Master's Thesis

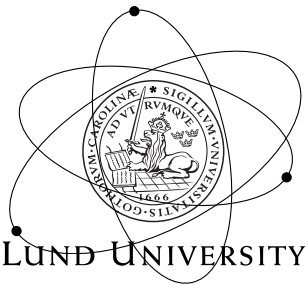
Studying Electron Dynamics using Attosecond Streaking

Classical Calculations with Monte Carlo Methods

Diploma Student:
Stefanos Carlström

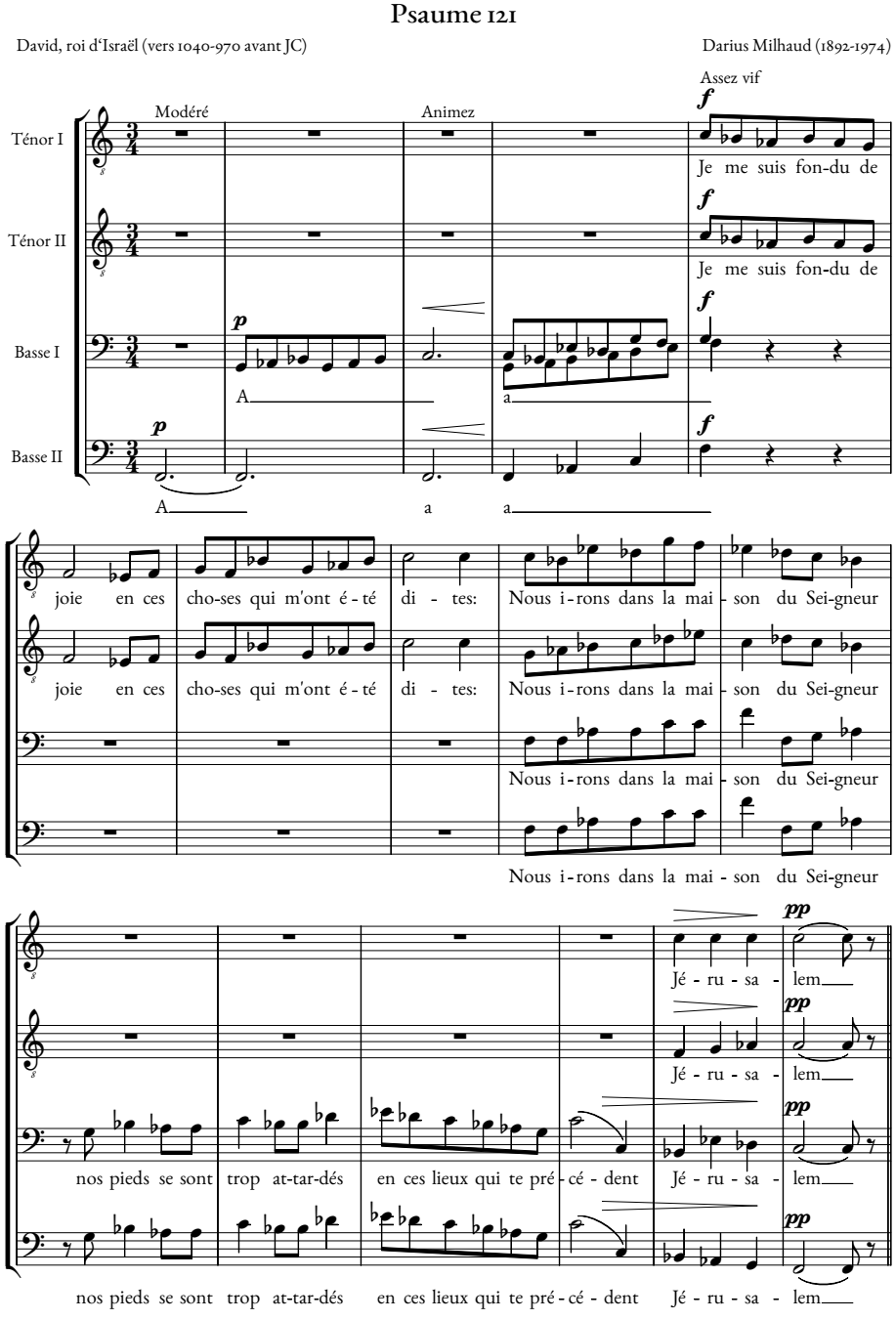
Supervisor:
Johan Mauritsson

Assistant Supervisor:
Erik Alerstam



Lund University

Faculty of Engineering
Division of Atomic Physics


Psaume 121
David, roi d'Israël (vers 1040-970 avant JC)
Darius Milhaud (1892-1974)
Modéré
Animez
Assez vif
Ténor I
Ténor II
Basse I
Basse II
Je me suis fon-du de joie en ces cho-ses qui m'ont é-té di-tes:
Nous i-rons dans la mai-son du Sei-gneur
Jé-ru-sa-lem
nos pieds se sont trop at-tar-dés en ces lieux qui te pré-cé-dent

Abstract

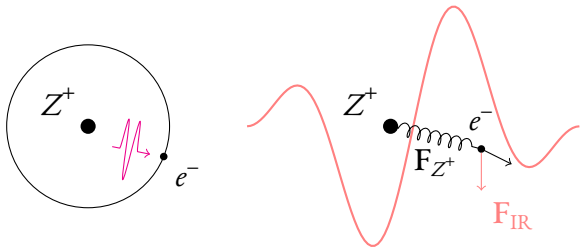
In this thesis, a program was implemented to study the electron dynamics of photoionization. These dynamics are probed by a streaking infrared field, that modulates the electrons' trajectories. Analytical quantum mechanical calculations for such systems are impossible without approximations for atoms more complex than hydrogen. However, using classical mechanics and Monte Carlo methods to capture the statistical behaviour of quantum mechanics, it was possible to extract the temporal dynamics of hydrogen and once ionized helium, and get good agreement with recent articles published by [Nagele et al. 2011] (*Journal of Physics B: Atomic, Molecular and Optical Physics*, 44), [Klünder et al. 2011] (*Physical Review Letters*, 106.14) and [Ivanov and Smirnova 2011] (*Physical Review Letters*, 107.21).

The program was implemented on a Graphics Processing Unit, the computational unit of a graphics card, which allows for massive parallelization of computations using inexpensive computer hardware available to normal consumers.

Populärvetenskaplig sammanfattning

Om man fotojoniserar en atom, dvs. sliter loss en elektron genom att belysa atomen, kommer elektronen att skjuta iväg från atomen med en hastighet som beror på ljusets energi (se figur nedan). Dock kommer atomkärnan att försöka få elektronen att återvända, jfr gravitationen. Om dessutom ett elektriskt fält, såsom infrarött laserljus verkar på elektronen under jonisationen och efteråt, kommer elektronen rörelse ytterligare förändras.

