```
In[•]:= SetOptions[SelectedNotebook[],
         PrintingStyleEnvironment → "Printout", ShowSyntaxStyles → True]

In[•]:= << Notation`

In[•]:= (* Simplify the notation for the
         formulas: index vectors based on a subscript notation *)
        Notation[ x_n_ ⟺ x_⟦n_⟧ ]

In[•]:= α = Reverse@{13.00773, 1.962079, 0.444529, 0.1219492} & /@ Range[1, 2] // Flatten
Out[•]=
        {0.121949, 0.444529, 1.96208, 13.0077, 0.121949, 0.444529, 1.96208, 13.0077}

        matNorm[v_, M_] := v / Sqrt[v.M.v];

        scfLoop[d_, e_, α_] :=
          Module[{Z, K, R, S, T, V, H, Q, atomPts, G, F, evals, evecs, newEvals},
           atomPts = {{0, 0, 0}, {d, 0, 0}}; (* Atomic positions *)
           Z = Partition[Table[#, {n, 1, 4}] & /@ atomPts // Flatten, 3];

           (* Array of positions for the coefficients α *)
           K = Array[Exp[-α#1 α#2 / (α#1 + α#2) Norm[Z#1 - Z#2]^2] &, {8, 8}];
           (* Coefficients matrix K_{p,q} *)
           R = Array[(α#1 Z#1 + α#2 Z#2) / (α#1 + α#2) &, {8, 8}]; (* Distances matrix R_{p,q} *)
           S = Array[(π / (α#1 + α#2))^(3/2) K#1,#2 &, {8, 8}];
           (* Overlap matrix S_{p,q} *)

           T = Array[α#1 α#2 / (α#1 + α#2) (3 - 2 α#1 α#2 / (α#1 + α#2) Norm[Z#1 - Z#2]^2) S#1,#2 &, {8, 8}];
           (* Kinetic energy matrix T_{p,q} *)
           V = Total@Table[Array[If[Norm[R#1,#2 - U] == 0, -2 π / (α#1 + α#2) K#1,#2, -S#1,#2
                 1 / Norm[R#1,#2 - U] Erf[Sqrt[α#1 + α#2] Norm[R#1,#2 - U]]] &, {8, 8}], {U, atomPts}];
           (* Potential energy matrix V_{p,q} *)
           H = T + V; (* Hamiltonian matrix H_{p,q} *)

           Q = Array[S#1,#3 S#2,#4 If[Norm[R#1,#3 - R#2,#4] == 0, 2 / Sqrt[π] Sqrt[(α#1 + α#3) (α#2 + α#4) / (α#1 + α#3 + α#2 + α#4)],
                 1 / Norm[R#1,#3 - R#2,#4] Erf[Sqrt[(α#1 + α#3) (α#2 + α#4) / (α#1 + α#3 + α#2 + α#4)] Norm[R#1,#3 - R#2,#4]]] &,
             {8, 8, 8, 8}]; (* Q_{p,r,q,s} matrix, it's an 8 by 8
```

```
    matrix containing 8 by 8 matrices *)

  P = ConstantArray[0., {8, 8}];
  (* Initial guess for the density matrix *)
  (* Perform a reset before the while-loop to assert correctness *)
  evals = ConstantArray[0., {8}];
  newEvals = evals + ϵ;

  (* While the new and previous eigenvalues are still different enough... *)
  While[Norm[newEvals - evals] > ϵ,
   newEvals = evals; (* Overwrite the previous eigenvalues *)
   G = Array[∑_{r=1}^{8} ∑_{s=1}^{8} P_{r,s} (2 Q_{#1,r,#2,s} - Q_{#1,r,s,#2}) &, {8, 8}] (* G_{p,q} matrix *);
   F = H + 1/2 G; (* Compute the Fock operator from the G matrix *)
   {evals, evecs} = Eigensystem[{F, S}];
   (* Calculate the eigenvalues and eigenvectors *)
   {evals, evecs} = {evals[[#]], evecs[[#]]} &@Ordering[evals];
   (* Sort them in ascending order *)
   evecs = matNorm[#, S] & /@ evecs;
   (* Normalize them w.r.t. the overlap matrix S *)
   (* Perform a numerical divergence check *)
   If[AllTrue[Thread[#.S.# ≠ 1] & /@ evecs, TrueQ], Abort[], Nothing];
   (* Calculate the density matrix from the
    eigenvector evecs_1 associated with the minimal eigenvalue *)
   P = 2 TensorProduct[First@evecs, evecs_1];
  ];

  (* Return the first eigenvalue and the equilibrium bonding energy,
  where 1/d is the E_nucl term *)
  (* The `Total[#, 2]` function
   gives the sum of all elements of the matrix # *)
  {evals_1, Total[atomPts (H + 1/4 G), 2] + 1/d}
 ];

interv = Range[0.3, 10, 0.01] // N;
ϵ = 10^-4 // N;(* Accuracy level *)

{totalTime, res} =
  ParallelMap[Quiet@scfLoop[#, ϵ, α] &, interv] // Transpose // AbsoluteTiming;

Length@interv (* Number of distances used *)
Quantity[totalTime, "Seconds"]
(* Time it took the loop to finish, measured in seconds *)
```

