In [1]:
```
#Gerando os arquivos para cada caso mas omitindo a cada run

#Bibliotecas:
import numpy as np
import pandas as pd
from qutip import *
import csv
import latex

import matplotlib.pyplot as plt
plt.rcParams.update(plt.rcParamsDefault)
plt.rcParams.update({
    "text.usetex": True,
    "font.family": "serif",
    "font.size": "18",
    "font.serif": ["Times New Roman"]})
import matplotlib
matplotlib.rc('xtick', labelsize=18)
matplotlib.rc('ytick', labelsize=18)
matplotlib.rc('legend', fontsize=14)
import matplotlib.gridspec as gridspec
from palettable.colorbrewer.qualitative import Set1_5
```

In [51]:
```
#Parâmetros (defino o que parace no Hamiltoniano):

Nat=1    #caso para 1 átomo
kappa = 2.0    #taxa de dissipação da cavidade
g = 0.0*kappa/np.sqrt(Nat)    #força de interação/constante de acopla
#frequência de Rabi do campo de controle (omegac > omegap para satis
#essa desigualdade não pode ser muito grande, ultrapassando a largura
epsilon = np.sqrt(0.1)*kappa    #1 fóton na cavidade com 1.0*kappa
omegac = 1.0*kappa
Delta1 = 0.0*kappa
Delta2 = 0.0*kappa    #ressonante (delta2 = 0)
DeltaP_list = np.linspace(-10,10,201)*kappa

gamma = 1.0    #taxa de dissipação do átomo (espontâneo)
gamma_13 = 0.5*gamma
gamma_23 = 0.5*gamma

N = 6    #número de estados Focks na cavidade (truncamento)

tlist = np.linspace(0,200,100)/kappa
```

In [52]:
```
#Estados ( |k> = (N,k-1) , N úmero total de estados):

state1 = basis(3,0)
state2 = basis(3,1)
state3 = basis(3,2)

#Operadores (S representa o sigma - sigma_nm = |n><m|
#- simgma_nm=tensor(qeye(N),staten*statem.dag())):

S11 = tensor(qeye(N),state1*state1.dag())
S22 = tensor(qeye(N),state2*state2.dag())
S31 = tensor(qeye(N),state3*state1.dag())
S13 = tensor(qeye(N),state1*state3.dag())
S32 = tensor(qeye(N),state3*state2.dag())
S23 = tensor(qeye(N),state2*state3.dag())
S33 = tensor(qeye(N),state3*state3.dag())

#operador de aniquilação dos fótons (destrói uma estação do campo)
a = tensor(destroy(N),qeye(3))


#Operadores de colapso:

c_ops_cav = np.sqrt(kappa)*a

c_ops_atom_13 = np.sqrt(gamma_13)*S13

c_ops_atom_23 = np.sqrt(gamma_23)*S23

#Dissipação:

c_ops_list = [c_ops_cav, c_ops_atom_13, c_ops_atom_23]
```

In [53]:
```
trans_ss_list = []

for i in range(len(DeltaP_list)):

    DeltaP = DeltaP_list[i]


    #Hamiltoniano:

    H = Delta1*S33 + (Delta1-Delta2)*S22 + DeltaP*S11 - DeltaP*a.dag(


    #Evolução temporal: equação mestra

    psi0 = tensor(basis(N,0), state2)

    output = mesolve(H,psi0,tlist,c_ops_list,[a.dag()*a])

    trans_ss = output.expect[0]

    trans_ss_list.append(trans_ss[-1]/((4*(epsilon**2)/kappa**2)))
```

In [38]:
```
#pré-plot da etapa de geração de dados para a cavidade vazia
plt.plot(DeltaP_list,trans_ss_list)
plt.show()
```

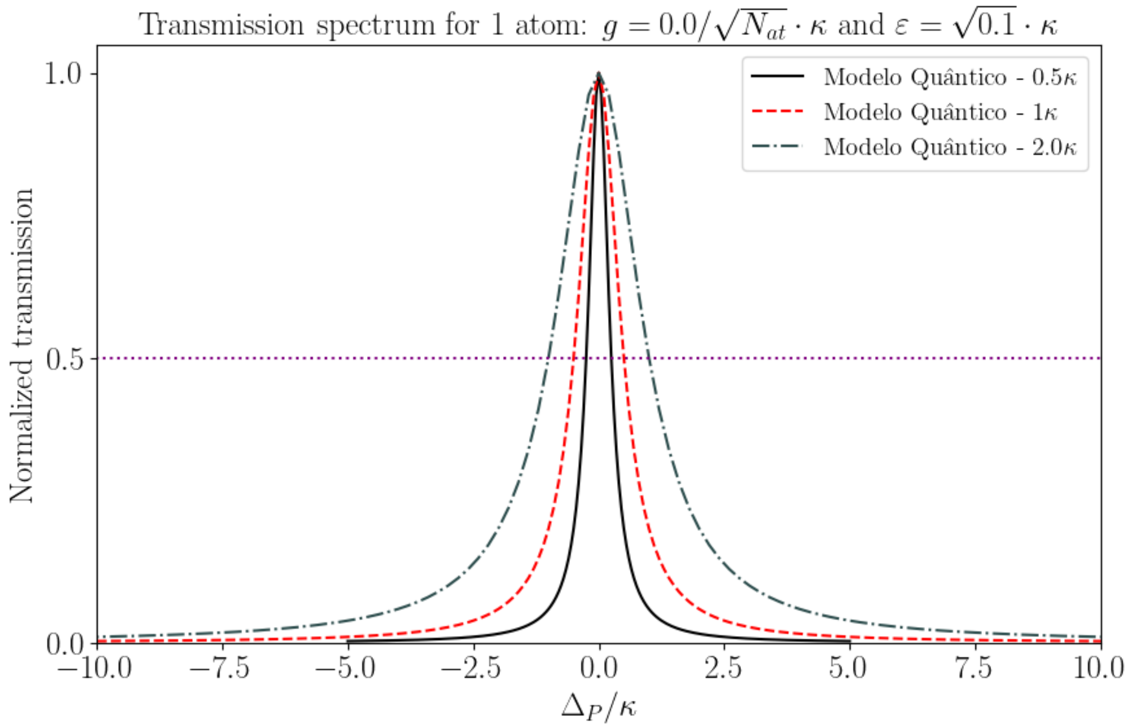
Out[38]:
```
[<matplotlib.lines.Line2D at 0x7fcae7f394f0>]
```

In [54]:
```
#Parâmetros a serem colocados e forma de salvar os arquivos:
#(Código feito para rodar três vezes)

#CEIT:
#omegac=2*kappa, gamma23=gamma13=0.5*kappa, psi0 = tensor(basis(N,0),
output_data = np.vstack((DeltaP_list,trans_ss_list))
file_data_store("CEIT_quantum_Nat1_g0_ep01_2kappa.csv",output_data.T,
#Átomo de dois níveis:
#omegac=0.0, gamma23=0.0, gamma13=0.5*kappa, psi0 = tensor(basis(N,0),
#output_data = np.vstack((DeltaP_list,trans_ss_list))
#file_data_store("atomo_2niveis.dat", output_data.T, numtype = "real"

#Cavidade vazia:
#omegac=0.0, gamma23=gamma13=0.5*kappa, psi0 = tensor(basis(N,0), sta
#output_data = np.vstack((DeltaP_list,trans_ss_list))
#file_data_store("cavidade_vazia.dat", output_data.T, numtype = "real
```