Elektronerna detekteras sedan och genom att mäta deras energi kan man beräkna vilken styrka det infraröda laserljuset hade vid jonisationstillfället (denna teknik kallas *streaking*). Det visar sig dock att detta värde inte helt stämmer överens med den styrka fältet faktiskt hade vid jonisationstillfället. Snarare passar värdet med fältets styrka vid en tidpunkt strax innan. Denna tidsskillnad beror på två saker: 1) genom att atomkärnan drar i elektronen fördröjs jonisationen och 2) genom att det infraröda fältet är närvarande under jonisationen påverkas elektronens rörelse i förhållande till atomkärnan på ett komplext sätt. Dessa tidsaspekter på fotojonisationen är mycket intressanta att studera för grundläggande forskning i atomfysik.



I (a) joniseras en atom av en inkommande ljuspuls och elektronen skjuter iväg. I (b) påverkas den lösa elektronen av två krafter. Den första kommer från atomkärnan som försöker drar i elektronen med kraften F_{Z^+}*. Ett elektriskt fält som accelererar elektronen ger upphov till den andra kraften,* F_{IR}.

Om det infraröda laserfältet är mycket starkt, kan det tvinga tillbaka elektronen till atomkärnan, förutsatt att det är riktat åt rätt håll under tillräckligt lång tid. När elektronen närmar sig kärnan, kan den förra spridas mot den senare, ”studsa”, och röra sig bort med en högre hastighet än den med vilken den närmade sig. Något liknande inträffar när en komet närmar solen och slungas iväg när den passerat.

I mitt projekt har jag tittat närmare på dessa fenomen med hjälp av klassiska (i motsats till kvantmekaniska, som annars är vanligast i atomfysiken), statistiska beräkningar av ett slag som kallas *Monte Carlo*-metoder (namnet kommer från statistiska studier av tärningskastande som ju inte är en obekant företeelse på kasinot i denna stad). Dessa har utförts på ett grafikkort i en vanlig dator. Sådana grafikkort är speciellt lämpade för sådana här statistiska beräkningar, där många oberoende försök måste göras för att ett tillförlitligt resultat skall uppnås.

Contents

Abbreviations and notation used

API Application programming interface

as Attosecond

CDF Cumulative distribution function

CPU Central processing unit

CTMC Classical trajectory Monte Carlo

FWHM Full width at half maximum

GPGPU General-purpose computing on graphics processing units

GPU Graphics processing unit

IR Infrared

ODE Ordinary differential equation

PDF Probability distribution function

QM Quantum mechanics

RK45 Runge–Kutta ODE solver of 5^{th} order with embedded 4^{th} order error estimator

SI Système international d'unités

SIMD Single-instruction multiple data

TDSE Time dependent Schrödinger equation

VMIS Velocity map imaging spectroscopy

XUV Extreme ultraviolet

$(\mathbf{a}|\mathbf{b})$ Scalar product of vectors $\mathbf{a}$ and $\mathbf{b}$

$\mathbf{e}_z$ Unit vector in the direction of $\mathbf{z}$

I_P The binding energy of the potential

τ is a time found by analytical QM derivation

t_S is a time found by numerical QM calculations

t_{CTMC} is a time found by (numerical) CTMC calculations

Ω The frequency of the ionizing XUV pulse

ω_{IR} The frequency of the streaking IR field

The primary interest of this thesis is to study the behaviour of photoionized electrons and how that behaviour is affected by changes in the surroundings. An electron bound to an atom is governed by quantum mechanics and must be treated using methods from this theory. For simple systems like hydrogen, exact solutions of the wavefunction can be found. Likewise, it is simple to find solutions for free particles in an external field. However, when both an atomic potential and an external field is present, quantum mechanics yield no easily found analytic solution. It is possible to acquire solutions numerically, but it can be difficult to separate which factors are responsible for the observable effects, since the calculations are performed with one bulk Hamiltonian. In classical mechanics it is easy to turn on and off different constituents, and draw conclusions on their influence. On the other hand, classical mechanics are deterministic, whereas quantum mechanics are probabilistic. It is possible to make up for this difference (to some extent and for certain problems), by using statistical methods called *Monte Carlo* methods. In this special application, a variant known as the *Classical Trajectory Monte Carlo* method is used.

The foundation for all different Monte Carlo techniques is the independent measurement that is repeated sufficiently many times, until a conclusion can be reached. This is known as the *law of large numbers*. Since the measurements are independent, they are very well suited for parallel computing, which is the second major interest of this thesis. Especially, how these calculations are efficiently implemented on graphics processing units, the computational cores of graphics cards in computers.

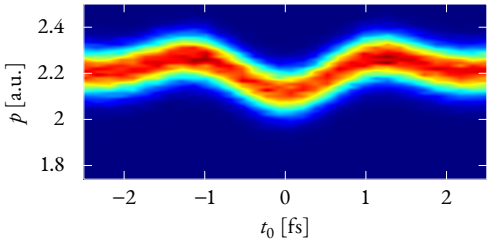


Figure 1.1: *A typical streaking spectrogram with the imprint of the infrared field clearly visible.*

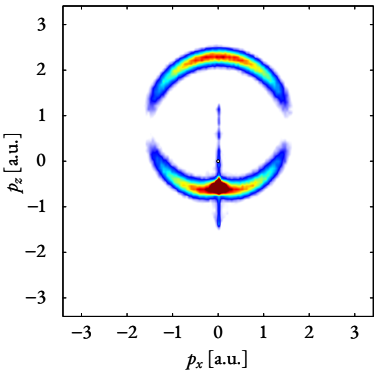


Figure 1.2: *A typical momentum distribution of the electron after scattering.*

With a working program it is then possible to study the dynamics of the electrons; the main calculations revolve around how the momenta of the electrons change when subjected to an oscillating infrared (IR) laser field. When the electrons hit a detector and their kinetic energies are measured, an imprint of the IR field is visible in the spectrogram, as well as an imprint left by the nuclei the electrons were bound to. The former imprint is clearly visible (see figure 1.1), whereas the latter is so small, it has to be extracted using numerical methods.

When the electrons travel close to the nucleus, the imprint left is more emphasized; the electrons

scatter and leave with higher velocity than what they had when they were incident on the nucleus. An example of this is shown in figure 1.2.

Much effort has been put into creating a program that can calculate these dynamics reliably and yet quickly enough to be useful. Nonetheless, for some of the scattering calculations mentioned above, they could easily last more than 48 hours for a single ionization time.

In section 2, an overview of the computational part of project is given, while section 3 deals with the physical aspects. Lastly, the conclusions are found in section 4.

GPGPU calculations and their suitability to MC methods

2

GPGPU — *General-purpose computing on graphics processing units* — is an evolving area in computational science where the processing power of graphics cards, originally targeted at running the increasingly realistic and demanding computer games, is leveraged for general-purpose calculations. The power of the graphics cards lies not in the speed of every single microprocessor on the card, but the huge amount of processors working at the same time. Below follows a brief description of how programming and execution of code on the GPU works. In this project, a GPU manufactured by Nvidia was used, along with the *application programming interface* (API) provided by the company, CUDA. For a more in-depth description of how the GPUs work and in the special case of Nvidia GPUs, see *e.g.* [NVIDIA 2011].

