```
# construct two point correlation matrix
def construct_CM(U,L):
    m1=np.array([[1,0],[0,0]])
    smat=np.kron(m1,np.identity(int(L/2)))
    CM = np.dot(U,np.dot(smat,U.transpose()))
    return CM

# calculate LE using |det(1-C+C*exp(-iHt))|
def calc_detLE(v,U,CM,t):
    LE=np.zeros(len(t))
    for i in t:
        Ut = construct_U(v,U,i)
        k=t.tolist().index(i)
        LE[k]=np.abs(np.linalg.det(np.identity(args.L)-CM+np.dot(CM,Ut)))
    return LE

# Run the program for both cases
t=np.arange(args.tint,args.tmax+args.dt/2,args.dt)

# calculate single-particle Hamiltonian
SPHi = construct_SPH(args.delta,args.L,args.openbc)
SPHf = construct_SPH(-args.delta,args.L,args.openbc)

Store1=0
for samp in range(int(args.sample)):
    APDW = construct_APDW(args.L,args.W)
    SPHiW = SPHi + APDW
    SPHfW = SPHf + APDW
    vsi,Usi = np.linalg.eigh(SPHiW)
    vsf,Usf = np.linalg.eigh(SPHfW)
    CM = construct_CM(Usi,args.L)
    Store1 += calc_detLE(vsf,Usf,CM,t)

LE1=np.squeeze(Store1/args.sample)
RR1=-2*np.log(LE1)/args.L # return rate

for item in RR1:
    print(item)
```