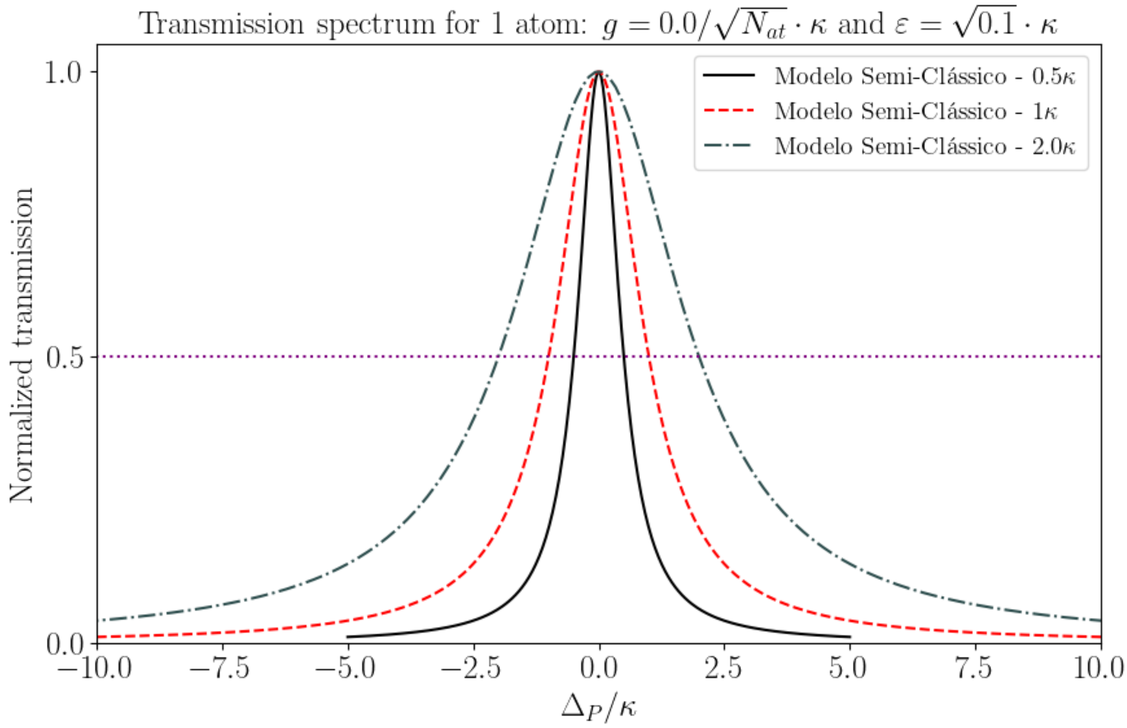
In [2]:
```
#Importando os arquivos:

CEIT_dados = np.loadtxt("CEIT.dat")
Delta_list = CEIT_dados[:,0]  #toma as linhas da primeira coluna
CEIT_list = CEIT_dados[:,1]   ##toma as linhas da segunda coluna

dois_niveis_dados = np.loadtxt("atomo_2niveis.dat")
dois_niveis_list = dois_niveis_dados[:,1]

cavidade_vazia_dados = np.loadtxt("cavidade_vazia.dat")
cavidade_vazia_list = cavidade_vazia_dados[:,1]

M1 = np.loadtxt('CEIT_semiclassical_Nat1_g0_ep01_05kappa.csv', delimi
M2 = np.loadtxt('CEIT_semiclassical_Nat1_g0_ep01_1kappa.csv', delimit
M3 = np.loadtxt('CEIT_semiclassical_Nat1_g0_ep01_2kappa.csv', delimit
M4 = np.loadtxt('CEIT_semiclassical_Nat3_g0_ep01.csv', delimiter=',')
M5 = np.loadtxt('CEIT_semiclassical_Nat1_g01_ep01_05kappa.csv', delin
M6 = np.loadtxt('CEIT_semiclassical_Nat1_g1_ep01_05kappa.csv', delimi
M7 = np.loadtxt('CEIT_semiclassical_Nat1_g5_ep01_05kappa.csv', delimi
M8 = np.loadtxt('CEIT_semiclassical_Nat1_g10_ep01_05kappa.csv', delin

N1 = np.loadtxt('CEIT_quantum_Nat1_g0_ep01_05kappa.csv', delimiter=',
N2 = np.loadtxt('CEIT_quantum_Nat1_g0_ep01_1kappa.csv', delimiter=',
N3 = np.loadtxt('CEIT_quantum_Nat1_g0_ep01_2kappa.csv', delimiter=',
N4 = np.loadtxt('CEIT_quantum_Nat1_g01_ep01_1kappa.csv', delimiter=',
N5 = np.loadtxt('CEIT_quantum_Nat1_g1_ep01_1kappa.csv', delimiter=',
N6 = np.loadtxt('CEIT_quantum_Nat1_g5_ep01_1kappa.csv', delimiter=',
N7 = np.loadtxt('CEIT_quantum_Nat1_g10_ep01_1kappa.csv', delimiter=',


#test: multiplicando todos os parâmetros por 2 (2*kappa)
O1 = np.loadtxt('CEIT_semiclassical_Nat1_g0_ep01_2vz05kappa.csv', del
O2 = np.loadtxt('CEIT_semiclassical_Nat1_g01_ep01_2vz05kappa.csv', de
O3 = np.loadtxt('CEIT_semiclassical_Nat1_g1_ep01_2vz05kappa.csv', del
O4 = np.loadtxt('CEIT_semiclassical_Nat1_g5_ep01_2vz05kappa.csv', del
O5 = np.loadtxt('CEIT_semiclassical_Nat1_g10_ep01_2vz05kappa.csv', de

#test: Oc=2*kappa
P1 = np.loadtxt('CEIT_semiclassical_Nat1_g0_ep01_1Oc_05kappa.csv', de
P2 = np.loadtxt('CEIT_semiclassical_Nat1_g1_ep01_1Oc_05kappa.csv', de
P3 = np.loadtxt('CEIT_semiclassical_Nat1_g1_ep01_1Oc_05kappa.csv', de
P4 = np.loadtxt('CEIT_semiclassical_Nat1_g5_ep01_1Oc_05kappa.csv', de
P5 = np.loadtxt('CEIT_semiclassical_Nat1_g10_ep01_1Oc_05kappa.csv', d

#Integração:
I1 = np.loadtxt('semiclassical_int_Nat1_g01_ep01_1kappa.csv', delimit
I2 = np.loadtxt('semiclassical_int_Nat1_g1_ep01_1kappa.csv', delimite
I3 = np.loadtxt('semiclassical_int_Nat1_g5_ep01_1kappa.csv', delimite
I4 = np.loadtxt('semiclassical_int_Nat1_g10_ep01_1kappa.csv', delimit
I5 = np.loadtxt('semiclassical_int_Nat1_g01_ep01_2vsparameters_05kapp
I6 = np.loadtxt('semiclassical_int_Nat1_g1_ep01_2vsparameters_05kappa
I7 = np.loadtxt('semiclassical_int_Nat1_g5_ep01_2vsparameters_05kappa
I8 = np.loadtxt('semiclassical_int_Nat1_g10_ep01_2vsparameters_05kapp
I9 = np.loadtxt('semiclassical_int_Nat1_g01_ep01_qutipdef.csv', delin
I10 = np.loadtxt('semiclassical_int_Nat1_g1_ep01_qutipdef.csv', delin
I11 = np.loadtxt('semiclassical_int_Nat1_g5_ep01_qutipdef.csv', delin
I12 = np.loadtxt('semiclassical_int_Nat1_g10_ep01_qutipdef.csv', deli
```

In [15]:

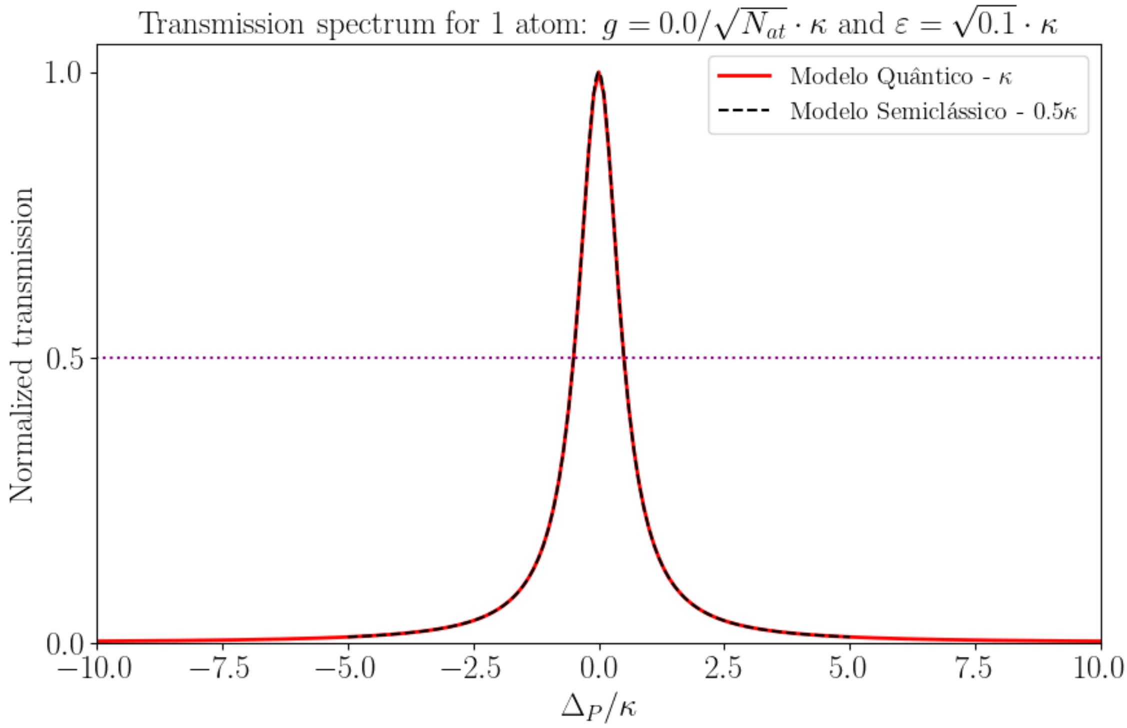
```
#Espectro de transmissão: Modelo Quântico variando kappa

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N1[:,0], N1[:,1], color = "k", linestyle = "-", linewidth =
ax1.plot(N2[:,0], N2[:,1], color = "red", linestyle = "--", linewidth
ax1.plot(N3[:,0], N3[:,1], color = "darkslategray", linestyle = "-.",
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["Modelo Quântico - $0.5\kappa$", "Modelo Quântico - $1\ka
plt.title(r'Transmission spectrum for 1 atom: $g=0.0/\sqrt{N_{at}}\co
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N1[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_quantum_allkappa.png",format='png', dpi
plt.show()
```



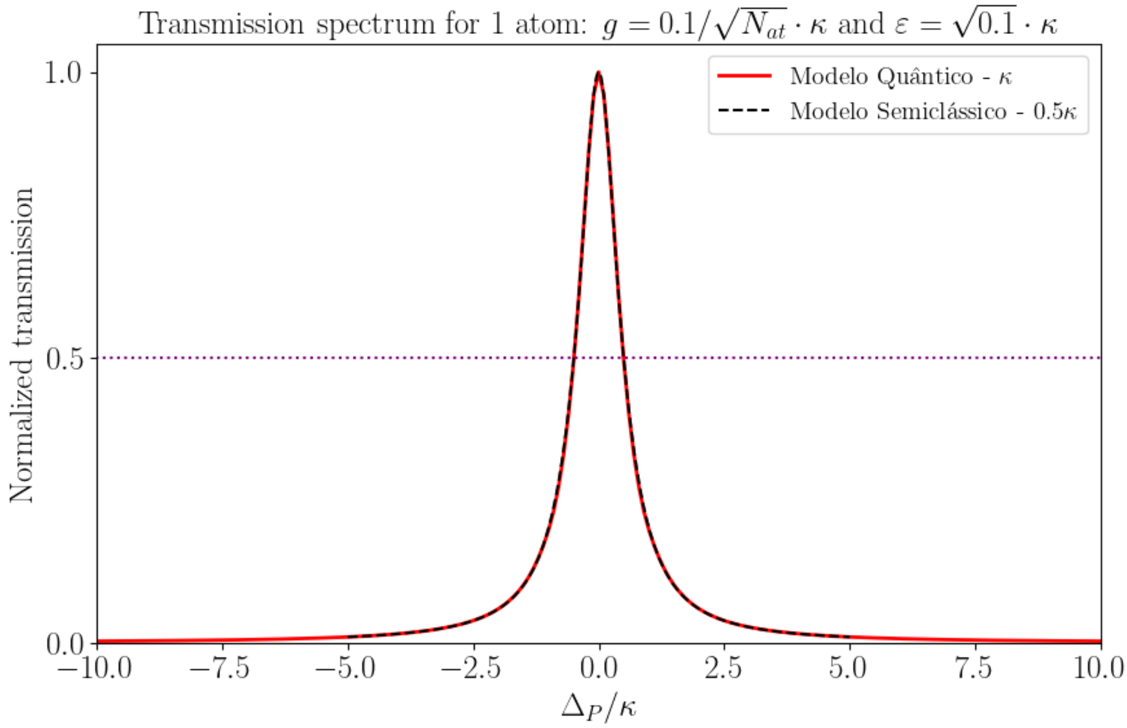
In [58]:
```
#Espectro de transmissão: Modelo Semi-clássico variando kappa

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(M1[:,0], M1[:,1], color = "k", linestyle = "-", linewidth =
ax1.plot(M2[:,0], M2[:,1], color = "red", linestyle = "--", linewidth
ax1.plot(M3[:,0], M3[:,1], color = "darkslategray", linestyle = "-.",
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
#ax1.legend(["Cavity-EIT", "Two-level atom", "Empty cavity"])
ax1.legend(["Modelo Semi-Clássico  - $0.5\kappa$", "Modelo Semi-Cláss
plt.title(r'Transmission spectrum for 1 atom: $g=0.0/\sqrt{N_{at}}\c
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(M1[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_semiclassical_allkappa.png",format='png
plt.show()
```