Each processor executes a *thread*, which is a processing unit that executes independently of the other threads of the program (but they all share the same *context* — access to the same memory blocks and so on). On the GPU a certain number of threads is executed together, in what is called a *warp*. A larger group of threads is called a *block*. All threads in a block is executed simultaneously (but each warp within the block is independent from the others), and they all have access to fast, on-die *shared memory* (typically ~10 kB in size). Lastly the blocks are organized in a *grid*, from which a *scheduler*[1] chooses the next block to execute. All threads have, at any time, access to a slower *global memory*, of considerably larger size (~1 GB).

[1] A scheduler is an algorithm, that given a batch of threads, chooses when to execute them and, in the case of a GPU, on which processor

If, for instance, one would like to calculate the Euler norm of a vector of length N

$$||\mathrm{v}||_2 = \sqrt{\sum_{i=1}^{N} v_i^2}$$

on a single processor, an implementation could be as follows.[2]

[2] Throughout the text, the code presented will be in C/C++. Some parts are taken directly from the program implemented in this project.

```
float vec[N] = { ... }; //A vector of N elements
float sum = 0;
for(int i=0; i<N; i++)
   sum += vec[i]*vec[i];
float norm = sqrt(sum);
```

If one, instead would like to implement the calculation in a parallel manner, one would construct a *kernel*, *i.e.* a small function, that calculates the square of a given number and places the result in a designated destination:

```
void square(float* elements, float* squares){
   int tId = getTid(); //Calculate the thread index
   squares[tId] = elements[tId]*elements[tId];
}
```

This kernel is executed by the GPU for each and every element in the vector `elements`, and every instance of the kernel runs on a separate processor. The thread index is the same as the processor number, *i.e.* which processor the kernel is executed on. In this way, the square of m (where m is the number of processors) elements is calculated, and the results are placed at the corresponding positions in the vector `squares`.

The sum is then calculated using a *parallel reduction*[3] algorithm and the square root is calculated on the CPU, since it is only performed once.

[3] Parallel reduction is an algorithm that is performed recursively in a tree-like manner, starting from the lowest level. For instance, to calculate a sum of a vector, the first step is to construct a vector of half the original size, with every element containing the sum of two elements in the original vector. If $N = 2^n$ where $n \in \mathbb{Z}^+$, the operation is done in n steps.

* * *

There are some aspects one must consider, in order to use GPUs for calculations. These are discussed on the following pages.

2.1 Orthogonality of the problem

The degree to which the problem can be broken down into smaller units that can be computed independently of each other, determines how successful the parallelization of the computations can be. If the units are not fully independent, that is, they have to communicate with each other to some degree, computational speed will be sacrificed, yielding longer calculation times.

2.2 Divergence yields sequentialization

There is an inherent limit in the parallelization available in a GPU. Separate threads are indeed run separately, but the hardware can be made more efficient if some assumptions are made. If a group of threads run identically the same code (*instructions*), but with different data some optimizations can be made (for instance using *Single-Instruction, Multiple Data* (SIMD) constructs). This is indeed the case for real-time graphics applications, but is not necessarily so for general-purpose calculations. What happens, is that when branching (also called divergence) occurs, *e.g.* an *if* clause, the different branches are executed in sequence, not in parallel. The threads that do not follow a certain branch are idle while that branch is executing, as in figure 2.1.

In many cases this can be solved by making the assumption that the calculations in all the branches together take less time than the sequentialization process does. For instance, calculating the parametric potential

$$V(r) = \begin{cases} -1/|r|, & |r| \geq \delta \\ \frac{1}{2\delta^3}\left(r^2 - \delta^2\right) - 1/\delta, & |r| < \delta \end{cases}, \quad V(r) \in \mathscr{C}^2$$

could be done in the conventional fashion:

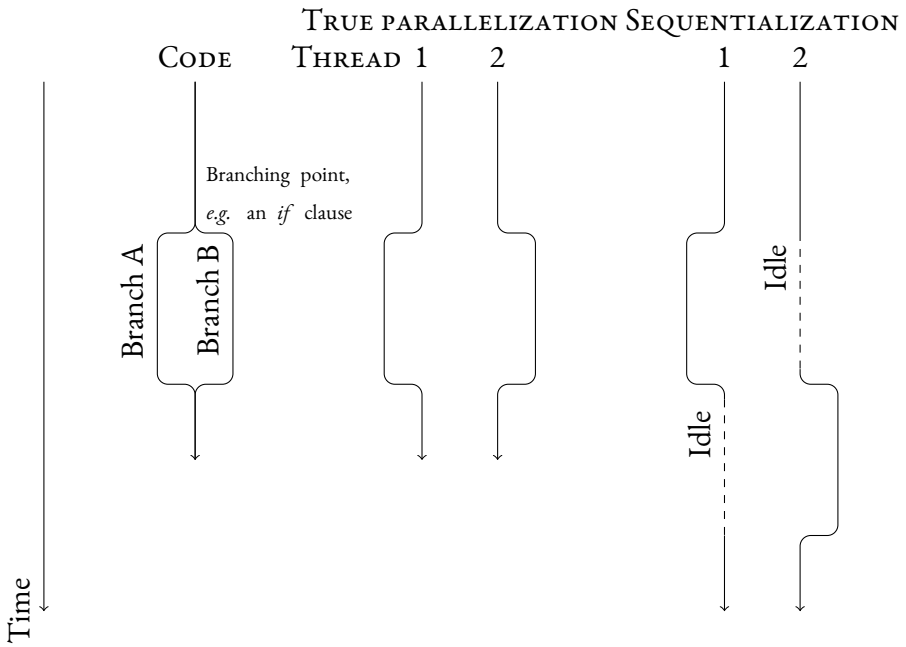


Figure 2.1: *An example of branching code is shown in this figure. After some time of execution, the path is divided into two, and in this example, the threads 1 and 2 take the A and B branches, respectively. If each and every thread were truly independent of the others, in terms of hardware implementation, these two branches would be executed in parallel. However, on a GPU all threads in a warp must execute exactly the same instruction, albeit with different data values. Therefore, branching constructs are split up as shown on the right, and one thread executes its branch while the other thread is idle.*

```
if(r>=delta)
  V=1.0f/abs(r);
else
  V=1/(2*delta3)*(r2-delta2)-1/delta;
```

or in a more GPU friendly way:

```
bool outside=(r>=delta);
V=(1.0f/abs(r))*outside +
  (1/(2*delta3)*(r2-delta2)-1/delta)*(!outside);
```

2.3 Random number generation

Generation of *pseudo-random numbers* is an art in itself, and is of utmost concern in Monte Carlo calculations. If the random numbers are used as stochastic variables on which the computations depend to capture a statistical behaviour, it is crucial that they do not repeat, since getting the same series of numbers more than once could potentially mean performing exactly the same computations more than once. This is even more complicated in parallel programs, where a repeated series of random numbers would make the parallelization pointless. To avoid this problem, this project takes the same approach as [Alerstam et al. 2009],[4] *i.e.* every thread is assigned its own unique random number ring, based on a *multiplier* and a starting *seed*. The ring will repeat itself, but only after a very long time (the period is $> 2^{60}$ numbers). This method is called *multiply-with-carry* and it was described by [Marsaglia and Zaman 1991].

