Computational Physics with R and Maxima
Project 1
Classical Particle Scattering in a Central Potential *

Edwin (Ted) Woollett

March 3, 2014

Contents

*The code examples use **R ver. 3.0.2** and **Maxima ver. 5.31.2** using **Windows XP**. This is a live document which will be updated when needed. Check `http://www.csulb.edu/~woollett/` for the latest version of these notes. Send comments and suggestions for improvements to `woollett@charter.net`

```
project1.pdf describes the major project proposed in
 Chapter 1 of Computational Physics with R and Maxima,
 and is made available to encourage the use of the R and
 Maxima languages for computational physics projects of modest size.

R language free and open-source software:
  http://www.r-project.org/

Maxima language free and open-source software:
  http://maxima.sourceforge.net/

Code files available on the author's webpage are

1. rutherford_repulse.mac  :use in Maxima: e.g., load("c:/k1/rutherford_repluse.mac") to load
2. rutherford_repulse.R    :use in R: e.g., source("c:/k1/rutherford_repulse.R") to load
3. rutherford_attract.mac
4. rutherford_attract.R
5. lennard_jones.mac
6. lennard_jones.R
7. cmass.R
8. coord.R
9. scatt.R
10. k1util.mac
11. sho1.R

The author uses the XMaxima interface to Maxima exclusively, with the startup
  file setting: display2d:false$, which allows denser screen output.

The author normally uses the default RGui interface when coding in R.

COPYING AND DISTRIBUTION POLICY:
NON-PROFIT PRINTING AND DISTRIBUTION IS PERMITTED.

You may make copies of this document and distribute them
to others as long as you charge no more than the costs of printing.

Keeping a set of notes about using Maxima up to date is easier
than keeping a published book up to date, especially in view of
the regular changes introduced in the Maxima software updates.
```

Feedback from readers is the best way for this series of notes to become more helpful to users of **R** and **Maxima**. *All* comments and suggestions for improvements will be appreciated and carefully considered.

1 Introduction

The major project proposed in Chapter 1 of Steven Koonin's text **Computational Physics** applies root finding and quadrature methods to the task of finding the values of the scattering angle (in classical scattering by a central potential) as functions of the energy and impact parameter of the scattered particle.

We use the resources of the free and open source software **R** (`http://www.r-project.org/`) and Maxima (`http://maxima.sourceforge.net/`) to write code which helps to solve this type of problem.

The use of such modern powerful "command interpreters" encourages a "bottom-up" style of code development, in which small jobs are coded first, checked interactively for correct behavior, and then used as part of slightly larger coding jobs in an iterative fashion. Our discussion provides explicit examples (in both languages) of this coding style.

The first `project1.pdf` section explains our notation for classical scattering. The following two sections consider two examples which can be solved analytically, classical Rutherford scattering, both repulsive and attractive cases. When developing numerical approaches, it is always good to start with an exactly soluble case since when your numerical methods are producing the wrong answers, it will be obvious. The following section considers scattering by the Lennard-Jones potential. The last section displays the **R** code used to create the first three figures.

2 Classical Scattering Kinematics and Dynamics

2.1 Decomposition into Relative and Center of Mass Motion

We include **R** scripts used to draw some of the diagrams in Sec 6.

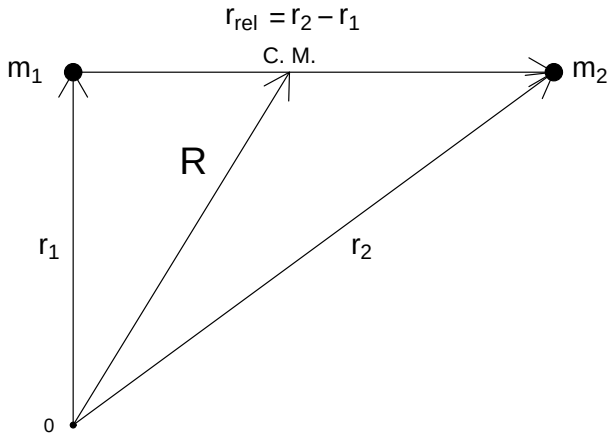

Figure 1: Center of Mass Diagram

Given two particles: mass m_1 with position vector $\mathbf{r}_1$ and mass m_2 with position vector $\mathbf{r}_2$, the position vector $\mathbf{R}$ of the location of the center of mass of the two particle system is defined by

$$\mathbf{R} = \frac{1}{M}(m_1\mathbf{r}_1 + m_2\mathbf{r}_2), \tag{2.1}$$

where $M = m_1 + m_2$ is the system mass. The position of particle 2 relative to particle 1 is defined by

$$\mathbf{r} = \mathbf{r_{rel}} = \mathbf{r}_2 - \mathbf{r}_1. \tag{2.2}$$

We define the position of particle 1 relative to the location of the system center of mass by

$$\mathbf{r}'_1 = \mathbf{r}_1 - \mathbf{R}, \tag{2.3}$$

which can be written, using Eqs. (2.1) and (2.2), as

$$\mathbf{r}'_1 = -\frac{m_2}{M}\mathbf{r}. \tag{2.4}$$

Likewise, the position of particle 2 relative to the center of mass is

$$\mathbf{r}'_2 = \mathbf{r}_2 - \mathbf{R}, \tag{2.5}$$

which becomes

$$\mathbf{r}'_2 = \frac{m_1}{M}\mathbf{r}. \tag{2.6}$$

From Eqs. (2.3) and (2.4) the velocity of particle 1 can be written

$$\mathbf{v}_1 = \dot{\mathbf{r}'_1} + \dot{\mathbf{R}} = \mathbf{V} - \frac{m_2}{M}\mathbf{v} \tag{2.7}$$

in terms of the velocity of the center of mass $\mathbf{V} = d\,\mathbf{R}/d\,t$ and the velocity of particle 2 relative to particle 1 $\mathbf{v} = d\,\mathbf{r}/d\,t$. In a similar manner, the velocity of particle 2 can be written as

$$\mathbf{v}_2 = \dot{\mathbf{r}'_2} + \dot{\mathbf{R}} = \mathbf{V} + \frac{m_1}{M}\mathbf{v}. \tag{2.8}$$

We can now transform the kinetic energy of the system

$$T = \frac{1}{2}m_1\mathbf{v}_1^2 + \frac{1}{2}m_2\mathbf{v}_2^2 \tag{2.9}$$

into the form

$$T = \frac{1}{2}M\mathbf{V}^2 + \frac{1}{2}\mu\mathbf{v}^2 \tag{2.10}$$

which exhibits a clean separation of the kinetic energy associated with motion $\mathbf{V}$ of the location of the center of mass, and a kinetic energy associated with the relative motion $\mathbf{v}$. We have introduced the symbol μ for the so-called "reduced mass" of the system, $\mu = (m_1 m_2)/M$. In the absence of external forces, $\mathbf{V}$ is a constant vector, and a reference frame with origin at the center of mass is an inertial frame in which $\mathbf{V} = 0$. *In such a frame*, the original lagrangian of the system

$$L = \frac{1}{2}m_1\mathbf{v}_1^2 + \frac{1}{2}m_2\mathbf{v}_2^2 - V(|\mathbf{r}_2 - \mathbf{r}_1|) \tag{2.11}$$

becomes

$$L = \frac{1}{2}\mu\mathbf{v}^2 - V(r) \tag{2.12}$$

where $r = |\mathbf{r}|$.

Once a solution $\mathbf{r}(t)$ has been found using the lagrangian Eq. (2.12), solutions for $\mathbf{r}_1(t)$ and for $\mathbf{r}_2(t)$ can be written down using Eqs. (2.4) and (2.6) (bearing in mind that, once we have adopted the center of mass frame, $\mathbf{r}_1 = \mathbf{r}'_1$ and $\mathbf{r}_2 = \mathbf{r}'_2$). In this sense, the two-body problem has been reduced to a fictitious problem of a single particle of mass μ and position vector $\mathbf{r}$ which experiences a potential (energy) $V(r)$ with the associated "force" given by $\mathbf{f} = -\nabla V(r)$. The solution involves motion in a plane with two degrees of freedom, which can be taken to be plane polar coordinates $\mathbf{r}(\mathbf{t})$ and $\boldsymbol{\theta}(\mathbf{t})$.

In the same center of mass frame, the system angular momentum vector

$$\mathbf{L} = \mathbf{r}_1 \times \mathbf{p}_1 + \mathbf{r}_2 \times \mathbf{p}_2 \tag{2.13}$$

where the momenta are $\mathbf{p}_1 = m_1\mathbf{v}_1$ and $\mathbf{p}_2 = m_2\mathbf{v}_2$, can be written as

$$\mathbf{L} = \mathbf{r} \times \mathbf{p} = \mu \mathbf{r} \times \mathbf{v}. \tag{2.14}$$

The system angular momentum vector $\mathbf{L}$ is a constant vector since

$$\frac{d\,\mathbf{L}}{d\,t} = \mu \mathbf{v} \times \mathbf{v} + \mathbf{r} \times \dot{\mathbf{p}}, \tag{2.15}$$

and the lagrange equation of motion

$$\frac{d}{d\,t}\frac{\partial\, L}{\partial\, \mathbf{v}} = \frac{\partial L}{\partial\, \mathbf{r}} \tag{2.16}$$

which is used with the lagrangian form Eq. (2.12) implies that

$$\dot{\mathbf{p}} = -\nabla V(r) \propto \mathbf{r}. \tag{2.17}$$

Since $\mathbf{L}$ is a fixed vector with a fixed direction in space, Eq. (2.14) implies that $\mathbf{r}$ is always perpendicular to $\mathbf{L}$ and thus lies in a fixed plane, which we take to be the $(\mathbf{x}, \mathbf{y})$ plane, with initial velocity vector of the incident particle in the $+\mathbf{x}$ direction.

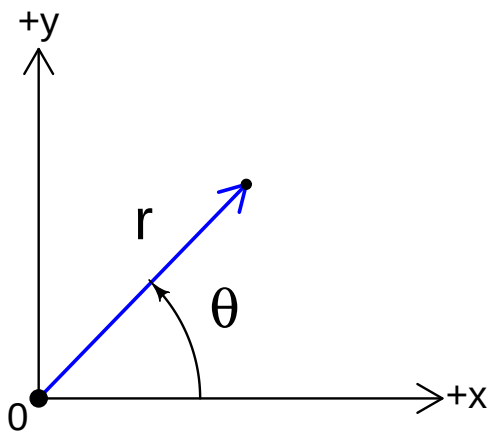

Figure 2: Coordinates Used Here

Taking plane polar coordinates (r, θ) in this fixed plane, with

$$\mathbf{r}(t) = r(t)\,\hat{\mathbf{r}}(t), \tag{2.18}$$

and the orthogonal time dependent unit vectors

$$\hat{\mathbf{r}}(t) = \hat{\mathbf{i}}\cos\theta(t) + \hat{\mathbf{j}}\sin\theta(t), \quad \hat{\boldsymbol{\theta}}(t) = -\hat{\mathbf{i}}\sin\theta(t) + \hat{\mathbf{j}}\cos\theta(t), \quad \hat{\mathbf{r}} \cdot \hat{\boldsymbol{\theta}} = 0, \tag{2.19}$$

and the resulting unit vector derivatives

$$\frac{d\,\hat{\mathbf{r}}}{d\,t} = \dot{\theta}\,\hat{\boldsymbol{\theta}}, \quad \frac{d\,\hat{\boldsymbol{\theta}}}{d\,t} = -\dot{\theta}\,\hat{\mathbf{r}}, \tag{2.20}$$

we find for the relative velocity $\mathbf{v}$

$$\mathbf{v} = \frac{d\,\mathbf{r}}{d\,t} = \dot{r}\,\hat{\mathbf{r}} + r\,\dot{\theta}\,\hat{\boldsymbol{\theta}}, \quad \mathbf{v}^2 = \dot{r}^2 + r^2\,\dot{\theta}^2. \tag{2.21}$$

We can express $\dot{\theta}$ in terms of the magnitude of the relative angular momentum and r using Eq. (2.14) and $\hat{\mathbf{r}} \times \hat{\boldsymbol{\theta}} = \hat{\mathbf{k}}$:

$$\mathbf{L} = \mu\, \mathbf{r} \times \mathbf{v} = \mu\, r^2\, \dot{\theta}\, \hat{\mathbf{k}}. \tag{2.22}$$

θ is initially equal to π, and continuously decreases with time:

$$\dot{\theta} = -\frac{|\mathbf{L}|}{\mu\, r^2}. \tag{2.23}$$

We see that in general, as $r \to \infty$, at the end of the scattering event, $\dot{\theta} \to 0$, and $\theta \to$ some constant. Eliminating $\dot{\theta}$ in Eq. (2.21), we get

$$\mathbf{v}^2 = \dot{r}^2 + \frac{|\mathbf{L}|^2}{\mu^2\, r^2}. \tag{2.24}$$

The total relative energy E is then

$$E = \frac{1}{2}\, \mu\, \mathbf{v}^2 + V(r) = \frac{1}{2}\, \mu\, \dot{r}^2 + V_{eff}, \tag{2.25}$$

where

$$V_{eff} = V(r) + \frac{|\mathbf{L}|^2}{2\, \mu\, r^2}. \tag{2.26}$$

Before the collision event, the incident particle has only kinetic energy, and the total constant energy E has the value $\frac{1}{2}\, \mu\, v_0^2$, where v_0 is the initial speed. The constant angular momentum $\mathbf{L}$ of this particle of mass μ is given by (decomposing $\mathbf{r}$ along and perpendicular to the z axis and using $|\mathbf{r}_\perp| \to b$ before the incident particle feels the central force, where b is the impact parameter of the particle considered).

$$\mathbf{L} = \mu\, \mathbf{r} \times \mathbf{v}_0 = \mu\, (\mathbf{r}_\parallel + \mathbf{r}_\perp) \times \mathbf{v}_0 = \mu\, \mathbf{r}_\perp \times \mathbf{v}_0 = -\mu\, b\, v_0\, \hat{\mathbf{k}} = -b\, \sqrt{(2\, \mu\, E)}\, \hat{\mathbf{k}}. \tag{2.27}$$

Thus $|\mathbf{L}| = b\, \mu\, v_0$. Using $v_0 = \sqrt{2\, E/\mu}$, we then get

$$\dot{\theta} = -\sqrt{\frac{2\, E}{\mu}}\, \frac{b}{r^2} = -v_0\, \frac{b}{r^2} \tag{2.28}$$

Solving Eq. (2.25) for $d\, r/d\, t = \dot{r}$

$$\dot{r} = \pm\, v_0\, (1 - V_{eff}(r)/E)^{1/2}. \tag{2.29}$$

From Eq. (2.23) we then have

$$d\, \theta = -\left(\frac{|\mathbf{L}|}{\mu\, r^2}\right)\, d\, t = -\left(\frac{|\mathbf{L}|}{\mu\, r^2}\right)\, \frac{d\, r}{\dot{r}}. \tag{2.30}$$

Combining Eqs. (2.29) and (2.30) yields

$$d\, \theta = \mp \left(\frac{|\mathbf{L}|}{\sqrt{2\, \mu}}\right)\, \frac{d\, r}{r^2\, \sqrt{E - V_{eff}(r)}} \tag{2.31}$$

We will later need to use the relation of the x and y components of the vector velocity to the radial and theta components, so we discuss that relation here. Using Eq. (2.19) and Eq. (2.21), and the equality

$$\mathbf{v} = \hat{\mathbf{i}}\, v_x + \hat{\mathbf{j}}\, v_y = \hat{\mathbf{r}}\, \dot{r} + \hat{\boldsymbol{\theta}}\, r\, \dot{\theta} \tag{2.32}$$

we find

$$v_x = \dot{r}\, \cos(\theta) - r\, \dot{\theta}\, \sin(\theta), \quad v_y = \dot{r}\, \sin(\theta) + r\, \dot{\theta}\, \cos(\theta) \tag{2.33}$$

2.2 The Differential Scattering Cross Section

We here specialize the two body problem to the case of scattering, in which one of the pair of particles is at rest at $r = 0$, and the other particle approaches the scattering center with some intial speed and hence initial energy equal to the inital kinetic energy. In general, as the incident particle approaches the scattering center, the particle will have both a radial velocity $\dot{r}$ and an angular velocity $r\,\dot{\theta}$. We see from Eq. (2.29) that the radial velocity $\dot{r}$ is zero at a radius r_{min} defined by the equation

$$E = V_{eff}(r_{min}). \tag{2.34}$$

For $r > r_{min}, E > V_{eff}$, and since $V_{eff}(r) \to 0$ as $r \to \infty$, a positive energy particle will approach the scattering center, reaching a minimum radial distance r_{min} and (in the absence of a capture mechanism) return to radial infinity. The polar angle θ is measured from the forward direction (the positive x axis).

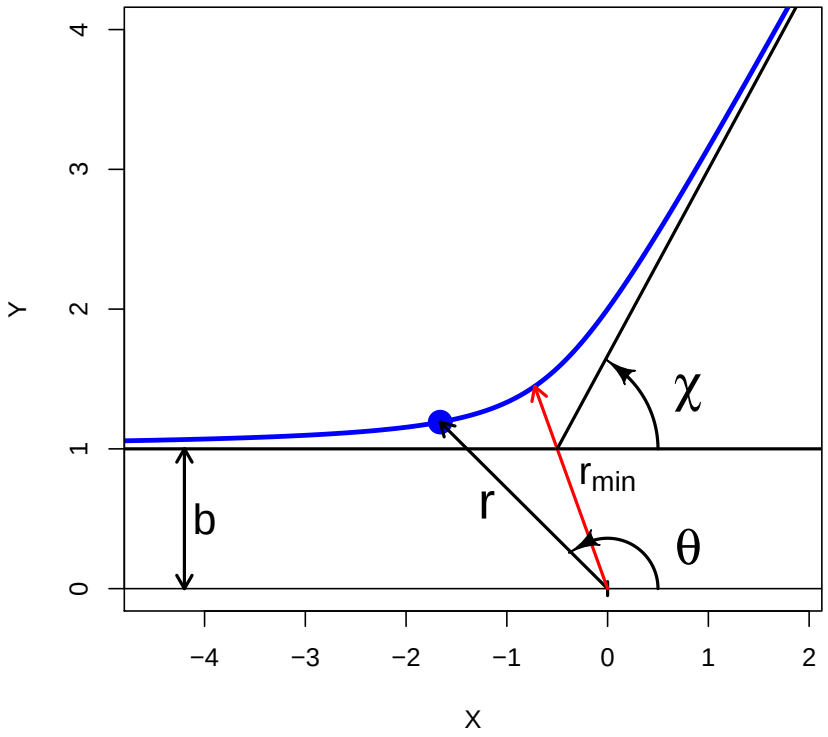

Figure 3: Scattering Angle χ Illustrated

Since the scattering of a given particle in a central potential takes place in a plane, we are free to choose the $z = 0$ plane, and the position of a incident particle can be described in terms of the coordinates (x, y), in which y is the perpendicular distance of the incident particle from the x axis. When the incident particle is at $x = -\infty, y = b$ (the particle's impact parameter), $r = \infty$, θ has a value arbitrarily close to π, and the value of θ decreases with time, with the scattered particle finally having either a positive value of θ, as shown in the figure, or a negative value of θ, in which case the asymptotic direction of the trajectory is below the forward direction.

We will denote the scattering angle (the final value of θ) by χ, which can be either positive or negative. Consider a beam of identical particles which approach the scattering center. We consider first only those particles which have the same kinetic energy when far away, (ie., a monoenergetic beam). We assume that the beam has a large diameter cross section; the relations derived will be correct for particles with small enough values of impact parameters b. We assume the number of beam particles per unit area (at $x \to -\infty$) is independent of ρ, the perpendicular distance from the x-axis; this assumption is clearly related to the previous one. If J is the incident beam flux density with units $number/(cm^2 sec)$,

then we are assuming that J is independent of the distance ρ from the beam axis.

We locate each beam particle during the scattering process using two different coordinate systems. We tag each beam particle with the pair of numbers (b, ϕ), choosing the positive x axis to be in the initial direction of motion of the beam particles and passing through the scattering center. The impact parameter b is the initial distance of the beam particle from the x axis, and ϕ is a cylindrical angle around the beam axis. A beam particle will maintain a constant value of ϕ and will be scattered into a final direction labeled by the two angles (ψ, ϕ). Here we are using spherical coordinates (r, ψ, ϕ), with r the radial distance from the scattering center, ψ is the (positive) polar angle measured from the positive x axis in the (x, y) plane, and ϕ is the azimuthal angle about the x axis.