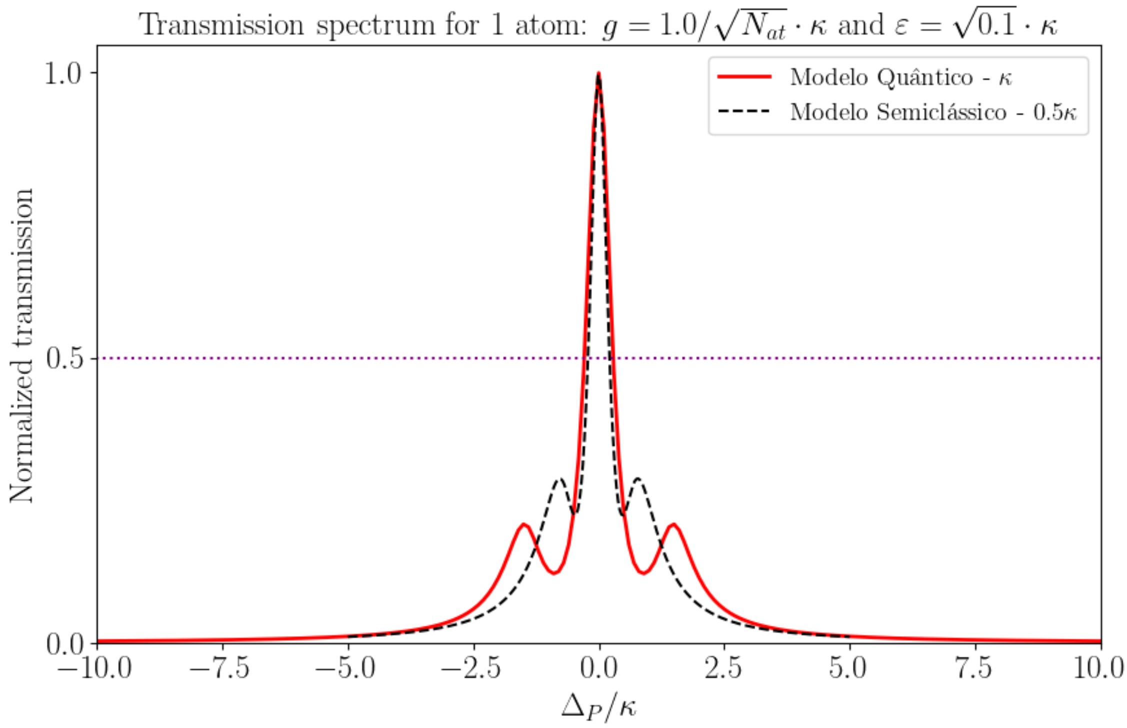
In [13]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=0

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N2[:,0],N2[:,1], color = "red", linewidth = 2)
ax1.plot(M1[:,0], M1[:,1], color = "k", linestyle = "--", linewidth =
#ax1.plot(M2[:,0], M2[:,1], color = "darkslategray", linestyle = "--'
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
#ax1.legend(["Cavity-EIT", "Two-level atom", "Empty cavity"])
ax1.legend(["Modelo Quântico - $\kappa$", "Modelo Semiclássico - $0.5
plt.title(r'Transmission spectrum for 1 atom: $g=0.0/\sqrt{N_{at}}\cd
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N2[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g0_ep01_compare_QSC.png",format='p
plt.show()
```



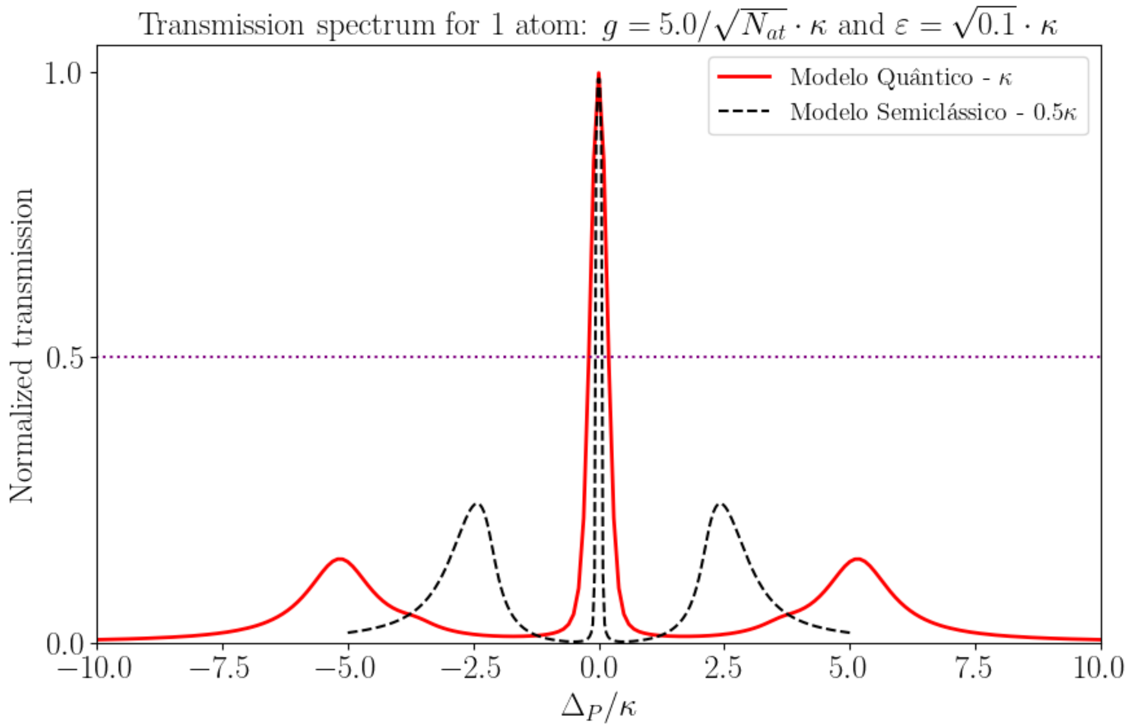
In [14]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=0.1

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N4[:,0],N4[:,1], color = "red", linewidth = 2)
ax1.plot(M5[:,0], M5[:,1], color = "k", linestyle = "--", linewidth =
#ax1.plot(M2[:,0], M2[:,1], color = "darkslategray", linestyle = "--'
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
#ax1.legend(["Cavity-EIT", "Two-level atom", "Empty cavity"])
ax1.legend(["Modelo Quântico - $\kappa$", "Modelo Semiclássico - $0.5
plt.title(r'Transmission spectrum for 1 atom: $g=0.1/\sqrt{N_{at}}\cd
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N4[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g01_ep01_compare_QSC.png",format=
plt.show()
```



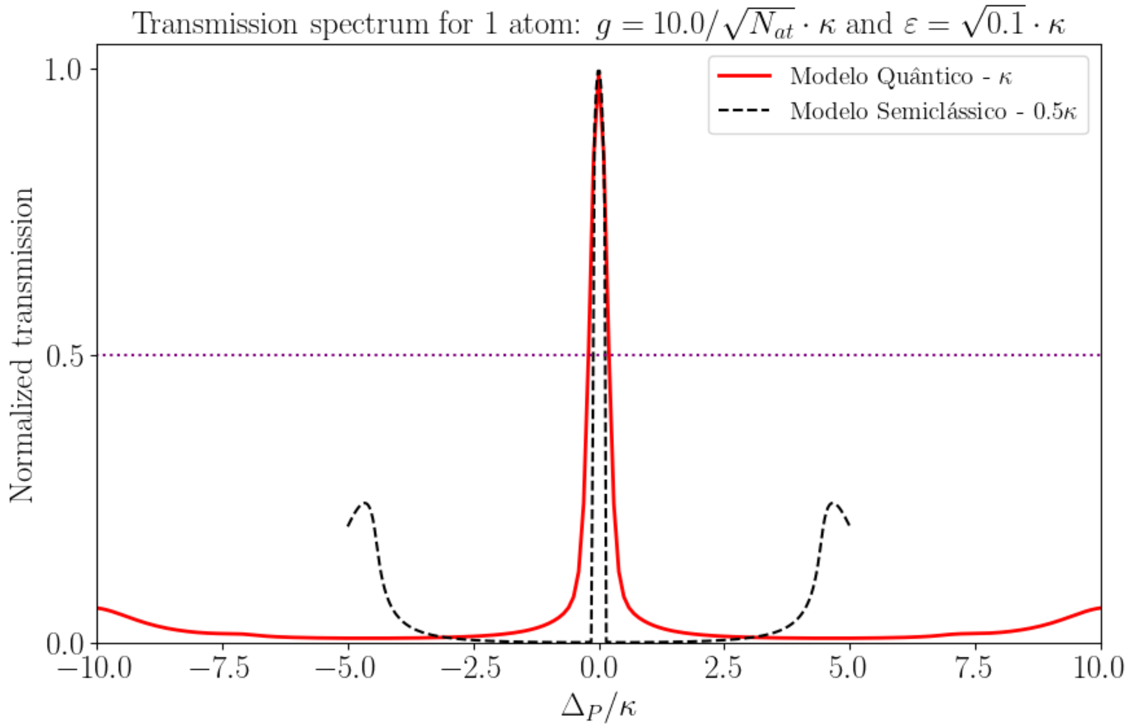
In [16]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=1.0

