Pseudopotential Units (Schmidt et al.)

$$\hat{H}_s = -\frac{1}{2\mu_1}\Delta r_1 - \frac{1}{2\mu_2}\Delta r_2 - \frac{1}{m_3}\nabla_{r_1}\cdot\nabla_{r_2} + q_1 q_3 V_k(r) + q_2 q_3 V_k(r_2) + q_1 q_2 V_k(|r_1 - r_2|) \quad (1)$$

2D:

Schmidt – Keldysh potential:

$$V_k(r) = \frac{\pi}{2r_0}\left(H_0\left(\frac{r}{r_0}\right) - Y_0\left(\frac{r}{r_0}\right)\right)$$

- IGNORES qs
- and sets $K_e = 1$

this is **Gaussian Units**

$q_1 q_3 V_k(r)$

ACTUAL Keldysh $V_k(r) = \frac{\pi k_e}{(\epsilon_1+\epsilon_2)r_0}\left(H_0\left(\frac{r}{r_0}\right) - Y_0\left(\frac{r}{r_0}\right)\right)$, $k_e = \frac{1}{4\pi\epsilon_0}$ $[N m^2 C^{-2}]$

The units of $V_k(r)$:

$$\frac{[N\,m^2\,C^{-2}]}{[m]} \rightarrow \frac{q_1 q_2}{m}\frac{N}{C^2} = [J]$$

V_k actual: $\left[\frac{Nm}{C^2}\right]$

Schmidt au = $10^{-22}\ eV/(m_e v)^2$ typo

$$V_{Xc} = \begin{cases} V_0 & r < r^* \\ -\frac{\alpha}{2}\left(\frac{dV_k(r)}{dr}\right)^2 & r > r^* \end{cases}$$

$\left[\frac{dV_k(r)}{dr}\right]$: $[J^2 C^{-2} m^{-2}]$

where $\alpha = 52\times10^3 \times 10^{-22}\left(\frac{eV\cdot V^2}{m^2}\right)$

au

$(J/C)^2$

au: $\left[\frac{eV}{m^2 V^2}\right]$

Schmidt: $\left[\frac{J\,V^2}{m^2}\right] = \left[\frac{J\cdot J^2}{C^2 m^2}\right] = \left[\frac{J}{C^2 m^2}\right]$

This doesn't make sense ...

Note:

- $1\,J = 6.242\times10^{18}\ eV$
- $[V] = \left[\frac{J}{C}\right] = \left[\frac{J}{As}\right]$

✗ Schmidt ref. [75]

MoS_2: $\alpha = 4.6\times10^{-18}\ eV m^2 V^{-2}$

fold. Importantly, the polarizability increases by an order of magnitude with increased screening. Hence, at $\kappa = 10$ the polarizability reaches values in the range of $\alpha = 45\times10^{-18}$ eV(m/V)^2 to 70×10^{-18} eV(m/V)^2. Focusing on freely suspended ($\kappa = 1$) MoS_2 and WSe_2, the polarizabilities are $\alpha = 4.6\times10^{-18}$ eV(m/V)^2 and $\alpha = 6.3\times10^{-18}$ eV(m/V)^2,

The au. of polarisability is

NIST units $\alpha = 52\times10^3 \times 1.648\,777\,274\,36\times10^{-41}\ C^2 m^2 J^{-1}$

$\therefore \alpha =$

```
>>> 52e3*1.64877727436e-41
8.573641826672e-37
>>>
```

$C^2 m^2 J^{-1}$

Units When $r > r^*$:

$$\therefore -\frac{\kappa}{2}\left(\frac{dV_k(r)}{dr}\right)^2 = \left[\frac{C^2 m^2}{J}\right]\left[\frac{J^2}{m^2 C^4}\right] = \left[\frac{J}{C^2}\right]$$

This makes sense

NIST

Fundamental Physical Constants

atomic unit of electric polarizability

$e^2 a_0^2 / E_h$

Numerical value	1.648 777 274 36 x 10^{-41} C^2 m^2 J^{-1}
Standard uncertainty	0.000 000 000 50 x 10^{-41} C^2 m^2 J^{-1}
Relative standard uncertainty	3.0 x 10^{-10}
Concise form	1.648 777 274 36(50) x 10^{-41} C^2 m^2 J^{-1}

Finding Bound States

Direct diagonalisation (computationally expensive)

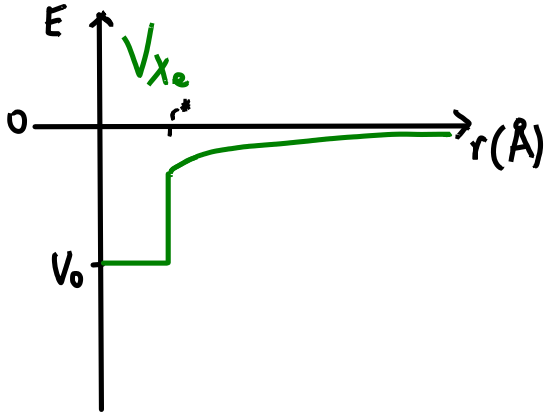
(2D s-wave Radial Schrödinger equation)

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{dR}{dr}\right)+\left(k^2-\frac{2\mu V(r)}{\hbar^2}\right)R=0$$

k is wavenumber: $\psi_{in}=e^{ikr}$

$$u_l(r)\equiv r\,R_l(k,r)$$

how $r\to 0$ the asymptotic solution $u_l(kr\to 0)\propto r^{l+1}$



$$\frac{1}{r}\frac{d}{dr}\left(r\frac{dR}{dr}\right)+\left(k^2-\frac{2\mu V(r)}{\hbar^2}\right)R=0$$

$$\therefore \frac{1}{r}\frac{d}{dr}\left(r\frac{dR}{dr}\right)-\frac{2\mu V(r)}{\hbar^2}R=-k^2R$$

$$\therefore \frac{1}{r}\left(\frac{dr}{dr}\frac{dR}{dr}+r\frac{d^2R}{dr^2}\right)-\frac{2\mu V_{xe}(r)}{\hbar^2}R=-k^2R$$

$$\therefore \frac{d^2R}{dr^2}+\frac{1}{r}\frac{dR}{dr}-\frac{2\mu V_{xe}(r)}{\hbar^2}R=-k^2R$$

(AKA: Bessel's eqn)