Choose db such that beam particles having impact parameters in the interval $(b, b + db)$ and also having azimuthal angles in the interval $(\phi, \phi + d\phi)$ will be scattered into a small positive solid angle

$$d\,\Omega = sin\psi\, d\psi\, d\phi \tag{2.35}$$

subtended by the scattering center at the origin of coordinates. The number of beam particles scattered per second into a particular $d\Omega$ is written as $dN = J\, d\sigma$ and this will also be equal to $JdA = J\, b\, db\, d\phi$. Hence the positive quantity $d\sigma$, which has the dimension cm^2, can be written as

$$d\sigma = b\, db\, d\phi = \left(\frac{d\sigma}{d\Omega}\right)\, d\Omega \tag{2.36}$$

Using our expression for $d\Omega$ and cancelling $d\phi$, we obtain

$$\frac{d\sigma}{d\Omega} = \left|\frac{b}{sin\,\psi}\frac{db}{d\psi}\right| \tag{2.37}$$

At this stage, all reference to ϕ has dropped out, and we replace the positive angle ψ by the signed scattering angle χ, to get

$$\frac{d\sigma}{d\Omega} = \left|\frac{b}{sin\,\chi}\frac{db}{d\chi}\right| \tag{2.38}$$

For those impact parameters for which $d\chi/db = 0$, the differential scattering cross section is infinitely large.

2.3 The Scattering Angle Integral

The effective potential (Eq. 2.26) can be written (using Eq.2.27),

$$V_{eff} = V(r) + \frac{E\, b^2}{r^2}, \tag{2.39}$$

and the change in θ (Eq. 2.31) becomes

$$d\theta = \pm\left(b\sqrt{E}\right)\, \frac{dr}{r^2\sqrt{E - V_{eff}(r)}}. \tag{2.40}$$

Integrating both sides over corresponding intervals, during the first part of the trip from $r = \infty$ to $r = r_{min}$, $d\,\boldsymbol{\theta}$ is negative and $d\,r$ is also negative, while on the second part of the trip, from $r = r_{min}$ to $r = \infty$, $d\,\theta$ is negative and $d\,r$ is positive:

$$\int_{\pi}^{\chi} d\,\theta = \chi - \pi = \int_{\infty}^{r_{min}} f(r)\, dr - \int_{r_{min}}^{\infty} f(r)\, dr = -2\int_{r_{min}}^{\infty} f(r)\, dr. \tag{2.41}$$

The scattering angle χ as a function of (E, b) is then written as

$$\chi = \pi - 2\,\phi_{\infty} \tag{2.42}$$

where

$$\phi_{\infty} = b\int_{r_{min}}^{\infty} \frac{dr}{r^2\sqrt{1 - \frac{V_{eff}(r)}{E}}}. \tag{2.43}$$

The argument of the square root function in the denominator of the integrand vanishes at the lower limit of the integral, by definition of r_{min} (see Eq. 2.34).

2.4 Reduction to Dimensionless Form

By choosing a length scale a_0 either suggested by the form of the potential or the physical regime considered, we can define a dimensionless radial distance via $\bar{r} = r/a_0$ and a dimensionless impact parameter $\bar{b} = b/a_0$.

A dimensionless energy $\bar{E}$ can be defined in terms of the value of the potential energy evaluated at $r = a_0$, or $\bar{E} = E/V(a_0)$. A dimensionless potential energy as a function of $\bar{r}$ is $\bar{V}(\bar{r}) = V(r)/V(a_0)$, and a dimensionless effective potential energy is $\bar{V}_{eff}(\bar{r}) = V_{eff}(r)/V(a_0)$, in terms of which ϕ_∞ becomes

$$\phi_\infty = \bar{b} \int_{\bar{r}_{min}}^{\infty} \frac{d\bar{r}}{\bar{r}^2 \sqrt{1 - \bar{V}_{eff}(\bar{r})/\bar{E}}}. \tag{2.44}$$

where $\bar{r}_{min}$ is the value of $\bar{r}$ at which the argument of the square root function vanishes, and

$$\bar{V}_{eff}(\bar{r}) = \bar{V}(\bar{r}) + \bar{E}\,\frac{\bar{b}^2}{\bar{r}^2} \tag{2.45}$$

Since $E = \frac{1}{2}\,\mu\,v_0^2$, a natural unit of speed is $v_0 = \sqrt{\frac{2\,E}{\mu}}$. A dimensionless component of the velocity vector can then be defined as $\bar{v}_x = v_x/v_0$. A natural unit of time is then $t_0 = a_0/v_0$ and a dimensionless time is $\bar{t} = t/t_0 = v_0\,t/a_0$.

We can then write the x-component of the equation of motion

$$\mu\,\frac{d\,v_x}{d\,t} = -\frac{\partial\,V(r)}{\partial\,x} = -\frac{\partial\,r}{\partial\,x}\,\frac{d\,V(r)}{d\,r} = -\frac{x}{r}\,\frac{d\,V(r)}{d\,r} \tag{2.46}$$

in dimensionless form:

$$\frac{d\,\bar{v}_x}{d\,\bar{t}} = -\frac{\bar{x}}{2\,\bar{E}\,\bar{r}}\,\frac{d\,\bar{V}(\bar{r})}{d\,\bar{r}} \tag{2.47}$$

We can also write the first order differential equation for $\theta(t)$, Eq. (2.28) in dimensionless form:

$$\frac{d\,\theta}{d\,\bar{t}} = -\bar{b}/\bar{r}^2 \tag{2.48}$$

and the first order differential equation for $r(t)$, Eq. (2.29), in dimensionless form:

$$\frac{d\,\bar{r}}{d\,\bar{t}} = \pm\,\left(1 - \bar{V}_{eff}(\bar{r})/\bar{E}\right)^{\frac{1}{2}} \tag{2.49}$$

Using Eq. (2.33) we also have

$$\bar{v}_x = \cos(\theta)\,\frac{d\,\bar{r}}{d\,\bar{t}} - \sin(\theta)\,\bar{r}\,\frac{d\,\theta}{d\,\bar{t}}, \quad \bar{v}_y = \sin(\theta)\,\frac{d\,\bar{r}}{d\,\bar{t}} + \cos(\theta)\,\bar{r}\,\frac{d\,\theta}{d\,\bar{t}}. \tag{2.50}$$

The starting point for the calculation of the differential scattering cross section, Eq. (2.38), is written in dimensionless form by noting that σ has the dimensions of area, so we define $\bar{\sigma} = a_0^2\,\sigma$ to get

$$\frac{d\bar{\sigma}}{d\Omega} = \left|\frac{\bar{b}}{sin\,\chi}\frac{d\,\bar{b}}{d\chi}\right| \tag{2.51}$$

3 Repulsive Rutherford Scattering

Here we treat the case in which the potential energy is

$$V(r) = \frac{\alpha}{r}, \quad \alpha > 0, \quad V_{eff}(r) = \frac{\alpha}{r} + \frac{E\,b^2}{r^2}, \tag{3.1}$$

$$\bar{V}(\bar{r}) = \frac{1}{\bar{r}}, \quad \bar{V}_{eff}(\bar{r}) = \frac{1}{\bar{r}} + \frac{\bar{E}\,\bar{b}^2}{\bar{r}^2} \tag{3.2}$$

Since the potential energy contains no natural length, we make use of the fact that quantum theory predicts that the z component of angular momentum is quantized and can only have the discrete values $L_z = n\,\hbar$, where $n = 0, \pm 1, \pm 2 \ldots$. $\hbar$ is related to the famous "Planck's constant" h by $\hbar = h/(2\,\pi)$. If we use square brackets to denote "dimensions of",

$$[\hbar] = [r\,m\,v] = [r\,m\,v^2/v] = [t\,m\,v^2] = \text{erg-sec} \tag{3.3}$$

(using cgs (centimeter-gram-sec) units). The measured value of $\hbar$ is $1.055\ \times\ 10^{-27}$ erg-sec.

In cgs units, all physical quantities can be expressed as numbers with units which are some combination of mass (m), length (r), and time (t). Since $[\hbar] = [m\,r^2/t]$ and $[\alpha/r] = [E] = [m\,v^2]$, then $[\alpha] = [m\,r^3/t^2]$. If we then try to construct a quantity a_0 with the units of length out of the reduced mass μ of the two particle system, represented by m for the balancing, the strength of the interaction α, and $\hbar$, we need

$$[a_0] = [r] = [\hbar^x\,m^y\,\alpha^z] = [(m\,r^2/t)^x\,m^y\,(m\,r^3/t^2)^z] = [r^{2\,x+3\,z}\,m^{x+y+z}/t^{x+2\,z}] \tag{3.4}$$

Balancing dimensions between the left and right-hand sides leads to three equations

$$2\,x + 3\,z = 1, \quad x + y + z = 0, \quad x + 2\,z = 0 \tag{3.5}$$

Just for practice, we use Maxima's **`solve`** function:

```
(%i1) solve([2*x+3*z=1,x+y+z=0,x+2*z],[x,y,z]);
(%o1) [[x = 2,y = -1,z = -1]]
```

Then the combination

$$a_0 = \frac{\hbar^2}{\mu\,\alpha} \tag{3.6}$$

has the dimensions of a length, and we can define $\bar{x} = x/a_0$, for example.

3.1 Plot of Effective Potential

The function **`Veff_plot`** (from **`rutherford_repulse.mac`**) has the definition

```
Veff_plot(e,b,r0,r1,xmin,xmax,ymin,ymax) :=
block([w,r,rmin,energy_line,numer],numer:true,
    w : 1/e/b,
    rmin : b*(w + sqrt(4+w^2))/2,
    energy_line : [discrete,[[rmin,e],[r1,e]]],
    plot2d([1/r + e*b^2/r^2, energy_line],[r,r0,r1], [x,xmin,xmax],
           [y,ymin,ymax],[style,[lines,3,1],[lines,3,2]],
           [legend,false],[xlabel,"r"],[ylabel,"Veff"]))$
```

The parameter **rmin** is the minimum radial distance between the incident particle and the scattering center. The invocation

```
(%i3) Veff_plot(1,1,0.9,3,0.8,3,0,2)$
```

chooses $\bar{E} = 1$, $\bar{b} = 1$, and chooses to plot the expression defined by **veff** over the range **[r,0.9,3]**. The canvas horizontal extent is determined by the arguments **xmin** and **xmax**. The canvas vertical extent is determined by the arguments **ymin** and **ymax**. This produces a plot of both the effective potential energy and also the constant total energy of the incident particle (in red).

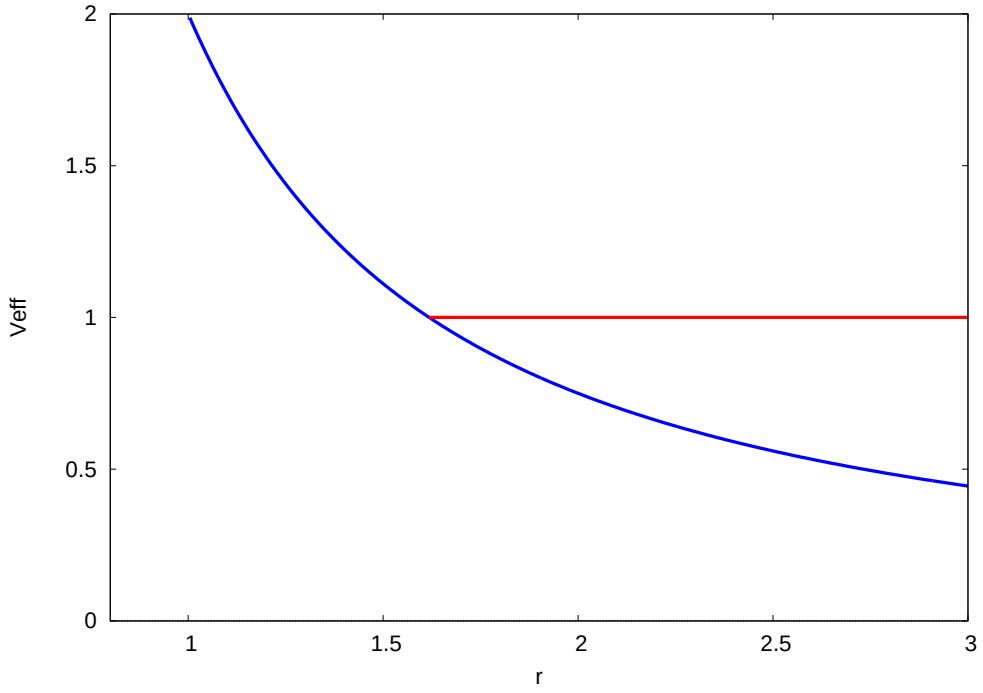

Figure 4: Effective Potential for $\bar{E} = 1$, $\bar{b} = 1$

3.2 Analytic Scattering Angle Using Maxima

From Eq. (2.44) we then have

$$\phi_\infty = \bar{b} \int_{\bar{r}_{min}}^{\infty} \frac{d\bar{r}}{\bar{r}^2 \sqrt{1 - 1/(\bar{E}\,\bar{r}) - \bar{b}^2/\bar{r}^2}}. \tag{3.7}$$

We replace the integration variable $\bar{r}$ by $z = \bar{r}/\bar{b}$, and define $w = 1/(\bar{E}\,\bar{b})$ to get

$$\phi_\infty = \int_{z_{min}}^{\infty} \frac{d\,z}{z\sqrt{z^2 - w\,z - 1}} \tag{3.8}$$

in which z_{min} is the positive value of z for which the argument of the square root is zero. This gives the result (see Maxima session just below)

$$\phi_\infty = \cos^{-1}\left(\frac{w}{\sqrt{4+w^2}}\right) = \frac{\pi}{2} - \sin^{-1}\left(\frac{w}{\sqrt{4+w^2}}\right), \quad w = 1/(\bar{E}\,\bar{b}). \tag{3.9}$$

and the scattering angle χ:

$$\chi = \pi - 2\,\phi_\infty = 2\,\sin^{-1}\left(\frac{w}{\sqrt{4+w^2}}\right) = 2\,\sin^{-1}\left(\frac{1}{\sqrt{1+4\,\bar{E}^2\,\bar{b}^2}}\right). \tag{3.10}$$

For given $\bar{E}$ as $\bar{b} \to \infty$, $w \to 0$ and $\chi \to 0$. And as $\bar{b} \to 0$, $w \to \infty$, and $\chi \to \pi = 180\,\mathrm{deg}$.

We find $z_{min} = \frac{1}{2}\left(w + \sqrt{4 + w^2}\right)$ and hence $\bar{r}_{min} = \frac{\bar{b}}{2}\left(w + \sqrt{4 + w^2}\right)$.

We let θ_0 be the positive angle between the positive x-axis and the direction of $\mathbf{r}(\mathbf{t})$ at the moment the incident particle is at its minimum distance from the scattering center (the origin of the coordinate system). From the figure, $\theta_0 + \phi_\infty = \pi$, so $\theta_0 = \pi - \phi_\infty$.

Here is a short Maxima session to find the above results.

```
(%i1) rs : solve(z^2 - w*z -1,z);
(%o1) [z = -(sqrt(w^2+4)-w)/2,z = (sqrt(w^2+4)+w)/2]
(%i2) zmin : rhs(rs[2]);
(%o2) (sqrt(w^2+4)+w)/2
(%i3) assume(w>0);
(%o3) [w > 0]
(%i4) phi_inf : integrate(1/(z*sqrt(z^2 - w*z -1)),z,zmin,inf);
(%o4) %pi/2-asin(w/sqrt(w^2+4))
```

The coordinates of the point of closest approach (C) to the scattering center (O) are $\bar{x}_c = \bar{r}_{min}\cos(\theta_0)$ and $\bar{y}_c = \bar{r}_{min}\sin(\theta_0)$.

The intersection (A) of the line OC and the line $\bar{y} = \bar{b}$ is defined by the coordinates $\bar{x}_a = \bar{b}\,\bar{x}_c/\bar{y}_c$ and $\bar{y}_a = \bar{b}$.

The hyperbolic orbit is symmetric about the point of closest approach. The asymptote of the incident particle as it approaches is the line $\bar{y} = \bar{b}$. The asymptote of the particle as it retreats to positive infinity is the line AG (in the limit $\bar{r} \to \infty$) $(\bar{x} = \bar{x}_a + \cos(\chi), \bar{y} = \bar{b} + \sin(\chi))$ which has its origin at point A and makes an angle χ with the positive x-axis.

Scattering Angle as a Function of Impact Parameter Using Maxima

Using Eq. (3.10), we can find the scattering angle for different (dimensionless) impact parameters, for a given (dimensionless) energy. In our code we represent $\bar{E}$ by **e**, and represent $\bar{b}$ by **b**.

```
(%i1) fpprintprec:8$
(%i2) angle_a(e,b) :=
block([numer],numer:true,
  2*asin(1/sqrt(1 + 4*e^2*b^2))*180/%pi)$
(%i3) for b in [10,5,1.5,1,0.8,0.5,0.3,0.1,0.01,0.001] do
      print(" ",b," ",angle_a(1,b))$
  10   5.7248105
  5   11.421186
  1.5   36.869898
  1   53.130102
  0.8   64.010766
  0.5   90.0
  0.3   118.07249
  0.1   157.38014
  0.01   177.70847
  0.001   179.77082
```

and we can then make a simple plot of the scattering angle in degrees versus the (dimensionless) impact parameter.

```
(%i4) plot2d(angle_a(1,b),[b,0.001,10],[ylabel,"chi-degrees"],
        [style,[lines,3]])$
```

which produces (roughly) the plot

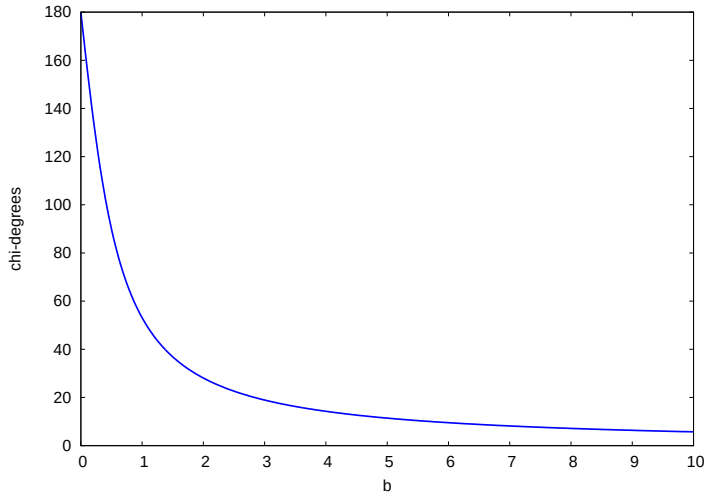

Figure 5: Analytic Scattering Angle Versus b for $e = 1$

3.3 Numerical Scattering Angle Using Maxima

Although we have analytic results for the case of Rutherford scattering, it will be instructive to pretend that no analytic solution can be found and develop numerical methods which allow calculation of the numerical scattering angle as a function of energy and impact parameter.

Here is an example function, **`angle_n(e,b)`**, which is used to make a table similar to the analytic table above.

```
(%i1) angle_n(e,b) :=
block([z,w,zmin,phi_inf,rexpr,iexpr,numer],numer:true,
    w : 1/(b*e),
    rexpr : z^2 - w*z -1, /* root -> zmin */
    iexpr : 1/(z*sqrt(rexpr)),        /* integrate to get phi_inf */
    zmin : find_root(rexpr,z,1e-4,1e6),
    phi_inf : quad_qagi(iexpr,z,zmin,inf)[1],
    (%pi - 2*phi_inf)*180/%pi)$
(%i2) bval : [10,5,1.5,1,0.8,0.5,0.3,0.1,0.01,0.001]$
(%i3) for b in bval do
      print(" ",b," ",angle_n(1,b))$
  10   5.7248105
  5   11.421186
  1.5   36.869898
  1   53.130102
  0.8   64.010766
  0.5   90.0
  0.3   118.07249
  0.1   157.38014
  0.01   177.70847
  0.001   179.77082
```

We can then make a plot of scattering angle as a function of impact parameter.

```
(%i4) chival : map(lambda([x],angle_n(1,x)),bval)$
(%i5) fll(chival);
(%o5) [5.724810451867294,179.7708171875211,10]
(%i6) plot2d([discrete,bval,chival],[style,[lines,3]],[xlabel,"b"],
          [ylabel,"chi-deg"])$
```

which produces the less than perfect plot:

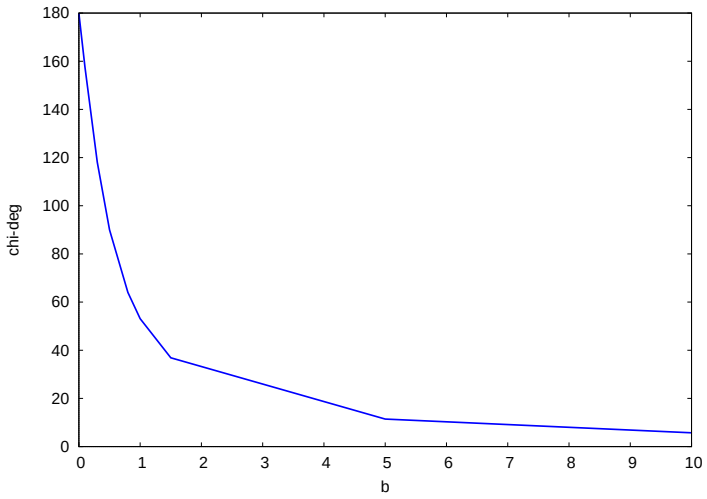

Figure 6: Numerical Scattering Angle Versus $\bar{b}$ for $\bar{E} = 1$

Clearly, we can add more sample points (by choosing more $\bar{b}$ values) to get a smoother plot.