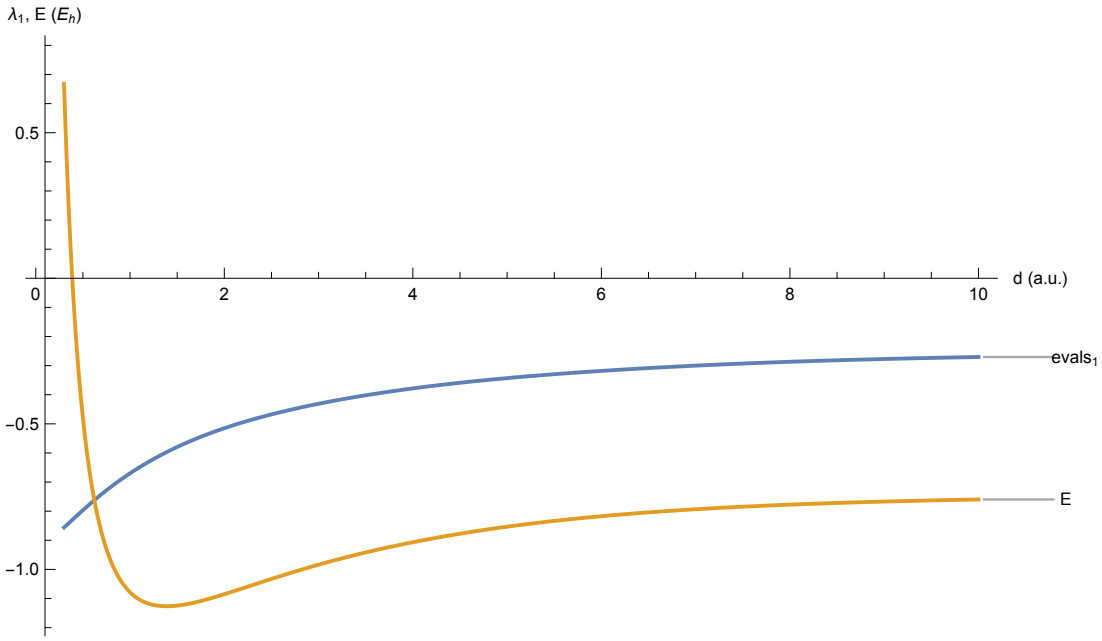
Out[ ]=

971

Out[ ]=

8.97029 s

```
(* Smallest eigenvalue evals1 and equilibrium bonding energy E *)
ListLinePlot[Thread[{interv, #}] & /@ res, AxesLabel → {"d (a.u.)", "λ1, E (Eh)"},
 PlotRange → Full, ImageSize → Large, PlotLabels → {"λ1", "E"}]
```




```
Quantity[Min[Last@res], "HartreeEnergy"] (* Smallest energy value*)
UnitConvert[%, "Electronvolts"](* ...converted in electronvolts *)
```

Out[ ]=

−1.12654 E_h

Out[ ]=

−30.6548 eV