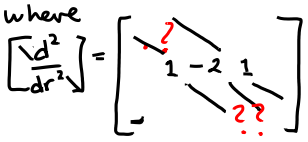
Note: $E=\frac{\hbar^2k^2}{2\mu}$

$\therefore \frac{2\mu E}{\hbar^2}=k^2$

Numerics ($A\vec{R}=\lambda\vec{R}$)

we can write our ODE as a linear system:

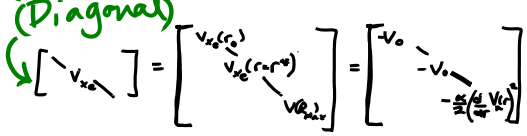
$$\left(\begin{bmatrix}\frac{d^2}{dr^2}\end{bmatrix}+\begin{bmatrix}\frac{1}{r}\end{bmatrix}\begin{bmatrix}\frac{d}{dr}\end{bmatrix}-\begin{bmatrix}V(r)\end{bmatrix}\right)\begin{bmatrix}|\\R\\|\end{bmatrix}=-k^2\begin{bmatrix}|\\R\\|\end{bmatrix}$$

where $\begin{bmatrix}\frac{d^2}{dr^2}\end{bmatrix}=$ 

?? BCS

$\begin{bmatrix}\frac{1}{r}\end{bmatrix}\begin{bmatrix}\frac{d}{dr}\end{bmatrix}=$ 

(Diagonal)

$\begin{bmatrix}V_{xe}\end{bmatrix}=$ 

using centered differences (2nd order)

$$\frac{d^2R}{dr^2}\approx\frac{R_{j+1}-2R_j+R_{j-1}}{\Delta r^2}$$

$$\frac{dR}{dr}\approx\frac{R_{j+1}-R_{j-1}}{2\Delta r}$$

if $A\vec{R} = \lambda\vec{R}$:

BCs?

$$A = \begin{bmatrix} ?? & & & \\ 1 & -2 & 1 & \\ & \ddots & \ddots & \ddots \\ & & ?? & \end{bmatrix} + \begin{bmatrix} \frac{1}{r} & & & \\ & \frac{1}{r} & & \\ & & \ddots & \\ & & & \frac{1}{r} \end{bmatrix} \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix} - \frac{2\mu}{\hbar^2} \begin{bmatrix} -V_0 & & & \\ & -V_0 & & \\ & & \ddots & \\ & & & -\frac{\alpha}{2}\left(\frac{d}{dr}V_{xe}\right)^2 \end{bmatrix}$$

$N+1$

$$A = \frac{d^2}{dr^2} + \left(\frac{1}{r} @ \frac{d}{dr}\right) - \frac{2\mu}{\hbar^2} V_{xe}$$

FINDING Boundary conditions $(l=0)$, $\frac{-2\mu E}{\hbar^2} = k^2$

ODE: $\frac{d^2R}{dr^2} + \frac{1}{r}\frac{dR}{dr} - \frac{2\mu V_{xe}(r)}{\hbar^2}R = -k^2 R$

$$R'' + \frac{1}{r}R' = \left(-k^2 + \frac{2\mu V_{xe}}{\hbar^2}\right)R$$

As $r \to 0$, $\frac{1}{r}R'$ dominates

$$\therefore \frac{1}{r}R' = 0 \rightarrow R' = 0$$ (Neumann BC / zero derivative BC.)

