Payoff Replication

Based on "Replication Strategy and Algorithm" - Financial Engineering Tool

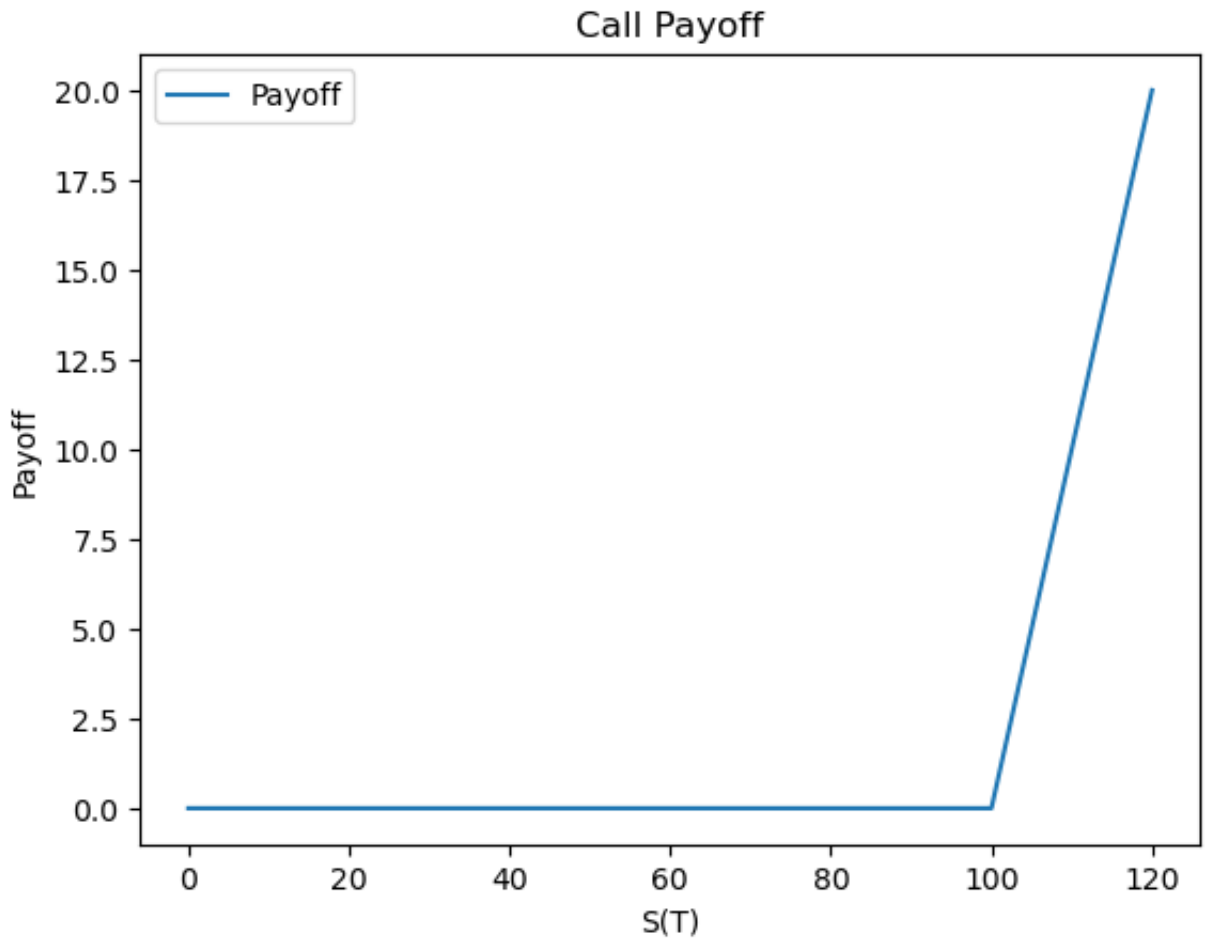
In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn
```

In [51]:
```
class replication:
    def __init__(self, data, name):
        self.name = name
        self.payoff_df = pd.DataFrame()
        self.df_parameters = pd.DataFrame(columns=['lambda', 'lambda_valu
        self.len_df = 0
        self.get_df_payoff(data)
        self.portfolios = pd.DataFrame('', columns=['ZCouponBonds', 'Nomi
        self.estimate_parameters()

    def get_df_payoff(self, data):
        self.payoff_df = pd.DataFrame(data)
        self.payoff_df['sign_payoff'] = self.payoff_df.apply(lambda row:
        self.len_df = self.payoff_df.shape[0]