[4] In fact, the implementation used is that of CUDAMCML, the program described in the reference.

* * *

There are a lot of things that have not been mentioned in this short overview, which has to be done to successfully utilize the computational power of a GPU.

In appendix C, an overview of how the calculations were implemented in code is given.

THE GOAL OF the attosecond (1 as $= 10^{-18}$ s) streak camera technique (first described by [Itatani et al. 2002]) was initially to measure and characterize attosecond pulses, in a manner analogue to that of classical streaking ([O'Brien 1936]). The idea is to probe the system dynamics with a *streaking field* that modulates the measurable quantities in a predictable manner. It is then possible to extract information of other processes affecting the measurements by comparing with the theoretically predicted result. However, it was quickly realized that the system used to measure (usually an ionization) was being measured at the same time. The attosecond streaking technique is thus able to take "pictures" of systems that evolve very rapidly.

In figures 3.1 and 3.2, the system studied is described. A single atom is ionized by a XUV pulse at a certain time t_0. After having escaped the potential, according to Einstein's formula for the photoelectric effect ([Einstein 1905]), the electron has attained a final kinetic energy of $W_k = \hbar\Omega - I_P$, where Ω is the frequency of the XUV pulse and I_P the binding energy. However, if simultaneous to the ionization a wiggling IR field is applied, the electron will be accelerated by the *Lorentz force*, in a predictable manner. This IR field is used as the streaking field, since its effects on the electron are known.

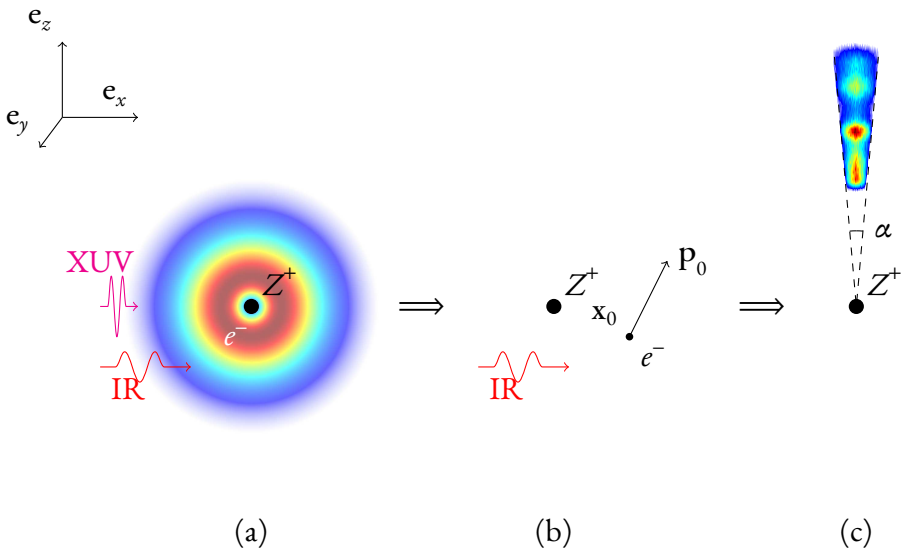


Figure 3.1: *(a) The electron is initially bound to the atom and is distributed according to its wavefunction. A XUV photon photoionizes the atom, and the electron is ejected. (b) The electron is now a free particle in the combined electric field of the IR pulse and the Coulomb potential. The initial position* x_0 *is a stochastic variable with the same distribution as the bound electron had, and its initial momentum is* $\mathrm{p}_0 = \sqrt{2(\Omega - I_P + V(\mathrm{x}_0))}$*, where* Ω *is the frequency of the XUV pulse. (c) Detection. If the electron has a final momentum inside a cone of* α*, it is detected. Electrons with higher energy arrive sooner at the detector.*

By, in a controlled fashion, ionizing at different times with regard to the IR field, and measuring the final velocity, a set of data is acquired (see figure 3.3). On this data one can then do computations to extract interesting information.

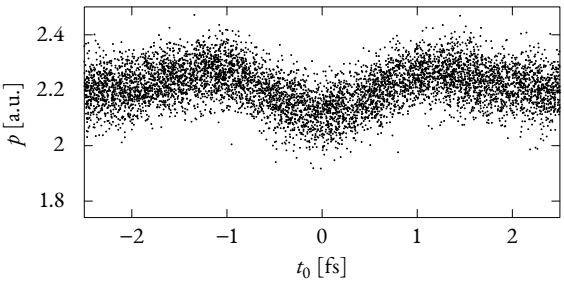


Figure 3.3: *Measured momenta for 10^4 electrons with different ionization times t_0.*

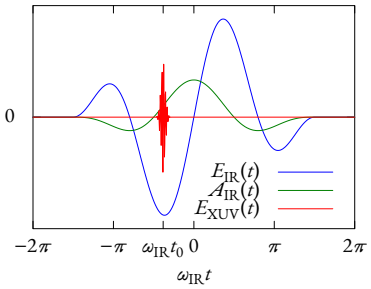


Figure 3.2: *In this figure the two pulses are shown: the IR pulse and the XUV pulse. In the calculations, ionization happens at time t_0, when the XUV pulse hits the atom. The electron is then ejected into the combined field of the nucleus and the IR pulse. $A_{IR}(t)$ is the vector potential of the IR pulse and arises with a negative sign in eq. (3.6).*

Until recently, the escape process could be regarded as instantaneous. However, [Wigner 1955] showed that there actually is an intrinsic time delay on the order of attoseconds. Using the streak camera technique mentioned above, it is possible to measure this *Wigner delay*, τ_W, provided that one can account for the effects of the measurement process.

In this project, one of the goals has been to evaluate to which extent a computational method using classical physics (known as the *Classical trajectory Monte Carlo* (CTMC) method) can be used to extract these time delays.

3.1 Equations of motion

To perform the streaking calculations, the effect of the streaking IR field must be known. The aim is to extract the time delays involved as a time shift. An electron that is ejected from an atom ionized in an external field, is subjected to the Lorentz force, originating from both atomic potential and the external field:

$$\mathrm{F} = q\,(\mathrm{E} + \mathrm{v} \times \mathrm{B})\,, \tag{3.1}$$

where E is the combined electric field of the atom and the external field and B is the corresponding magnetic field. $q = -e$ is the charge of the electron and v is its instantaneous velocity. According to Newton's second law, the force on a body is related to the acceleration it experiences via $\mathrm{F} = m\mathrm{a}$. This gives the equations of motion

$$\ddot{\mathrm{x}} = \mathrm{a} = \frac{q}{m_\mathrm{e}}\,(\mathrm{E} + \mathrm{v} \times \mathrm{B})\,. \tag{3.2}$$

This is a second-order differential equation which can be transformed into two coupled first-order differential equations:

$$\begin{cases} \dot{\mathbf{v}} = \frac{q}{m_e}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \\ \dot{\mathbf{x}} = \mathbf{v}. \end{cases} \tag{3.3}$$

This system is readily solved numerically using stepwise integration, but some simplifications can be made; using *atomic units* (see appendix A), many important quantities are set to unity, thereby simplifying equations and making it easier to avoid numerical round-off errors.[5] Furthermore, the influence of the magnetic field **B** can be neglected:

[5] Hereafter, all quantities will be in atomic units, unless explicitly stated otherwise.

$$B_0 = E_0/c_0 \implies \dot{\mathbf{v}} = -\left(E_0\mathbf{e}_E + \frac{v}{c_0}\mathbf{e}_v \times E_0\mathbf{e}_B\right) \approx -E_0\mathbf{e}_E = -\mathbf{E}, \quad v \ll c_0.$$

Only the external field is contributing, as on these time scales, the nucleus is stationary.