In the above calculation, we made use of Maxima's **`lambda`** function which allows us to avoid defining a named function. The alternative, "named function" method could look like:

```
(%i7) func(x) := angle_n(1,x)$
(%i8) chival : map(func,bval);
```

Both of the above methods of turning a Maxima list into another list avoid the less efficient loop solution, which might look like:

```
(%i9) chival:[]$
(%i10) for b in bval do (
         chival : cons(angle_n(1,b),chival))$
(%i11) chival : reverse(chival)$
(%i12) fll(chival);
(%o12) [5.724810451867294,179.7708171875211,10]
```

3.4 Numerical Scattering Angle Using R

Here we will first translate the Maxima function **`angle(e,b)`** into **`R`**. In the file **`rutherford_repulse.mac`** we have the Maxima code

```
/* some (e,b) dependent angles and rmin.
    theta0 defines the angle of closest approach
       (counter-clockwise from positive x axis)
    phi_inf defines the rotation angle of r-vec as the
       incident particle comes from theta=%pi, r = inf to
       the point of closest approach, so
         theta0 = %pi - phi_inf.
    chi is the scattering angle
        chi = %pi - 2*phi_inf
    rmin is the distance of closest approach
     to the scattering center = ( x = 0, y = 0 )
  */
```

```
angles(e,b) :=
block([w,phi_inf,theta0,chi,oldfp,rmin,numer],numer:true,
    oldfp : fpprintprec,
    fpprintprec : 8,
    w : 1/(e*b),
    phi_inf : acos(w/sqrt(4+w^2)),
    print(" phi_inf = ",phi_inf," rad, or ",phi_inf*180/%pi," deg"),
    theta0 : %pi - phi_inf,
    print(" theta0 = ",theta0," rad, or ",theta0*180/%pi," deg"),
    chi : %pi - 2*phi_inf,
    print(" chi = ",chi," rad, or ",chi*180/%pi," deg"),
    rmin : b*(w + sqrt(4+w^2))/2,
    print(" rmin = ",rmin),
    fpprintprec : oldfp,
    done)$
```

To translate this function, which makes use of the analytic results derived using Maxima above, we can follow the steps in the code: 1.) replace **`:`** with **`=`**, 2.) remove commas at the ends of complete statements, 3.) replace **`print(...)`** with **`cat(... "\n")`**, 4.) replace **`%pi`** with **`pi`**.

Of course, there is the more basic replacement: **`angles(e,b) := block([local variables], code)$`** replaced with **`angles = function(e,b) {code }`**.

We don't need to declare local variables in the **R** function definition, and the default number of digits printed out for floating point numbers needs no attention.

We write this translation inside Notepad2 (or Notepad++) first, and then paste the result into Rgui.

```
> angles = function(e,b) {
+    w = 1/e/b
+    phi_inf = acos(w/sqrt(4+w^2))
+    cat(" phi_inf = ",phi_inf," rad, or ",phi_inf*180/pi," deg\n")
+     theta0 = pi - phi_inf
+     cat(" theta0 = ",theta0," rad, or ",theta0*180/pi," deg\n")
+     chi = pi - 2*phi_inf
+     cat(" chi = ",chi," rad, or ",chi*180/pi," deg\n")
+     rmin = b*(w + sqrt(4+w^2))/2
+     cat(" rmin = ",rmin,"\n")}
> angles(1,1)
 phi_inf =  1.107149  rad, or  63.43495  deg
 theta0 =  2.034444  rad, or  116.5651  deg
 chi =  0.9272952  rad, or  53.1301  deg
 rmin =  1.618034
```

Next we translate **`angle_n(e,b)`**, which produces, using numerical means, the scattering angle converted into degrees, into R syntax and paste it into **R**. We can then print out a simple table of corresponding values of $\bar{b}$ and χ.

```
> angle_n = function(e,b) {
+      w = 1/(b*e)
+      fr = function(z) z^2 - w*z -1  #  root -> zmin
+      zmin = uniroot(fr, c(1e-4, 1e6),tol=1e-10)$root
+      phi_inf = integrate(function(z) 1/z/sqrt(fr(z)), zmin, Inf)$val
+      (pi - 2*phi_inf)*180/pi}
> angle_n(1,1)
[1] 53.1301
```

```
> bval = c(10,5,1.5,1,0.8,0.5,0.3,0.1,0.01,0.001)
> for (b in bval) cat(" ",b," ",angle_n(1,b),"\n")
  10   5.72481
  5   11.42119
  1.5   36.8699
  1   53.1301
  0.8   64.01077
  0.5   90
  0.3   118.0725
  0.1   157.3801
  0.01   177.7085
  0.001   179.7709
```

We can then make a simple plot of the above values of the scattering angles, using the **R** function **sapply** to apply a function to a vector, returning a vector. (**newvec = sapply (oldvec, func)**.)

```
> chival = sapply(bval,function(x) angle_n(1,x))
> head(chival)
[1]  5.723923 11.420227 37.289532 53.351958 64.009976 90.253342
> angle_n(1,10)
[1] 5.723923
> angle_n(1,5)
[1] 11.42023
> tail(chival)
[1]  64.00998  90.25334 118.07137 157.37981 177.70850 179.77094
> plot(bval,chival,type="l",lwd=3,col="blue",xlab = "b",
+       ylab = "chi")
> abline(v=0)
```

which produces the same crude plot (because we don't have enough samples)

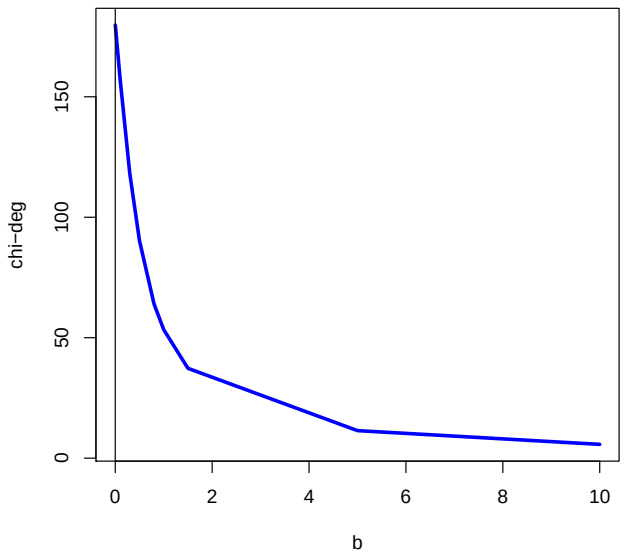

Figure 7: χ in degrees versus $\bar{b}$, $\bar{E} = 1$

3.5 Review of Maxima's Runge-Kutta rk() with the Simple Harmonic Oscillator

We will use Maxima's Runge-Kutta integrator **rk()** for plotting the scattering trajectory. Here we review the use of **rk()** in the context of the simple harmonic oscillator with unit period and intial conditions $x(0) = 1$ and $v_x(0) = 0$. The utility functions **fll**, **head**, **tail**, and **take** are loaded by **maxima-init.mac** and are in the file **k1util.mac**. The syntax used with **rk** here to describe the pair of first order differential equations $d\,x/d\,t = v_x,\ d\,v_x/d\,t = -4\,\pi^2\,x$ is

```
rk( [dx/dt,dvx/dt],[x,vx],[x0,vx0], [t,t0,tmax,dt] )
```

and in this example, **rk()** returns a list of the form **[[t0, x0, vx0],[t0 + dt, x1, vx1],...]**.

```
(%i1) pts : rk([vx,-4*%pi^2*x],[x,vx],[1,0],[t,0,1,0.01])$
(%i2) fll(pts);
(%o2) [[0.0,1.0,0.0],[1.0,0.99999995729235,5.1201813129342355E-6],101]
(%i3) tL : take(pts,1)$
(%i4) fll(tL);
(%o4) [0.0,1.0,101]
```

Here we make a plot of $x(t)$ versus t.

```
(%i5) xL : take(pts,2)$
(%i6) fll(xL);
(%o6) [1.0,0.99999995729235,101]
(%i7) plot2d([discrete,tL,xL],[x,0,1],[xlabel,"T"],[ylabel,"X"],
          [style,[lines,3]])$
```

Here we make a plot of $v_x(t)$ versus t.

```
(%i8) vxL : take(pts,3)$
(%i9) fll(vxL);
(%o9) [0.0,5.1201813129342355E-6,101]
(%i10) plot2d([discrete,tL,vxL],[x,0,1],[xlabel,"T"],[ylabel,"Vx"],
            [style,[lines,3]])$
```

3.6 Scattering Trajectory Plot Using Maxima

The file **rutherford_repulse.mac** contains the function **orbit_plot1(e,b,tm,tp,dt,rchi,xmin,xmax,ymin,ymax)** which provides maximum flexibility for the task of drawing a scattering orbit. The code uses the Maxima fourth-order Runge-Kutta integrator **rk**.

Here are three examples of use, first for the case $e = 1$ and $b = 1$ (χ, the scattering angle, is represented by **chi** in the code).

```
(%i1) orbit_plot1(1,1,5,5,0.01,5,-4.54,1.87,0,4)$
 rmin =  1.618034
 phi_inf =  1.1071487  rad or  63.434949  deg
 theta0 =  2.0344439  rad or  116.56505  deg
 chi =  0.927295  rad, or  53.130102  deg
 xc =  -0.723607  yc =  1.4472136
 vcx =  0.552786  vcy =  0.276393
 xa =  -0.5
 backwards from xc, yc
 xfirst =  -4.536487
 forwards from xc, yc
 xlast =  1.8729016  ylast =  4.2659325
 vx_last =  0.54218  vy_last =  0.701
 xf =  2.5  yf =  5.0
```

which produces:

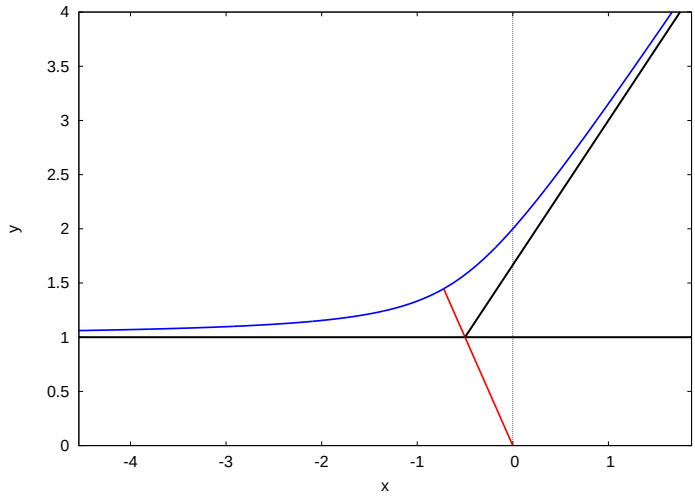

Figure 8: $e = 1$, $b = 1$, $\chi = 53\,\mathrm{deg}$

Next for $e = 1$ and $b = 0.6$

```
(%i2) orbit_plot1(1,0.6,5.4,5,0.01,5,-4.54,1.87,0,4)$
 rmin =  1.281025
 phi_inf =  0.876058  rad or  50.194429  deg
 theta0 =  2.2655346  rad or  129.80557  deg
 chi =  1.3894766  rad, or  79.611142  deg
 xc =  -0.820092  yc =  0.984111
 vcx =  0.359816  vcy =  0.299846
 xa =  -0.5
 backwards from xc, yc
 xfirst =  -4.7366598
 forwards from xc, yc
 xlast =  0.162247  ylast =  4.4261007
 vx_last =  0.167226  vy_last =  0.86386
 xf =  0.401639  yf =  5.5180328
```

which produces

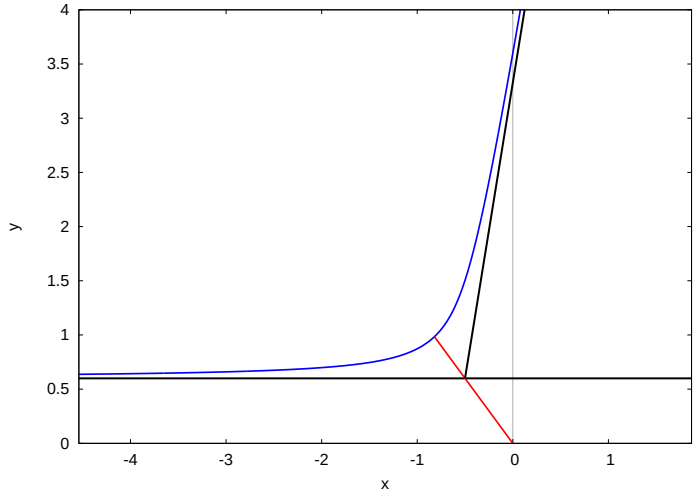

Figure 9: $e = 1$, $b = 0.6$, $\chi = 79.6\,\mathrm{deg}$

Next for $e = 1$ and $b = 0.3$

```
(%i3) orbit_plot1(1,0.3,5.4,5.5,0.01,5,-4.54,1.87,0,4)$
 rmin =  1.0830952
 phi_inf =  0.54042  rad or  30.963757  deg
 theta0 =  2.6011732  rad or  149.03624  deg
 chi =  2.0607537  rad, or  118.07249  deg
 xc =  -0.928746  yc =  0.557248
 vcx =  0.142507  vcy =  0.237512
 xa =  -0.5
 backwards from xc, yc
 xfirst =  -4.6088375
 forwards from xc, yc
 xlast =  -2.4910817  ylast =  3.9952223
 vx_last =  -0.414274  vy_last =  0.784846
 xf =  -2.8529412  yf =  4.7117647
```

which produces

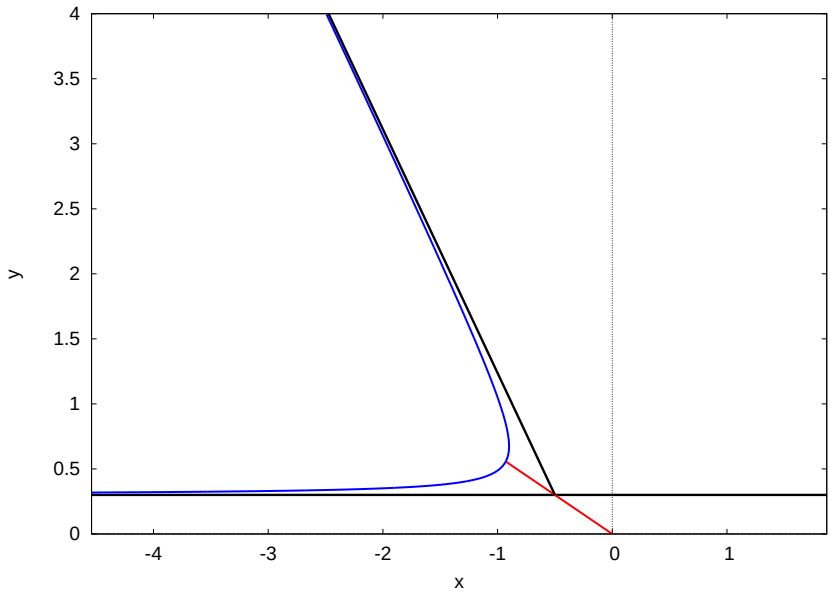

Figure 10: $e = 1$, $b = 0.3$, $\chi = 118\,\mathrm{deg}$

The code used to plot the Rutherford scattering trajectory given a value of $\bar{E}$ (represented by **e** in the code) and $\bar{b}$ (represented by **b** in the code), first calls **init(e,b)** to calculate local (to the calling function) values of χ (represented by **chi**), the cartesian coordinates $(\bar{x}_c, \bar{y}_c)$ of the point of closest approach (C), the x-coordinate $\bar{x}_a$ of the intersection of the line $\bar{y} = \bar{b}$ and the line OC, and finally the cartesian velocity components at point C (the point of closest approach). Since $d\bar{r}/d\bar{t} = 0$ at the point C, Eq. (2.50) then implies that

$$\bar{v}_{cx} = \bar{b}\,\sin(\theta_0)/\bar{r}_{min}, \quad \bar{v}_{cy} = -\bar{b}\,\cos(\theta_0)/\bar{r}_{min} \tag{3.11}$$

The code then uses Maxima's **rk** function to integrate both forward in time from point C for the part of the orbit in which $d\bar{r}/d\bar{t} > 0$ and also backward in time from point C for the part of the orbit in which $d\bar{r}/d\bar{t} < 0$.

Thus, inside the function **orbit_plot1** are the lines ($\bar{x}$ is represented by **x**, etc.)

```
    init(e,b),

    /* symbolic expressions for acceleration components */
    rmag : sqrt(x^2 + y^2),  /* rmag in terms of x and y */
    dvxdt : x/2/e/rmag^3,  /* dvx/dt repulsive rutherford case */
    dvydt : y/2/e/rmag^3,  /* dvy/dt */


    /* integrate backwards from xc, yc */
    rkpts : rk([vx, dvxdt, vy, dvydt ],
             [x,vx,y,vy],[xc,vcx,yc,vcy],[t,0,-tm,-dt]),
    xL : take(rkpts,2),
    yL : take(rkpts,4),
    pm : [discrete, xL, yL],
```

which will be the points plotted for the earlier part of the trajectory, and then the lines

```
/* integrate forwards from xc,yc */

    rkpts : rk([vx, dvxdt, vy, dvydt ],
             [x,vx,y,vy],[xc,vcx,yc,vcy],[t,0,tp,dt]),
    xL : take(rkpts,2),
    yL : take(rkpts,4),
    pp : [discrete, xL, yL],
```

which will be the points plotted for the later part of the trajectory. In both functions, **(xc, yc)** represent $(\bar{x}_c, \bar{y}_c)$ and **(vcx, vcy)** represent $(\bar{v}_{cx}, \bar{v}_{cy})$ and these are locally available after calling **init(e,b)**.

In this function, a homemade function **take(mlist,n)** is used to create a list **xL** of the x coordinates and a list **yL** of the y coordinates, using the syntax **xL : take(pts, 2)** and **yL : take(pts, 4)** since **rk()**, in this example, returns a list with elements of the form **[t, x, vx, y, vy]**.

The function **take(mL,n)** has the definition

```
take(%aL,%nn) := (map(lambda([x],part(x,%nn)), %aL))$
```

The syntax used with **rk** here is

```
rk( [dx/dt,dvx/dt,dy/dt,dvy/dt],[x,vx,y,vy],[x0,vx0,y0,vy0], [t,t0,tmax,dt] )
```

in which the first argument is the list of the right-hand sides of four first-order differential equations and is based on Eqs. (2.47) and (3.2), leading to, for example

$$\frac{d\,\bar{v}_x}{d\,\bar{t}} = \frac{\bar{x}}{2\,\bar{E}\,\bar{r}^3} = \frac{\bar{x}}{2\,\bar{E}\,\left(\bar{x}^2 + \bar{y}^2\right)^{3/2}} \tag{3.12}$$

3.7 Review of R's ode() with the Simple Harmonic Oscillator

The file **sho1.R** contains **R** code to integrate the simple harmonic oscillator system with unit period and intial conditions **x0 = 1, vx0 = 0** and make a plot of both the position **x** and the velocity component **vx** as functions of the time.

```
## sho1.R
## simple harmonic oscillator with period = 1
## produces side by side plots of
##  x vs t and vx vs t
yini = c(x = 1, vx = 0)
times = seq(0, 1, 0.001)
sho = function(t, y, parms) {
     with( as.list(y), {
        dx = vx
        dvx = -4*pi^2*x
        list( c(dx, dvx) ) } ) }
out = ode(times = times, y = yini, func = sho, parms = NULL)
plot(out, lwd = 2)
```

We first load in the **deSolve** package, and then load and run this script with **source**, as usual:

```
> library(deSolve)
> source("c:/k1/sho1.R")
> head(out)
      time         x          vx
[1,] 0.000 1.0000000  0.00000000
[2,] 0.001 0.9999803 -0.03947803
[3,] 0.002 0.9999210 -0.07895450
[4,] 0.003 0.9998224 -0.11842785
[5,] 0.004 0.9996842 -0.15789672
[6,] 0.005 0.9995066 -0.19735930
```

which produces the plot

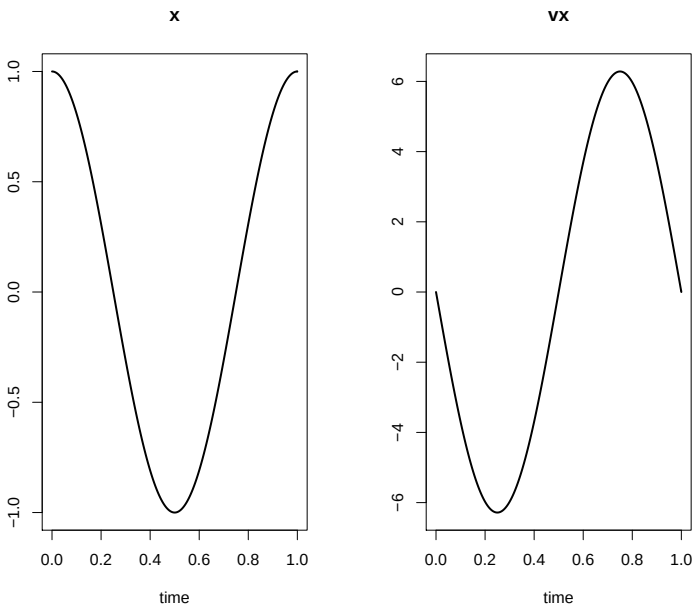

Figure 11: Position and Velocity for SHO with Unit Period

The side effect of loading in the package **deSolve** is to set up some plot defaults which have been accepted in the above example.