$R(r) = C$, C is constant

$$\frac{d^2R}{dr^2} \approx \frac{R_{j-1} - 2R_j + 2R_{j+1}}{\Delta r^2}$$

Because this is ghost point

Even Symmetry

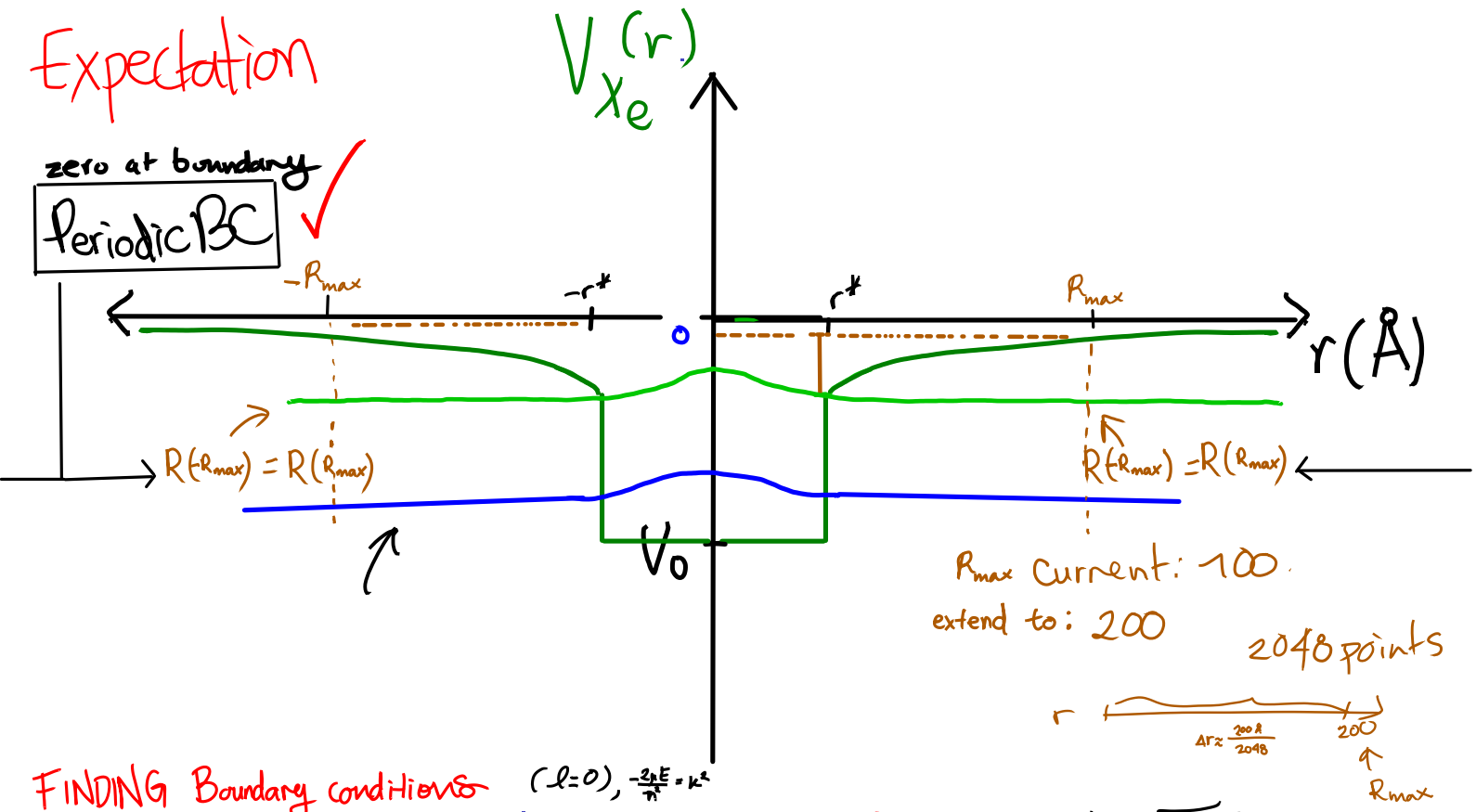
matrix coefficients at the Boundary

$$\frac{dR}{dr} \approx \frac{R_{j+1} - R_{j-1}}{2\Delta r} = \text{zero}$$ ← NOT WORKING

TRY: EVEN extension & periodic BCs.

Even extension to Schmidt $V_{Xe}(r)$

Expectation



R_{max} current: 100.
extend to: 200

2048 points

FINDING Boundary conditions $(l=0)$, $-\frac{2\mu E}{\hbar^2} = k^2$

ODE: $\frac{d^2R}{dr^2} + \frac{1}{r}\frac{dR}{dr} - \frac{2\mu V_{Xe}(r)}{\hbar^2}R = -k^2 R$ ⇝ CHANGE of variables $u(r) = \sqrt{r}\,R(r)$

① extending r- array to larger R_{max} ✓

② change $r = \sqrt{x^2+y^2}$ ✓
for $|x|$

modify in
Matrix:

$A = \frac{1}{\Delta r} D2 + V_{Xe}$

where $\Delta r = r[1] - r[0]$

[eigenvectors, unitary] = np.linalg.eigh(A)
↑ of A ↑

$u = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1N} \\ u_{21} & u_{22} & & u_{2N} \\ u_{31} & u_{32} & & u_{3N} \\ \vdots & \vdots & & \vdots \\ u_{N1} & u_{N2} & & u_{NN} \end{bmatrix}$ columns of u are eigenvectors of A.

eventually decided against:

X · periodic BCS
X · extension

$R = \frac{u_\ell(r)}{r}$

Change of variables

2) radial Schrödinger equation $(\ell = 0)$

$$E\psi = \frac{-\hbar^2}{2\mu}\left(\frac{1}{r}\frac{d}{dr}r\frac{d}{dr}\right)\psi + V(r)\psi$$

using $\psi = u(r) = \sqrt{r}\,R(r)$ ←

$E = \frac{k^2\hbar^2}{2\mu}$ $\therefore -k^2 = \frac{-2\mu E}{\hbar^2}$

$$\therefore -k^2\frac{u(r)}{\sqrt{r}} = \left(\frac{1}{r}\frac{d}{dr}r\frac{d}{dr}\right)\frac{u}{\sqrt{r}} - \frac{2\mu}{\hbar^2}V(r)\frac{u}{\sqrt{r}}$$

$$= \frac{1}{r}\frac{d}{dr}r\left(\frac{du}{dr}r^{-1/2} - \frac{u}{2}r^{-3/2}\right) - \frac{2\mu}{\hbar^2}V(r)\frac{u}{\sqrt{r}}$$

$$= \frac{1}{r}\frac{d}{dr}\left(\frac{du}{dr}r^{1/2} - \frac{u}{2}r^{-1/2}\right) - \frac{2\mu}{\hbar^2}V(r)\frac{u}{\sqrt{r}}$$

$$= \frac{1}{r}\left(\frac{d^2u}{dr^2}r^{1/2} + \frac{1}{2}\frac{du}{dr}r^{-1/2} - \frac{1}{2}\left(\frac{du}{dr}r^{-1/2} - \frac{u}{2}r^{-3/2}\right)\right) - \frac{2\mu}{\hbar^2}V(r)\frac{u}{\sqrt{r}}$$

$$= \frac{d^2u}{dr^2}r^{-1/2} + \frac{1}{2}\frac{du}{dr}r^{-3/2} - \frac{1}{2}\left(\frac{du}{dr}r^{-3/2} - \frac{u}{2}r^{-5/2}\right) - \frac{2\mu}{\hbar^2}V(r)\frac{u}{\sqrt{r}}$$

$$= \frac{1}{\sqrt{r}}\frac{d^2u}{dr^2} + \frac{1}{2}\frac{du}{dr}\cdot\frac{1}{\sqrt{r}} - \frac{1}{2}\frac{du}{dr}\cdot\frac{1}{\sqrt{r}} + \frac{u}{4}r^{-5/2} - \frac{2\mu}{\hbar^2}V(r)\frac{u}{\sqrt{r}}$$

$$\therefore -k^2\frac{u(r)}{\sqrt{r}} = \frac{1}{\sqrt{r}}\frac{d^2u}{dr^2} + \frac{u}{4r^{5/2}} - \frac{2\mu}{\hbar^2}V(r)\frac{u}{\sqrt{r}}$$

$$\therefore -k^2u(r) = \frac{d^2u(r)}{dr^2} + \frac{u(r)}{4r^2} - \frac{2\mu}{\hbar^2}V(r)u$$

Schmidt matrix

increase order approximation.