Thus equation (3.3) reduces to

$$\begin{cases} \dot{\mathbf{v}} = -\mathbf{E} \\ \dot{\mathbf{x}} = \mathbf{v}. \end{cases} \tag{3.4}$$

If only the external field had been present, the solution of eq. (3.4) would be simple:

$$\mathbf{v}_f = \mathbf{v}_0 - \int_{t_0}^{\infty} \mathrm{d}t\, \mathbf{E}_{\text{ext}} = \mathbf{v}_0 + [\mathbf{A}_{\text{ext}}]_{t_0}^{\infty} = \mathbf{v}_0 - \mathbf{A}_{\text{ext}}(t_0), \tag{3.5}$$

as $\mathbf{E}_{\text{ext}} = -\frac{\partial \mathbf{A}_{\text{ext}}}{\partial t}$ and $\mathbf{A}_{\text{ext}}(\infty) = 0$ ($\mathbf{A}_{\text{ext}}(t)$ is the vector potential of the electric field $\mathbf{E}_{\text{ext}}(t)$). In figure 3.4, it is clearly illustrated how the final velocity of the electron only depends on the initial velocity and the vector potential at the instant of ionization, as eq. (3.5) predicts. With a Coulomb field present, the solution is approximately ([Nagele et al. 2011])

$$\mathbf{v}_f \approx \mathbf{v}_0 - \alpha\mathbf{A}_{\text{ext}}(t_0 + t_S). \tag{3.6}$$

The factor α is close to unity and t_S is a time shift related to the Wigner delay, as well as measurement induced delays. This equation is an approximate solution, which holds as long as the Coulomb field can be seen as a perturbation, *i.e.* the electrons are never really close to the core, which is true as long as the IR field is "weak" ($\tilde{\gamma} = \sqrt{W_k/2U_p} > 1$, where W_k is the kinetic energy of the electron and $U_p = E_0^2/4\omega_{\text{IR}}$ is the quiver energy induced by the Ponderomotive force of the IR field. See [Mauritsson et al. 2008]). Another approximation is that the electrons are ejected instantaneously into the continuum at time t_0 when the XUV pulse arrives (thus implying that the XUV pulse is a

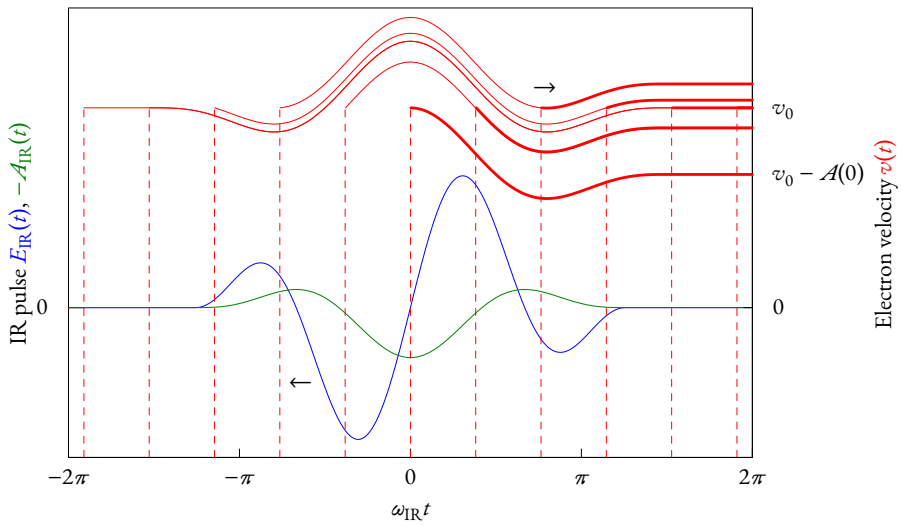


Figure 3.4: *In this figure equation (3.5) is illustrated. When an electron is ejected in an IR field and the Coulomb potential is neglected, it will transfer momentum according to* $\mathrm{F} = q\mathrm{E}_{IR} \implies v(t) = q\int_{t_0}^{t} \mathrm{d}t'\, E_{IR}(t')$. *When the pulse has ended, no forces are acting on the electron; it has thus reached its final velocity. Shown in blue is the electric field of the IR pulse, in green its associated vector potential, but with a negative sign, and in red the attained velocity of the electron for some different ionization times. One thing that is easily seen in this figure, is the fact that electrons released at time* $-t_0$ *and* t_0 *attain the same final velocity* v_f *as the vector potential in this case is symmetric.*

Dirac δ function; the finite temporal length of the pulse is accounted for via the spectrum of energies it contains and can transfer to the electrons).

Even though there are a lot of approximations leading to eq. (3.6), there is a huge benefit in having a solution on this form; the measurements of these very small times are reduced to a simple numerical fit of data. The key is the precise timing between the XUV and the IR pulse, t_0, that has to be done. This is still difficult to do in a laser lab, but it is possible to ionize from two different shells and extract a differential shift (see *e.g.* [Schultze et al. 2010]).

3.2 1D streaking

To find out what the influence of the Coulomb field is, equation (3.4) has to be solved exactly. Since quantum mechanics is probabilistic, Monte Carlo methods turn out to work rather well for this purpose; the initial values for position and momentum are randomized from the wavefunction's distribution, and subsequently, the problem is treated fully classically. It is important that the initial conditions are sampled randomly enough,[6] so as to not introduce patterns in the data, which may lead to false conclusions. The manner in which this is done is described above in section 2.3.

[6] On a computer, random numbers are only *pseudo-random*. It is important that the method used to generate them, does not yield repetitions, *i.e.* a sequence of numbers appearing more than once.

3.2.1 Procedure

The general procedure that is followed during one calculation is as follows:

1. Sample initial parameters such as ionization times t_0, initial positions $\mathbf{x}_0$ and initial velocities $\mathbf{v}_0$ of the electrons.
2. Propagate the electrons using eq. (3.4) until they have attained their final momenta, *i.e.* there are no more forces acting on them $\iff$ the IR pulse has ended and the electrons are far away from the nucleus.
3. Select those electrons that are considered as detected according to some criterion (see below) and store their final momenta.
4. Do a Least Squares fit of the spectrogram[7] to eq. (3.6) to extract the t_{CTMC} that corresponds to t_S.
5. Save the data.

[7] A typical result is seen in figure 3.5.

This procedure is typically repeated for a lot of different ionization times t_0 (on the order 10^7).

