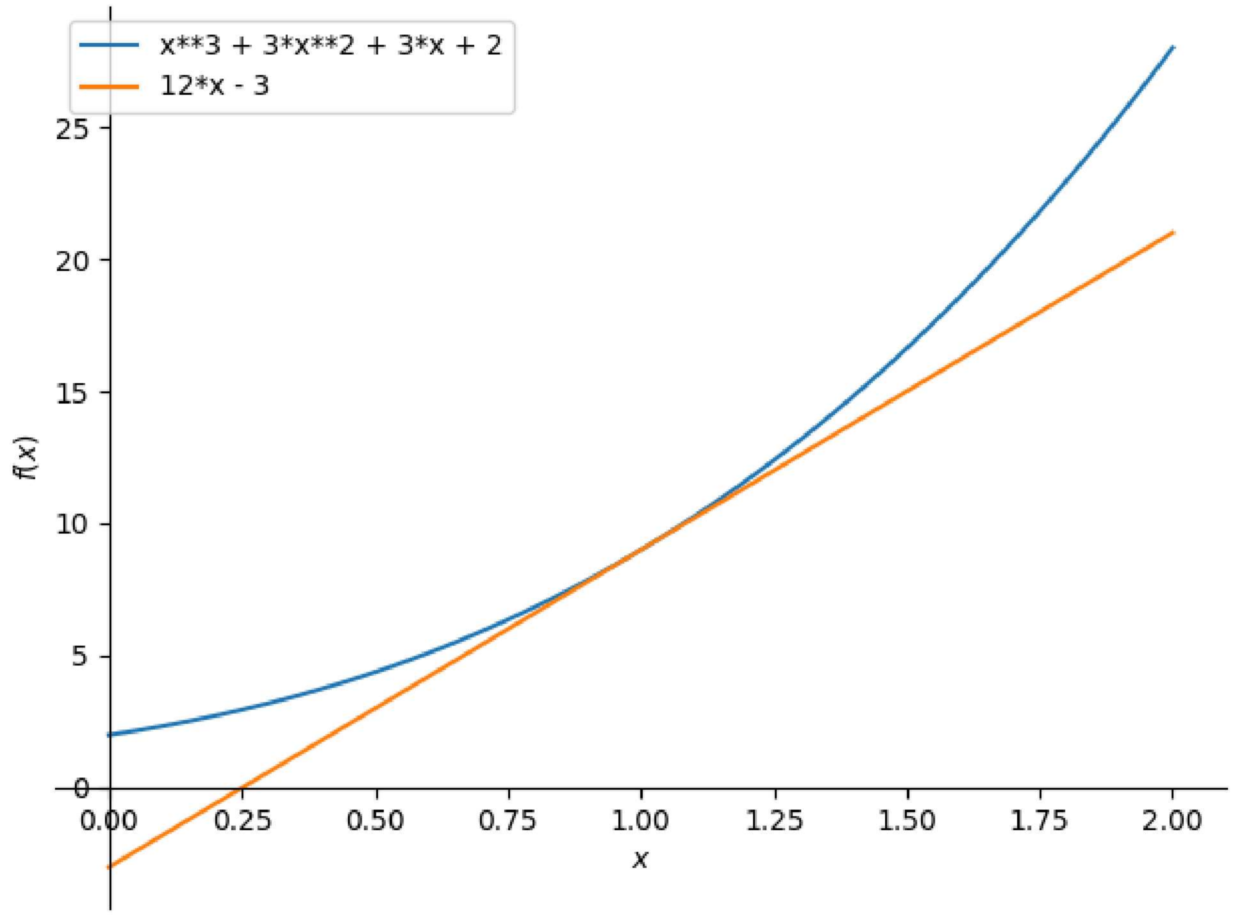
```
# Enter your code here.
y0 = f.subs(x, x0)
```

```
m = fprime.subs(x, x0)
tangent = sp.expand(y0 + m*(x-x0))

print(tangent)

plot(f,tangent, (x,0,2),
     show=True,
     legend=True,)
```

```
12*x - 3
```



```
<sympy.plotting.backends.matplotlibbackend.matplotlib.MatplotlibBackend at
0x7ddc07d60c80>
```

1c

```
# Enter your code here.
sols = sp.solve(sp.Eq(f, tangent), x)
other = [s for s in sols if s != x0]
[(s, f.subs(x, s)) for s in other]
```

```
[(-5, -63)]
```

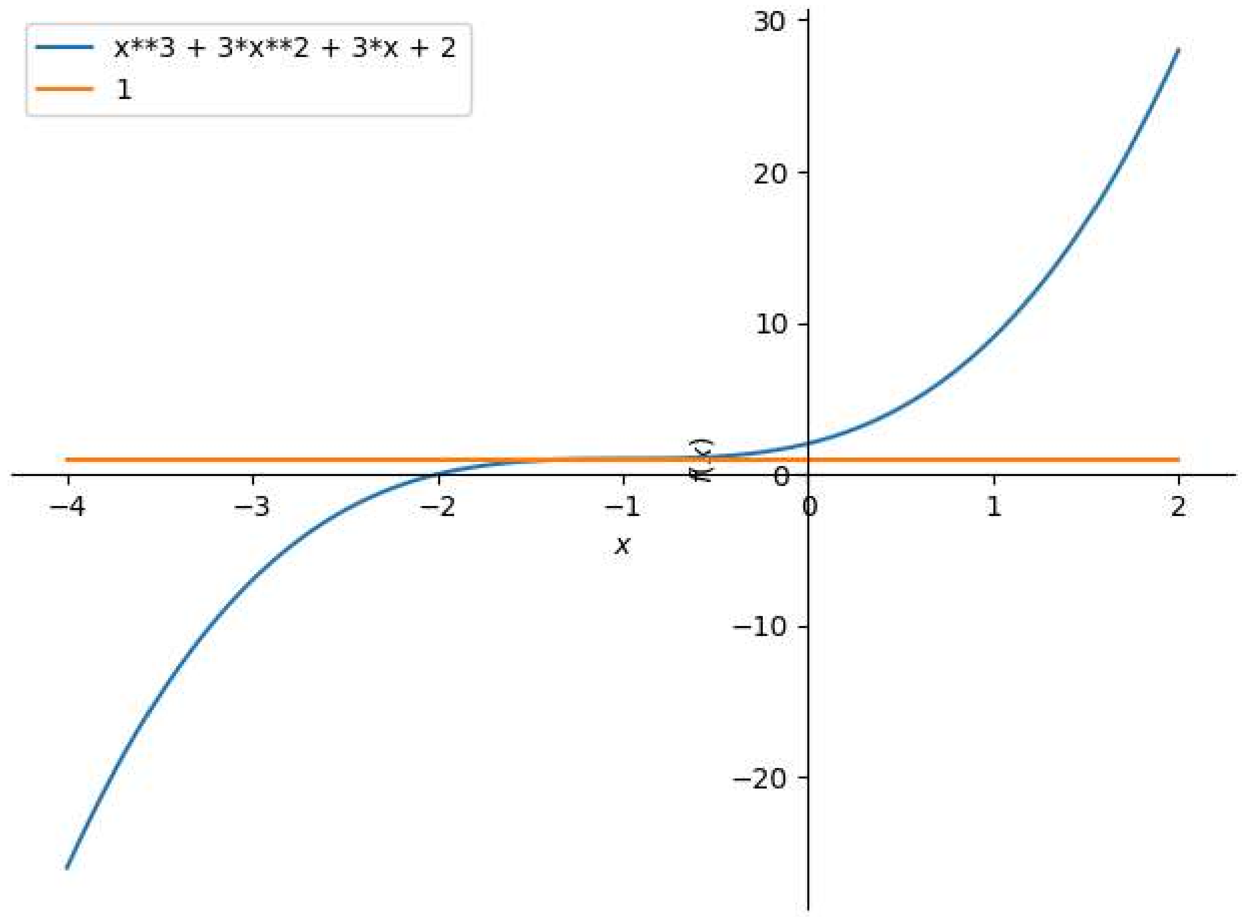
1d

```
# Enter your code here.
# inflection point
f2 = sp.diff(fprime, x)
infl_x = sp.solve(sp.Eq(f2, 0), x)[0]
infl_y = f.subs(x, infl_x)
(infl_x, infl_y)
```

```
(-1, 1)
```

1e

```
# Enter your code here.
# plot f(x) and tangent line at inflection point
matP = fprime.subs(x, infl_x)
tangent_infl = sp.expand(infl_y + matP*(x-infl_x))
tangent_infl
plot(
    f, tangent_infl, (x, -4, 2),
    show=True,
    legend=True
)
```



```
<sympy.plotting.backends.matplotlibbackend.matplotlib.MatplotlibBackend at
0x7ddc00608c80>
```

1f

```
# Enter your code here.
sols_P = sp.solve(sp.Eq(f, tangent_infl), x)
[s for s in sols_P if s != infl_x]
```

```
[]
```

Question 2

2a

```
# Enter your code here.
def usub_convert(integrand, gx, u_symb, x_symb):
    """
    Given f(g(x)) * g'(x),
        return f(u) by dividing out g'(x) and substituting g(x) -> u.
        """
        gprime = sp.diff(gx, x_symb)
        integrand_modified = integrand / gprime
        integrand_u = integrand_modified.subs(gx, u_symb)
        return integrand_u
```

2b

```
# Enter your code here.
integrand = sp.exp(x)/(1 + sp.exp(2*x))
integrated = sp.integrate(integrand, x)
integrated
```

$$\operatorname{RootSum}\left(4z^2 + 1, \left(i \mapsto i \log\left(2i + e^x\right)\right)\right)$$

2c

```
# Enter your answer here as either a comment or print statement.
# Let u = e^x
```

2d

```
# Enter your code here.
fu = usub_convert(integrand, sp.exp(x), u, x)
fu
integrated = sp.integrate(fu, u)
integrated
subs_back = integrated.subs(u, sp.exp(x))
subs_back
```

$$\operatorname{atan}\left(e^x\right)$$