Tests
- No BCS ✓
- HIGH ORDER Approximation to derivative ✓
- zero BCS: $(r \to 0\ ,\ r \to R_{max})$ ✓

(Mixed) FINITE Differences (~4th order accuracy)

Finite Difference Equations

- $\frac{\partial^{(2)} f}{\partial x^{(2)}} \approx \frac{35f(x+0h) - 104f(x+1h) + 114f(x+2h) - 56f(x+3h) + 11f(x+4h)}{12h^2}$
- $\frac{\partial^{(2)} f}{\partial x^{(2)}} \approx \frac{11f(x-1h) - 20f(x+0h) + 6f(x+1h) + 4f(x+2h) - 1f(x+3h)}{12h^2}$

using 4th order centered finite differences

$$\frac{d^2 u(r)}{dr^2} \approx \frac{-\frac{1}{12}u_{j-2} + \frac{4}{3}u_{j-1} - \frac{5}{2}u_j + \frac{4}{3}u_{j+1} - \frac{1}{12}u_{j+2}}{\Delta r^2}$$

	0	1	2	3	4	...	N-1	
0	$\frac{35}{12}$	$\frac{-104}{12}$	$\frac{114}{12}$	$\frac{-56}{12}$	$\frac{11}{12}$	0 ...	0	custom
1	$\frac{11}{12}$	$\frac{-20}{12}$	$\frac{6}{12}$	$\frac{4}{12}$	$\frac{-1}{12}$	0 ...	⋮	custom
2	$\frac{-1}{12}$	$\frac{4}{3}$	$-\frac{5}{2}$	$\frac{4}{3}$	$\frac{-1}{12}$	0 ...		
3	0	$\frac{-1}{12}$	$\frac{4}{3}$	$-\frac{5}{2}$	$\frac{4}{3}$	$\frac{-1}{12}$ 0 ...		
⋮		0	$\frac{-1}{12}$	$\frac{4}{3}$	$-\frac{5}{2}$	$\frac{4}{3}$ $\frac{-1}{12}$ ...		
					⋮			
	... 0	$\frac{-1}{12}$	$\frac{4}{3}$	$-\frac{5}{2}$	$\frac{4}{3}$	$\frac{-1}{12}$ 0		
	... 0	$\frac{-1}{12}$	$\frac{4}{3}$	$-\frac{5}{2}$	$\frac{4}{3}$	$\frac{-1}{12}$ 0		
N-3	... 0	$\frac{-1}{12}$	$\frac{4}{3}$	$-\frac{5}{2}$	$\frac{4}{3}$	$\frac{-1}{12}$		Centered diff 4th order
N-2	... 0	$\frac{-1}{12}$	$\frac{4}{12}$	$\frac{6}{12}$	$\frac{20}{12}$	$\frac{11}{12}$		custom
N-1	... 0	$\frac{11}{12}$	$\frac{-56}{12}$	$\frac{114}{12}$	$\frac{-104}{12}$	$\frac{35}{12}$		custom

- $\frac{\partial^{(2)} f}{\partial x^{(2)}} \approx \frac{-1f(x-3h) + 4f(x-2h) + 6f(x-1h) - 20f(x+0h) + 11f(x+1h)}{12h^2}$
- $\frac{\partial^{(2)} f}{\partial x^{(2)}} \approx \frac{11f(x-4h) - 56f(x-3h) + 114f(x-2h) - 104f(x-1h) + 35f(x+0h)}{12h^2}$

Then $A = \frac{1}{\Delta r^2}$ second_D + Schmidt_matrix

A appears to be non-Hermitian due to second_D

This is likely why I get complex phases

Python3 -i

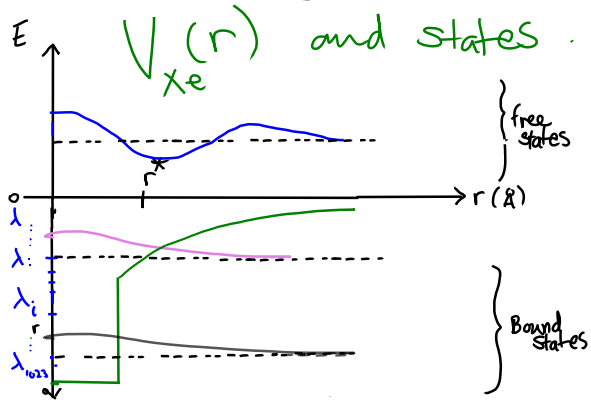
```
ana@ana-XPS-13-9343:~/scatterbrain$ python3 -i scatterbrain.py
/usr/lib/python3/dist-packages/numpy/core/_asarray.py:85: ComplexWarning: C
g complex values to real discards the imaginary part
  return array(a, dtype, copy=False, order=order)
eigenenergy 0: +2.01-0.00j meV
eigenenergy 1: -2.65+3.34j meV
eigenenergy 2: -2.65-3.34j meV
eigenenergy 3: +14.77+11.90j meV
eigenenergy 4: +14.77-11.90j meV
eigenenergy 5: +60.05+20.82j meV
eigenenergy 6: +60.05-20.82j meV
eigenenergy 7: -24.37+0.00j meV
>>> E = -eigenvalues * hbar**2 / (2 * M_red) / meV
>>> for item in np.sort_complex(E)[:100]:
...     print(item)
```

→ Lots of complex energy eigenstates as N → larger. ☹

Decided against non-centered differences on the Boundaries!

Result :

Direct diagonalisation method



need to translate the states by their corresponding λ_i

Binding energy stuff

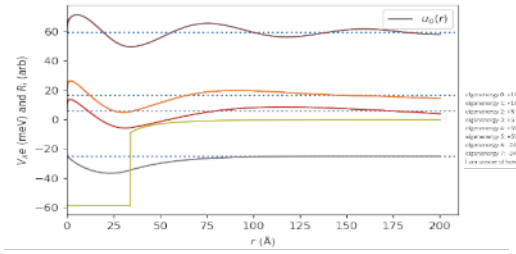
CONVERSION: $\frac{1\,J}{6.242 \cdot 10^{21}} = 1\,meV$

ENERGY from eigenvalues

$\lambda_i = k_i^2 = \frac{2\mu_{red} E_i}{\hbar^2}$

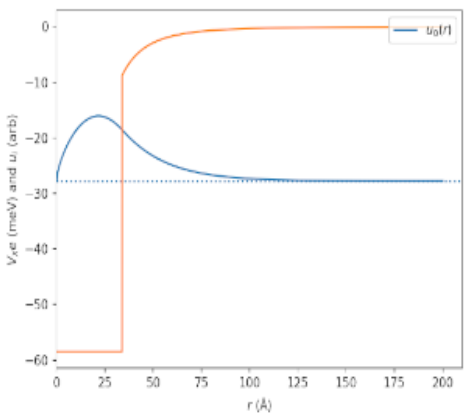
GROUND STATE $\therefore \frac{-k_0^2 \hbar^2}{2\mu_{red}} = E_0$ $(\therefore E_0/meV)$

expected $E_0 \sim 31.7\,meV$ (Schmidt et al. Fig. 6.)