We can then return to one plot per row, and plot the velocity component versus the position (a phase space plot) with

```
> par(mfrow = c(1,1))
> plot(out[,"x"], out[,"vx"],type = "l",lwd = 3,col = "blue",
+    xlab = "X", ylab = "Vx")
```

which produces the single plot

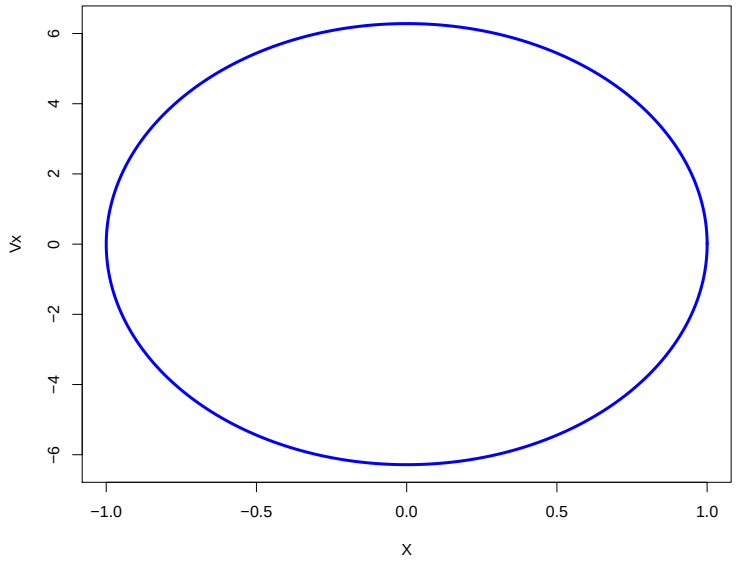

Figure 12: Velocity vs. Position for SHO with Unit Period

R vectors containing the discrete positions and velocity components can be extracted from the **`data.frame`** produced by **`ode`**.

```
> xL = out[,"x"]
> head(xL)
[1] 1.0000000 0.9999803 0.9999210 0.9998224 0.9996842 0.9995066
> vxL = out[,"vx"]
> head(vxL)
[1]  0.00000000 -0.03947803 -0.07895450 -0.11842785 -0.15789672
[6] -0.19735930
```

The **R** vector containing the discrete times can also be extracted

```
> tL = out[,1]
> head(tL)
[1] 0.000 0.001 0.002 0.003 0.004 0.005
```

and we could have used **`out[,2]`** to get the positions, etc.

We are free to use **y** as a variable inside the derivative function, as in

```
## sho3.R
## simple harmonic oscillator with period = 1
## produces side by side plots of
##  y vs t and vy vs t
 yini = c(y = 1, vy = 0)
 times = seq(0, 1, 0.001)
 sho = function(t, y, parms) {
     with( as.list(y), {
        dy = vy
        dvy = -4*pi^2*y
        list( c(dy, dvy) ) } ) }

 out = ode(times = times, y = yini, func = sho, parms = NULL)
 plot(out, lwd = 2)
```

We get the same plots as before, and we can access the elements of the **data.frame** as before. Here we also define the **R** function **fll** for use.

```
> library(deSolve)
> source("c:/k1/sho3.R")
> fll = function(xL) {
+     xlen = length(xL)
+     cat(" ",xL[1]," ",xL[xlen]," ",xlen,"\n") }
> head(out)
     time         y          vy
[1,] 0.000 1.0000000  0.00000000
[2,] 0.001 0.9999803 -0.03947803
[3,] 0.002 0.9999210 -0.07895450
[4,] 0.003 0.9998224 -0.11842785
[5,] 0.004 0.9996842 -0.15789672
[6,] 0.005 0.9995066 -0.19735930
> tL = out[,1]
> fll(tL)
  0   1   1001
> yL = out[,"y"]
> fll(yL)
  1   1   1001
> vyL = out[,"vy"]
> fll(vyL)
  0   3.247125e-07   1001
> tail(yL,n=2)
[1] 0.9999803 1.0000000
```

3.8 Scattering Trajectory Plot Using R

The file **rutherford_repulse.R** contains the function

```
    orbit_plot1(e,b,tm,tp,dt,rchi,xmin,xmax,ymin,ymax)
```

which provides much flexibility for the task of drawing a scattering orbit. The code uses the **deSolve R** package integrator **ode()**, and the **deSolve** package must be loaded, using **library(deSolve)** before using the part of the code which calls **ode()**.

Here is one example, for the case $e = 1$ and $b = 1$. The scattering angle χ is represented by **chi** in the code.

```
> source("c:/k1/rutherford_repulse.R")
> library(deSolve)
> orbit_plot1(1,1,5.5,5,0.01,5,-4.54,1.87,0,4)
 rmin =  1.618034
 phi_inf =  1.107149  rad or  63.43495  deg
 theta0 =  2.034444  rad or  116.5651  deg
 chi =  0.9272952  rad, or  53.1301  deg
 xc =  -0.7236068  yc =  1.447214
 vcx =  0.5527864  vcy =  0.2763932
 xa =  -0.5
  backwards from xc, yc
 xfirst =  -4.982187
  forwards from xc, yc
 xlast =  1.872902  ylast =  4.265932
 vx_last =  0.5421801  vy_last =  0.7009998
 xf =  2.5  yf =  5
```

which produces the scattering trajectory plot

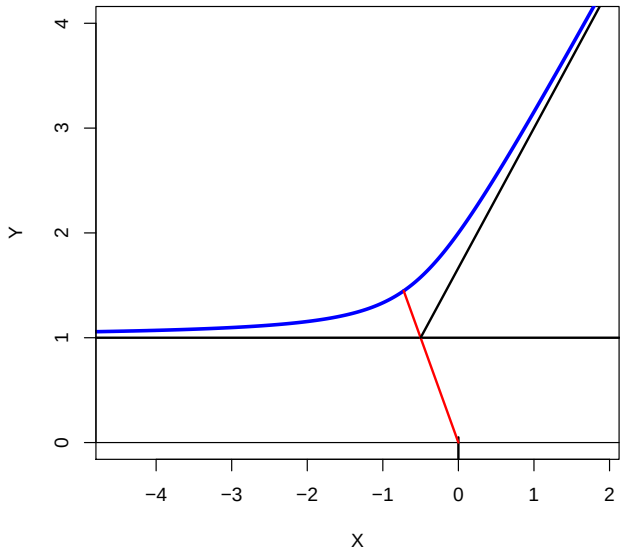

Figure 13: $e = 1$, $b = 1$, $\chi = 53\,\mathrm{deg}$

See the discussion in Sec. (3.6), in which Maxima is used for plots of trajectories, for some details, which still form the basis for the **R** code, with some obvious translations and differences, especially in how the final plot is built up in stages.

The **R** version of **init(e,b)** defines some parameters and makes them visible to the calling environment (in this case **orbit_plot1**) by using the special assignment syntax **a <<- 10**, for example. (In the code comments, the word 'global' refers to parameter visibility in the calling environment.)

```
##  init(e,b)  specific to repulsive rutherford case,
##    uses analytic expressions to produce global definitions
##    of chi,xc,yc, vcx,vcy,and xa. Besides printing out
##    these values, init(e,b) also prints
##    out the values of rmin, phi_inf, and theta0.
```

```
init = function(e,b) {
    w = 1/e/b
    rmin = b*(w + sqrt(4+w^2))/2   # repulsive case
    cat(" rmin = ",rmin,"\n")
    phi_inf = acos(w/sqrt(4+w^2))   # repulsive case
    cat(" phi_inf = ", phi_inf," rad or ", phi_inf*180/pi," deg\n")
    theta0 = pi - phi_inf
    cat(" theta0 = ",theta0," rad or ", theta0*180/pi," deg\n")
    chi <<-  pi - 2*phi_inf   # global parameter chi in radians
    cat(" chi = ", chi, " rad, or ", chi*180/pi," deg\n")
    # point of closest approach
    xc <<-  rmin*cos(theta0)    # global
    yc <<-   rmin*sin(theta0)      # global
    vcx <<-   b*sin(theta0)/rmin    # global
    vcy <<-    -b*cos(theta0)/rmin    # global
    cat(" xc = ", xc," yc = ", yc,"\n")
    cat(" vcx = ", vcx," vcy = ", vcy,"\n")
    #   x-intersection of rmin line with y=b line
    xa <<-   xc*b/yc    # global
    cat(" xa = ",xa,"\n") }
```

Then, inside the **R** function **orbit_plot1** are the lines:

```
    # define local xc, yc, vcx, vcy, xa, chi
    init(e,b)

    trajec = function(t, y, parms) {
      with( as.list(y), {
        r = sqrt(x^2 + y^2)
        dx = vx
        dvx = x/2/e/r^3   # repulsive case
        dy = vy
        dvy = y/2/e/r^3   # repulsive case
        list( c(dx, dvx, dy, dvy) ) } ) }

    #   integrate backwards from xc, yc

    yini = c(x = xc, vx = vcx, y = yc, vy = vcy)
    times = seq(0, -tm, -dt)
    out = ode(times = times, y = yini, func = trajec, parms = NULL)
    xL = out[,"x"]
    yL = out[,"y"]
    plot(xL, yL, xlim = c(xmin, xmax), ylim = c(ymin, ymax),
          type = "l", col = "blue", lwd = 3, xlab = "X",
          ylab = "Y")

    #   integrate forwards from xc,yc

    times = seq(0, tp, dt)
    out = ode(times = times, y = yini, func = trajec, parms = NULL)
    xL = out[,"x"]
    yL = out[,"y"]
    lines(xL, yL, lwd = 3, col = "blue")
```

as well as lines for other graphic elements and orbit detail printouts.

3.9 Analytic Differential Cross Section vs. Scattering Angle Using Maxima

Using Eqs. (2.51) and (3.10), we have

$$\sin(\chi/2) = \frac{1}{\left(1 + 4\,(\bar{E}\,\bar{b})^2\right)^{1/2}} \tag{3.13}$$

so a right triangle has 1 on the side opposite $\chi/2$ and $\left(1 + 4\,(\bar{E}\,\bar{b})^2\right)^{1/2}$ on the hypotenuse, and hence $2\,\bar{E}\,\bar{b}$ on the side adjacent to $\chi/2$. Hence

$$\cot(\chi/2) = 2\,\bar{E}\,\bar{b}, \quad \bar{b} = \frac{\cot(\chi/2)}{2\,\bar{E}} \tag{3.14}$$

We then need the first derivative $d\,\bar{b}/d\,\chi$.

```
(%i7) diff(cot(x/2)/2/e,x);
(%o7) -csc(x/2)^2/(4*e)
```

so

$$\frac{d\,\bar{b}}{d\,\chi} = -\frac{1}{4\,\bar{E}\,\sin^2(\chi/2)} \tag{3.15}$$

Hence

$$\frac{d\,\bar{\sigma}}{d\,\Omega} = \frac{1}{16\,\bar{E}^2\,\sin^4(\chi/2)}, \quad \text{or} \quad \frac{d\,\sigma}{d\,\Omega} = \left(\frac{\alpha}{4\,E}\right)^2 \frac{1}{\sin^4(\chi/2)} \tag{3.16}$$

If we plot the natural log of the differential scattering cross section (using dimensionless units) versus the scattering angle χ, with $\bar{E} = 1$, using the code

```
(%i6) plot2d(log(1/(16*sin(x/2)^4)),[x,0.01,1],[xlabel,"chi"],
          [ylabel,"ln(dsigma/do)"],[style,[lines,3]])$
```

we get:

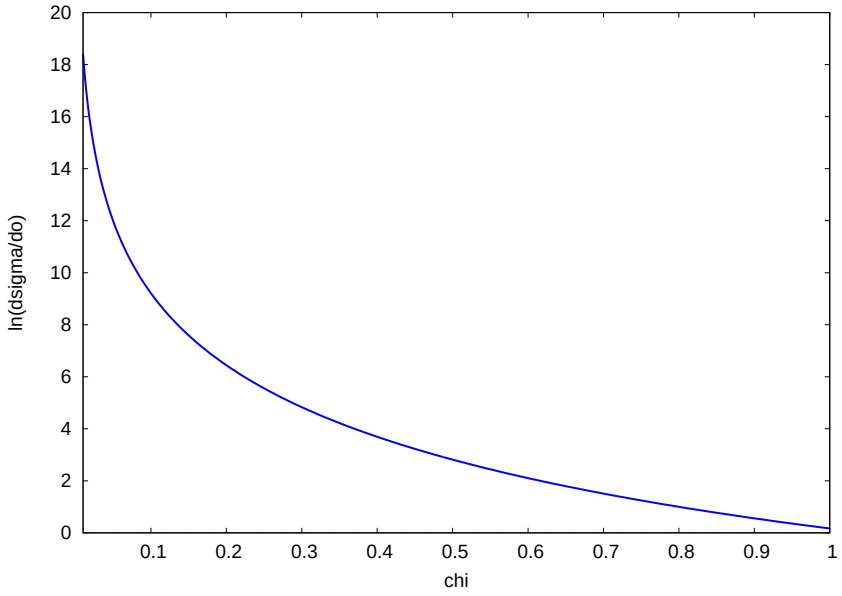

Figure 14: $\ln(d\,\sigma/d\,\Omega)$ versus χ, $\bar{E} = 1$

3.10 Numerical Differential Cross Section vs. Scattering Angle Using Maxima

We pretend we do not have the analytic result for the scattering angle as a function of the energy and impact parameter, and use numerical methods to plot the natural logarithm of the predicted (dimensionless) differential cross section versus the scattering angle. This provides an opportunity to use our results for the numerical approximation of a first derivative.

sigma_points calls **achi(e,b)** to construct a list **chiL** of scattering angles (in radians) corresponding to a set of values of the dimensionless impact parameter $\bar{b}$. The energy **e**, the starting impact parameter value **b0**, the maximum impact parameter value **bmax**, and the impact parameter increment **db** are inputs. The function **f1d(num,dx,funcL)** is called with the syntax **f1d(nb,db,chiL)** to obtain a list of first derivatives $d\,\chi/d\,\bar{b}$ at the impact parameter grid points and stored in the list **dchi_dbL**. **sigma_points** returns a list of lists: **[chi-list, log(dsig)-list]**. The value of each element of **sigL** in the code is based on

$$d\,\bar{\sigma}/d\,\Omega = \frac{\bar{b}}{\sin(\chi)}\,\frac{1}{|\,d\,\chi/d\,\bar{b}\,|} \tag{3.17}$$

The three functions just mentioned are defined in **rutherford_repulse.mac**.

```
/* sigma_points(e,b0,bmax,db) produces a list of two lists:
    [chi-list, log(d_sig/d_omega)-list] using numerical methods.
    Typical list arithmetic Maxima methods replace
    conventional loop methods here.
    calls achi() and f1d() */

sigma_points(ee,b0,bmax,db):=
block([nb,bL,chiL,dchi_dbL,sigL,numer],numer:true,
    bL : makelist(b,b,b0,bmax,db),
    nb : length(bL),
    chiL : map(lambda([x], achi(ee,x)), bL),
    dchi_dbL : f1d(nb,db,chiL),
    sigL : abs(bL/sin(chiL)/dchi_dbL),
    [chiL,log(sigL)])$

/* achi(e,b) returns the scattering angle in radians
     using numerical, rather than analytical, methods. */

achi(e,b) :=
block([z,w,zmin,phi_inf,rexpr,iexpr,numer],numer:true,
    w : 1/(b*e),
    rexpr : z^2 - w*z -1, /* root -> zmin */
    iexpr : 1/(z*sqrt(rexpr)),        /* integrate to get phi_inf */
    zmin : find_root(rexpr,z,1e-4,1e6),
    phi_inf : quad_qagi(iexpr,z,zmin,inf)[1],
    (%pi - 2*phi_inf))$

/*
(%i7) achi(1,1);
(%o7) 0.927295
(%i8) deg(%);
(%o8) 53.130102
*/
```

```
/* f1d(nv,hh,gL) returns a list of first derivatives
    at the grid points for a function whose
     nv values at the grid points separated by hh
      are in the list gL */

f1d(nv,hh,gL) :=
block([j,fpL:[],fp0,fpl,numer],numer:true,
   for j:2 thru nv-1 do
       fpL : cons( (gL[j+1] - gL[j-1])/2/hh,fpL),
   fpL : reverse(fpL),
   /* use linear interpolation to define first and
       last elements of fpL */
   fp0 : 2*fpL[1] - fpL[2],
   fpl : 2*fpL[nv-2] - fpL[nv-3],
   fpL : cons(fp0,fpL),
   fpL : append(fpL,[fpl]),
   fpL)$
```

Here is an example of use which compares the strictly numerical approach to the analytic.

```
(%i1) load(rutherford_repulse);
(%o1) "c:/k1/rutherford_repulse.mac"
(%i2) [chival,sigval] : sigma_points(1,0.1,50,0.1)$
(%i3) time(%);
(%o3) [15.44]
(%i4) fll(chival);
(%o4) [2.7468015,0.0199993,500]
(%i5) fll(sigval);
(%o5) [-2.712689,15.648312,500]
(%i6) chi_min : lmin(chival);
(%o6) 0.0199993
(%i7) chi_max : lmax(chival);
(%o7) 2.7468015
(%i8) plot2d([[discrete, chival,sigval], log(1/16/sin(x/2)^4)],
            [x,chi_min,chi_max],[xlabel,"chi"],
            [ylabel,"ln(dsigma/do)"],[style,[lines,3]],
            [legend,"numerical","analytic"])$
```

produces the plot

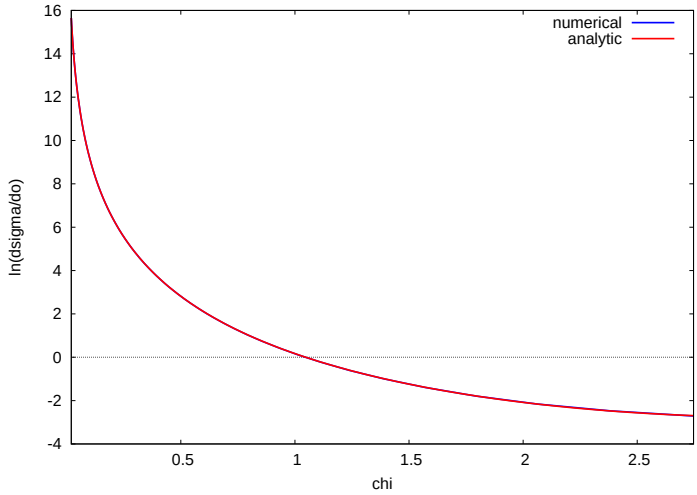

Figure 15: $\ln(d\sigma/d\Omega)$ versus χ, $\bar{E} = 1$

We see that if we sample enough points, we get agreement between the analytic result and the numerical method.