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N5[:,0],N5[:,1], color = "red", linewidth = 2)
ax1.plot(M6[:,0], M6[:,1], color = "k", linestyle = "--", linewidth =
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
#ax1.legend(["Cavity-EIT", "Two-level atom", "Empty cavity"])
ax1.legend(["Modelo Quântico - $\kappa$", "Modelo Semiclássico - $0.5
plt.title(r'Transmission spectrum for 1 atom: $g=1.0/\sqrt{N_{at}}\cd
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N5[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g1_ep01_compare_QSC.png",format='p
plt.show()
```



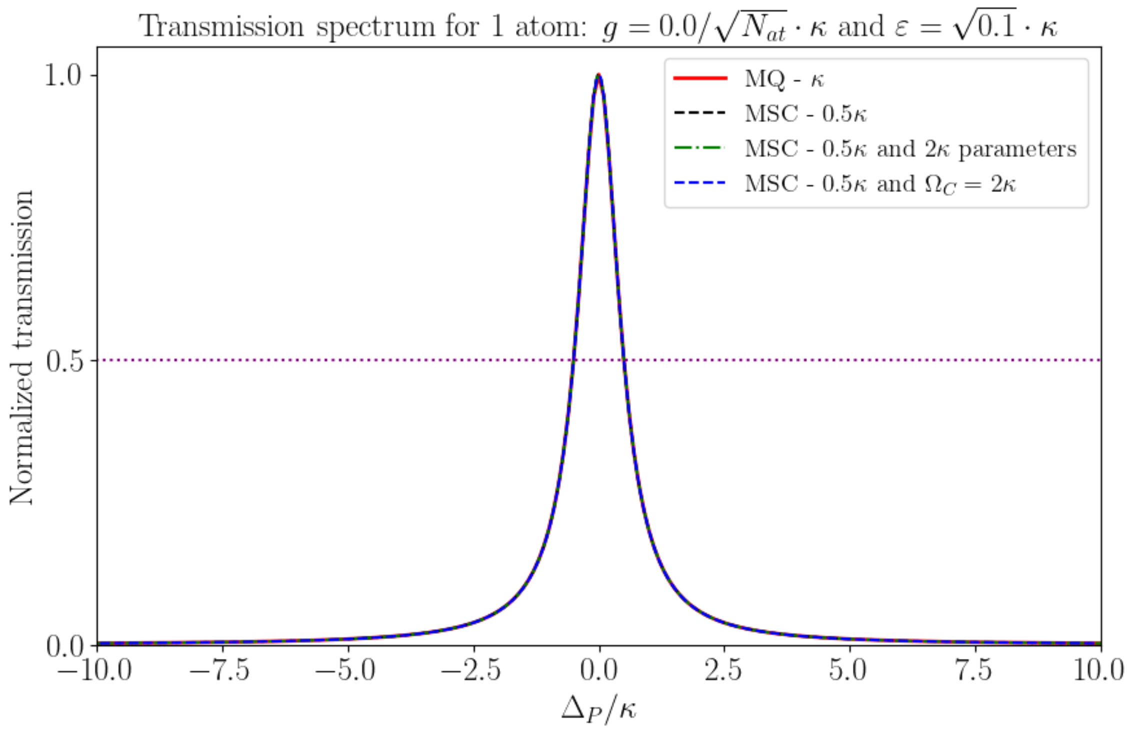
In [17]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=5.0

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N6[:,0],N6[:,1], color = "red", linewidth = 2)
ax1.plot(M7[:,0], M7[:,1], color = "k", linestyle = "--", linewidth =
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
#ax1.legend(["Cavity-EIT", "Two-level atom", "Empty cavity"])
ax1.legend(["Modelo Quântico - $\kappa$", "Modelo Semiclássico - $0.5
plt.title(r'Transmission spectrum for 1 atom: $g=5.0/\sqrt{N_{at}}\cd
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N6[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g5_ep01_compare_QSC.png",format='p
plt.show()
```



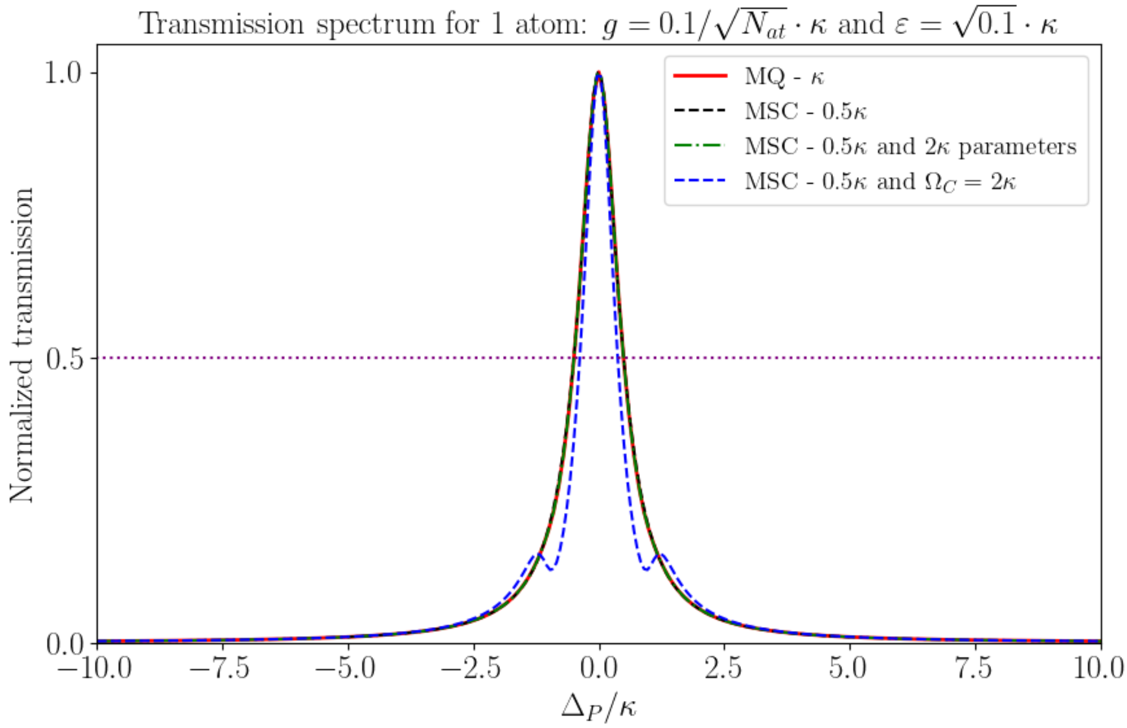
In [18]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=10.0