    def plot_payoff(self):
        self.payoff_df.plot(x='S(T)', y='Payoff')
        plt.title(self.name + " Payoff")
        plt.xlabel("S(T)")
        plt.ylabel("Payoff")
        #plt.axis([0,150,-10,50])
        plt.show()

    def estimate_parameters(self):
        for i in range(self.len_df - 1):
            denum = (self.payoff_df['S(T)'][i+1] - self.payoff_df['S(T)']
            lambda_value = (self.payoff_df['Payoff'][i+1] - self.payoff_d
            lambda_name = 'lambda_' + str(i)
            lambda_sign = 1 if lambda_value >=0 else -1
            self.df_parameters.loc[self.df_parameters.size] = [lambda_nam
        self.df_parameters.reset_index(drop=True, inplace=True)
        self.len_df = self.df_parameters.shape[0]

    def replicate_payoff(self):
        i = 0
        i_r = 0
        i_l = 0
        while i < self.len_df:
            if self.payoff_df['Payoff'][i] != 0:
                self.portfolios["ZCouponBonds"][i] = self.payoff_df['sign
                self.portfolios["Nominal"][i] = abs(self.payoff_df['Payof
            while i_r < self.len_df:
                if i_r == i:
                    if self.df_parameters['lambda_value'][i] != 0:
                        self.portfolios["Calls"][i] = str(self.portfolios
                        self.portfolios["Calls_strike"][i] = str(self.por
                elif i_r < self.len_df:
```

```
                    if self.df_parameters['lambda_value'][i_r] - self.df_
                        self.portfolios["Calls"][i] = str(self.portfolios
                        self.portfolios["Calls_strike"][i] = str(self.por
                i_r += 1
            while i_l != 0:
                if i_l == i:
                    if self.df_parameters['lambda_value'][i-1] != 0:
                        self.portfolios["Puts"][i] = str(self.portfolios[
                        self.portfolios["Puts_strike"][i] = str(self.port
                else:
                    if self.df_parameters['lambda_value'][i_l] - self.df_
                        self.portfolios["Puts"][i] = str(self.portfolios[
                        self.portfolios["Puts_strike"][i] = str(self.port
                i_l -= 1
            i+= 1
            i_r = i
            i_l = i
        self.portfolios = self.portfolios.drop_duplicates(keep='first').f
        self.portfolios.reset_index(drop=True, inplace=True)
        return self.portfolios
```

Call

In [58]:
```
payoff = {'S(T)': [0., 100, 110., 120.],
          'Payoff': [0., 0., 10., 20.]}

Call = replication(payoff, name="Call")
Call.plot_payoff()
Call.replicate_payoff()
```



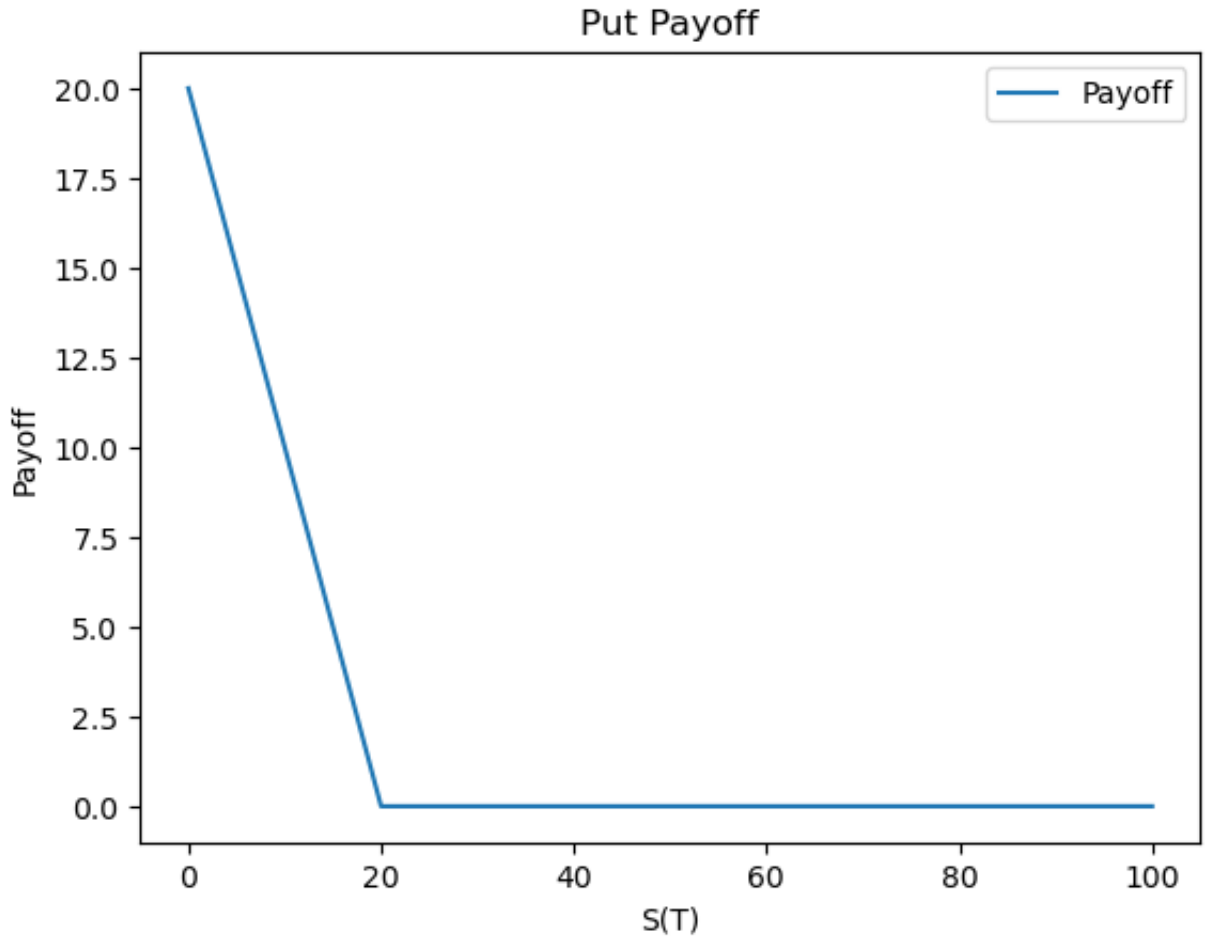
Out[58]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0			1.0	100.0		
1	1	10.0	1.0	110.0	-1.0 1.0	110.0 100.0
2						

Put

In [62]:

```
payoff = {'S(T)': [0., 20 , 100.],
          'Payoff': [20., 0., 0.]}

Put = replication(payoff, name="Put")
Put.plot_payoff()
Put.replicate_payoff()
```

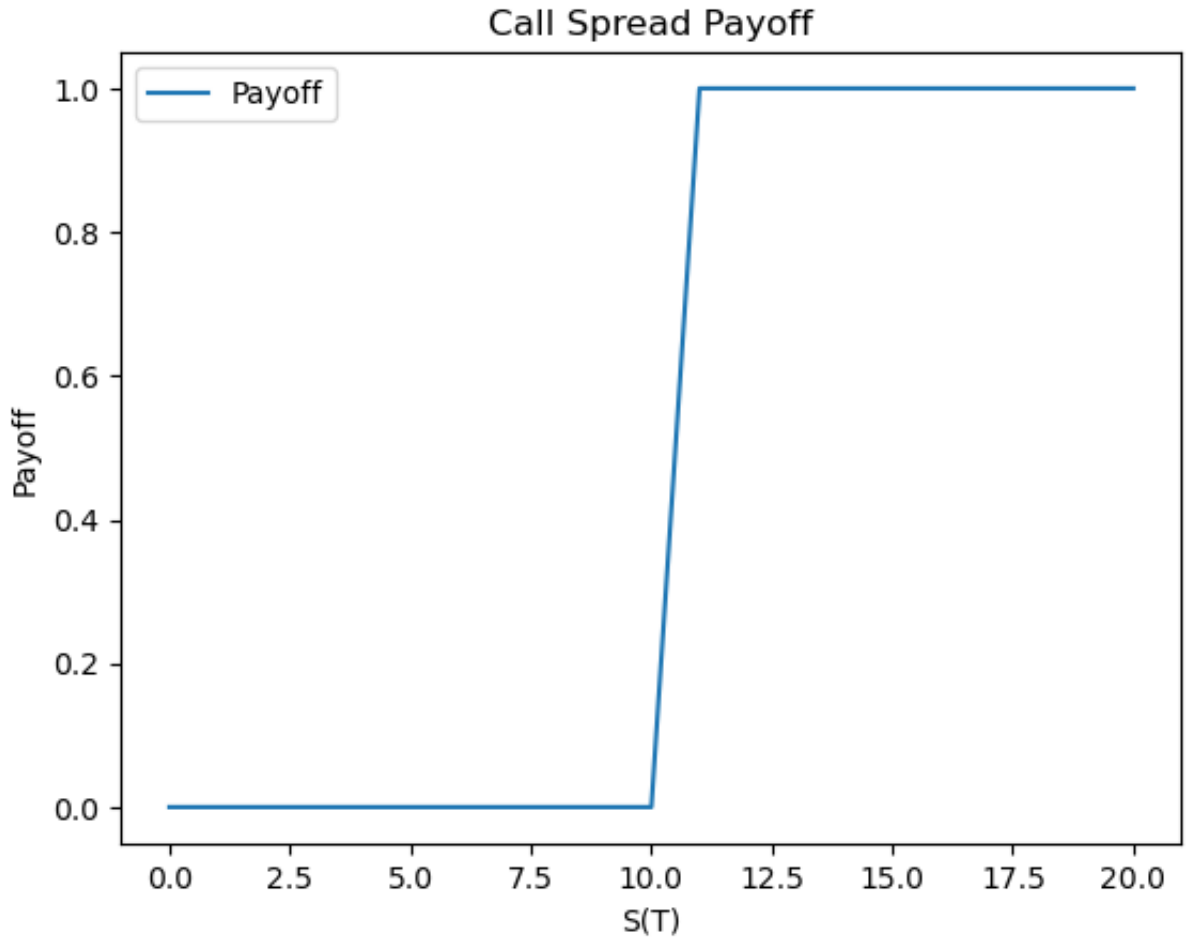


Out[62]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0	1	20.0	-1.0 1.0	0.0 20.0		
1					1.0	20.0
2						

Call spread

In [71]:
```
payoff = {'S(T)': [0., 10., 11., 20.],
          'Payoff': [0., 0., 1., 1.]}