Sampling According to the Copenhagen interpretation of quantum mechanics, the absolute square of the wavefunction is to be interpreted as the probability to find the corresponding particle p at a certain point $\mathbf{r}_p$ in space:

$$\mathbb{P}\left(\mathbf{r}_p \in V\right) = \int_V \mathrm{d}^3\mathbf{r}' \, |\Psi(\mathbf{r}')|^2.$$

In this work, it was chosen to use the absolute square of the wavefunctions as the *probability distribution functions* (PDFs) from which the initial values are drawn for the calculations. A different approach taken by many, is to initially place the electron in randomly chosen Kepler orbits around the nucleus (see *e.g.* [Abrines and Percival 1966]). Not doing it this way may and may not have had implications for the variance of the results; see appendix E for a discussion on this.

The wavefunction for an s state in a hydrogen-like system in the ground state is

$$\Psi(\mathbf{r}) = R_{1,0}(r)Y_0^0(\vartheta,\phi) = 2Z^{3/2}\exp(-Zr)\sqrt{\frac{1}{4\pi}}.$$

The angular part, Y_0^0, is isotropic in this state, and the angles can therefore be sampled uniformly. The radial part is sampled using the *inverse transformation method* ([Devroye 1986]):

If a stochastic variable X has a certain PDF $f(x)$ and a *cumulative distribution function* (CDF) $F(x) = \int_{-\infty}^{x} \mathrm{d}x' f(x')$, then $Y = F(X)$ is uniformly distributed on the interval $[0, 1]$. Conversely, if the stochastic variable Y is uniformly distributed on the interval $[0, 1]$, then $X = F^{-1}(Y)$ has the PDF $f(x)$.

In the case of the radial function $R_{1,0}$ the inverse transformation method amounts to finding the r for which $U = \int_0^r \mathrm{d}r'\, r'^2 |R_{1,0}(r')|^2$, with U picked uniformly from the interval $[0, 1]$.

$$U = \int_0^r \mathrm{d}r'\, r'^2 |R_{1,0}(r')|^2 = 4Z^3 \int_0^r \mathrm{d}r'\, r'^2 \exp(-2Zr') =$$
$$\ldots = 1 - \exp(-2Zr)\,(1 + 2Z\,(Zr^2 + r)) \qquad (3.7)$$

Equation (3.7) is a non-linear equation, which can be solved using Newton–Raphson's method ([Bonnans et al. 2003]), which also needs the derivative.

The initial velocity of the electron is decided from two factors:

1. The initial speed, given by Einstein's formula for the photoelectric effect $W_k = \Omega - I_p$, where Ω is the frequency of the XUV photon and I_p is the ionization potential. To preserve momentum, the potential energy at the initial position, $V(r)$, has to be added as well. As the XUV pulse is of Gaußian shape in the temporal domain, it has a Gaußian shape in the frequency domain as well. The frequency Ω is therefore randomized from a Gaußian distribution via the Box–Müller transform ([Box and Müller 1958]).

2. The initial direction is in the one-dimensional case randomized isotropically, as in [Nagele et al. 2011]. In reality, when the electron is ionized from a s state, the velocity distribution should be randomized according to a p distribution. This is done in the two-dimensional case.

Finally, the time of ionization (t_0 in eq. (3.6)) is randomized uniformly over an interval around the extents of the IR pulse.

Propagation After sampling the different distributions for all electrons, the electrons are propagated according to eq. (3.4) using an adaptive Runge–Kutta ODE solver of the 5^{th} order with an embedded 4^{th} order error estimator (RK45, see [Palmer 2003]). This solver adapts it step size if the estimated truncation error is larger than a set tolerance, and can thus be used for differential equations that have singularities at points in space. Nevertheless, it is not a stiff solver, which means that for trajectories close to the nucleus, the step size will be so small that the required calculation time is very large.

It is in this step of the procedure that the strengths and the weaknesses of parallelization on a GPU are the most prevalent. All other steps are very deterministic.

Detection When the experiment is carried out in the lab, the electron energies are measured using *time-of-flight spectroscopy*, in which the time from ionization until the electrons hit the detector (usually a *micro-channel plate*, MCP) is registered. This time is then translated into a kinetic energy. Only those electrons that are emitted inside one of the two hemispheres are detected, and they are bent towards the detector using a *magnetic bottle*.[8]

In the calculations, the electrons are considered to be detected simply if they have a velocity vector inside a small cone some time after the IR pulse has ended. For instance, if the detector is placed in the positive e_z direction, and the detection cone angle is α, the electron with velocity v is detected if

$$(\mathrm{v}|\mathrm{e}_z) \geq |\mathrm{v}| \cos\alpha \iff \frac{v_z}{|\mathrm{v}|} \geq \cos\alpha.$$

Instead of registering the kinetic energy, the length of the momentum $\equiv m_\mathrm{e}|\mathrm{v}|$ is stored.

[8] If one were to detect isotropically, *i.e.* all electrons emitted, that would correspond to 4π (compare with area of sphere). In this case, only 2π is detected (area of hemisphere)

Fitting Finally, to extract the time shift t_{CTMC} corresponding to t_S in eq. (3.6), a non-linear least squares fit (see *e.g.* [Seber and Wild 2003]) of the measured momenta to the same equation is made.

Saving The data is saved to a binary file consisting of

- a header containing the calculation parameters
- the ionization times $\mathbb{R}$
- final momenta lengths $\mathbb{R}^+$
- final positions $\mathbb{R}^3$
- final velocities $\mathbb{R}^3$

This file is easily loaded into any mathematical software for further processing and plotting.

3.2.2 Results

In figure 3.5, a typical example of a streaking spectrogram can be seen. One can clearly see that equation (3.6) is a good approximation, since the difference is virtually indiscernible. There is however a time shift between the solid white line that represents equation (3.5) and the calculated momentum distribution. This time shift is readily extractable using the method described above, and for this certain calculation it is approximately −6.6 as (numerical solution of the *time dependent Schrödinger equation* (TDSE) yields −6.9 as, [Nagele et al. 2011]).

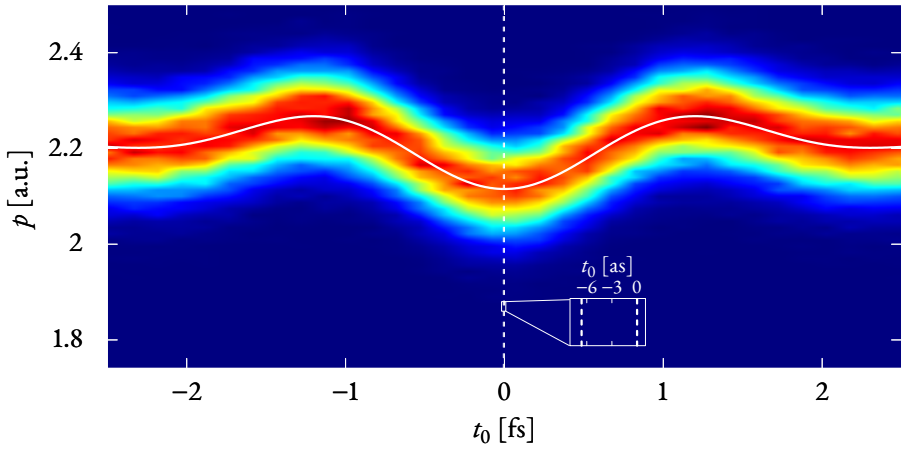


Figure 3.5: *Streaking spectrogram for a hydrogen atom streaked by an IR field. This calculation was done using $2^{24}(\approx 1.7\cdot 10^7)$ electrons. Shown also, as a solid white line, is equation (3.5). The right dashed vertical line corresponds to t_S=0 and the left dashed vertical line to the time shift induced by the Coulomb potential. In this case a XUV pulse of 80 eV was used with a duration of 200 as (see section 3.2.3 for a discussion on this subject).*

