Exercise 9

I denne oppgaven skal vi se på HeH^+ i minimal basis {1s(H),1s(He)}. Lag et program som minimerer energien med hensyn til molekylorbital (MO) koeffisientene C, dvs. løs følgende ligning iterativt:

$$\mathbf{FC} = \mathbf{SC}\boldsymbol{\epsilon}$$

I kodecellen under blir en-elektron integral matrisen ($\mathbf{h}$) og atomorbital (AO) ovelapp matrisen ($\mathbf{S}$) definert. Videre har vi definert en funksjon (get_two_electron_matrix) som kan brukes til å finne to-elektron bidraget ($\mathbf{G}(\mathbf{C})$) og elektrontettheten ($\mathbf{D}$), der MO koeffisientene må sendes inn som en 2x2 matrise ($\mathbf{C}$) og antall okkuperte orbitaler (n) må sendes inn. En funksjon (get_gradient_norm) som beregner normen til gradienten er også definert, der inputet er Fock matrisen ($\mathbf{F}$), elektrontettheten ($\mathbf{D}$) og AO overlapp matrisen ($\mathbf{S}$). Gradientnormen sier noe om hvor forskjelling MOer i den nye iterasjonen er fra MOer i forrige iterasjon.

In [21]:
```
from scipy.linalg import eigh
import numpy as np
from numpy import linalg as LA

# Defines the integrals between basis functions in
# the h, s and g matrix. units of energy quantities are a.u.
def get_hhe_integrals():
    h = np.array([[-1.73467046, -1.43701378], [-1.43701378, -2.60005224]])
    s = np.array([[1.00000000, 0.53841475], [0.53841475, 1.00000000]])
    g = np.array([[[[ 1.54921186,  0.73768334],
                    [ 0.73768334,  1.18373171]],

                   [[ 0.73768334,  0.45159774],
                    [ 0.45159774,  0.89116906]]],


                  [[[ 0.73768334,  0.45159774],
                    [ 0.45159774,  0.89116906]],

                   [[ 1.18373171,  0.89116906],
                    [ 0.89116906,  2.11142590]]]])
    nuclear_repulsion = 1.370925410104
    return h, s, g, nuclear_repulsion

# Function to calculate G[D].
# Input c = 2x2 matrix containing MO-coefficients
# Input n = number of occupied orbitals
def get_two_electron_matrix(c,n):
    if n != 1:
        print('Are you sure the number of occupied orbitals is {}'.format(n))
        return None

    d = density(c)
    gd=np.zeros((2,2))
    for i in range(2):
        for j in range(2):
            for k in range(2):
                for l in range(2):
                    gd[i,j] += (g[i,j,k,l]-0.5*g[i,l,k,j])*d[k,l]
    return gd

# AO density D
def density(c):
    d = np.zeros((2,2))
    for i in range(2):
        for j in range(2):
            d[i,j] = c[i,0]*c[j,0]
    return d

# Calculates the gradient  |FDS - SDF|
def get_gradient_norm(F,d,s):
    gradient = LA.norm(F@d@s-s@d@F)
    return gradient
```

Nyttig informasjon for å løse oppgaven:

Fock matrisen er gitt ved

$$\mathbf{F} = \mathbf{h} + \mathbf{G}(\mathbf{C})$$

Energien kan beregnes som

$$E = 2\mathrm{Tr}\mathbf{hD} + \mathrm{Tr}\mathbf{DG}(\mathbf{C}) + \mathrm{kjernerepulsjon}$$

eller som

$$E = 2\mathrm{Tr}\mathbf{FD} - \mathrm{Tr}\mathbf{DG}(\mathbf{C}) + \mathrm{kjernerepulsjon}$$

Roothan-Hall ligningen $\mathbf{FC} = \mathbf{SC}\epsilon$ er et generalisert egenverdiproblem. Scipy funksjonen eigh kan være til hjelp.

Nyttig informasjon for å forstå oppgaven:

Matriseelementene i en-elektron operatoren er gitt ved (i a.u.)

$$h_{\mu\nu} = \langle \chi_\mu | -\frac{1}{2}\nabla^2 - \sum_I \frac{Z_I}{r_I} | \chi_\nu \rangle$$

der $-\frac{1}{2}\nabla^2$ er operator for kinetisk energi for et elektron og $-\sum_I \frac{Z_I}{r_I}$ er tiltrekking mellom elektrona og kjernene (summen over I er en sum over de to atomkjernene i HeH+).