N = 6144
r = np.linspace(1e-6, 200, N)

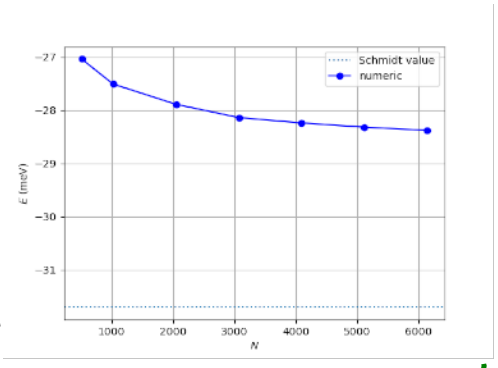
Imposing zero BCS $\begin{cases} r \to 0 : u(r) = 0 \\ r \to R_{max} : u(r) = 0 \end{cases}$



A Matrix is Hermitian

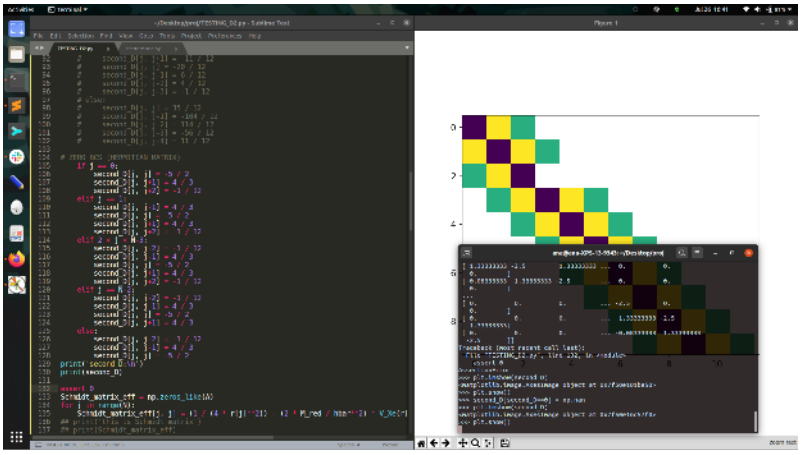
N = 3072

$E_0 = 28.14\,meV$



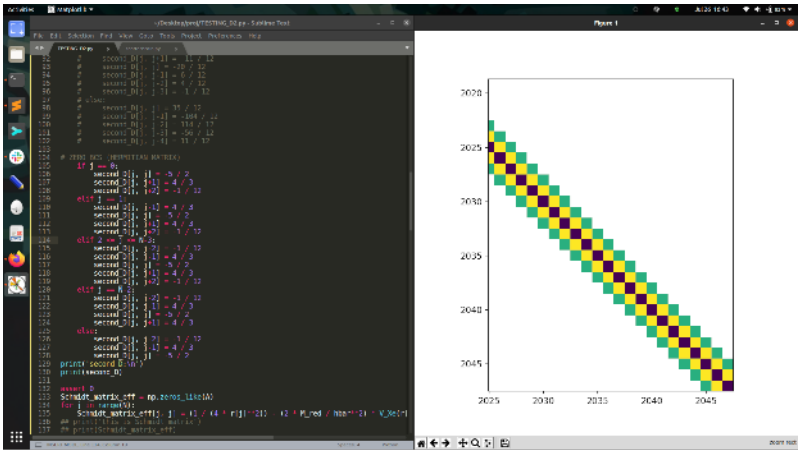
as we increase N value converges $\sim -29\,meV$

* Computer cannot handle more points



BUG: else statement applied to the non-defined cases: j=2, j=N-2

fixed:



only plot states that satisfy

$E = -(\text{eigenvalues} * \hbar^2 / 2 M_{red}$

iterate over eigenvalues

```
if not v_0 <= E[i] <= 0:
    continue
```

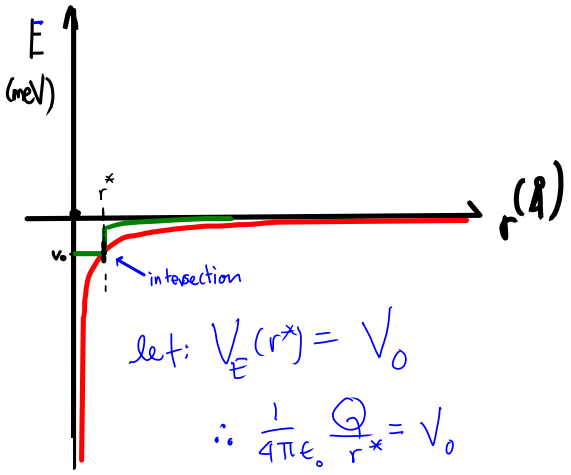
plot: { CODE

plt.show()