3.2.3 On pulse duration

The temporal width of a *Fourier limited pulse* is usually defined as the width of the intensity profile. The spectral width is related to the temporal width as (in SI units)

$$\Delta E_{\mathrm{FWHM}} = \frac{4\hbar \ln 2}{\Delta t_{\mathrm{FWHM}}}.$$

From this, it is seen that a spectral width of 9.1 eV is required to generate a XUV pulse of 200 as duration. However, this leads to a large distribution of the experimental data points to which the fit is made, thus yielding a large statistical variance of the extracted time shifts. It was therefore decided to use pulses of narrower spectral width in the calculations to yield a more precise time shift, although the approximation that the ionization by the XUV pulse is instantaneous does not hold any longer; a spectral width

of only 1.65 eV (which was the width most often used) cannot generate pulses shorter than 1.1 fs, which is of the order of the streaking IR pulse.

3.2.4 Sweeping the XUV center frequency

By giving the XUV photon different energies, the time shift in eq. (3.6) changes, as more energetic electrons escape the potential more easily. In the limit of infinite energy the ionization process is truly instantaneous. By measuring the time shift in this fashion, the *Wigner delay* can be extracted, if all other effects can be accounted for. A lot of work (see *e.g.* [Klünder et al. 2011] and [Dahlström 2011]) has been invested in this fundamental quantity in atomic physics, since this delay is needed to verify a lot of theory. Recently, an analytical approximation was derived by [Ivanov and Smirnova 2011] which seems to fit very well with the numerical solution to the TDSE from [Nagele et al. 2011] and [Klünder et al. 2011]. However, it was derived only for hydrogen, so a rederivation for hydrogen-like systems, with an attractive Coulomb potential Z/r was made (see appendix R). The approximation, as well as the numerical data are used as rulers for the quality of the classical results of this work. In figure 3.6, the time shifts for H and He^+ for different XUV energies can be seen, as well as some references.

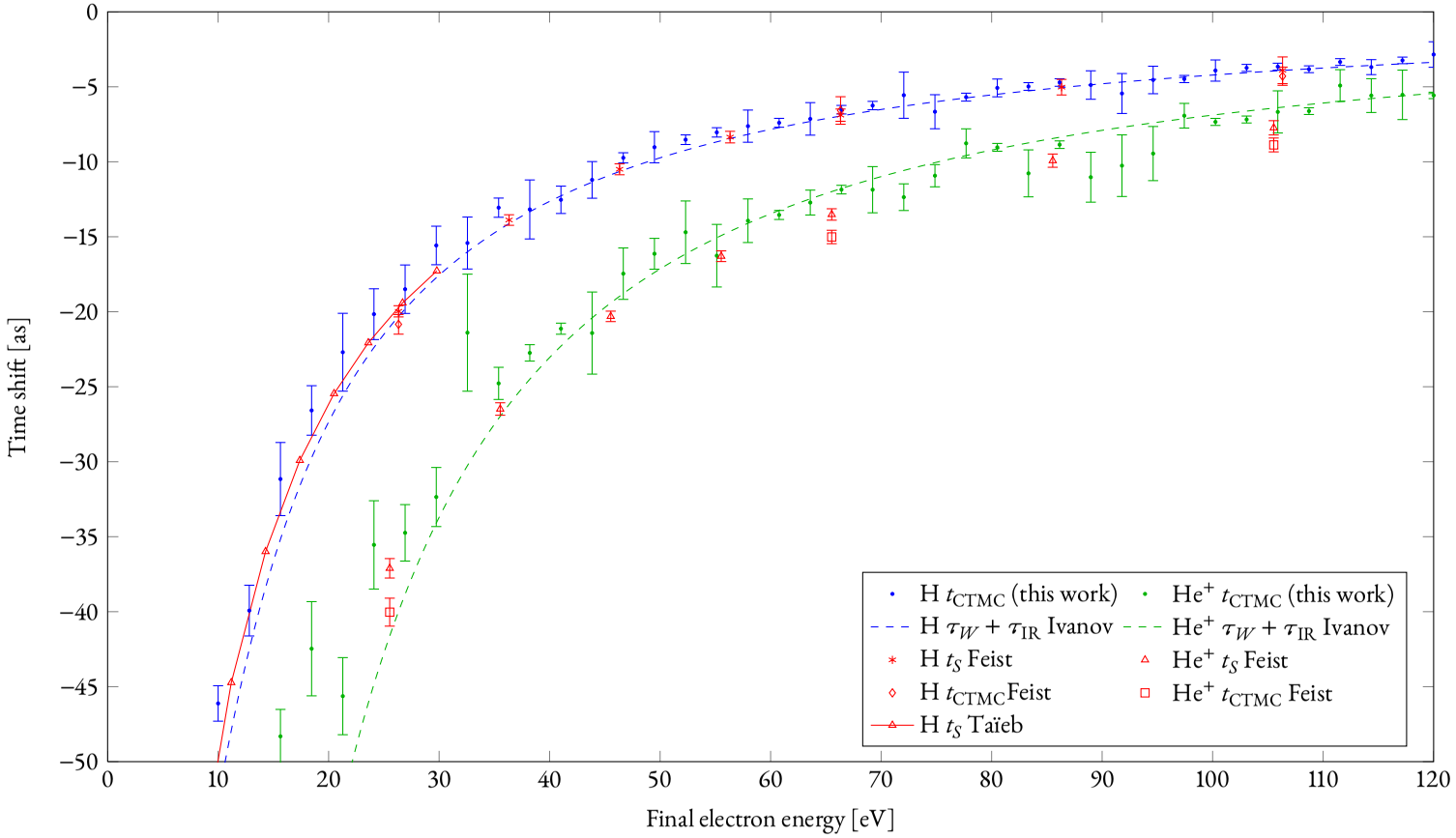


Figure 3.6: *Temporal shift t_{CTMC} for different XUV ionizing energies, as well as some reference data (all reference data in red colour, data calculated by J. Feist taken from [Nagele et al. 2011], data calculated by R. Taïeb taken from [Klünder et al. 2011]). Shown also as dashed lines is the analytical approximation by [Ivanov and Smirnova 2011].*