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N7[:,0],N7[:,1], color = "red", linewidth = 2)
ax1.plot(M8[:,0], M8[:,1], color = "k", linestyle = "--", linewidth =
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
#ax1.legend(["Cavity-EIT", "Two-level atom", "Empty cavity"])
ax1.legend(["Modelo Quântico - $\kappa$", "Modelo Semiclássico - $0.5
plt.title(r'Transmission spectrum for 1 atom: $g=10.0/\sqrt{N_{at}}\c
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N7[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g10_ep01_compare_QSC.png",format=
plt.show()
```



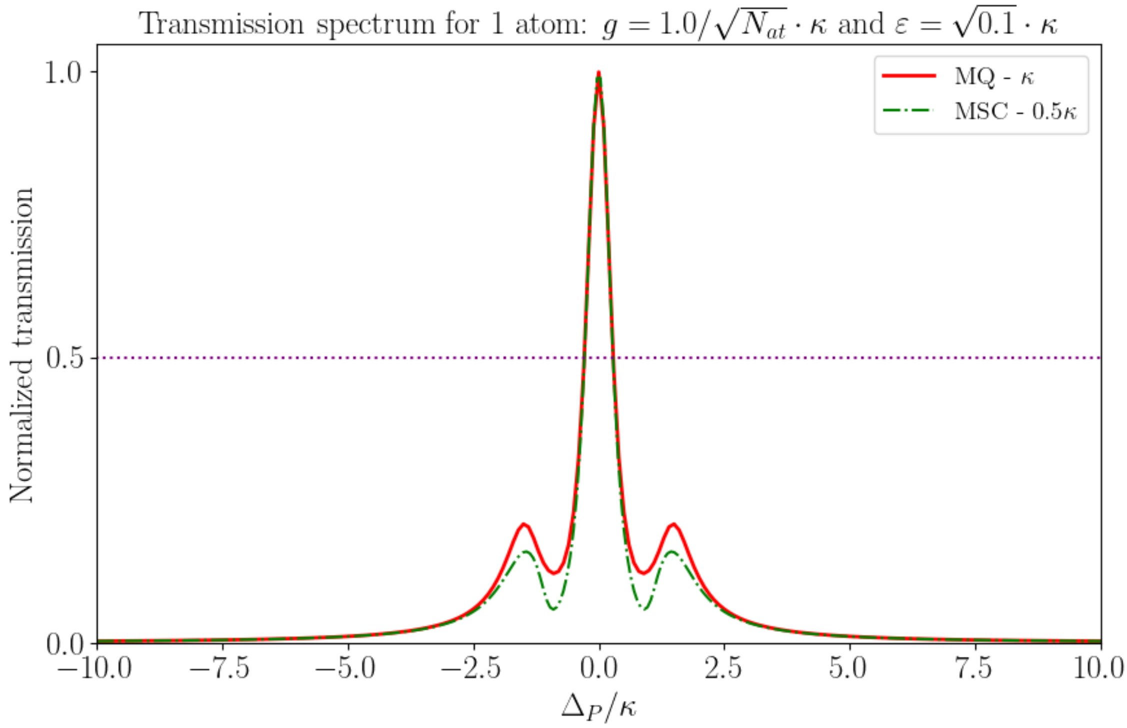
In [6]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=0 e teste pa

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N2[:,0],N2[:,1], color = "red",linestyle = "-", linewidth =
ax1.plot(M1[:,0], M1[:,1], color = "k", linestyle = "--", linewidth =
ax1.plot(O1[:,0], O1[:,1], color = "green", linestyle = "-.", linewic
ax1.plot(P1[:,0], P1[:,1], color = "blue", linestyle = "--", linewidt
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC - $0.5\kappa$","MSC - $0.5\kappa$ a
plt.title(r'Transmission spectrum for 1 atom: $g=0.0/\sqrt{N_{at}}\cc
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N2[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g0_ep01_compare_QSC_tests.png",fo
plt.show()
```



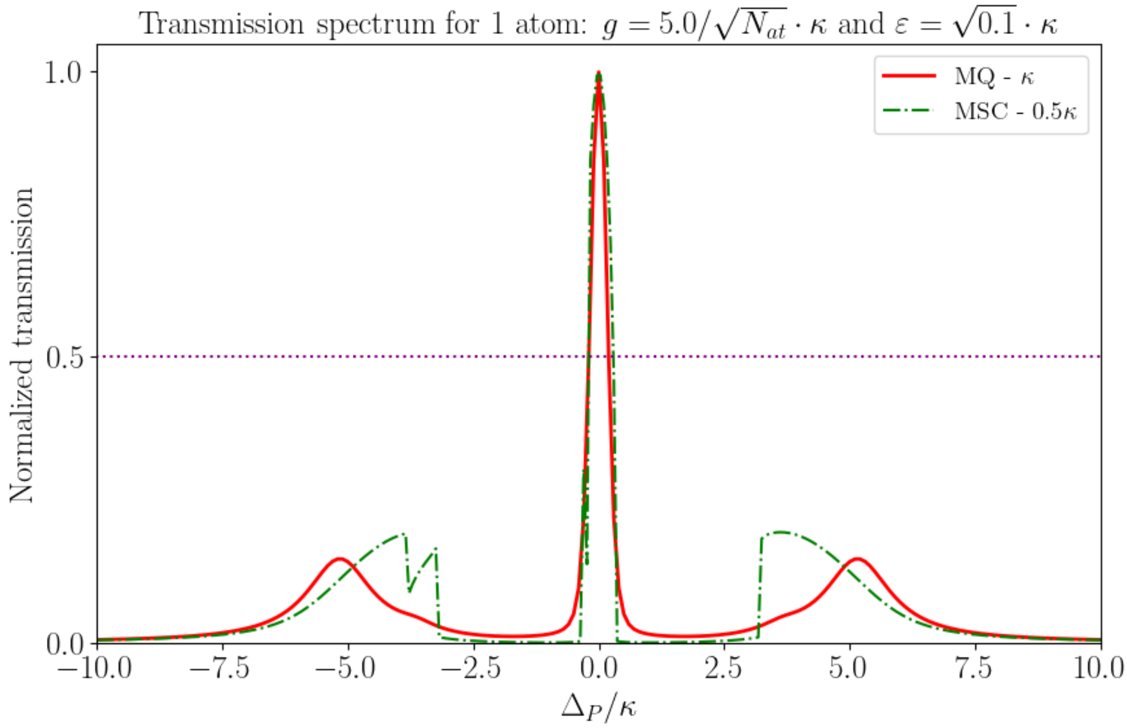
In [7]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=0.1 e teste