3.11 Numerical Differential Cross Section vs. Scattering Angle Using R

We use numerical methods with **R** to calculate the scattering angle as a function of the energy and impact parameter, and use numerical methods to plot the natural logarithm of the predicted (dimensionless) differential cross section versus the scattering angle.

sigma_points calls **achi(e,b)** to construct a **R** vector **chiL** of scattering angles (in radians) corresponding to a set of values of the dimensionless impact parameter $\bar{b}$. The energy **e**, the starting impact parameter value **b0**, the maximum impact parameter **bmax**, and the impact parameter increment **db** are inputs. The function **f1d(num,dx,funcL)** is called with the inputs **f1d(nb,db,chiL)** to obtain a **R** vector of first derivatives $d\,\chi/d\,\bar{b}$ at the impact parameter grid points and stored in the vector **dchi_dbL**. **sigma_points** returns a **R** list of two **R** vectors: **{chi-vec, log(dsig)-vec}**. The value of each element of **sigL** in the code is based on

$$d\,\bar{\sigma}/d\,\Omega = \frac{\bar{b}}{\sin(\chi)}\,\frac{1}{|\,d\,\chi/d\,\bar{b}\,|} \tag{3.18}$$

The following four functions are defined in **rutherford_repulse.R**.

```
##    sigma_points(e,b0,bmax,db) produces a list of two lists:
##    [chi-list, log(d_sig/d_omega)-list] using numerical methods.
##    Typical list arithmetic Maxima methods replace
##    conventional loop methods here.
##    calls achi() and f1d()


sigma_points = function(ee,b0,bmax,db) {
    bL = seq(from=b0, to=bmax, by=db)
    nb = length(bL)
    chiL = sapply(bL, function(x) achi(ee,x))
    dchi_dbL = f1d(nb, db, chiL)
    sigL = abs(bL/sin(chiL)/dchi_dbL)
    list( chiL,log(sigL))}

## scattering angle in radians

achi = function(e,b) {
     w = 1/(b*e)
     fr = function(z) z^2 - w*z -1  #  root -> zmin
     zmin = uniroot(fr, c(1e-4, 1e6),tol = 1e-10)$root
     phi_inf = integrate(function(z) 1/z/sqrt(fr(z)), zmin, Inf)$val
     pi - 2*phi_inf}

##    f1d(nv,hh,gL) returns a vector of first derivatives
##    at the nv grid points for a function whose
##    nv values at the grid points separated by hh
##    are in the vector gL. Would be more accurate if
##    we used quadratic interpolation to define the
##    end of grid derivatives.

f1d = function(nv,hh,gL) {
   fpL = vector(mode = "numeric", length = nv)
   for (j in 2:(nv-1)) fpL[j] = (gL[j+1] - gL[j-1])/2/hh
   fpL[1] = 2*fpL[2] - fpL[3]
   fpL[nv] = 2*fpL[nv-1] - fpL[nv-2]
   fpL}
```

```
##  print out first, last and length of a vector

fll = function(xL) {
    xlen = length(xL)
    cat(" ",xL[1]," ",xL[xlen]," ",xlen,"\n") }
```

Here is an example of using these functions to make a plot of the natural log of the differential scattering cross-section versus the scattering angle χ.

```
> source("rutherford_repulse.R")
> spts = sigma_points(1,0.1,50,0.1)
> chival = spts[[1]]
> sigval = spts[[2]]
> fll(chival)
  2.746802   0.01999933   500
> fll(sigval)
  -2.712689   15.64831   500
> chi_min = min(chival); chi_min
[1] 0.01999933
> chi_max = max(chival); chi_max
[1] 2.746802
> plot(chival, sigval, xlim = c(chi_min,chi_max),
+           type="l", col = "blue", lwd = 3,
+           xlab="chi", ylab="log(sigma)")
> abline(h=0,v=0)
> curve(log(1/16/sin(x/2)^4),chi_min,chi_max,n=200,
+            add=TRUE,col = "red",lwd = 3)
```

which produces the plot

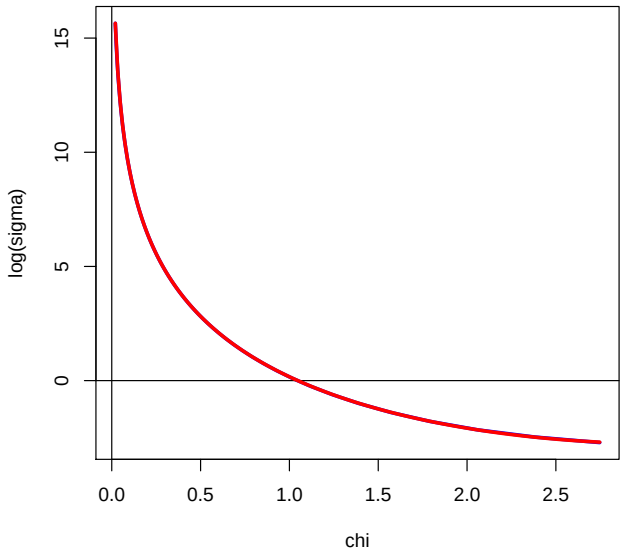

Figure 16: $\ln(d\sigma/d\Omega)$ versus χ, $\bar{E} = 1$

The analytic curve (in red) lies on top of the numerical curve (in blue), provided we take enough samples using the numerical method.

4 Attractive Rutherford Scattering

Here we treat the case in which the potential energy is

$$V(r) = -\frac{\alpha}{r}, \quad \alpha > 0, \quad V_{eff}(r) = -\frac{\alpha}{r} + \frac{E\,b^2}{r^2}, \tag{4.1}$$

$$\bar{V}(\bar{r}) = -\frac{1}{\bar{r}}, \quad \bar{V}_{eff}(\bar{r}) = -\frac{1}{\bar{r}} + \frac{\bar{E}\,\bar{b}^2}{\bar{r}^2} \tag{4.2}$$

4.1 Plot of Effective Potential - Attractive Rutherford

The function **Veff_plot** (from **rutherford_attract.mac**) has the definition

```
Veff_plot(e,b,r0,r1,xmin,xmax,ymin,ymax) :=
block([w,rmin,energy_line,numer],numer:true,
    w : 1/e/b,
    rmin : b*(-w + sqrt(4+w^2))/2,  /* attractive case */
    print(" rmin = ",rmin),
    energy_line : [discrete,[[rmin,e],[r1,e]]],
    plot2d([-1/r + e*b^2/r^2, energy_line],[r,r0,r1], [x,xmin,xmax],
           [y,ymin,ymax],[style,[lines,3,1],[lines,3,2]],
           [legend,false],[xlabel,"r"],[ylabel,"Veff"]))$
```

The parameter **rmin** is the minimum radial distance between the incident particle and the scattering center. The invocation

```
(%i1) Veff_plot(1,1,0.4,8,0.4,8,-1,2)$
 rmin =  0.618034
plot2d: some values were clipped.
```

chooses $\bar{E} = 1$, $\bar{b} = 1$, and chooses to plot the expression defined by **veff** over the range **[r,0.4,8]**. The canvas horizontal extent is determined by the arguments **xmin** and **xmax**. The canvas vertical extent is determined by the arguments **ymin** and **ymax**. This produces a plot of both the effective potential energy and also the constant total energy of the incident particle (in red).

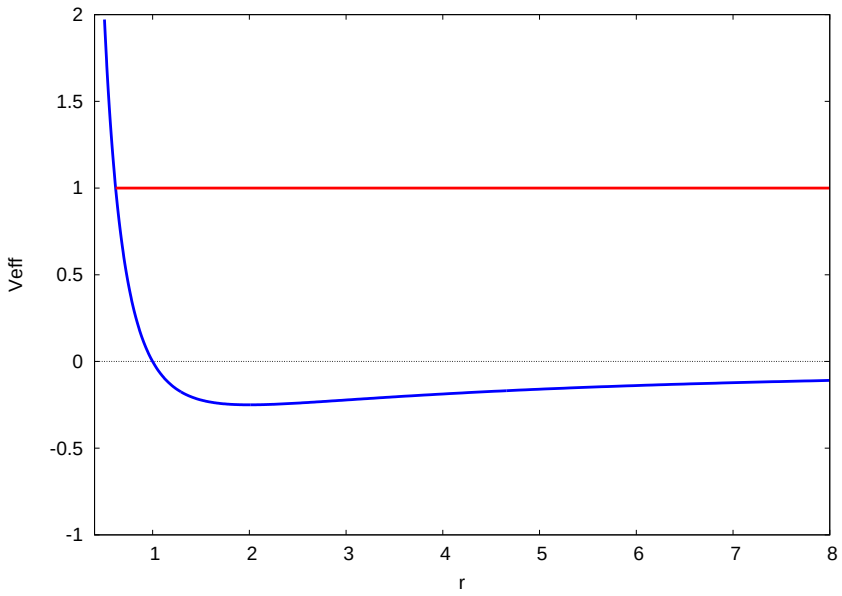

Figure 17: Effective Potential for $\bar{E} = 1$, $\bar{b} = 1$

4.2 Analytic and Numeric Scattering Angle Using Maxima - Attractive Rutherford

Continuing with the attractive Rutherford case, from Eq. (2.44) we then have

$$\phi_\infty = \bar{b} \int_{\bar{r}_{min}}^{\infty} \frac{d\bar{r}}{\bar{r}^2 \sqrt{1 + 1/(\bar{E}\,\bar{r}) - \bar{b}^2/\bar{r}^2}}. \tag{4.3}$$

We replace the integration variable $\bar{r}$ by $z = \bar{r}/\bar{b}$, as before, and define $w = 1/(\bar{E}\,\bar{b})$, as before, to get

$$\phi_\infty = \int_{z_{min}}^{\infty} \frac{d\,z}{z\sqrt{z^2 + w\,z - 1}} \tag{4.4}$$

in which z_{min} is the positive value of z for which the argument of the square root is zero. This gives the result (see Maxima session just below)

$$\phi_\infty = \frac{\pi}{2} + \sin^{-1}\left(\frac{w}{\sqrt{4 + w^2}}\right), \quad w = 1/(\bar{E}\,\bar{b}). \tag{4.5}$$

For attractive scattering, $\phi_\infty > \pi/2$ and

$$\phi_\infty + (\phi_\infty - |\chi|) = \pi, \quad \chi = \pi - 2\,\phi_\infty < 0 \tag{4.6}$$

$$\chi = -2\,\sin^{-1}\left(\frac{w}{\sqrt{4 + w^2}}\right) \tag{4.7}$$

We take χ to be a negative number here with the magnitude of χ the deviation below the forward line $y = b$ (or the line $\bar{y} = \bar{b}$).

We find $z_{min} = \frac{1}{2}\,(\sqrt{4 + w^2} - w)$ and hence $\bar{r}_{min} = \frac{\bar{b}}{2}\,(\sqrt{4 + w^2} - w)$.

We let θ_0 be the positive angle between the positive x-axis and the direction of $\mathbf{r}(\mathbf{t})$ at the moment the incident particle is at its minimum distance from the scattering center (the origin of the coordinate system). As before, $\theta_0 + \phi_\infty = \pi$, so $\theta_0 = \pi - \phi_\infty$.

Here is a short Maxima session to find the above results.

```
(%i1) rs : solve(z^2 + w*z -1,z);
(%o1) [z = -(sqrt(w^2+4)+w)/2,z = (sqrt(w^2+4)-w)/2]
(%i2) zmin : rhs(rs[2]);
(%o2) (sqrt(w^2+4)-w)/2
(%i3) assume(w>0);
(%o3) [w > 0]
(%i4) phi_inf : integrate(1/(z*sqrt(z^2 + w*z -1)),z,zmin,inf);
(%o4) asin(w/sqrt(w^2+4))+%pi/2
```

Using the function **angles(e,b)** from **rutherford_attract.mac**,

```
(%i5) angles(1,1)$
 phi_inf =  2.0344439  rad, or  116.56505  deg
 theta0 =  1.1071487  rad, or  63.434949  deg
 chi =  -0.927295  rad, or  -53.130102  deg
 rmin =  0.618034
```

Since the magnitude of the scattering angle as a function of the dimensionless parameters $\bar{E}$ and $\bar{b}$ is the same as in the case of repulsive Rutherford scattering, the plots of the magnitude of the scattering angle as a function of $\bar{b}$ for given $\bar{E}$ are the same, and the plots of the natural logarithm of the magnitude of the differential scattering cross section as a function of the magnitude of the scattering angle are the same as the repulsive case.

In **rutherford_attract.mac** is the function **angle_a(e,b)** which incorporates the analytic formula for the scattering angle:

```
/* analytic scattering angle in degrees for given
   values of Ebar and bbar */

angle_a(e,b) :=
block([numer],numer:true,
     -2*asin(1/sqrt(1 + 4*e^2*b^2))*180/%pi)$
```

Using this function, we can make a simple plot of scattering angle versus impact parameter for the case $\bar{E} = 1$.

```
(%i6) plot2d(angle_a(1,b),[b,0.01,6],[ylabel,"chi-deg"],
         [style,[lines,3]])$
```

which produces the plot

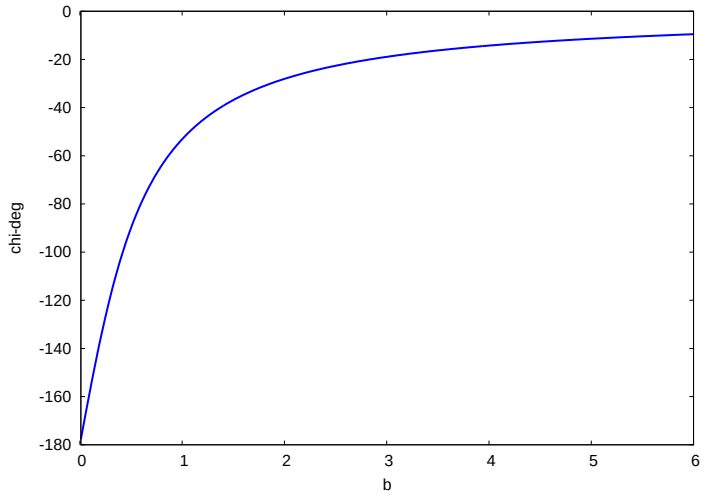

Figure 18: Scattering Angle Versus $\bar{b}$ for $\bar{E} = 1$

Also in the file **rutherford_attract.mac** are the three numerical functions:

```
deg(z) := block([numer],numer:true, z*180/%pi)$

achi(e,b) :=
block([z,w,zmin,phi_inf,rexpr,iexpr,numer],numer:true,
    w : 1/(b*e),
    rexpr : z^2 + w*z -1, /* root -> zmin for attractive case */
    iexpr : 1/(z*sqrt(rexpr)),        /* integrate to get phi_inf */
          zmin : find_root(rexpr,z,1e-4,1e6),
    phi_inf : quad_qagi(iexpr,z,zmin,inf)[1],
    (%pi - 2*phi_inf))$

angle_n(e,b) := (deg(achi(e,b)))$
```

Using this numerical approach instead, we can also make a continuous plot by using the **lambda** function:

```
(%i7) plot2d(lambda([b],angle_n(1,b)),[b,0.01,6],[ylabel,"chi-deg"],
           [style,[lines,3]])$
(%i8) time(%);
(%o8) [25.86]
```

which produces the same plot as produced using **angle_a(e,b)**, but takes a longer time.

Of course a faster method would be to define a list of values of the impact parameter, and map the conversion to scattering angle onto that list, producing a list of corresponding scattering angles in a shorter time, as in

```
(%i9) bL : makelist(b,b,0.1,5,0.1)$
(%i10) fll(bL);
(%o10) [0.1,5.0,50]
(%i11) chiL : map(lambda([b],angle_n(1,b)), bL)$
(%i12) time(%);
(%o12) [1.61]
(%i13) fll(chiL);
(%o13) [-157.38014,-11.421186,50]
(%i14) plot2d([discrete,bL,chiL],[x,0,5],[xlabel,"b"],[ylabel,"chi-deg"],
          [style,[lines,3]])$
```

which produces

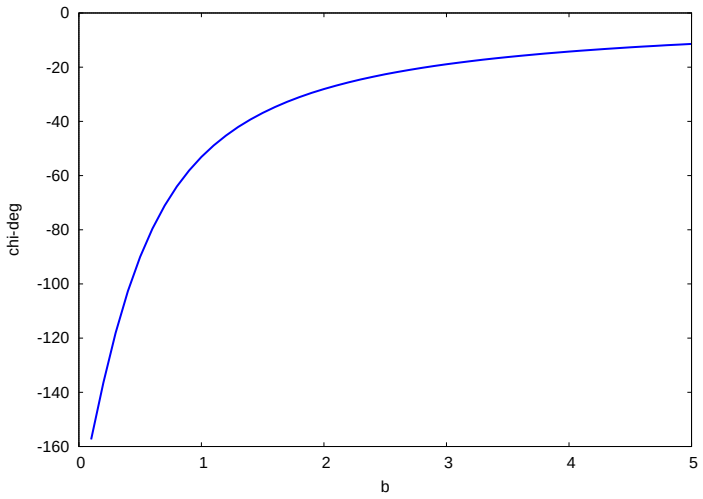

Figure 19: Scattering Angle Versus $\bar{b}$ for $\bar{E} = 1$

4.3 Numerical Scattering Angle Versus Impact Parameter Using R - Attractive Rutherford

In the file **rutherford_attract.R** are the three functions

```
##  convert radians to degrees
deg = function(z) z*180/pi

## scattering angle in radians via numerical methods
## for attractive Rutherford scattering
achi = function(e,b) {
     w = 1/(b*e)
     fr = function(z) z^2 + w*z -1  #  root -> zmin
     zmin = uniroot(fr, c(1e-4, 1e6),tol = 1e-10)$root
     phi_inf = integrate(function(z) 1/z/sqrt(fr(z)), zmin, Inf)$val
     pi - 2*phi_inf}

##  angle_n(e,b) attractive rutherford case
##    numerical scattering angle in degrees for given
##      values of Ebar and bbar

angle_n = function(e,b) deg(achi(e,b))
```

Using these functions, we can make a simple plot of scattering angle versus impact paramter for the case of attractive Rutherford scattering.

```
> bL = seq(0.01,6,0.1)
> fll(bL)
  0.01   5.91   60
> chiL = sapply(bL,function(x) angle_n(1,x))
> fll(chiL)
  -177.7085   -9.671686   60
> plot(bL,chiL,type="l",col="blue",lwd=3,ylab="chi-deg",xlab="b")
> abline(v=0)
```

which produces the plot

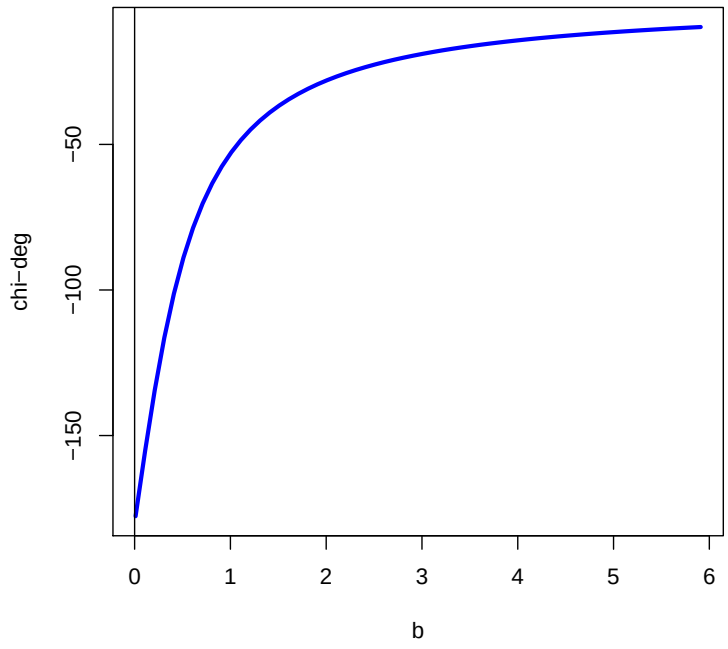

Figure 20: Scattering Angle Versus $\bar{b}$ for $\bar{E} = 1$

4.4 Scattering Trajectory Plot Using Maxima and R - Attractive Rutherford

The file **rutherford_attract.mac** contains the function

```
orbit_plot1(e,b,tm,tp,dt,rchi,xmin,xmax,ymin,ymax,nextend)
```

which provides much flexibility for the task of drawing a scattering orbit. The code uses the Maxima fourth-order Runge-Kutta integrator **rk**.