To-elektron matrisa $\mathbf{G}(\mathbf{C})$ inneholder informasjon om elektron-elektron frastøyting (integral over operatoren $\frac{1}{r_{12}}$)

Dere skal bruke programmet dere lager til å svare på følgende spørsmål:

In [22]:
```
h, s, g, nuclear_repulsion = get_hhe_integrals()
#Finner første forslag til c ved å se på koeffisientene til mulige lineærkombi
nasjoner av de to 1s orbitalene
c_guess = np.array([[1/np.sqrt(2), 1/np.sqrt(2)], [1/np.sqrt(2), -1/np.sqrt(2
)]])

def solve_it():
    F = h + get_two_electron_matrix(c_guess,1)
    d = density(c_guess)
    while get_gradient_norm(F,d,s)>1e-10:
        epsilon, c = eigh(F,s)
        d = density(c)
        F = h + get_two_electron_matrix(c,1)
    return F, c, d, epsilon

def get_energy():
    F, c, d, epsilon = solve_it()
    E = 2*np.trace(np.dot(h,d)) + np.trace(np.dot(d,get_two_electron_matrix(c,
1))) + nuclear_repulsion
    return E
```

1) Hva er den elektroniske energien til systemet (10+ desimaler)?

In [23]:
```
# Definer energien i variabelen electronic_energy med tall (E = -2.8...)
electronic_energy = get_energy()
# YOUR CODE HERE
print(electronic_energy)
```

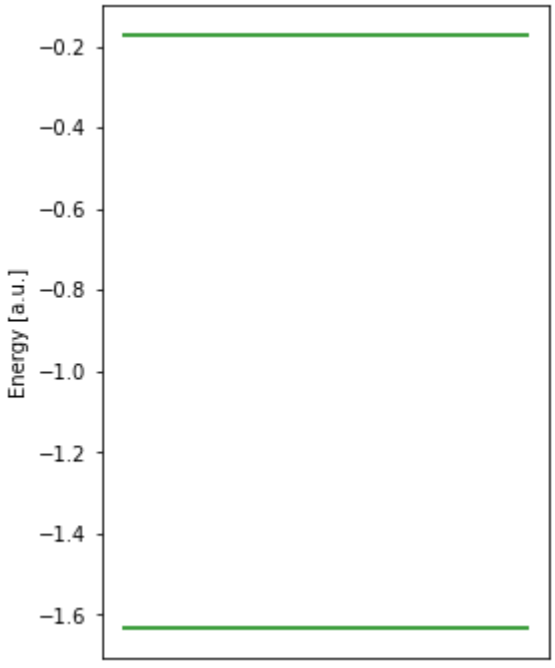
```
-2.841382482410755
```

In [24]:
```
# Test: Sjekker electronic_energy mot fasit til 10s desimaler
assert(abs(electronic_energy - -2.841382482410756) < 1e-10)
```

2) Hva er orbitalenergiene (6+ desimaler)? Skisser et MO diagram

In [25]:
```
import matplotlib.pyplot as plt
# Definer orbital energiene med tall (orbital_energies = [-1.7..., -0.4...])
F, c, d, epsilon = solve_it()
orbital_energies = epsilon

fig = plt.figure(figsize=(4, 6))
plt.ylabel('Energy [a.u.]')
plt.hlines(epsilon, -1, 1, color = 'green')
plt.xticks([])
plt.show()
# YOUR CODE HERE
```



In [26]:
```
# Hemmelig test: Sjekker orbital_energies mot fasit til 6 desimaler
```

3) Er HeH^+ stabilt i følge beregningen?

In [27]:
```
# Svar enten heh_stabil=True eller heh_stabil=False
heh_stabil = True
print("To elektroner i bindende orbital (lavest i energi).")
```

```
To elektroner i bindende orbital (lavest i energi).
```

In [28]:
```
# Hemmelig test: Sjekker heh_stabil mot fasit
```

4) Bruk C til å skrive opp lineærkombinasjoner av AOer eksplisitt, dvs. som en ekspansjon av 1s(H) og 1s(He)

In [29]:
```
# Svar med koeffisientene definert i to vektorer med 6+ desimaler
# Første element er bidrag fra 1s(H) og andre element er bidrag fra 1s(He)
# Feks MO1 = [0.2, 0.9] = 0.2 * 1s(H) + 0.9 * 1s(He)
MO1 = [c[0][0], c[1][0]]
MO2 = [c[0][1], c[1][1]]

print(MO1)
print(MO2)
# YOUR CODE HERE
```

```
[0.20252251963808915, 0.8762885757343425]
[-1.1692809905883976, 0.8002198060217096]
```

In [30]:
```
# Hemmelig test: Sjekker MO1 og MO2 mot fasit til 6 desimaler
```