Call_spread = replication(payoff, name="Call Spread")
Call_spread.plot_payoff()
Call_spread.replicate_payoff()
```



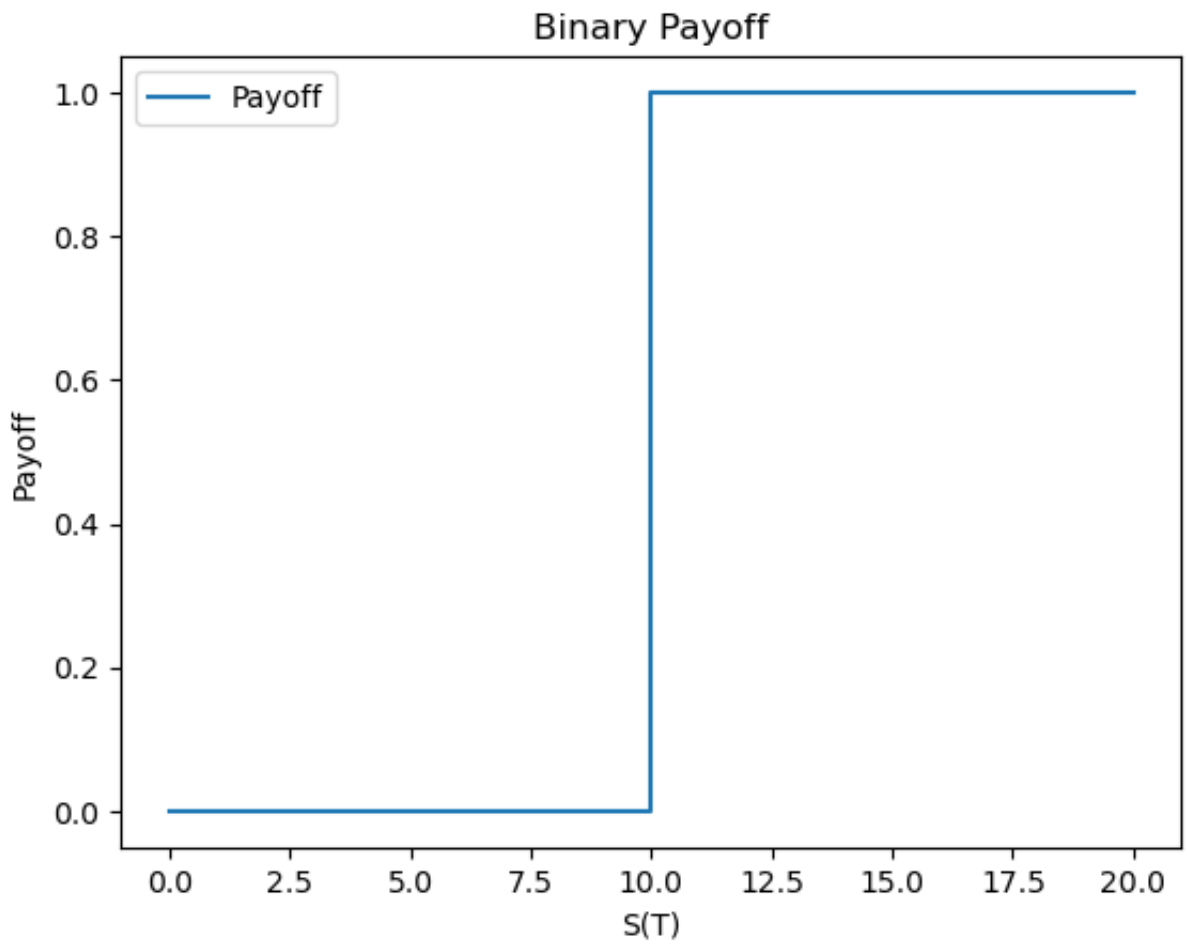
Out[71]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0			1.0 -1.0	10.0 11.0		
1	1	1.0			-1.0 1.0	11.0 10.0
2						

Binary

doesn't work because can't be write as call/put/bonds

In [72]:
```
payoff = {'S(T)': [0., 10., 10., 20.],
          'Payoff': [0., 0., 1., 1.]}