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N4[:,0],N4[:,1], color = "red",linestyle = "-", linewidth =
ax1.plot(M5[:,0], M5[:,1], color = "k", linestyle = "--", linewidth =
ax1.plot(O2[:,0], O2[:,1], color = "green", linestyle = "-.", linewic
ax1.plot(P2[:,0], P2[:,1], color = "blue", linestyle = "--", linewidt
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC - $0.5\kappa$","MSC - $0.5\kappa$ a
plt.title(r'Transmission spectrum for 1 atom: $g=0.1/\sqrt{N_{at}}\cc
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N4[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g01_ep01_compare_QSC_tests.png",fc
plt.show()
```



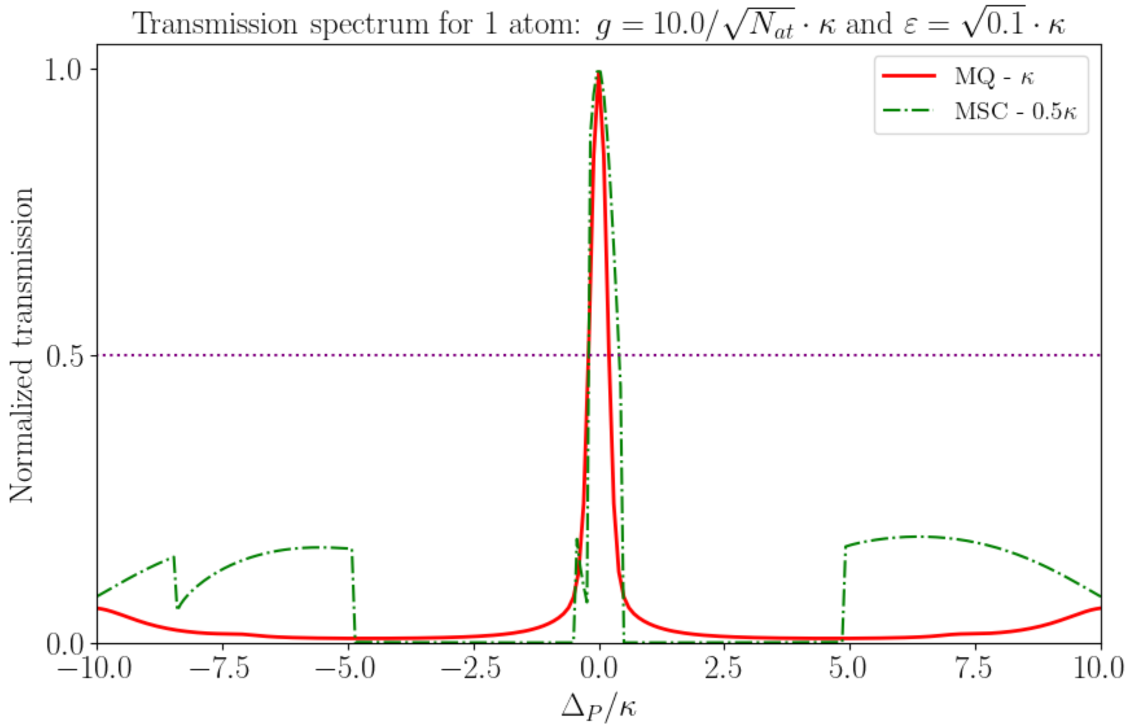
In [4]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=1.0 e teste

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N5[:,0],N5[:,1], color = "red",linestyle = "-", linewidth =
#ax1.plot(M6[:,0], M6[:,1], color = "k", linestyle = "--", linewidth
ax1.plot(O3[:,0], O3[:,1], color = "green", linestyle = "-.", linewic
#ax1.plot(P3[:,0], P3[:,1], color = "blue", linestyle = "--", linewic
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC - $0.5\kappa$","MSC - $0.5\kappa$ a
plt.title(r'Transmission spectrum for 1 atom: $g=1.0/\sqrt{N_{at}}\cc
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N5[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g1_ep01_compare_QSC_tests.png",foi
plt.show()
```



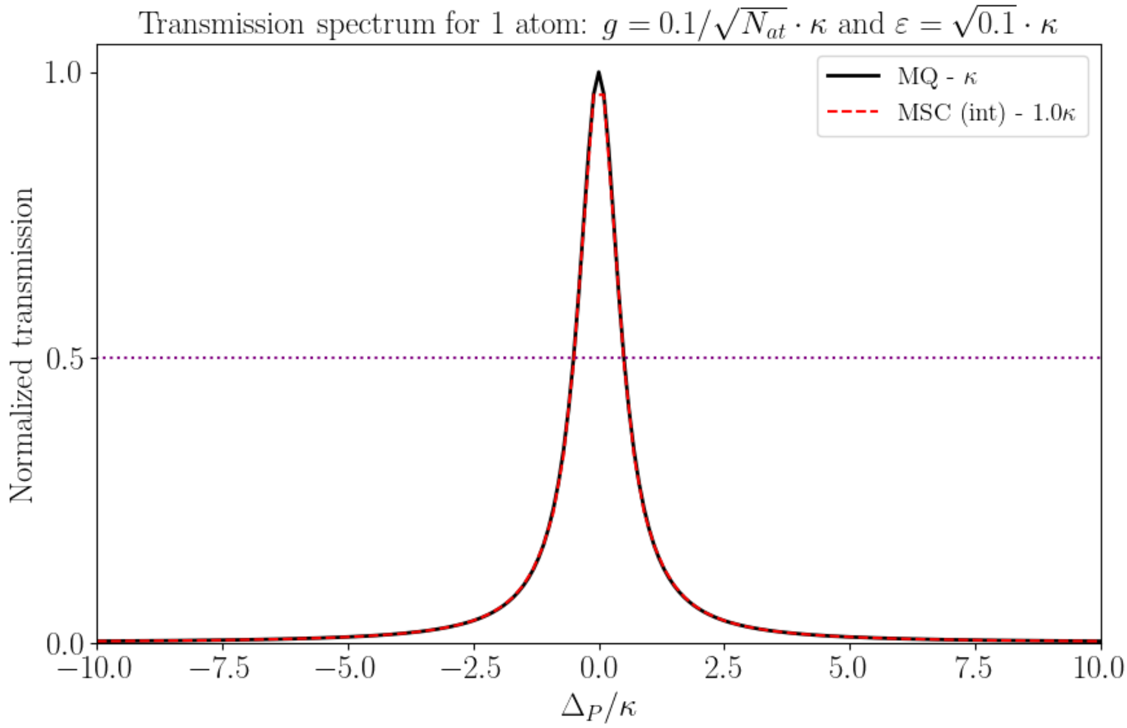
In [5]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=5.0 e teste

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N6[:,0],N6[:,1], color = "red",linestyle = "-", linewidth =
#ax1.plot(M7[:,0], M7[:,1], color = "k", linestyle = "--", linewidth
ax1.plot(O4[:,0], O4[:,1], color = "green", linestyle = "-.", linewic
#ax1.plot(P4[:,0], P4[:,1], color = "blue", linestyle = "--", linewic
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC - $0.5\kappa$","MSC - $0.5\kappa$ a
plt.title(r'Transmission spectrum for 1 atom: $g=5.0/\sqrt{N_{at}}\cc
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N6[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g5_ep01_compare_QSC_tests.png",fo
plt.show()
```



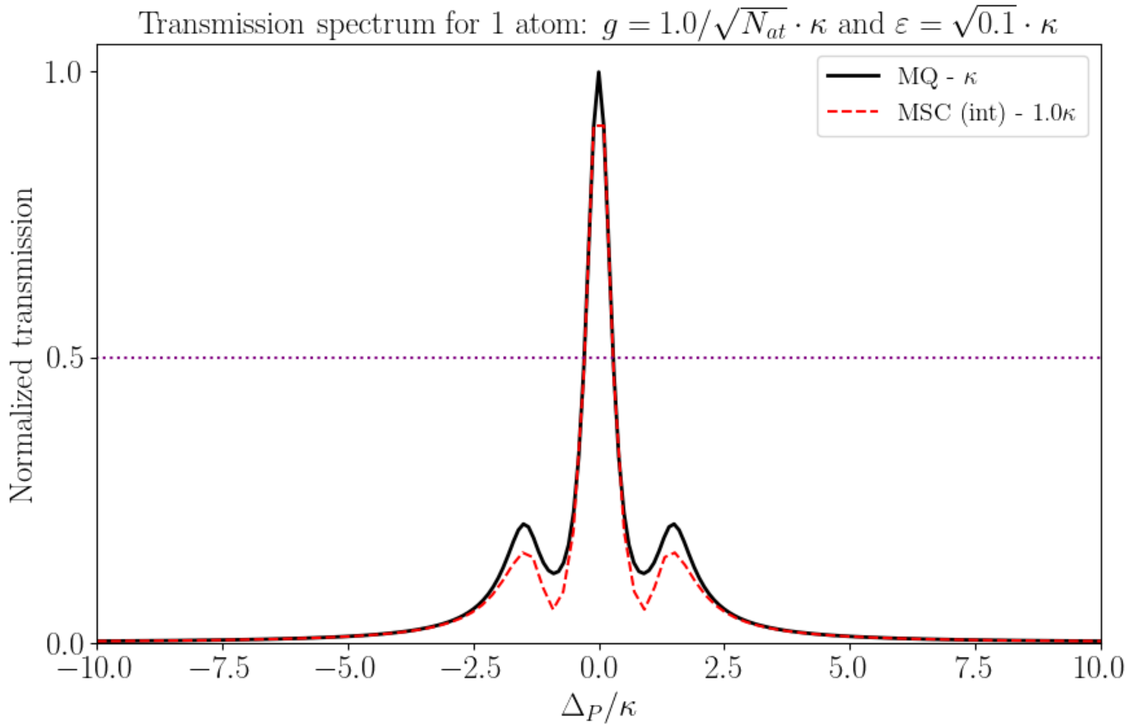
In [6]:
```
#Comparando espectros de cada modelo pelo acoplamento: g=10.0 e teste