A translation into R with the syntax

```
orbit_plot1(e,b,tm,tp,dt,rchi,xmin,xmax,ymin,ymax)
```

is in the file **rutherford_attract.R** and produces the same plots illustrated here using the Maxima version. The Maxima version uses the small integer **nextend** to determine the number of points to use for the extension of the red line

which joins the origin to the closest point of the trajectory. Here are three examples of use of the Maxima version, first for the case $e = 1$ and $b = 1$ (χ, the scattering angle, is represented by **chi** in the code). The code for **orbit_plot1** causes printouts (to the screen) of much diagnostic information about the trajectory.

```
(%i1) orbit_plot1(1,1,5,5,0.01,5,-3,3.4,-2,2,5,1)$
 rmin =  0.618034
 phi_inf =  2.0344439  rad or  116.56505  deg
 theta0 =  1.1071487  rad or  63.434949  deg
 chi =  -0.927295  rad, or  -53.130102  deg
 xc =  0.276393  yc =  0.552786
 vcx =  1.4472136  vcy =  -0.723607
 xa =  0.5
 backwards from xc, yc
 xfirst =  -5.7276639
 forwards from xc, yc
 xlast =  4.2046373  ylast =  -4.0061018
 vx_last =  0.655096  vy_last =  -0.861996
 xf =  3.5  yf =  -3.0
```

produces

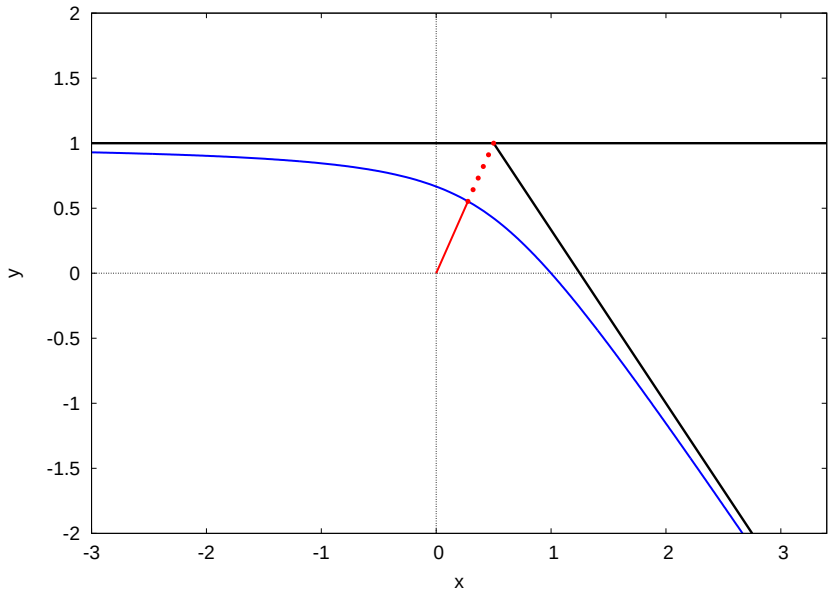

Figure 21: $\bar{E} = 1, \bar{b} = 1$

and then for $\bar{E} = 1$ and $\bar{b} = 0.6$,

```
(%i2) orbit_plot1(1,0.6,5,5,0.01,5,-3,3.4,-2,2,5,1)$
 rmin =  0.281025
 phi_inf =  2.2655346  rad or  129.80557  deg
 theta0 =  0.876058  rad or  50.194429  deg
 chi =  -1.3894766  rad, or  -79.611142  deg
 xc =  0.179908  yc =  0.215889
 vcx =  1.6401844  vcy =  -1.3668203
 xa =  0.5
 backwards from xc, yc
 xfirst =  -5.9196184
 forwards from xc, yc
 xlast =  1.6346676  ylast =  -5.7185896
 vx_last =  0.198759  vy_last =  -1.0623693
 xf =  1.4016393  yf =  -4.3180328
```

produces

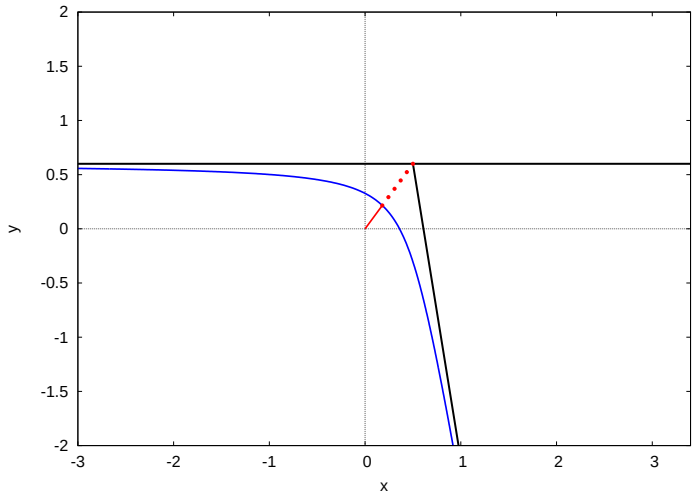

Figure 22: $\bar{E} = 1, \bar{b} = 0.6$

and for $\bar{E} = 1$ and $\bar{b} = 0.3$,

```
(%i3) orbit_plot1(1,0.3,5,5,0.01,5,-3,3.4,-2,2,5,1)$
rmin =  0.0830952
phi_inf =  2.6011732  rad or  149.03624  deg
theta0 =  0.54042  rad or  30.963757  deg
chi =  -2.0607537  rad, or  -118.07249  deg
xc =  0.0712535  yc =  0.0427521
vcx =  1.8574929  vcy =  -3.0958215
xa =  0.5
backwards from xc, yc
xfirst =  -6.0729577
forwards from xc, yc
xlast =  -2.6061534  ylast =  -5.4927368
vx_last =  -0.506288  vy_last =  -0.951945
xf =  -1.8529412  yf =  -4.1117647
```

we get

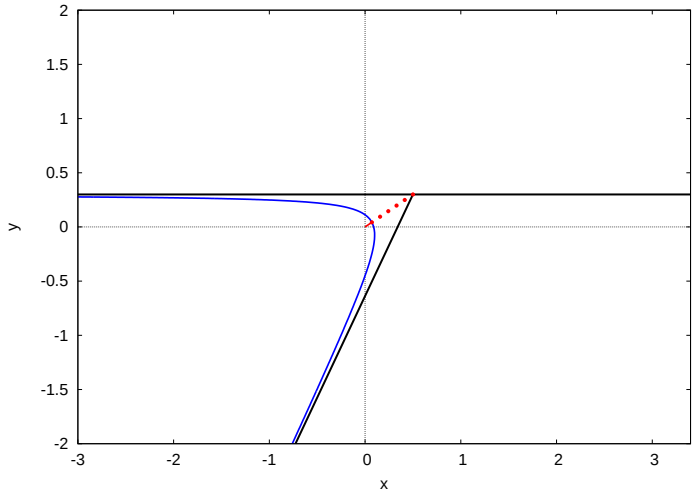

Figure 23: $\bar{E} = 1, \bar{b} = 0.3$

An example of the use of the **R** version (which uses a more sophisticated adaptive algorithm than does Maxima) is:

```
> source("rutherford_attract.R")
> library(deSolve)
> orbit_plot1(1,1,5,5,0.01,5,-3,3.4,-2,2)
 rmin =  0.618034
 phi_inf =  2.034444  rad or  116.5651  deg
 theta0 =  1.107149  rad or  63.43495  deg
 chi =  -0.9272952  rad, or  -53.1301  deg
 xc =  0.2763932  yc =  0.5527864
 vcx =  1.447214  vcy =  -0.7236068
 xa =  0.5
  backwards from xc, yc
 xfirst =  -5.727665
  forwards from xc, yc
 xlast =  4.204644  ylast =  -4.006098
 vx_last =  0.6550977  vy_last =  -0.861996
 xf =  3.5  yf =  -3
```

which produces the plot

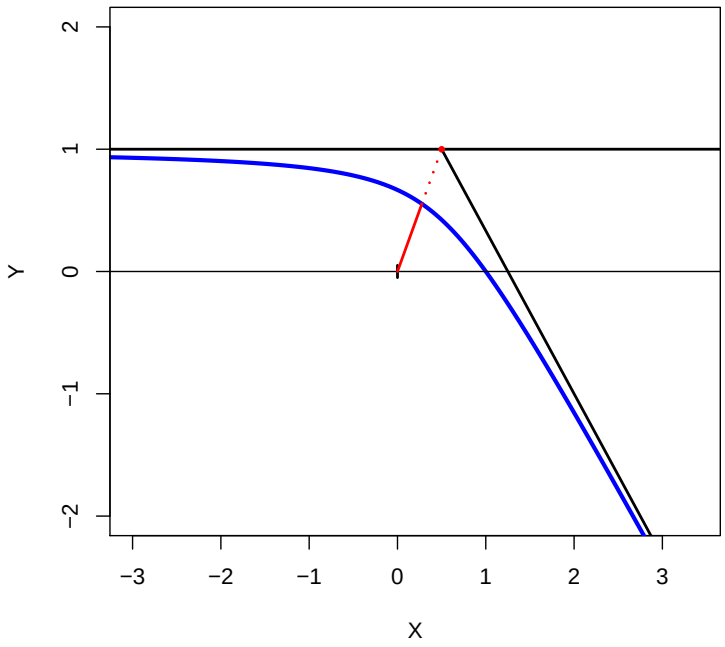

Figure 24: $\bar{E} = 1, \bar{b} = 1$

Using the Maxima version (which uses a fixed time step), for a given energy, as you decrease the impact parameter, you need to finally decrease the integration step size to achieve correct asymptotic behavior: approaching $y = b$ as $t \to -\infty$, and the $t \to +\infty$ orbital elements becoming tangent to the scattering angle line. We find that we must set $dt = 0.001$ when **`e = 1, b = 0.2`**, and we must set $dt = 0.0001$ when **`e = 1, b = 0.1`**, to get a resulting plot with the correct asymptotic behavior.

4.5 Differential Scattering Cross Section - Attractive Rutherford Case

In Section (4.2) we found the negative scattering angle for the attractive Rutherford potential case:

$$\chi = -2 \sin^{-1}\left(\frac{w}{\sqrt{4+w^2}}\right) \tag{4.8}$$

This implies

$$\sin(|\chi|/2) = (1 + 4\,\bar{E}^2\,\bar{b}^2)^{-1/2} \tag{4.9}$$

Using the same argument as before, this implies that

$$\cot(|\chi|/2) = 2\,\bar{E}\,\bar{b}, \quad \bar{b} = -\frac{\cot(\chi/2)}{2\,\bar{E}} \tag{4.10}$$

and hence

$$\frac{d\,\bar{b}}{d\,\chi} = \frac{1}{4\,\bar{E}\,\sin^2(\chi/2)} \tag{4.11}$$

We then get the same formal expression as in the repulsive Rutherford case, Eq.(3.16

$$\frac{d\,\bar{\sigma}}{d\,\Omega} = \frac{1}{16\,\bar{E}^2\,\sin^4(\chi/2)}. \tag{4.12}$$

We will use the same approximate numerical method as used in the repulsive Rutherford case, and compare the results with the analytic answer. Using Maxima for the case $\bar{E} = 1$,

```
(%i1) load(rutherford_attract);
(%o1) "c:/k1/rutherford_attract.mac"
(%i2) [chival,sigval] : sigma_points(1,0.1,50,0.1)$
(%i3) time(%);
(%o3) [16.25]
(%i4) fll(chival);
(%o4) [-2.7468015,-0.0199993,500]
(%i5) fll(sigval);
(%o5) [-2.712689,15.648312,500]
(%i6) chi_min : lmin(chival);
(%o6) -2.7468015
(%i7) chi_max : lmax(chival);
(%o7) -0.0199993
(%i8) plot2d([[discrete, chival,sigval], log(1/16/sin(x/2)^4)],
            [x,chi_min,chi_max],[xlabel,"chi"],
            [ylabel,"ln(dsigma/do)"],[style,[lines,3]],
            [legend,"numerical","analytic"])$
```

which produces the plot

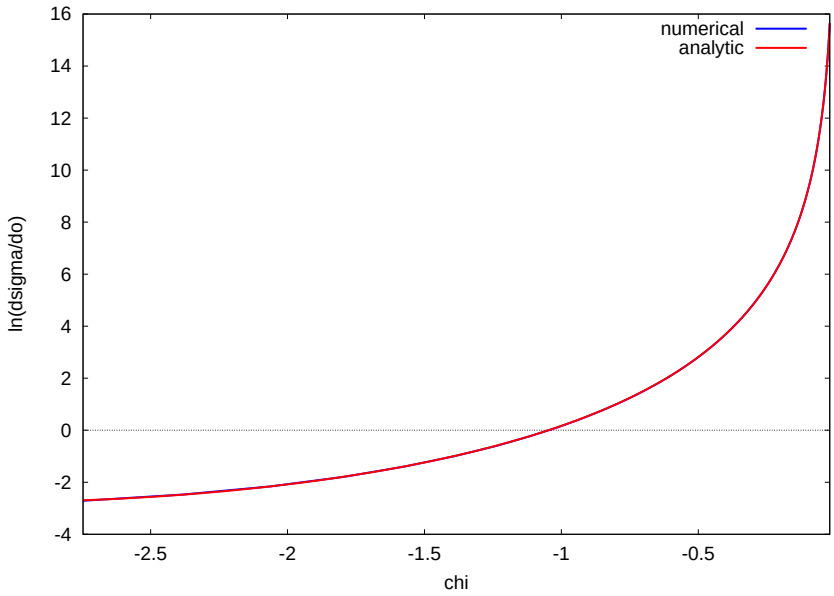

Figure 25: $\bar{E} = 1, \bar{b} = 1$

which implies agreement (again provided we use enough samples in the approximate numerical method).

Using **R** instead, we get the same agreement:

```
> source("rutherford_attract.R")
> spts = sigma_points(1,0.1,50,0.1)
> chival = spts[[1]]
> fll(chival)
  -2.746802   -0.01999933   500
> sigval = spts[[2]]
> fll(sigval)
  -2.712689   15.64831   500
> chi_min = min(chival); chi_min
[1] -2.746802
> chi_max = max(chival); chi_max
[1] -0.01999933
> plot(chival, sigval, xlim = c(chi_min,chi_max),
+           type="l", col = "blue", lwd = 3,
+           xlab="chi", ylab="log(sigma)")
> abline(h=0,v=0)
> curve(log(1/16/sin(x/2)^4),chi_min,chi_max,n=200,
+            add=TRUE,col = "red",lwd = 3)
> legend("topleft",col=c("blue","red"),legend=c("numerical","analytic"),
+       lwd = 2, cex = 1.5)
```

which produces

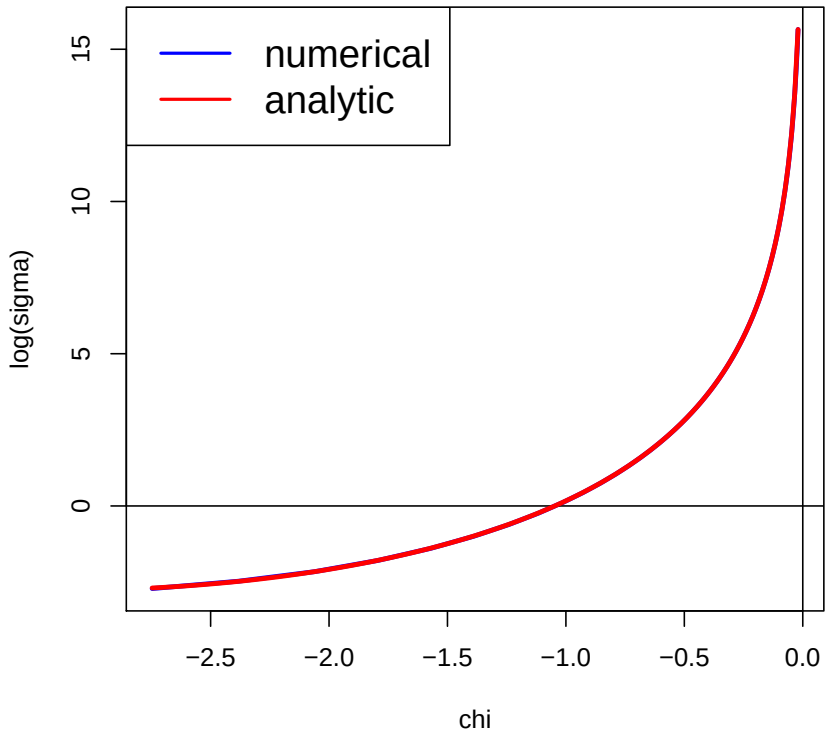

Figure 26: $\bar{E} = 1, \bar{b} = 1$

5 Scattering by the Lennard-Jones Potential

A central potential (energy) expression which implies an attractive force (on the scattered particle) when r is large and a repulsive force when r is small is the Lennard-Jones "6-12" potential

$$V(r) = 4\,V_0\,\left((a/r)^{12} - (a/r)^6\right) \tag{5.1}$$

which contains the adjustable parameter a with the dimension of length and the adjustable parameter V_0 with the dimension of energy. The factor of 4 assures that V_0 is the depth of the potential energy minimum (below the zero of energy). The switch from attractive to repulsive force occurs when $r = 2^{1/6}\,a$.

Defining dimensionless variables $\bar{r} = r/a$ and $\bar{V}(\bar{r}) = V/V_0$, we get

$$\bar{V}(\bar{r}) = 4\,\left(\frac{1}{\bar{r}^{12}} - \frac{1}{\bar{r}^6}\right). \tag{5.2}$$

The dimensionless energy is $\bar{E} = E/V_0$ and the dimensionless impact parameter is $\bar{b} = b/a$.

The dimensionless x component of acceleration is then, using Eq.(2.47)

$$\frac{d\,\bar{v}_x}{d\,\bar{t}} = -\frac{12\,\bar{x}}{\bar{E}}\,\left(\frac{1}{\bar{r}^8} - \frac{2}{\bar{r}^{14}}\right) \tag{5.3}$$

The dimensionless y component of acceleration is

$$\frac{d\,\bar{v}_y}{d\,\bar{t}} = -\frac{12\,\bar{y}}{\bar{E}}\,\left(\frac{1}{\bar{r}^8} - \frac{2}{\bar{r}^{14}}\right) \tag{5.4}$$

For large $\bar{r}$

$$\frac{d\,\bar{v}_x}{d\,\bar{t}} \approx -\frac{12\,\bar{x}}{\bar{E}\,\bar{r}^8} \tag{5.5}$$

and if $\bar{x} < 0$ (for large $\bar{r}$) then $\bar{v}_x$ is increasing with time (attractive region) and if $\bar{x} > 0$ (for large $\bar{r}$) then $\bar{v}_x$ is decreasing with time (repulsive region). With no restriction on the size of $\bar{r}$, $d\,\bar{v}_x/d\,\bar{t}$ changes sign when $\bar{r} = 2^{1/6} \approx 1.122462$.

```
(%i1) 2^(1/6),numer;
(%o1) 1.122462048309373
```

which determines the location of the minimum of $\bar{V}(\bar{r})$, where the dimensionless potential (energy) takes the value -1.

```
(%i2) 4*(1/x^(12) - 1/x^6), x = 2^(1/6);
(%o2) -1
```

The dimensionless **effective** potential energy is

$$\bar{V}_{eff}(\bar{r}) = \bar{V}(\bar{r}) + \bar{E}\,\bar{b}^2/\bar{r}^2 \tag{5.6}$$

The scattering angle χ is then

$$\chi = |\,\pi - 2\,\phi_\infty| \tag{5.7}$$

where

$$\phi_\infty = \bar{b}\,\int_{\bar{r}_{min}}^{\infty} \frac{d\bar{r}}{\bar{r}^2\,\sqrt{1 - \frac{4}{\bar{E}}\,\left(\frac{1}{\bar{r}^{12}} - \frac{1}{\bar{r}^6}\right) - \bar{b}^2/\bar{r}^2}}. \tag{5.8}$$

The lower limit $\bar{r}_{min} = (r_{min}/a)$ is the largest real (positive) value of $\bar{r}$ for which the argument of the square root vanishes, and hence the largest real root of the equation $f(r) = 0$, where in the code we replace $\bar{r}$ by r, $\bar{b}$ by b, and $\bar{E}$ by e, and

$$f(r) = e\,r^{12} - e\,b^2\,r^{10} + 4\,r^6 - 4 = 0. \tag{5.9}$$

If we make a plot of $f(r)$ for $e = 1$, $b = 1$, using the code (this and following code is in **lennard_jones.mac**)

```
(%i3) fplot(e,b,xmin,xmax,ymin,ymax) :=
block([f,numer], numer : true,
    f : e*x^12 - e*b^2*x^10 + 4*x^6 - 4,
    plot2d(f,[x, xmin, xmax],[y,ymin,ymax],
            [style, [lines, 3]], [ylabel, "f"],
            [xlabel, "r"],[nticks,200]))$
(%i4) fplot(1,1,0.01,2,-10,10)$
plot2d: some values were clipped.
```

we get the following plot which shows one root near $r = 1$

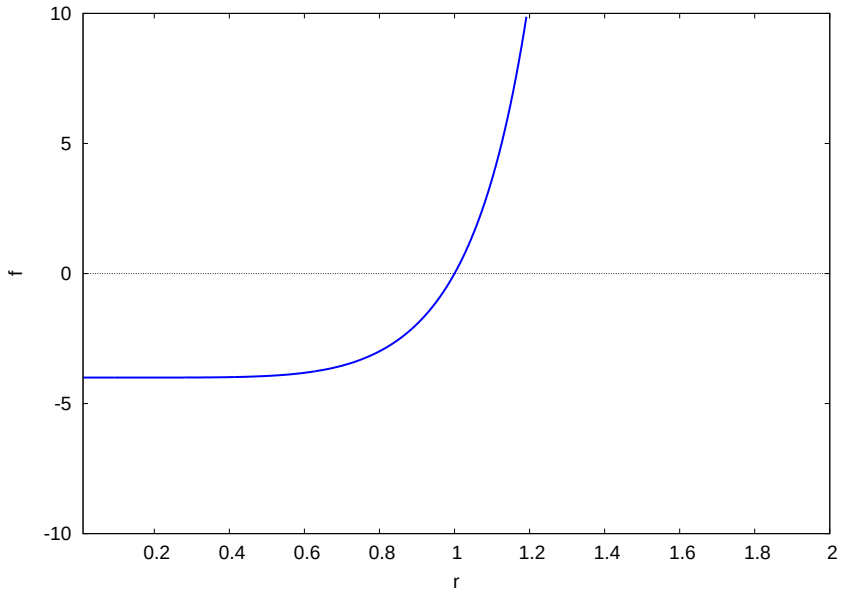

Figure 27: Arg. of Radical for $\bar{E} = 1, \bar{b} = 1$

which is confirmed by **find_root**:

```
(%i5) find_root(x^12 - x^10 + 4*x^6 -4,x,0.1,2);
(%o5) 1.0
```

and **realroots** shows that it is an exact solution:

```
 (%i6) realroots(x^12 - x^10 + 4*x^6 -4);
(%o6) [x = -1,x = 1]
```

as is obvious from inspection of the expression.

Using **R** instead of Maxima, we get the same plot with (this function is in **lennard_jones.R**):

```
> fplot = function(e,b,xmin,xmax,ymin,ymax) {
+     curve(e*x^12 - e*b^2*x^10 + 4*x^6 - 4, xmin, xmax,
+         n=200, col="blue", lwd=3, ylim = c(ymin,ymax),
+         xlab = "r", ylab = "f")
+     abline(h = 0)}
> fplot(1,1,0.01,2,-10,10)
```

Again, using **R** to find **rmin** locations

```
> e = 1;  b = 1
> uniroot (function(x) e*x^12 - e*b^2*x^10 + 4*x^6 - 4, c(0.1,2),
+           tol = 1e-10 )$root
[1] 1
> e = 1; b = 0.3
> uniroot (function(x) e*x^12 - e*b^2*x^10 + 4*x^6 - 4, c(0.1,2),
+           tol = 1e-10 )$root
[1] 0.9714185
```

5.1 Effective Lennard-Jones Potential Plots

A Maxima funciton, for the Lennard-Jones case, for a plot of the **effective** potential energy together with a line for the energy of the incident particle (in red) ending at **rmin** is:

```
Veff_plot(e,b,r0,r1,xmin,xmax,ymin,ymax) :=
block([r,veff,root_expr,rmin,energy_line,numer],numer:true,
    veff : 4*(r^(-12) - r^(-6)) + e*b^2/r^2,
    root_expr : 1 - 4*(1/r^12 - 1/r^6)/e - b^2/r^2,
    rmin : map('rhs, realroots(root_expr,1e-15)),
    rmin : apply('max, rmin),
    energy_line : [discrete,[[rmin,e],[r1,e]]],
    plot2d([veff, energy_line],[r,r0,r1], [x,xmin,xmax],
           [y,ymin,ymax],[style,[lines,3,1],[lines,3,2]],
           [legend,false],[xlabel,"r"],[ylabel,"Veff"]))$
```

Here are a few examples of use. First for **e = 1** and **b = 1**,

```
(%i1) Veff_plot(1,1,0.9,3,0.8,3,-0.5,2)$
```

which produces the plot

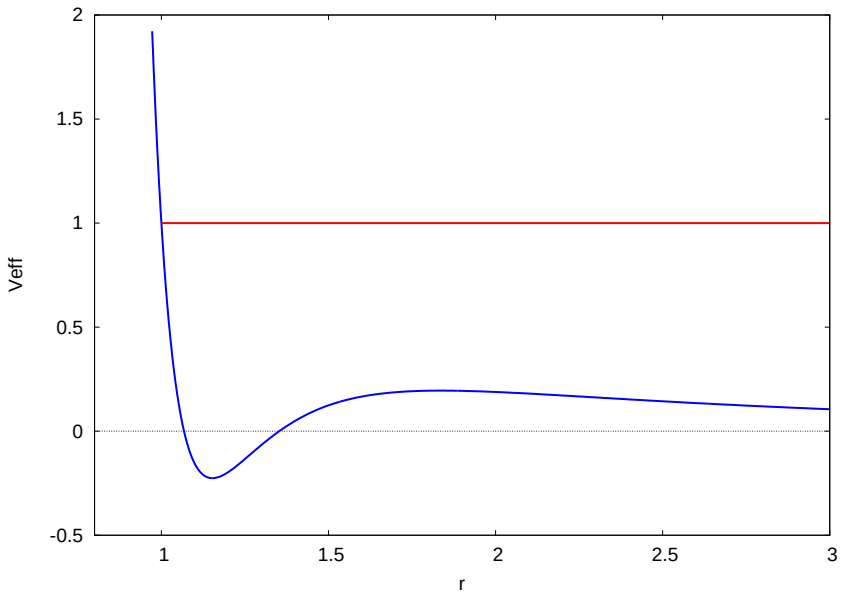

Figure 28: Effective Potential Energy for $\bar{E} = 1, \bar{b} = 1$

A second example at a value of the impact parameter which results in a “wrap-around” orbit is

```
(%i2) Veff_plot(1,1.66597,1,5,1,5,-0.5,2)$
```

which produces the plot

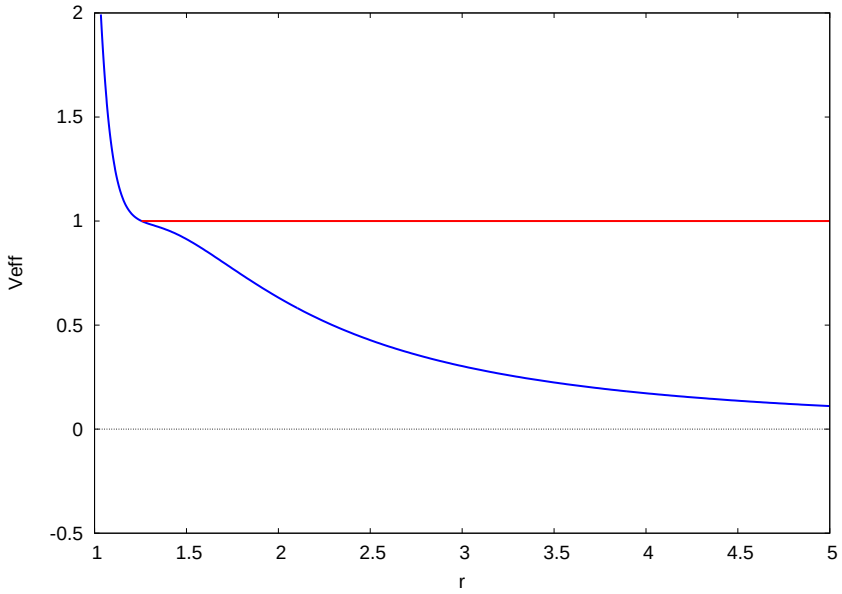

Figure 29: Effective Potential Energy for $\bar{E} = 1, \bar{b} = 1.66597$

5.2 Lennard-Jones Trajectory Plots

The Maxima function

```
orbit_plot1(e,b,tm,tp,dt,rchi,xmin,xmax,ymin,ymax,nextend,psize)
```

in the file **lennard_jones.mac** provides much flexibility in producing a plot of the trajectory for the Lennard-Jones potential.

Here is an example which prints to the screen diagnostic information about the details of the trajectory.

```
(%i1) orbit_plot1(1,1,5,5,0.01,5,-2,1.2,0,2,4,0.4)$
 rmin =  1
 phi_inf =  1.0723305  rad or  61.440014  deg
 theta0 =  2.0692621  rad or  118.55999  deg
 chi =  0.996932  rad, or  57.119973  deg
 xc =  -0.478079  yc =  0.878317
 vcx =  0.878317  vcy =  0.478079
 xa =  -0.544312
 backwards from xc, yc
 xfirst =  -5.6624496
 forwards from xc, yc
 xlast =  2.2342414  ylast =  5.2982522
 vx_last =  0.542905  vy_last =  0.839859
 xf =  2.1700966  yf =  5.1990458
```

and produces the plot

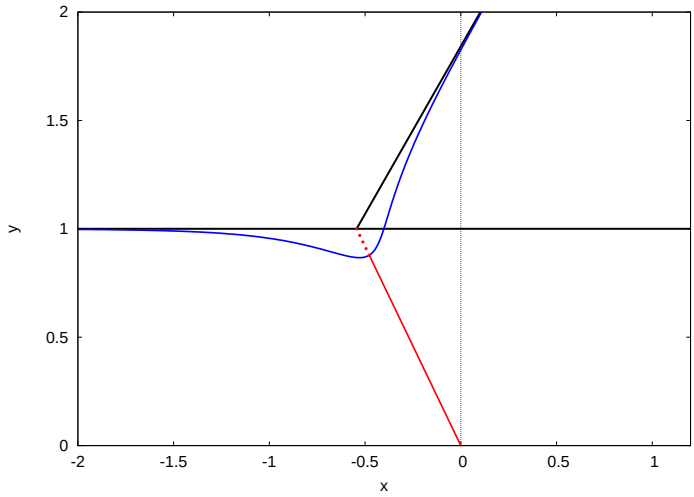

Figure 30: Lennard-Jones Trajectory for $\bar{E} = 1, \bar{b} = 1$

A case in which the trajectory wraps around the scattering center and has a negative scattering angle of about -185 degrees is produced by the parameters **`e = 1`** and **`b = 1.66597`**.

```
(%i2) orbit_plot1(1,1.66597,5,13,0.01,5,-4.9,1.5,-2,2,4,0.4)$
 rmin =  1.256028
 phi_inf =  3.1889565  rad or  182.71375  deg
 theta0 =  -0.0473639  rad or  -2.7137495  deg
 chi =  -3.2363204  rad, or  -185.4275  deg
 xc =  1.2546194  yc =  -0.0594681
 vcx =  -0.062799  vcy =  -1.3248922
 xa =  -35.147555
 backwards from xc, yc
 xfirst =  -3.3460942
 forwards from xc, yc
 xlast =  -11.453391  ylast =  -0.585262
 vx_last =  -0.995518  vy_last =  0.0945861
 xf =  -40.125139  yf =  2.1389006
```

which produces

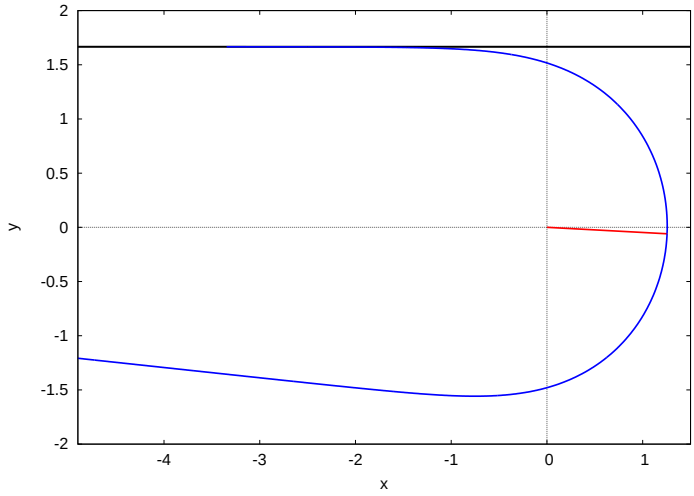

Figure 31: $\bar{E} = 1,\ \bar{b} = 1.66597,\ \chi = -185.4$ degrees

5.3 Lennard-Jones Scattering Angle Plots

The Maxima function **angle_n(e,b)** for the Lennard-Jones case uses numerical methods to return the scattering angle (in degrees), given the dimensionless energy **e** and impact parameter **b**. The scattering angle can be either positive or negative.

```
angle_n(e,b) :=
block([root_expr,rmin,phi_inf,numer],numer:true,
    root_expr : 1 - 4*(1/r^12 - 1/r^6)/e - b^2/r^2,
    rmin : map('rhs, realroots(root_expr,1e-15)),
    rmin : apply('max, rmin),
    phi_inf : b*quad_qagi(1/r^2/sqrt(root_expr),r,rmin,inf)[1],
    (%pi - 2*phi_inf)*180/%pi)$
```

with the behavior:

```
(%i1) angle_n(1,1);
(%o1) 57.119973
```

For given energy **e = 1**, we can use the Maxima **map** function with **angle_n** (provided we also use the **lambda** function) the produce a list of scattering angles from a list of values of the dimensionless impact paramter **b**.

```
(%i2) bval : [1,0.8];
(%o2) [1,0.8]
(%i3) map('lambda([x],angle_n(1,x)),bval);
(%o3) [57.119973,85.630912]
```

We can use the Maxima function **makelist** to make a list of values of **b** in an organised manner, and then a list of the corresponding scattering angles (in degrees).

```
(%i4) bval : makelist(b,b,0.1,3,0.5);
(%o4) [0.1,0.6,1.1,1.6,2.1,2.6]
(%i5) chival : map('lambda([x],angle_n(1,x)),bval);
(%o5) [168.87958,111.16737,41.099183,-109.10721,-9.3447236,-2.2820441]
```

and then make a simple plot which shows both the discrete points and also joins the points with straight lines:

```
(%i6) plot2d([[discrete,bval,chival],[discrete,bval,chival],
         [discrete,[[0,0],[3,0]]] ],
     [x,0,3],[y,-180,180],[xlabel,"b"],[ylabel,"chi"],
     [style,[lines,3,1],[points,2,2,1],[lines,3,5] ],
     [legend, false])$
```

which produces the plot

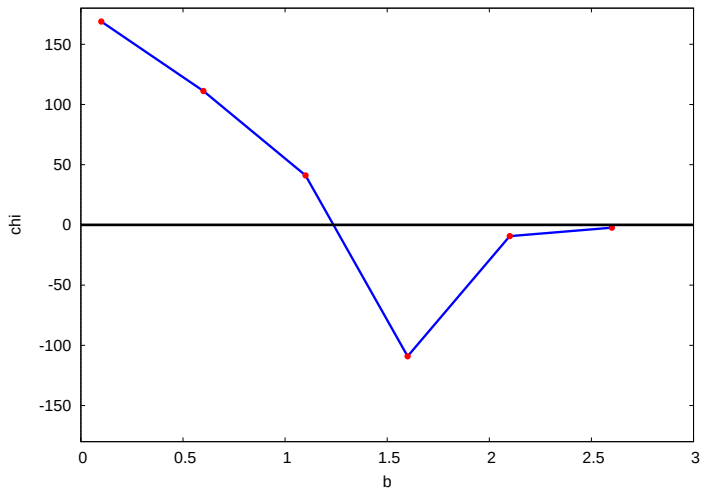

Figure 32: Scattering Angles for $\bar{E} = 1$

The region of impact parameters from **b = 0.4** to **b = 2** can then be looked at more carefully:

```
(%i7) bval : makelist(b,b,0.4,2,0.1);
(%o7) [0.4,0.5,0.6,0.7,0.8,0.9,1.0,1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9,2.0]
(%i8) chival : map('lambda([x],angle_n(1,x)),bval);
(%o8) [134.94076,123.22066,111.16737,98.679851,85.630912,71.853809,57.119973,
       41.099183,23.282176,2.8123902,-21.942679,-54.792783,-109.10721,
       -93.021101,-35.226049,-20.509053,-13.435124]
(%i9) plot2d([[discrete,bval,chival],[discrete,bval,chival],
         [discrete,[[0.3,0],[2.2,0]]] ],
     [x,0.3, 2.2],[y,-180,180],[xlabel,"b"],[ylabel,"chi"],
     [style,[lines,3,1],[points,2,2,1],[lines,3,5] ],
     [legend, false])$
```

which produces the plot

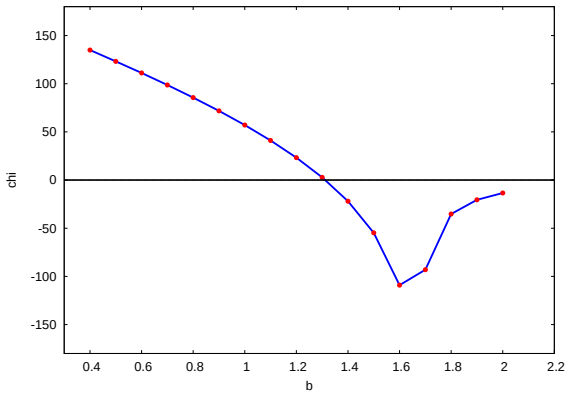

Figure 33: Scattering Angles for $\bar{E} = 1$

It is also possible to use the function **angle_n(e,b)** with the **lambda** function to get a continuous plot of scattering angle versus impact parameter, as in

```
(%i10) plot2d(lambda([x],angle_n(1,x)),[x,0.3,2.5],[y,-200,180],
        [xlabel,"b"],[ylabel,"chi"],
        [style,[lines,3,1]])$
(%i11) time(%);
(%o11) [33.62]
```

which produces the plot

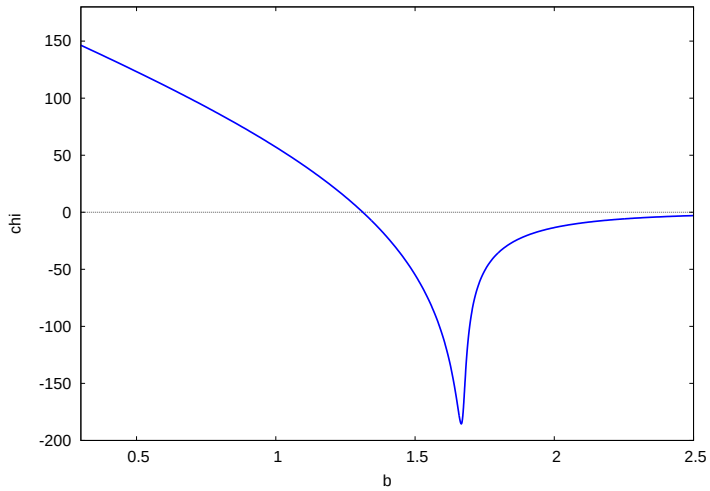

Figure 34: Scattering Angle Versus $\bar{b}$ for $\bar{E} = 1$

5.4 Lennard-Jones Differential Scattering Cross Section

The scattering angle in radians is returned by the Maxima function **achi(e,b)**, edited to describe the Lennard-Jones case:

```
achi(e,b) :=
block([root_expr,rmin,phi_inf,numer],numer:true,
    root_expr : 1 - 4*(1/r^12 - 1/r^6)/e - b^2/r^2,
    rmin : map('rhs, realroots(root_expr,1e-15)),
    rmin : lmax(rmin),
    phi_inf : b*quad_qagi(1/r^2/sqrt(root_expr),r,rmin,inf)[1],
        (%pi - 2*phi_inf))$
```

For example,

```
(%i1) achi(1,1);
(%o1) 0.996932
(%i2) map(lambda([x],achi(1,x)),[0.8,1,1.2]);
(%o2) [1.4945414,0.996932,0.406351]
```

As another example, we can calculate the finite value of $\bar{b}$ for which the scattering angle is zero (see Figure 34):

```
(%i3) find_root(lambda([x],achi(1,x)),x,1.3,1.4);
(%o3) 1.3124992
(%i4) achi(1,%);
(%o4) 1.27897692E-13
```

Thus, for $\bar{E} = 1$, the scattering angle is approximately zero for $\bar{b} = 1.3124992$.

In our plots of the scattering angle as a function of impact parameter, we have seen that there is a sharp negative minimum in the scattering angle for an impact parameter in the neighborhood of $\bar{b} = 1.6$.

