Stopping power with the ZBL model

We will demonstrate how to find the nuclear stopping power of an ion by using the ZBL (Ziegler, Biersack and Littmark) model.
Specifically modeling the impact of an hydrogen with isotope of Silicon 28Si matter, and 28Si with isotope of gold 179Au.

Introduction

Nuclear stopping power is a phenomenon where energy loss is experienced by charged particles as they travel through a material due to interactions with atomic nuclei.
The research of it is crucial because it allows us to model the behavior of charged particles when they interact with matter. The energy loss of charged particles determines their penetration characteristics (depth, range, ionization)
And it is relevant to various fields of science and technology.

Applications of nuclear stopping power are:

- *Radiation Therapy*: In medical applications, charged particles are used for cancer treatment. Modeling of stopping power helps in accurately delivering the required radiation dose to the tumor while minimizing damage to surrounding healthy tissue.
- *Particle Detectors:* By measuring the energy loss of charged particles as they pass through a detector material, information can be obtained about their type, energy, and trajectory by using stopping power knowledge.
- *Semiconductors*: Stopping power is important in ion implantation processes used in the semiconductor industry for doping or modifying the properties of materials. By accurately controlling the energy and range of ions implanted into a material, specific doping profiles can be achieved, leading to the creation of transistors, diodes, and other electronic devices.

Many more applications exist, such as: nuclear fusion, particle accelerators, nuclear forensics.

Challenges associated with nuclear stopping power include:

- *Material dependence*: Stopping power varies with the target material. Developing stopping power databases for different materials is a lot of work, and accuracy can be limited, especially for less-studied “rare” materials.
- *Complex Interactions*: The interaction of charged particles with atomic nuclei involves very complicated and detailed physical processes such as ionization, elastic scattering, and excitation.Taking all of these interactions into account accurately is difficult and requires sophisticated theoretical models and experimental data.
- *Energy dependence:* The stopping power of charged particles typically depends on their energy. Low and high energies

Overcoming those challenges requires a combination of theoretical models, computer simulations and experiment data to accurately predict the behavior of particle stopping power.

Implementation

The project is implemented using python3 with SciPy, NumPy and Matplotlib as prerequisites.
NumPy is a powerful library for numerical computations in Python, providing efficient array operations and mathematical functions.
SciPy is a comprehensive library for scientific computing in Python, offering tools for optimization, interpolation, linear algebra, and more.
Matplotlib is a versatile library for creating static, animated, and interactive visualizations in Python, supporting a wide range of plotting and graphing capabilities.

These were chosen mainly for ease of development and prototyping. They have a wide variety of tools to solve the required problems - root finding and integration.

The main entry point is called *main.py* and calculates the stopping power of 2 different cases - hydrogen to silicon and silicon to gold.
Plots a log-log comparison to the “universal nuclear stopping power formula” for both.
Also all distances are converted to Ångstroms, and reduced constants to prevent numbers being too close to machine epsilon.

Finding r_{min}

The first step is to find the root of the equation $g(r_{min}) = 0$

$$g(r) = \sqrt{1 - \left(\frac{b}{r}\right)^2 - \frac{V(r)}{E_{\text{com}}}}$$

Which happens at *root.py* `get_rmin(Z1, Z2, Ecom, b)` where we get the minimum approach distance.

First, we solve the equation squared for $g^2(r_{min}) = 0$ In order to get rid of the square root and negative values.
Using scipy’s optimize.root, which by default is a hybrid method using a modification of the Powell hybrid method as implemented in MINPACK.
Sanity checking that at higher b, the root r_{min} is equal to b shows that found root r_{min} is correct.

To visualize these, running the method `plot()` displays g as a function of r for logspaced E_{com}

Finding b_{max}

At *negligible_potential.py* it plots the screened Coulomb potential $V(r)$ with different energies to decide upon a suitable b_{max}.
This can be naturally found “dynamically” for each energy level later but just setting a static one will give good results as well.