Binary = replication(payoff, name="Binary")
Binary.plot_payoff()
Binary.replicate_payoff()
```



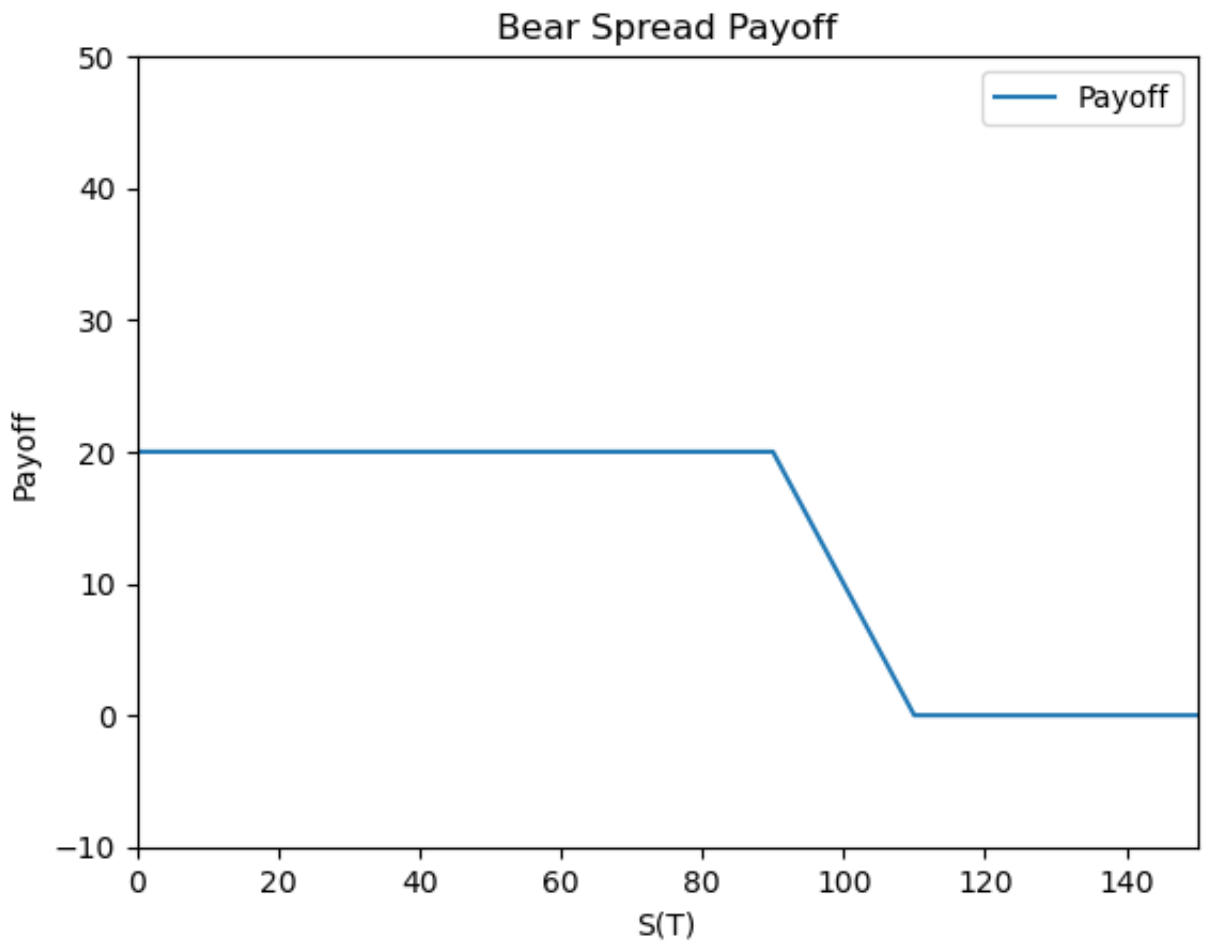
Out[72]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0						
1		1	1.0			

Bear Spread

In [50]:

```
payoff = {'S(T)': [0., 90., 110., 150.],
          'Payoff': [20., 20., 0., 0.]}

bear_spread = replication(payoff, name="Bear Spread")
bear_spread.plot_payoff()
bear_spread.replicate_payoff()
```



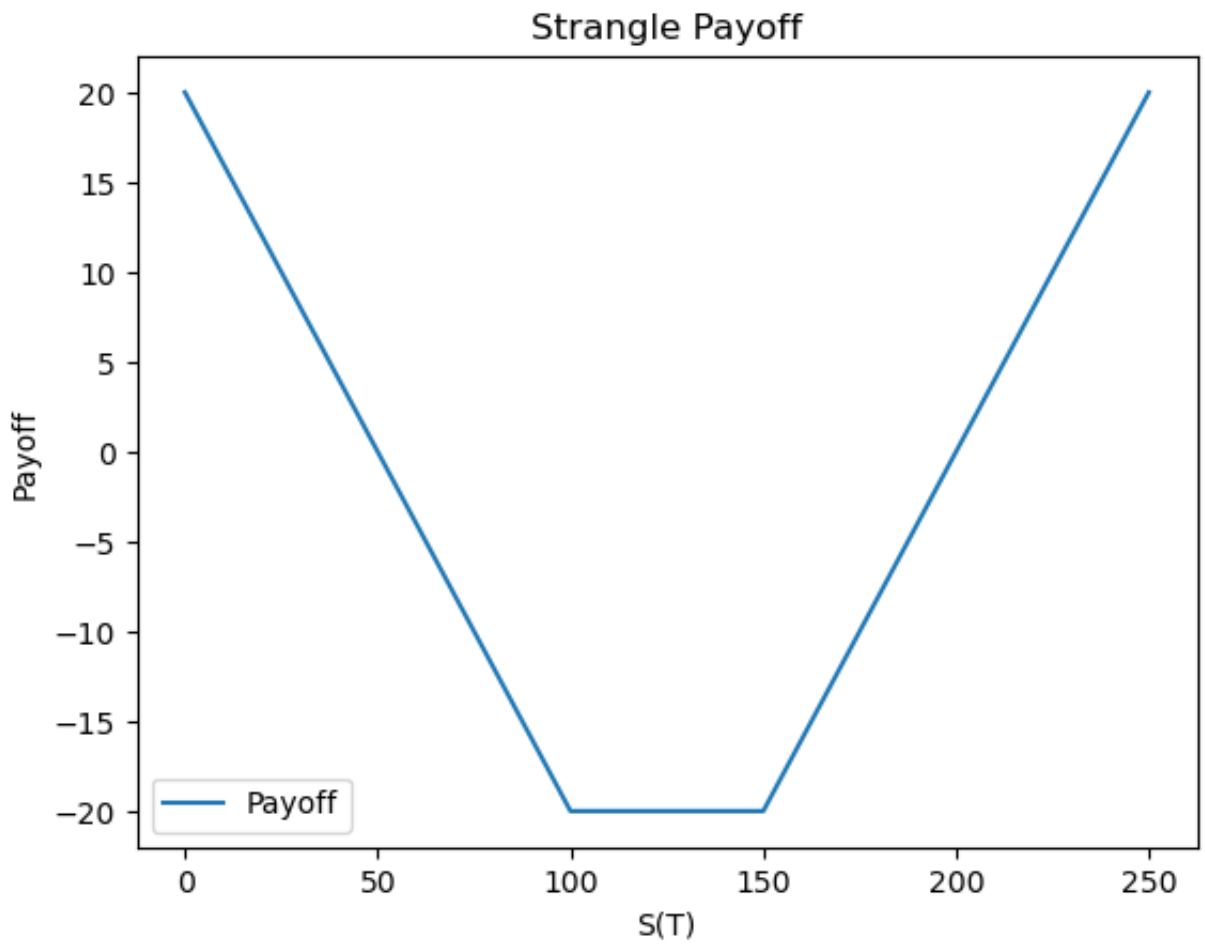
Out[50]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0	1	20.0	-1.0 1.0	90.0 110.0		
1					1.0 -1.0	110.0 90.0
2						

Strangle

In [53]:

```
payoff = {'S(T)': [0., 50., 100., 150., 200., 250.],
          'Payoff': [20., 0., -20., -20., 0., 20]}

Strangle = replication(payoff, name="Strangle")
Strangle.plot_payoff()
Strangle.replicate_payoff()
```



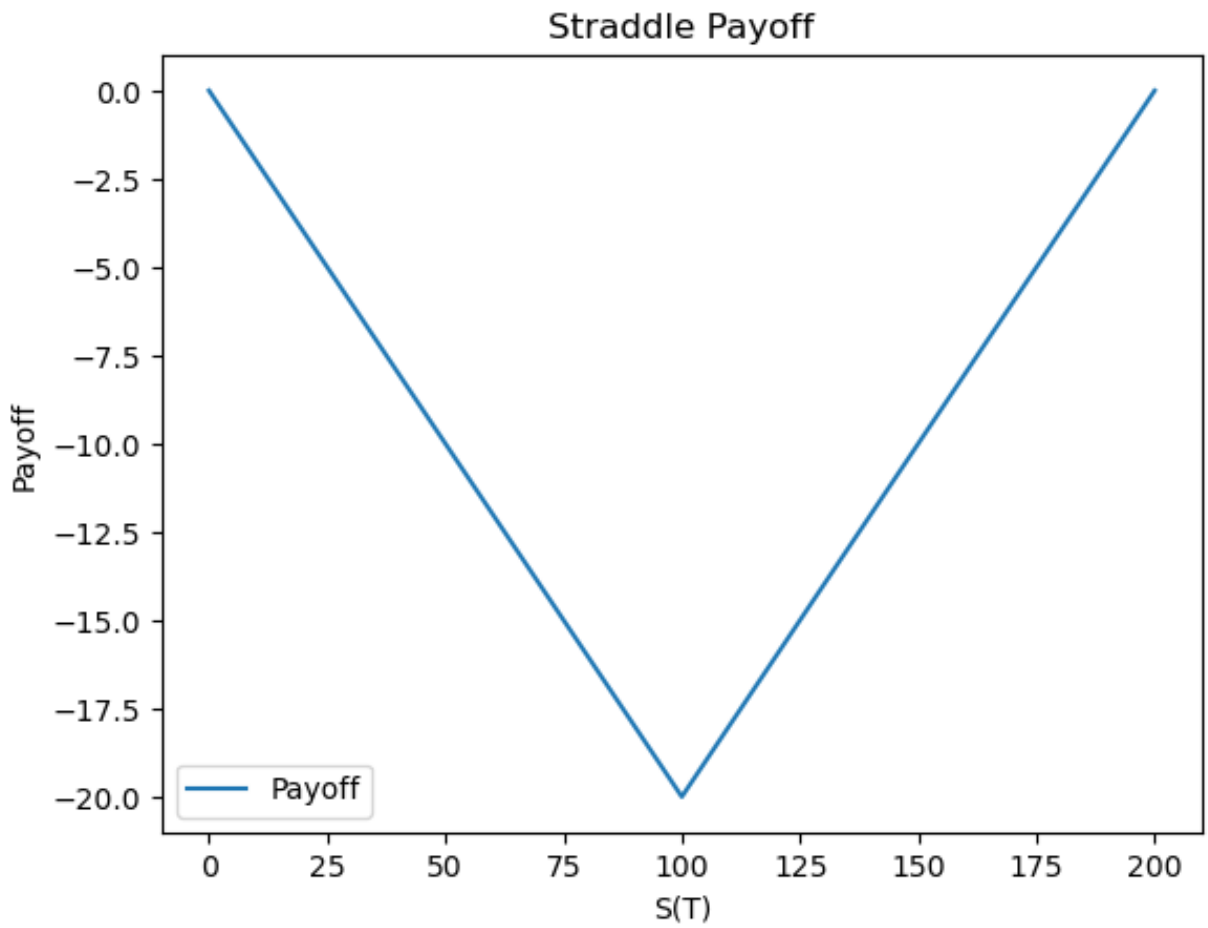
Out[53]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0	1	20.0	-0.4 0.4 0.4	0.0 100.0 150.0		
1			-0.4 0.4 0.4	50.0 100.0 150.0	0.4	50.0
2	-1	20.0	0.4	150.0	0.4	100.0
3			0.4	200.0	-0.4 0.4 0.4	200.0 150.0 100.0
4						

Straddle

In [55]:

```
payoff = {'S(T)': [0., 100., 200.],
          'Payoff': [0., -20., 0]}

Straddle = replication(payoff, name="Straddle")
Straddle.plot_payoff()
Straddle.replicate_payoff()
```



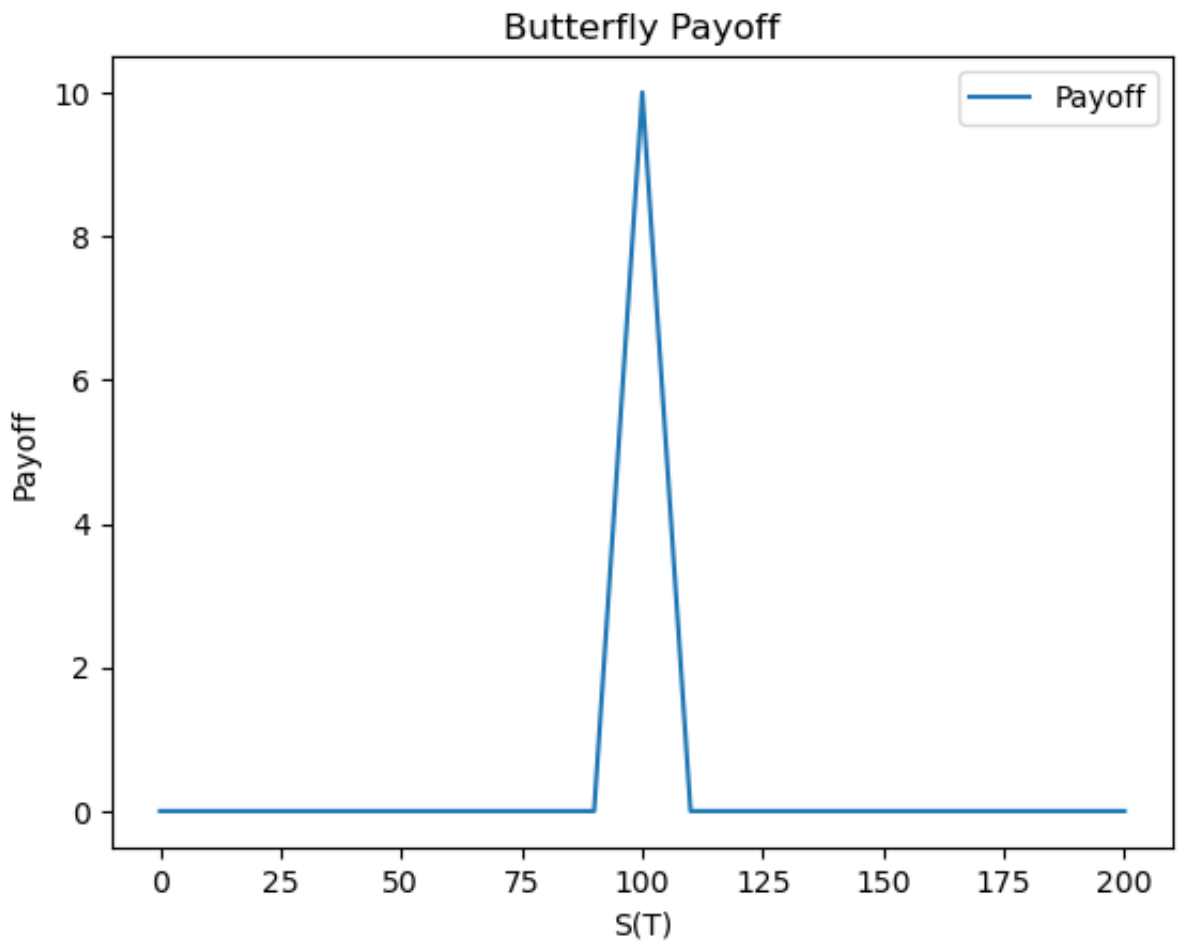
Out[55]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0			-0.2 0.4	0.0 100.0		
1	-1	20.0	0.2	100.0	0.2	100.0
2						

Butterfly

In [66]:

```
payoff = {'S(T)': [0., 90., 100., 110., 200],
          'Payoff': [0., 0., 10., 0., 0.]}

Butterfly = replication(payoff, name="Butterfly")
Butterfly.plot_payoff()
Butterfly.replicate_payoff()
```



Out[66]:

	ZCouponBonds	Nominal	Calls	Calls_strike	Puts	Puts_strike
0			1.0 -2.0 1.0	90.0 100.0 110.0		
1	1	10.0	-1.0 1.0	100.0 110.0	-1.0 1.0	100.0 90.0
2					1.0 -2.0 1.0	110.0 100.0 90.0
3						

In [ ]: