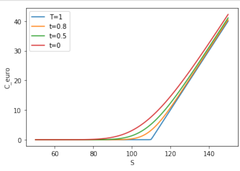
In [1]:
```
import numpy as np # numerical & math calculations
import matplotlib.pyplot as plt # graphing
import scipy.stats as sp # prob/stat functions
```

In [2]:
```
# define functions needed for European Call
def d1(S,K,T,t,r,sigma):
    d1 = (np.log(S/K) + (r+0.5*sigma**2)*(T-t))/(sigma*np.sqrt(T-t))
    return d1
def d2(S,K,T,t,r,sigma):
    d2 = (np.log(S/K) + (r-0.5*sigma**2)*(T-t))/(sigma*np.sqrt(T-t))
    return d2
def C_euro(S,K,T,t,r,sigma):
    C_euro = S*sp.norm.cdf(d1(S,K,T,t,r,sigma))-\
        K*np.exp(-r*(T-t))*sp.norm.cdf(d2(S,K,T,t,r,sigma))
    return C_euro
```

In [3]:
```
# test parameters
S = 100;#
K = 110;
T = 1;#
t = 0;
r = 0.02;#
sigma = 0.15;#
print(d1(S,K,T,t,r,sigma), d2(S,K,T,t,r,sigma), C_euro(S,K,T,t,r,sigma))
```

```
-0.42706786536216595 -0.577067865362166 3.0664675602182285
```

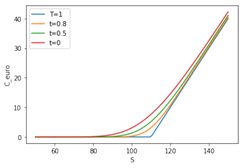
In [4]:
```
S = np.linspace(50,150,101);
eps = 0.0001; # a small number
C0 = C_euro(S,K,T,T-eps,r,sigma);
C1 = C_euro(S,K,T,T-0.2,r,sigma);
C2 = C_euro(S,K,T,T-0.5,r,sigma);
C3 = C_euro(S,K,T,0,r,sigma);
plt.plot(S,C0,label='T=1')
plt.plot(S,C1,label='t=0.8')
plt.plot(S,C2,label='t=0.5')
plt.plot(S,C3,label='t=0')
plt.xlabel('S')
plt.ylabel('C_euro')
plt.legend()
plt.show()
```



In [5]:
```
# test parameters
S = 100;#
K = 110;
T = 1;#
t = 0.25;
r = 0.02;#
sigma = 0.15;#
print(d1(S,K,T,t,r,sigma), d2(S,K,T,t,r,sigma), C_euro(S,K,T,t,r,sigma))

S = np.linspace(50,150,95);
eps = 0.0001; # a small number
C0 = C_euro(S,K,T,T-eps,r,sigma);
C1 = C_euro(S,K,T,T-0.2,r,sigma);
C2 = C_euro(S,K,T,T-0.5,r,sigma);
C3 = C_euro(S,K,T,0,r,sigma);
plt.plot(S,C0,label='T=1')
plt.plot(S,C1,label='t=0.8')
plt.plot(S,C2,label='t=0.5')
plt.plot(S,C3,label='t=0')
plt.xlabel('S')
plt.ylabel('C_euro')
plt.legend()
plt.show()
```

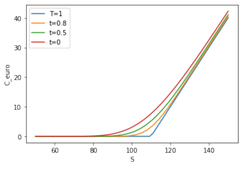
```
-0.5532761470987568 -0.6831799576664226 2.2115065341927576
```



In [6]:
```
# test parameters
S = 100;#
K = 110;
T = 1;#
t = 0.5;
r = 0.02;#
sigma = 0.15;#
print(d1(S,K,T,t,r,sigma), d2(S,K,T,t,r,sigma), C_euro(S,K,T,t,r,sigma))

S = np.linspace(50,150,55);
eps = 0.0001; # a small number
C0 = C_euro(S,K,T,T-eps,r,sigma);
C1 = C_euro(S,K,T,T-0.2,r,sigma);
C2 = C_euro(S,K,T,T-0.5,r,sigma);
C3 = C_euro(S,K,T,0,r,sigma);
plt.plot(S,C0,label='T=1')
plt.plot(S,C1,label='t=0.8')
plt.plot(S,C2,label='t=0.5')
plt.plot(S,C3,label='t=0')
plt.xlabel('S')
plt.ylabel('C_euro')
plt.legend()
plt.show()
```

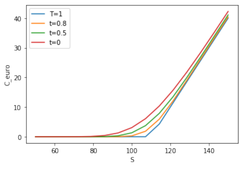
```
-0.7512790799960994 -0.8573450971740816 1.3193726996811534
```



In [7]:
```
# test parameters
S = 100;#
K = 110;
T = 1;#
t = 0.75;
r = 0.02;#
sigma = 0.15;#
print(d1(S,K,T,t,r,sigma), d2(S,K,T,t,r,sigma), C_euro(S,K,T,t,r,sigma))

S = np.linspace(50,150,15);
eps = 0.0001; # a small number
C0 = C_euro(S,K,T,T-eps,r,sigma);
C1 = C_euro(S,K,T,T-0.2,r,sigma);
C2 = C_euro(S,K,T,T-0.5,r,sigma);
C3 = C_euro(S,K,T,0,r,sigma);
plt.plot(S,C0,label='T=1')
plt.plot(S,C1,label='t=0.8')
plt.plot(S,C2,label='t=0.5')
plt.plot(S,C3,label='t=0')
plt.xlabel('S')
plt.ylabel('C_euro')
plt.legend()
plt.show()
```

```
-1.166635730724332 -1.241635730724332 0.4362756770366598
```

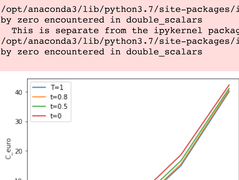


In [8]:
```
# test parameters
S = 100;#
K = 110;
T = 1;#
t = 1;
r = 0.02;#
sigma = 0.15;#
print(d1(S,K,T,t,r,sigma), d2(S,K,T,t,r,sigma), C_euro(S,K,T,t,r,sigma))

S = np.linspace(50,150,5);
eps = 0.0001; # a small number
C0 = C_euro(S,K,T,T-eps,r,sigma);
C1 = C_euro(S,K,T,T-0.2,r,sigma);
C2 = C_euro(S,K,T,T-0.5,r,sigma);
C3 = C_euro(S,K,T,0,r,sigma);
plt.plot(S,C0,label='T=1')
plt.plot(S,C1,label='t=0.8')
plt.plot(S,C2,label='t=0.5')
plt.plot(S,C3,label='t=0')
plt.xlabel('S')
plt.ylabel('C_euro')
plt.legend()
plt.show()
```

```
-inf -inf 0.0
```

```
/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:3: RuntimeWarning: divide by zero encountered in double_scalars
  This is separate from the ipykernel package so we can avoid doing imports until
/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:6: RuntimeWarning: divide by zero encountered in double_scalars
```



In [ ]:
```
```