Stopping power $S(E_{lab})$

At *main.py* for each logarithmically spaced energy value $Elab \in [10, 5e6]eV$
It finds r_{min} and calculates its stopping power.
To calculate the stopping power, first it solves the integral (11)

$$\Theta = \pi - 4b \int_0^1 F(u)\mathrm{d}u$$

$$F(u) = \left[b^2(2-u^2) + \frac{r_{\min}^2}{u^2 E_{\mathrm{com}}} \left(V(r_{\min}) - V\left(\frac{r_{\min}}{1-u^2} \right) \right) \right]^{-1/2}$$

Which is achieved by variable substitution from the original angular function (2)
This also diverges at 0 and 1 hence we integrate at the range $[eps,\ 1 - eps]$ when eps is the machine minimum.
The result is then plugged into the integral of the energy loss in a collision over all possible collision parameters b:

$$S_n(E_{\mathrm{lab}}) = 2\pi\gamma E_{\mathrm{lab}} \int_0^{b_{\max}} \sin^2\left(\frac{\Theta(b, E_{\mathrm{com}})}{2} \right) b\mathrm{d}b,$$

$$\gamma = \frac{4M_1M_2}{(M_1+M_2)^2}$$

This is calculated for both stated cases and plotted against the “universal nuclear stopping power formula” in a log-log plot while also calculating the mean square error.

Both integrals are calculated using scipy’s Simpson’s rule method (see Simpson integration section for elaboration) with a linearly spaced 200 samples. Which leads to a $1/200$ and $b_{max}/200$ step size respectively

“Expected results” - universal nuclear stopping power formula

At *expected.py* using the formulas

$$S_n(E_{\text{lab}}) = \frac{8.462 \times 10^{-15} Z_1 Z_2 M_1}{(M_1 + M_2)\left(Z_1^{0.23} + Z_2^{0.23}\right)} s_n(\epsilon)$$

$$s_n(\epsilon) = \begin{cases} \dfrac{\ln(1 + 1.138\epsilon)}{2\left(\epsilon + 0.01321\epsilon^{0.21226} + 0.19593\epsilon^{0.5}\right)}, & \epsilon \le 30 \\ \dfrac{\ln \epsilon}{2\epsilon}, & \epsilon > 30 \end{cases}$$

$$\epsilon = \frac{32.53 M_2 E_{\text{lab}}}{Z_1 Z_2 (M_1 + M_2)(Z_1^{0.23} + Z_2^{0.23})}$$

It can plot the stopping power as a function of logarithmic $Elab \in [10,\ 5e6]eV$ and also is used at *main.py* for direct visual comparison

Simpson’s rule integration

Scipy’s quad gives poor results, while a Simpson’s rule integration with not many samples provides nice results.
The integration interval [a, b] divided into smaller subintervals, each subinterval is approximated using parabolic curves. Then evaluate the function at the endpoints of each subinterval and multiply the function values at the endpoints by different weights and sum them up.
Our sub interval width is constant and no need to adjust the values according to the subinterval size.

Performance considerations

All the values depending on the atomic number, Ecom and Z are “memoized” once calculated and served again when requested.
Linspace variables as the u in theta calculation are defined once outside of the method as they are not dependent on u or b.

Equations including multiple powers of 10 have them reduced beforehand.
Some are very close to machine epsilon, and when combining them together we can get more accurate results and faster calculations
Also, at the beginning, the code was based on for loops. Once it was removed from `theta()` and used numpy’s calculation - it cut the running time by at least 10 times.

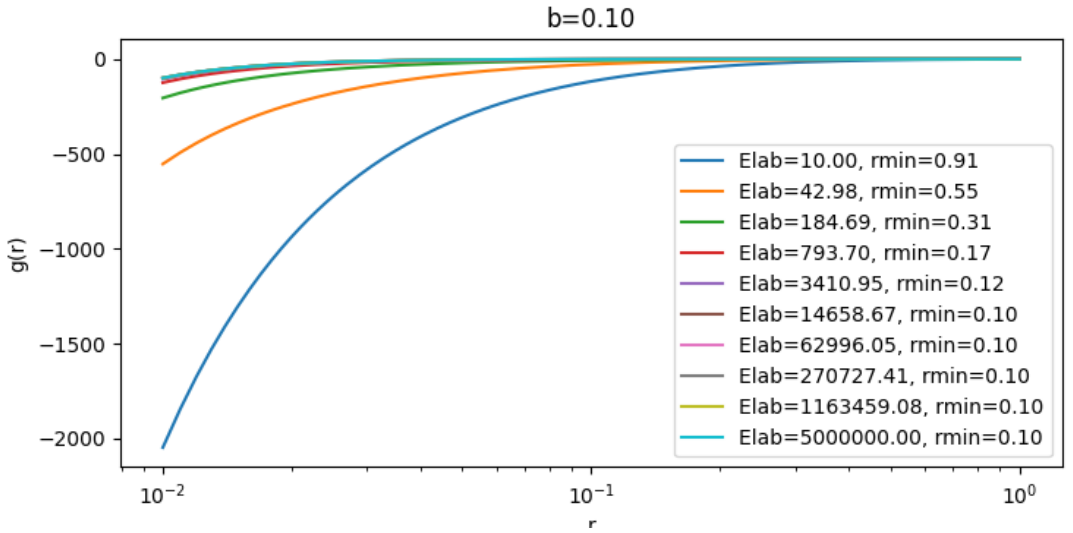
Accuracy considerations

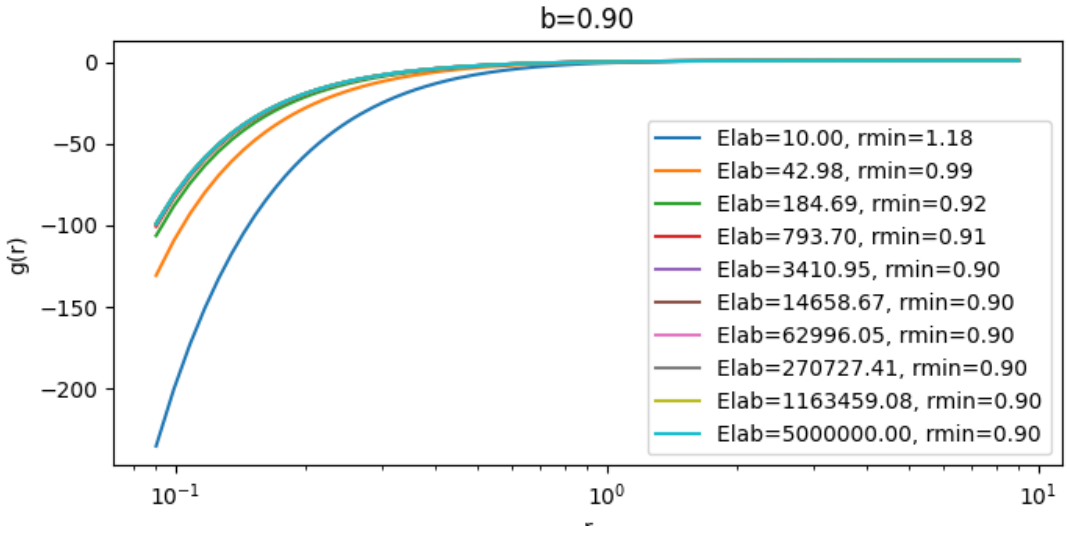
At the first iteration, the distance units were in cm and the constants had very small powers of 10 - such as the elementary charge 10^{-19}. Those were too small for the machine to consider. Changing it to Angstroms and “merging” the powers of 10; created results much closer to the universal formula. I believe more adjustments can be made to improve machine accuracy.

Results

Finding r_{min}

As theory suggests, r_{min} should converge to b as it gets bigger and further away from collision, while it should also converge to b when the energy increases.

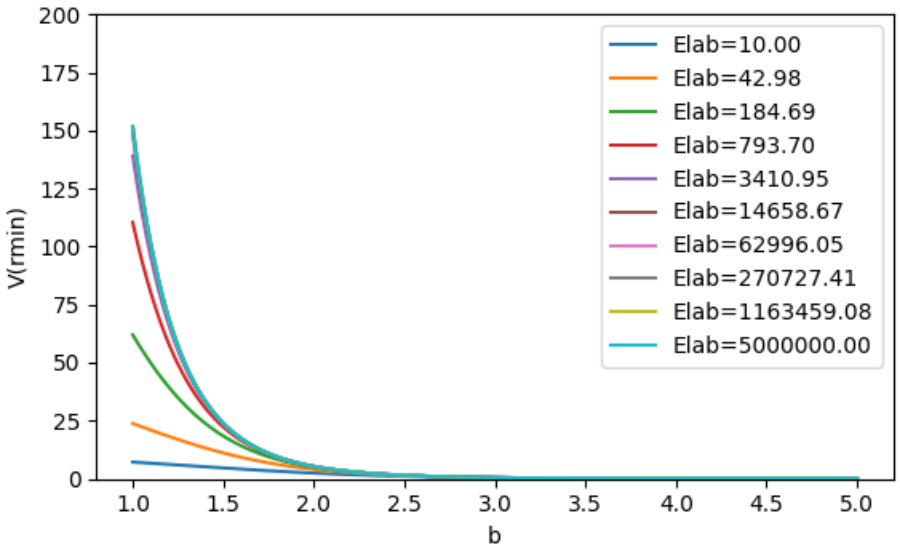




Indeed we see here faster convergence of $g(r)$ to b when $Elab$ is getting bigger and closer to $5MeV$.
Alternatively when $b = 0.9$ we see convergence to r_{min} already at smaller energies.
This is consistent with the units and theory, as $b = 1$ is a distance equal to size of a particle.