Testing Diagonalisation

Coulomb potential

$$V_E(r) = \frac{1}{4\pi\epsilon_0}\cdot\frac{Q}{r}$$

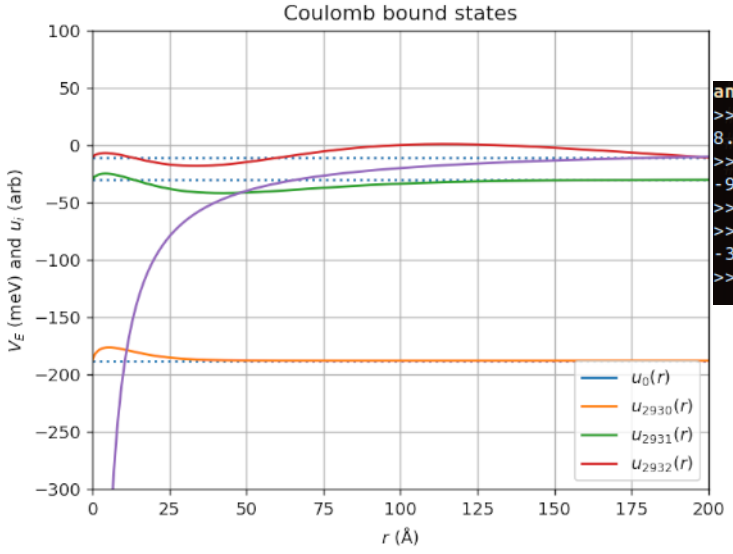


let: $V_E(r^*) = V_0$

$\therefore \frac{1}{4\pi\epsilon_0}\frac{Q}{r^*} = V_0$

$\therefore Q = V_0(4\pi\epsilon_0 r^*)$

Result



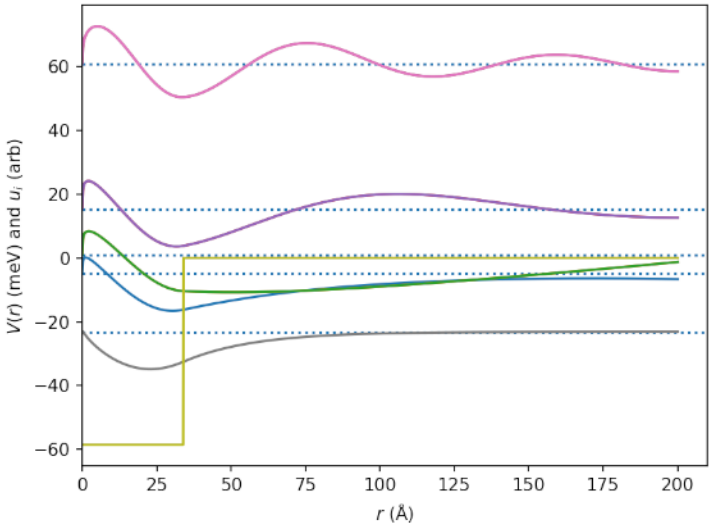

```
ana@ana-XPS-13-9343:~/Desktop/proj$ python3 -i TESTING_D2.py
>>> epsilon
8.854e-12
>>> v_0
-9.372733308899999e-21
>>> Q = v_0 * 4 * np.pi * epsilon * r_star
>>> Q
-3.5456393493840546e-39
>>>
```

Eigenenergies do not satisfy:

$E_n \propto \frac{1}{(2n-1)^2}$

where n is eigenstate index.

Diagonalisation using square well and free BCS



Scipy.integrate.solve_ivp(...)

$$\frac{dy}{dt} = f(t, y)$$

- t is 1-D independent variable (scalar)
- y is N-D vector valued ← ndarray (shape options): (n,) or (n,K) (vectorized=True)
- f(t, y) – determines the DE
- t_span is 2-tuple of floats $[t_0, t_f]$ (solver starts @ $t_0 \to t_f$)
- y_0 is IC. array_like, shape (n,)
- t_eval ← times at which to store the computed solution
  - Must be sorted and lie within t_span

f(t,y)

return np.shape(array_like(y)) = (n,K)

ie. each column in this corresponds to a single column in y

return

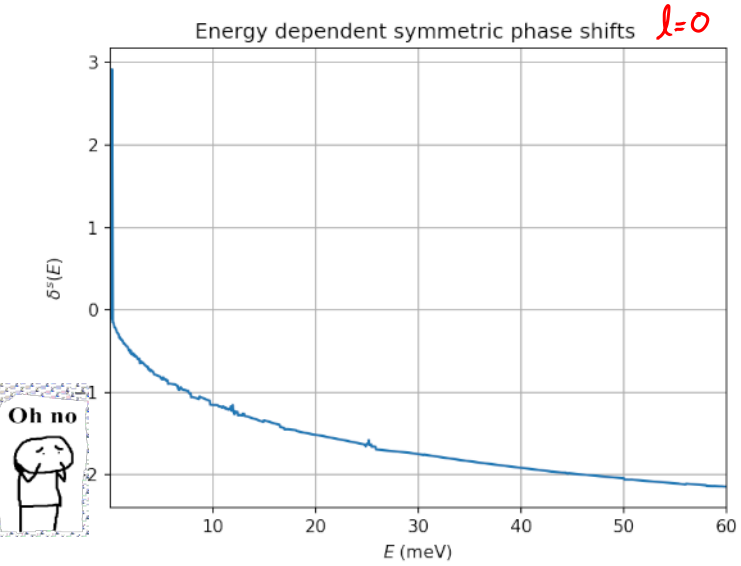
t : ndarray, shape (n_points,)

y : ndarray, shape (n_points,)

sol : Found solution at t

Defaults

- method = 'RK45'
- t_eval = None
- dense_output = False
- vectorised = False
- options...




```
#######################SOLVE_IVP#####################################
# ODE solver SOLVE_IVP
delta_0 = [r[0], 1]
delta = []
for i, k_i in enumerate(k):
    # print(i)
    solution_ODE_31 = solve_ivp(
                                lambda r, delta : ODE_31(r, delta, k_i),
                                 [r[0], r[-1]], delta_0, t_eval=r
                                 )
    delta.append(solution_ODE_31.y[0, -1]- np.pi)

# print(f'THIS IS SOLVE_IVP {delta=}\n ')
print(f'{np.shape(delta)=}\n')

plt.plot(E / meV,  delta)
plt.axis(xmin=E_min/ meV, xmax=E_max / meV)
plt.xlabel(r'$E$ (meV) ')
plt.ylabel(r'$\delta^{s} (E)$')
plt.title('Energy dependent symmetric phase shifts ')
plt.grid(True)
plt.savefig('solveivp_symmetric_phase_shifts.png')
plt.show()

#######################SOLVE_IVP#####################################
```

—Shooting Method—

The essence of the shooting method is to guess a complete $\vec{z}$ at one endpoint, use the relationship for $\frac{d\vec{z}}{dx}$ to propagate a solution to $\vec{z}(x)$ over to the other endpoint, and then compare how close the propagated solution is to known solution in the second boundary condition. You then update your guess and repeat the process to converge the propagated solution to the true solution at the other boundary.

ODE

$$\frac{d^2u}{dr^2} = -\frac{u}{4r^2} + \frac{2\mu}{\hbar^2}V_{Xe}(r)u - k^2u$$

Boundary conditions (for r=0)

$$\vec{y}_0 = \begin{bmatrix} \sqrt{r[0]} \\ \frac{1}{2\sqrt{r[0]}} \end{bmatrix}$$

← asymptotic solution as $r \to 0$

← 1st derivative

we must recast our ODE as a couple of first order ODEs

let $w = \frac{du}{dr}$

then $\frac{dw}{dr} = \frac{d^2u}{dr^2} = -\frac{u}{4r^2} + \frac{2\mu}{\hbar}V_{Xe}(r)u - k^2u$

Then $\dot{\vec{y}} = \begin{bmatrix} w \\ \frac{dw}{dr} \end{bmatrix} = \begin{bmatrix} \frac{du}{dr} \\ -\frac{u}{4r^2} + \frac{2\mu}{\hbar}V_{Xe}(r)u - k^2u \end{bmatrix} = f(r, \vec{y})$

Single vector ODE:

Shooting Method

ODE: $\frac{d^2u}{dr^2} + \frac{u}{4r^2} - \frac{2\mu}{\hbar^2}V(r)u + k^2u = 0$

$\therefore \frac{d^2u}{dr^2} + \frac{u}{4r^2} - \frac{2\mu V(r)}{\hbar^2}u + K^2u = 0$

$\frac{d^2u}{dr^2} = \underbrace{\left(-\frac{1}{4r^2} + \frac{2\mu V(r)}{\hbar^2} - K^2\right)}_{\text{pseudo-potential}}u$

BCS Solve analytically

when: $r \to 0$, $V_{\text{eff}} = V_0$

$\frac{d^2u(r)}{dr^2} + \frac{u(r)}{4r^2} = 0$

asymptotic solution — as $r \to R_{max}$ we wait... for shooting method

$u(r) = C_1\sqrt{r} + C_2\sqrt{r}\ln(r)$

$C_1\sqrt{r}$: use as asymptotic sol

$C_2\sqrt{r}\ln(r)$: Throw away

$$y_0 = \begin{bmatrix} \sqrt{r[0]} \\ \frac{1}{2}r[0]^{-\frac{1}{2}} \end{bmatrix}$$

Initialise: K argument (Guess ①)

1. If $V_0 = -58.5$ meV

Then $E_{min} = V_0 = \frac{\hbar^2k^2}{2M_{red}}$

$\therefore K^2 = \frac{2M_{red}E_{min}}{\hbar^2}$

$E_{max} = 0$

$-\frac{\hbar^2}{2\mu}\frac{d^2u}{dr^2} + V(r)u = Eu$

$\frac{d^2u}{dr^2} - \frac{2\mu}{\hbar^2}V(r)u = -\frac{2\mu E}{\hbar^2}u$

$\frac{d^2u}{dr^2} = \left(\frac{2\mu}{\hbar^2}V(r) - \frac{2\mu E}{\hbar^2}\right)u$

if $E > 0$:

$E_{free} = \frac{\hbar^2k^2}{2\mu}$

$k \in \mathbb{R}$

if $E < 0$:

let $k = iK$

$E_{bound} = -\frac{\hbar^2K^2}{2\mu}$

CODE + Algorithm

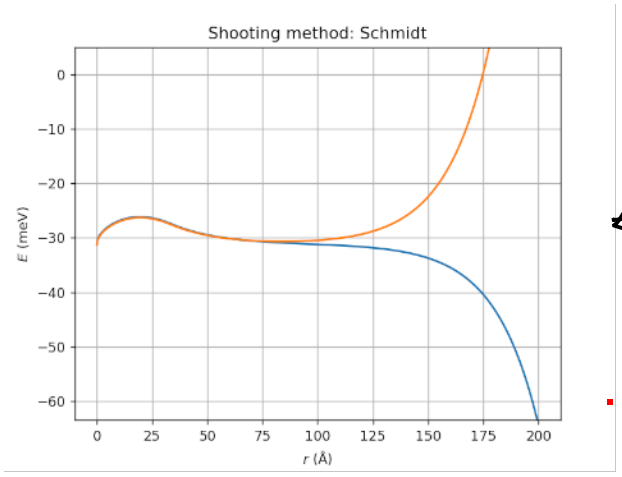
```
# Bound states search:
# SHOOTING method
# ENERGY
# E = -30 * meV #[J]

# ODE
def ODE(r, y, E):
    u, w = y
    result = np.zeros_like(y)
    if abs(u) > 1e100: # if blowing up return zero derivative
        return result

    else:
        result[0] = w
        result[1] = (- 1 / (4 * r**2) +  2 * M_red * (V_Xe(r) - E) / hbar**2) * u
        return result

# ICS
y_0 = [np.sqrt(r[0]), 1 / (2 * np.sqrt(r[0]))]
# ODEINT solver
for E in np.linspace(-31.15 * meV, -31.3 * meV, 2):
    solution_ODE = solve_ivp(lambda r, y: ODE(r, y, E), [r[0], r[-1]], y_0, t_eval=r)
    u = solution_ODE.y[0]
    plt.plot(r / angstrom,  (u / abs(u[r < r_star]).max() * 5) + E / meV)


plt.axis(ymin=v_0 / meV - 5, ymax=5)
plt.ylabel(r'$E$ (meV) ')
plt.xlabel(r'$r$ (Å)')
plt.title('Shooting method: Schmidt')
plt.grid(True)
plt.show()
```

① Initial guess

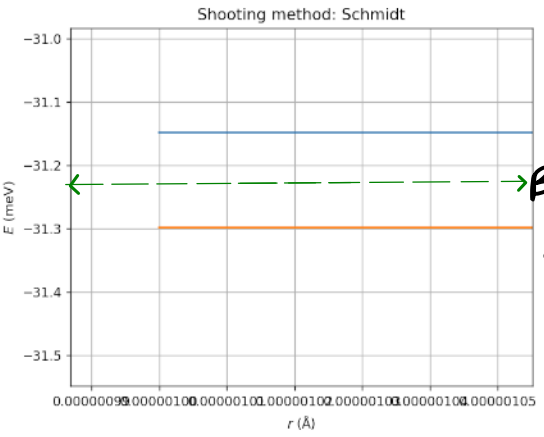
$E_{max} = -31.15$ meV — UB

$E_{min} = -31.3$ meV — LB

Plot:



zoom!