A useful function here is **find_b(b_list, chi_list)**, which, given two equal length lists, a list of $\bar{b}$ values and a list of the corresponding χ values, finds the value of $\bar{b}$ corresponding to the minimum value of χ.

```
which_min(aL) :=
block([amin,j,jval:0,numer ],numer:true,
    amin:lmin(aL),
    for j thru length(aL) do
       if is(equal(aL[j], amin)) then (
                jval : j,
                      return()),
               jval)$

find_b(xL,yL) := (xL[which_min(yL)])$
```

As a simple test example with a small number of elements:

```
(%i5) bL : makelist(b,b,0.1,0.6,0.1);
(%o5) [0.1,0.2,0.3,0.4,0.5,0.6]
(%i6) chiL : map('lambda([x],achi(1,x)), bL);
(%o6) [2.9475046,2.752433,2.5553542,2.3551605,2.1506062,1.9402366]
(%i7) lmin(chiL);
(%o7) 1.9402366
(%i8) find_b(bL, chiL);
(%o8) 0.6
```

To use **R**, we can define:

```
find_b = function(bvec,chi_vec) bvec[which.min(chi_vec)]
```

with the behavior

```
> bL = seq(0.1,0.6,0.1); bL
[1] 0.1 0.2 0.3 0.4 0.5 0.6
> chiL = sin(bL); chiL
[1] 0.09983342 0.19866933 0.29552021 0.38941834 0.47942554 0.56464247
> min(chiL)
[1] 0.09983342
> find_b(bL,chiL)
[1] 0.1
```

We use two different methods to make cross section plots.

We first use the more efficient **`sigma_points(e,b0,bmax,db)`** and **`f1d(nb,db,chiL)`** method to generate a list of scattering angles and the corresponding natural logarithms of the absolute value of the differential scattering cross section implied by the range of impact parameters **`(b0, bmax)`**. This method was used for the Rutherford scattering cases.

We use first use **`find_b( b_list, chi_list )`** to get the approximate location of the impact parameter which corresponds to the steep minimum of the scattering angle.

```
(%i9) bval : makelist(b,b,0.1,3,0.1)$
(%i10) fll(bval);
(%o10) [0.1,3.0,30]
(%i11) chival : map('lambda([x], achi(1,x)), bval)$
(%i12) fll(chival);
(%o12) [2.9475046,-0.016455,30]
(%i13) lmin(chival);
(%o13) -1.9042801
(%i14) find_b(bval,chival);
(%o14) 1.6
```

We then work toward a plot corresponding to the small values of impact parameters, from $\bar{b} = 0.1$ up to $\bar{b} = 1.6$. The values of χ indicated on the plots are in radians. In this range of impact parameters, the scattering angle starts off at a large positive value 2.9475046 (approximately $169\,\mathrm{deg}$) when $\bar{b} = 0.1$, steadily decreases in value, becomes negative and reaches -1.9042801 (approximately $-109\,\mathrm{deg}$) when $\bar{b} = 1.6$.

```
(%i15) achi(1,0.1);
(%o15) 2.9475046
(%i16) deg(%);
(%o16) 168.87958
(%i17) achi(1,1.6);
(%o17) -1.9042801
(%i18) deg(%);
(%o18) -109.10721
(%i19) [chival,sigval] : sigma_points(1,0.1,1.6,0.02)$
(%i20) time(%);
(%o20) [1.8]
(%i21) chi_min : lmin(chival);
(%o21) -1.9042801
(%i22) chi_max : lmax(chival);
(%o22) 2.9475046
(%i23) plot2d([discrete,chival,sigval],[x,chi_min,chi_max],
         [xlabel,"chi"],[ylabel,"ln(dsigma/do)"],
         [style,[lines,3]])$
```

which produces the plot

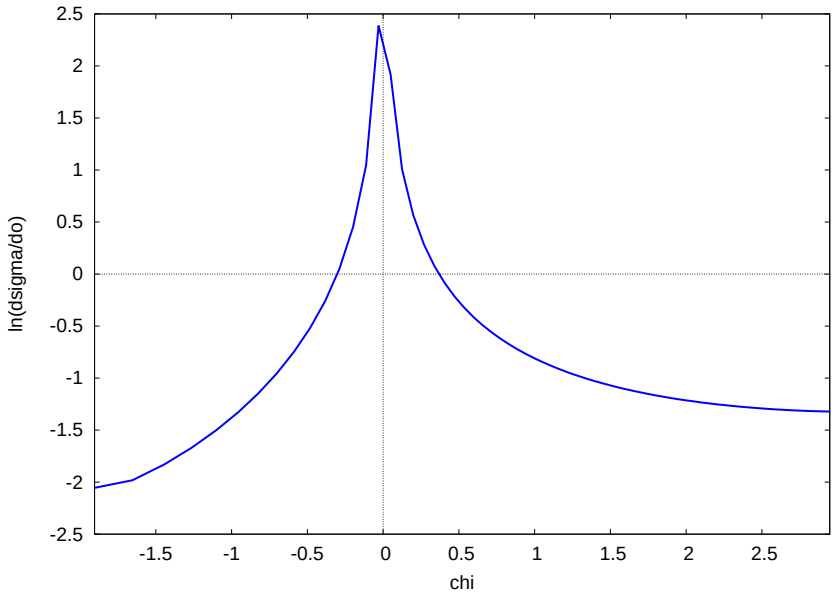

Figure 35: Log of Cross Section vs. χ for $\bar{E} = 1$ and $\bar{b} = (0.1, 1.6)$

In the above figure, the peak in the natural logarithm of the absolute value of the cross section occurs for $\chi = 0$, corresponding to $\bar{b} \approx 1.3124992$, due to the presence of $\sin(\chi)$ in the denominator of the formula for the differential cross section.

We then make a separate plot for $\bar{b}$ in the range $(1.7, 3)$. In this range of impact parameters, the scattering angle starts at a large negative value $\chi = -1.6235245 = -93\,\mathrm{deg}$ when $\bar{b} = 1.7$, and then becomes less negative steadily, ending at $\chi = -0.016455 = -0.94\,\mathrm{deg}$ when $\bar{b} = 3$.

```
(%i24) achi(1,1.7);
(%o24) -1.6235245
(%i25) deg(%);
(%o25) -93.021101
(%i26) achi(1,3);
(%o26) -0.016455
(%i27) deg(%);
(%o27) -0.942803
(%i28) [chival,sigval] : sigma_points(1,1.7,3,0.02)$
(%i29) time(%);
(%o29) [1.54]
(%i30) chi_min : lmin(chival);
(%o30) -1.6235245
(%i31) chi_max : lmax(chival);
(%o31) -0.016455
(%i32) plot2d([discrete,chival,sigval],[x,chi_min,chi_max],
         [xlabel,"chi"],[ylabel,"ln(dsigma/do)"],
         [style,[lines,3]])$
```

which produces the plot

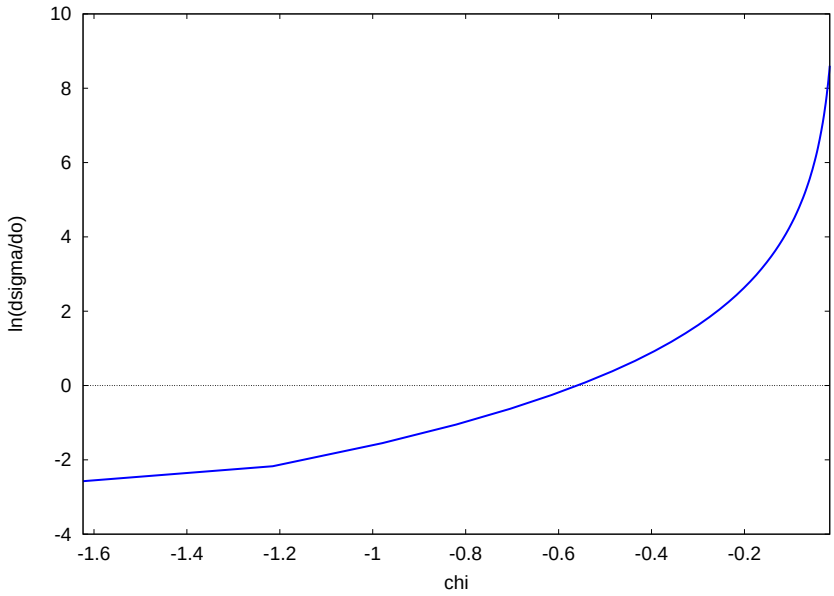

Figure 36: Log of Cross Section vs. χ for $\bar{E} = 1$ and $\bar{b} = (1.7, 3)$

Since the behavior of the scattering angle and differential cross section (as a function of the energy and impact parameter) is more complicated than the pure Rutherford scattering cases, we create a second somewhat slower and less efficient method of calculation here, which is able to produce smoother plots.

Instead of using **sigma_points** and **f1d** (used in the Rutherford scattering cases) as above, we define and use a function **chi_sigma1(e,b,db)**, which evaluates the approximate numerical first derivative of χ with respect to a single value of $\bar{b}$ and returns a list of the value of χ implied by the input value of $\bar{b}$, together with the natural logarithm of the absolute value of the differential cross section for that scattering angle.

```
chi_sigma1(e,b,db):=
block([chival,dchi_db,sig,numer],numer:true,
   chival : achi(e,b),
   dchi_db : (achi(e,b+db) - achi(e,b-db))/2/db,
   sig : abs(b/sin(chival)/dchi_db),
   [chival,log(sig)])$
```

For example,

```
(%i1) chi_sigma1(1,0.1,0.01);
(%o1) [2.9475046,-1.3216832]
(%i2) bL : [0.1,0.2]$
(%i3) map('lambda([x],chi_sigma1(1,x,0.01)), bL);
(%o3) [[2.9475046,-1.3216832],[2.752433,-1.3127418]]
```

Finally, we have constructed **sigma_plot(e,b0,bmax,db)**, which, given the energy **e**, the interval of **b** values to use (**b0** is lower limit, **bmax** is upper limit), and the separation **db** of **b** values used, makes a plot of the natural logarithm of the absolute value of the dimensionless differential scattering cross section as a function of the corresponding scattering angles.

```
sigma_plot(e,b0,bmax,db) :=
block([bL,pts,chiL,chi_min,chi_max, numer],numer:true,
    bL : makelist(b,b,b0,bmax,db),
    pts :  map('lambda([x],chi_sigma1(1,x,db)), bL),
    chiL : take(pts,1),
    chi_min : lmin(chiL),
    chi_max : lmax(chiL),
    plot2d([discrete, pts],[x,chi_min,chi_max],
         [xlabel,"chi"], [ylabel,"ln(dsigma/do)"],
         [style,[lines,3]]))$
```

We again make a separate plot for $\bar{b}$ in the range $(0.1, 1.6)$.

```
(%i4) sigma_plot(1,0.1,1.6,0.01);
(%i5) time(%);
(%o5) [152.06]
```

which produces

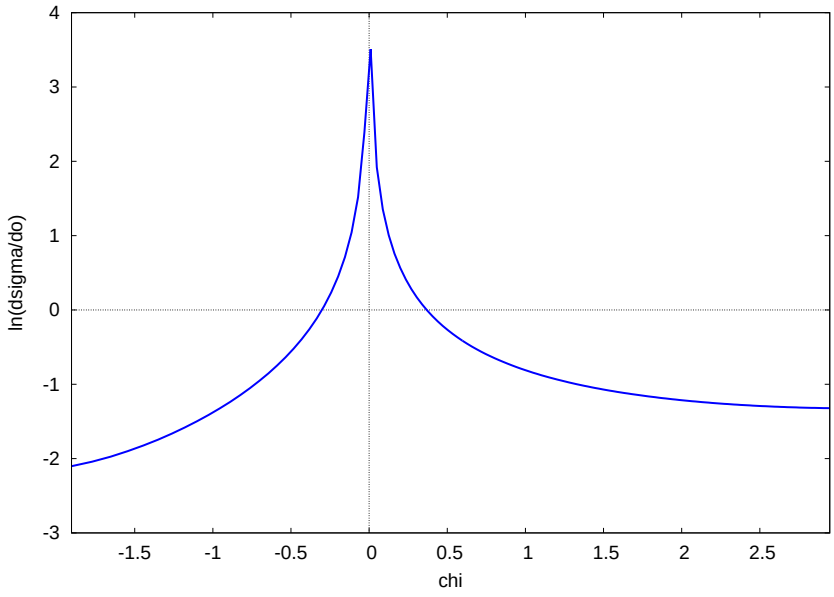

Figure 37: Log of Cross Section vs. χ for $\bar{E} = 1$ and $\bar{b} = (0.1, 1.6)$

We then make a separate plot for $\bar{b}$ in the range $(1.7, 3)$.

```
(%i6) sigma_plot(1,1.7,3,0.01)$
(%i7) time(%);
(%o7) [119.9]
```

which produces

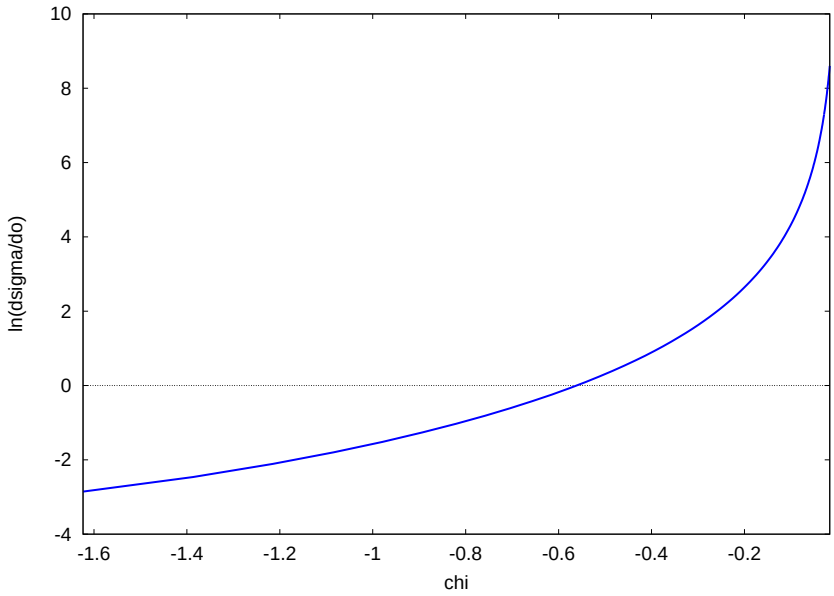

Figure 38: Log of Cross Section vs. χ for $\bar{E} = 1$ and $\bar{b} = (1.7, 3)$

6 R Scripts For The First Three Figures

The center of mass diagram (see 2.1) was produced with the **R** script **cmass.R**.

```
## cmass.R
plot(0:3,0:3,type="n",xlab="",ylab="",
           xaxt="n",yaxt="n",bty="n")  # no axes, no box
points(0.5,0.5,pch=20)
points(0.5,2.5,pch=19,cex=2)
text(0.3,2.5,expression(m[1]),cex=1.5)
arrows(0.5,0.5,0.5,2.5,code=2,cex=1.2)
text(0.4,1.5,expression(r[1]),cex=1.5)
text(0.4,0.5,"0")
points(2.5,2.5,pch=19,cex=2)
arrows(0.5,0.5,2.5,2.5,code=2,cex=1.2)
text(1.7,1.5,expression(r[2]),cex=1.5)
text(2.65,2.5,expression(m[2]),cex=1.5)
arrows(0.5,0.5,1.4,2.5,code=2,cex=1.2)
text(1,2,"R",cex=2)
arrows(0.5,2.5,2.5,2.5,code=2,cex=1.2)
text(1.4,2.6,"C. M.",cex=1.2)
text(1.2,2.8,expression(r[rel]),cex=1.5)
text(1.35,2.8,"=",cex=1.5)
text(1.45,2.8,expression(r[2]),cex=1.5)
text(1.55,2.8,"-",cex=1.5)
text(1.65,2.8,expression(r[1]),cex=1.5)
```

and executed via

```
> source("c:/k1/cmass.R")
```

The coordinate system diagram (see 2.1) was produced with the **R** script **coord.R**.

```
## coord.R

plot(0:3,0:3,type="n",xlab="",ylab="",
            xaxt="n",yaxt="n",bty="n")  # no axes, no box

arrows(0.25,0.25,2.75,0.25,code=2,lwd=2) # x axis
arrows(0.25,0.25,0.25,2.75,code=2,lwd=2)  # y axis
arrows(0.25,0.25,1.536,1.782,code=2,lwd=3,col="blue")  # position vector r
text(2.9,0.25,"+x",cex=2)
text(0.25,2.9,"+y",cex=2)
text(0.9,1.5,"r",cex=3)
points(1.536,1.782,pch=20,cex=1.75)
points(0.25,0.25,cex=2,pch=19)
text(0.12,0.12,"0",cex=2)
## arc of a circle with an arrow
plotcircle(r=1,mid=c(0.25,0.25),from=0,to=0.75,arrow=TRUE)
text(1.4,0.9,expression(theta),cex=3)  # greek letter theta
```

The function **plotcircle** is from the **shape** package which must be loaded before using the script.

```
> library(shape)
> source("c:/k1/coord.R")
```

The scattering angle definition Figure 3 was created using **rutherford_repulse.R**, **scatt.R**, and the **deSolve** and **shape** libraries (see the end comment in the **scatt.R** file, which is shown here:)

```
## scatt.R generic scatt angle plot

##   scatt_plot1 for generic scattering plot
##    repulsive rutherford potential.
##   calls init(e,b) to define local values of
##     xc,yc,xa,vcx,vcy,chi.


scatt_plot1 = function(e,b,tm,tp,dt,rchi,xmin,xmax,ymin,ymax) {

     n1 = 150  # vector element for incident rvec arrow

     # define local xc, yc, vcx, vcy, xa, chi

    init(e,b)

    #  symbolic expressions for acceleration components

    trajec = function(t, y, parms) {
     with( as.list(y), {
        r = sqrt(x^2 + y^2)
        dx = vx
        dvx = x/2/e/r^3   # repulsive case
        dy = vy
        dvy = y/2/e/r^3   # repulsive case
        list( c(dx, dvx, dy, dvy) ) } ) }

    #   integrate backwards from xc, yc

    cat("  backwards from xc, yc \n")
    yini = c(x = xc, vx = vcx, y = yc, vy = vcy)
    times = seq(0, -tm, -dt)
    out = ode(times = times, y = yini, func = trajec, parms = NULL)
```

```
##  make xLb and yLb global so they can be looked at outside program
xLb <<- out[,"x"]
yLb <<- out[,"y"]

x1 = xLb[n1]
y1 = yLb[n1]
cat(" x1 = ",x1," y1 = ",y1,"\n")
xfirst = tail(xLb, n=1)
cat(" xfirst = ", xfirst," \n")
plot(xLb, yLb, xlim = c(xmin, xmax), ylim = c(ymin, ymax),
      type = "l", col = "blue", lwd = 3, xlab = "X",
      ylab = "Y")

#   integrate forwards from xc,yc

cat("  forwards from xc, yc \n")
times = seq(0, tp, dt)
out = ode(times = times, y = yini, func = trajec, parms = NULL)

xL = out[,"x"]
yL = out[,"y"]
cat(" xlast = ", tail(xL, n=1)," ylast = ",tail(yL, n=1),"\n")
vxf = tail(out[,"vx"], n=1)
vyf = tail(out[,"vy"], n=1)
cat (" vx_last = ",vxf," vy_last = ",vyf,"\n")

lines(xL, yL, lwd = 3, col = "blue")

# add line y = b
abline(h = b, lwd=2)
abline(h = 0)
# add tick mark at origin
lines ( c(0,0), c(- 0.05,0.05), lwd=2)

#  add point at end of rvec arrow
points(x1,y1,pch=19,col="blue",cex=2)

#  add rvec arrow
arrows(0,0,x1,y1, code=2, length=0.1,  lwd = 2 )

# add rmin vector
arrows(0,0,xc,yc,code=2, length=0.1, col="red",lwd = 2)
text(0,0.8,expression(r[min]),cex=1.5)

#  show angle theta to rvec line, trial and error:
plotcircle(mid=c(0,0), from=0, to= 2.1, r=0.5, lwd = 2, arrow=TRUE)
text(0.8, 0.3, expression(theta),cex=2)
text(-1.2,0.6,"r",cex=2)

# add chi line
yf = b + rchi*sin(chi)
xf = xa + rchi*cos(chi)
cat(" xf = ",xf," yf = ",yf,"\n")
lines ( c(xa,xf), c(b, yf), lwd=2)

#  show angle chi
plotcircle(r=1,mid=c(xa,b), from=0,to=chi, arrow=TRUE)
text(0.8, 1.4, expression(chi),cex=2)
```

```
    # show impact parameter b
    arrows(-4.2,0,-4.2,1,code=3,length=0.1,lwd=2)
    text(-4,0.5,"b",cex=1.7)
    }

##   production of Fig.3, Scattering Angle Illustrated:
##
##  > source("c:/k1/rutherford_repulse.R")
##  > library(deSolve)
##  > library(shape)
##  > source("c:/k1/scatt.R")
##  > scatt_plot1(1,1,5.5,5,0.01,5,-4.54,1.87,0,4)
```