Finding b_{max}

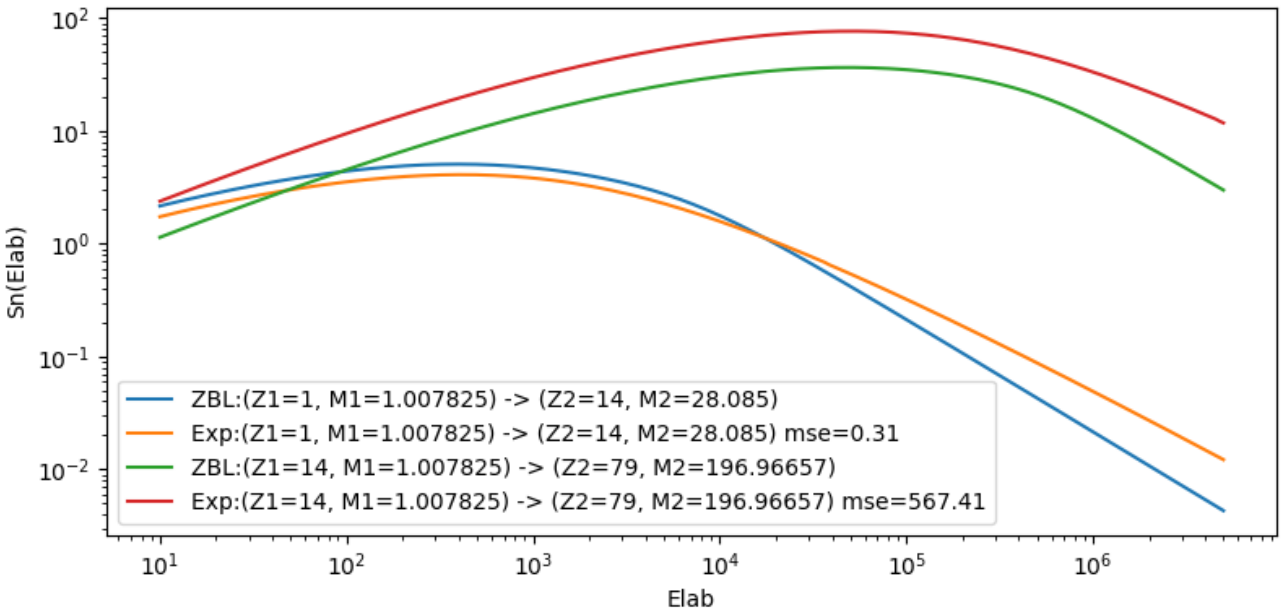
requires us to find the value where the interaction potential has negligible value.



At this plot, we see the potential sharply falls and converges to 0 already at `b=4`

Sn - Stopping power

Plotting (log-log) the stopping power of both cases, with 200 points for each Simpson's rule integration



For the case of hydrogen (blue for the numerical approximation, orange for the “expected” universal equation)
We get close results between the two, with a mean squared error of 0.31.
The stopping power peaks at 5.0 with $Elab = [10^2, 10^3]$ which is not far from the starting point of 2.1 at very low energy.
After the peak it falls almost linearly into 0

For the Silicon -> Gold (Green for the numerical approximation, red for the “expected”) case we get a much bigger mean squared error of 567.41 with much bigger stopping powers.
Increasing the number of integration points to 1000 while also increasing running time to over 1 minute gives us MSE of 543.47.
The stopping power peaks at $35\ eV$ around $Elab = 5e6$ and

Comparing the both, hydrogen being the lighter material the stopping power peaks at lower energies.
The bigger delta from the expected for the silicon -> gold case we can attribute the bigger value range. But when looking at normalized MSEs we still get a substantially bigger delta compared to hydrogen.

Conclusions

Numerical approximations with a ZBL model have been performed for an hydrogen hitting silicon, and silicon hitting gold. Then it was compared to the universal nuclear stopping power formula.

The model shows relatively good results. We get rather quick and good results for the hydrogen case, while the difference at the silicon -> gold case is pretty big from the universal equation - yet it follows the general tendency of the stopping power.
This of course can be expected, both methods are broad approximations.

Trying to give more samples to the Simpson’s integration does slightly better results, hence if we have patience we can use it.
While we can improve performance again by calculating a suitable b_{max} for each energy level.

Also, the integrated function diverges at the edges. For which we use epsilon at the singularity points. Instead we can use techniques to remove those and get better results.