② Bisection

creates new LB/UB

③ Repeat

Automating

```
if sign == -1:
    append to list
if sign == 1:
    append to list
```

BISECTION

new_bound = γ

$\beta - \alpha = \gamma$

① set γ as new LB/UB

(ELI5) Shooting method –

if: E = bound state energy

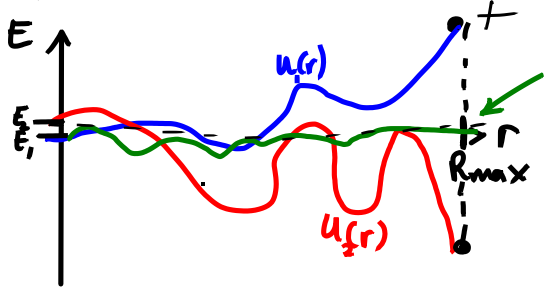
then: $u(R_{max}) = 0$

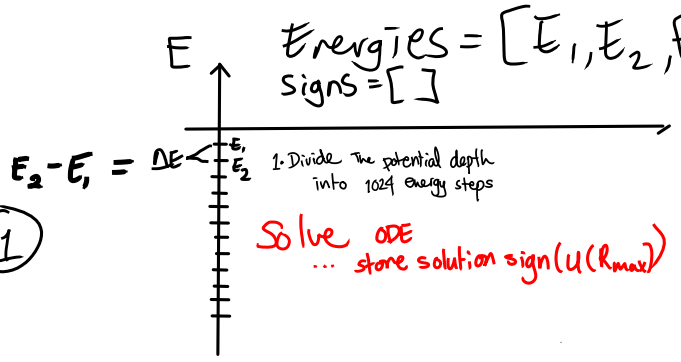
∴ if $E \neq$ bound state energy

then: $u(R_{max}) \neq 0$

say you have E_1, E_2

for E_1: $u_1(R_{max}) > 0$ and E_2: $u_2(R_{max}) < 0$





2 search signs for sign flips

[+, +, +, −, +, +, −, − ...] use these to extract

[E1, E2, E3, E4, E5, E6, ...] ← corresponding E_n

3 construct a List $E_{flip} = [[E_3, E_4], [E_5, E_6], [\ \ ...]]$

each of the pairs corresponds to $[E_{upper}, E_{lower}]$

The energy bounds for an eigenenergy

ITERATIVELY –

4 using E_{low} and E_{up}:

Solve ODE using $\frac{E_{low} + E_{up}}{2} = E_{mid}$

obtain sign(u(Rmax))

if sign(u(Rmax)) for E_{mid} == sign(u(Rmax)) for E_{low}

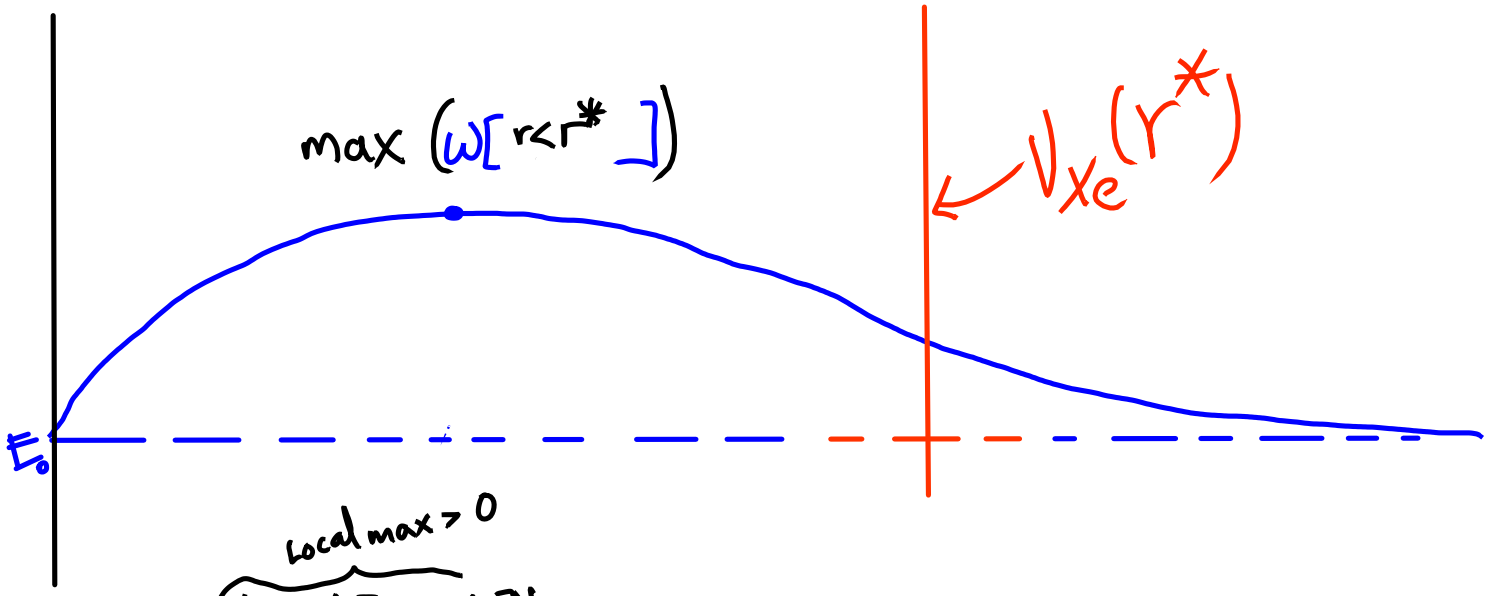
$E_{low} = E_{mid}$

else:

$E_{up} = E_{mid}$

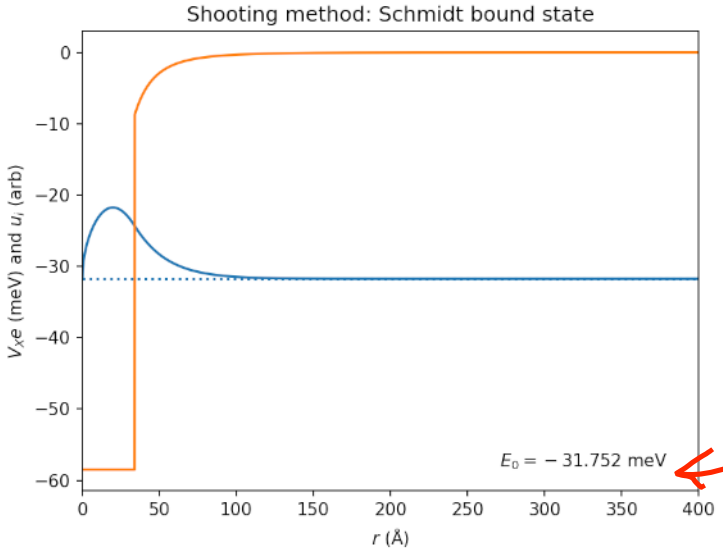
while abs$(E_{low} - E_{up}) > 1e\text{-}6$:

↑ we do this until we have scanned the E_{flip} list of Bounds entirely



$$S = \max\left(\overbrace{\text{abs}(W[r<r^*])}^{\text{local max} > 0}\right)$$

plot: $(W / S) * 10 + E_0/\text{meV}$



E_0 agrees with Schmidt et al.