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N7[:,0],N7[:,1], color = "red",linestyle = "-", linewidth =
#ax1.plot(M8[:,0], M8[:,1], color = "k", linestyle = "--", linewidth
ax1.plot(O5[:,0], O5[:,1], color = "green", linestyle = "-.", linewic
#ax1.plot(P5[:,0], P5[:,1], color = "blue", linestyle = "--", linewic
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC - $0.5\kappa$","MSC - $0.5\kappa$ a
plt.title(r'Transmission spectrum for 1 atom: $g=10.0/\sqrt{N_{at}}\c
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N7[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g10_ep01_compare_QSC_tests.png",fc
plt.show()
```



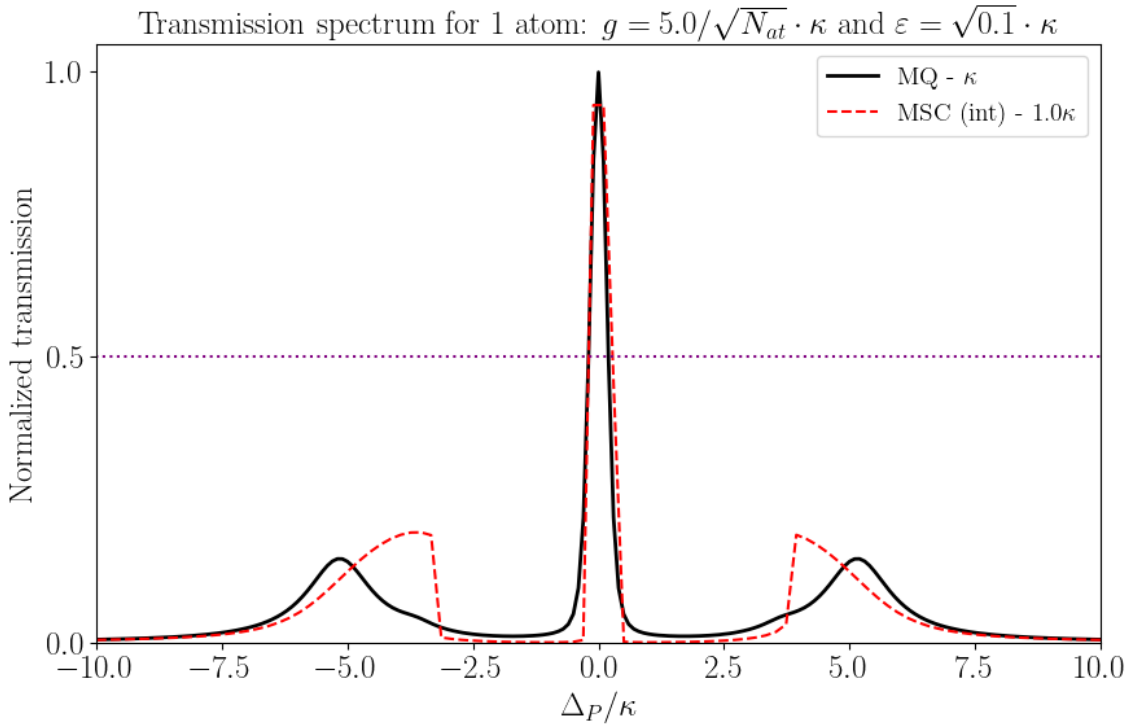
In [15]:
```
#Integração em SC: comparando espectros de cada modelo pelo acoplamen

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N2[:,0],N2[:,1], color = "k", linewidth = 2)
#ax1.plot(I1[:,0], I1[:,1], color = "k", linestyle = "--", linewidth
#ax1.plot(I5[:,0], I5[:,1], color = "b", linestyle = "-.", linewidth
ax1.plot(I9[:,0], I9[:,1], color = "r", linestyle = "--", linewidth =
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC (int) - $1.0\kappa$","MSC (int) - $
plt.title(r'Transmission spectrum for 1 atom: $g=0.1/\sqrt{N_{at}}\co
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N2[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g01_ep01_compare_int_QSC.png",forn
plt.show()
```



In [16]:
```
#Integração em SC: comparando espectros de cada modelo pelo acoplamen

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N5[:,0],N5[:,1], color = "k", linewidth = 2)
#ax1.plot(I2[:,0], I2[:,1], color = "k", linestyle = "--", linewidth
#ax1.plot(I6[:,0], I6[:,1], color = "b", linestyle = "-.", linewidth
ax1.plot(I10[:,0], I10[:,1], color = "r", linestyle = "--", linewidth
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC (int) - $1.0\kappa$","MSC (int) - $
plt.title(r'Transmission spectrum for 1 atom: $g=1.0/\sqrt{N_{at}}\cd
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N5[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g1_ep01_compare_int_QSC.png",forma
plt.show()
```



In [17]:
```
#Integração em SC: comparando espectros de cada modelo pelo acoplamen

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N6[:,0],N6[:,1], color = "k", linewidth = 2)
#ax1.plot(I3[:,0], I3[:,1], color = "k", linestyle = "--", linewidth
#ax1.plot(I7[:,0], I7[:,1], color = "b", linestyle = "-", linewidth =
ax1.plot(I11[:,0], I11[:,1], color = "r", linestyle = "--", linewidth
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC (int) - $1.0\kappa$","MSC (int) - $
plt.title(r'Transmission spectrum for 1 atom: $g=5.0/\sqrt{N_{at}}\cd
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N6[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g5_ep01_compare_int_QSC.png",forma
plt.show()
```



In [18]:
```
#Integração em SC: comparando espectros de cada modelo pelo acoplamen

fig, ax1 = plt.subplots(figsize=(10, 6))
ax1.plot(N7[:,0],N7[:,1], color = "k", linewidth = 2)
#ax1.plot(I4[:,0], I4[:,1], color = "k", linestyle = "--", linewidth
#ax1.plot(I8[:,0], I8[:,1], color = "y", linestyle = "-", linewidth =
ax1.plot(I12[:,0], I12[:,1], color = "r", linestyle = "--", linewidth
ax1.set_xlabel("$\Delta_P/\kappa$")
ax1.set_ylabel("Normalized transmission")
ax1.legend(["MQ - $\kappa$", "MSC (int) - $1.0\kappa$","MSC (int) - $
plt.title(r'Transmission spectrum for 1 atom: $g=10.0/\sqrt{N_{at}}\c
plt.xlim(-10,10)
plt.ylim(0, 1.05*np.max(N7[:,1]))
plt.axhline(y=0.5, xmin=-10.0, xmax=10.0, color="purple", linestyle =
plt.yticks([0,0.5,1.0])
plt.savefig("EIT_transmission_Nat1_g10_ep01_compare_int_QSC.png",form
plt.show()
```