3.2.5 Smooth potentials and their implications

The RK45 ODE solver is an explicit solver that does not deal well with singularities. Since it is adaptive, it will take shorter integration steps when the estimated truncation error is too large. This means that the amount of steps needed for trajectories close to the nucleus will be extremely large, which is equivalent with very long calculation times. To alleviate this, the Coulomb potential can be smoothed from a true singularity into a function that is $\mathscr{C}^2$ continuous, *i.e.* two times differentiable. A traditional way of doing this, is by approximating the Coulomb potential by $V(r) = (r^2 + \varepsilon)^{-1/2}$, $\quad 0 < \varepsilon \ll 1$. However, using this potential seems to actually increase the calculation times, probably because the problem becomes stiff over a larger volume of space. Instead the following approximation was chosen:

$$V(r) = \begin{cases} -1/|r|, & |r| \geq \delta \\ \frac{1}{2\delta^3}\left(r^2 - \delta^2\right) - 1/\delta, & |r| < \delta \end{cases}, \quad V(r) \in \mathscr{C}^2. \tag{3.8}$$

This potential retains the Coulomb potential behaviour outside the spherical volume of radius δ. A plot of the potential is seen in figure 3.7. To verify the meaningfulness of this potential, an estimation of its impact on calculation errors and calculation speed was made by making many XUV energy sweeps for different values of δ. The calculation error for each δ was taken to be

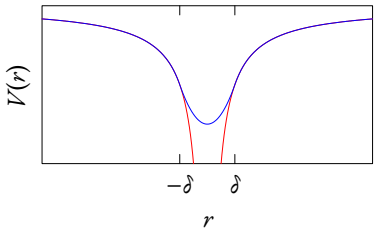


Figure 3.7: *The blue curve is the smooth potential (3.8) and the red curve is the original Coulomb potential.*

$$\frac{\sqrt{\sum_i^N \left| t_{\mathrm{CTMC}}^{(i)} - t_S\left(W_k^{(i)}\right)\right|^2}}{N},$$

where the values for t_S were the same as the numerical references in figure 3.6. The result can be seen in figure 3.8. It seems that the error is not very much affected by the choice of δ, but there is an optimum region for δ in terms of execution time.

3.3 2D streaking (VMIS)

Velocity Map Imaging Spectroscopy (VMIS) is two-dimensional technique for studying the dynamics of atomic and molecular systems (see [Eppink and Parker 1997]). The mode of operation is as follows: the sample is irradiated with laser pulses, whereupon it is ionized and the electrons are ejected. The electrons propagate as an expanding spherical shell, which is accelerated by an electric field generated by two electrodes. The spherical shell is projected onto a fluorescent screen, which is imaged for further analysis. Via an inverse Abel transform (or using iterative techniques, see [Dasch 1992] and

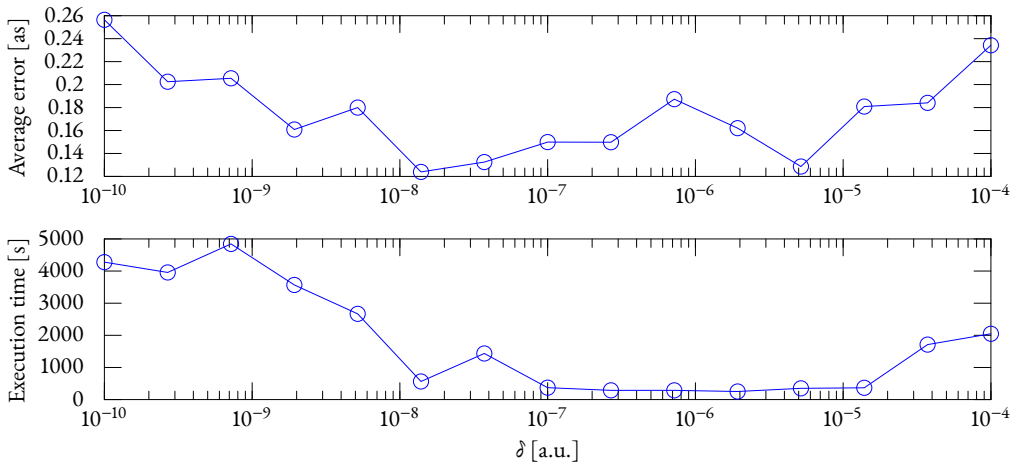


Figure 3.8: *Results from using a smooth potential with varying δ. In the plot on the top, it is seen that the error as defined above only varies within the same order of magnitude. However, the execution time varies dramatically.*

[Vrakking 2001]), the spherical shell is reconstructed, and a slice of thickness Δy along the xz-plane in the notation of figure 3.1 is extracted. From this slice it is possible to deduce the energies of the electrons as well as their angular distribution.

While doing theoretical calculations, it is not necessary to go through the steps of projecting the electron cloud and then do the inverse Abel transform, since the electron cloud is already represented in $\mathbb{R}^3$ coordinates in-memory. The only thing that has to be done is the slicing, and that step replaces the detection step described in section 3.2.1 above. Instead of specifying a cone angle α, a slicing thickness Δy has to be provided instead.

Another difference in the two-dimensional case as opposed to the one-dimensional procedure, is that (as mentioned above) the initial direction of the electrons are sampled according to the p distribution (Y_1^0) (if ionizing from an s state) instead, so as to yield results in agreement with reality.

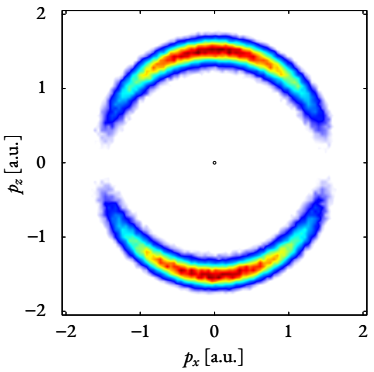


Figure 3.9: *Typical VMIS result. The two lobes of the* p *distribution are clearly visible. No IR field was present in the calculations.*

Lastly, the VMIS images are captured for one fixed delay between the XUV and the IR pulse, one fixed t_0, otherwise the image would be smeared as the electron clouds would have travelled different distances.

A typical result from a VMIS calculation performed on hydrogen can be seen in figure 3.9. What is shown is the momentum distribution of the electrons after pho-

toionization, which is, in effect, a Fourier transform of the spatial distribution. Normally, one is interested in the energy though, which is related to the momentum as $W_k = p^2/2m$.

3.3.1 Rocking the electron cloud

With an IR field present during ionization, the electron cloud will "float" in it, as a buoy does on the water surface. According to equation (3.6), the final acquired momentum will be shifted by the vector potential of the IR field at the ionization instant. However, in the time span between ionization and detection at a time at which the IR pulse has since long ended, the electron cloud will be swept back and forth, being rocked as a swing. If then the intensity of the streaking IR field is increased, *i.e.* $\tilde{\gamma} \to 1$, the approximation (3.6) is no longer valid as the electron cloud is swept back to the core. Instead other behaviour can be studied, such as electrons scattering from the nucleus. This scattering can be thought of as an analogue to the slingshot effect as used by space vehicles to escape our solar system — indeed, both the Coulomb force and the gravitational force have a $1/r^2$ dependence.

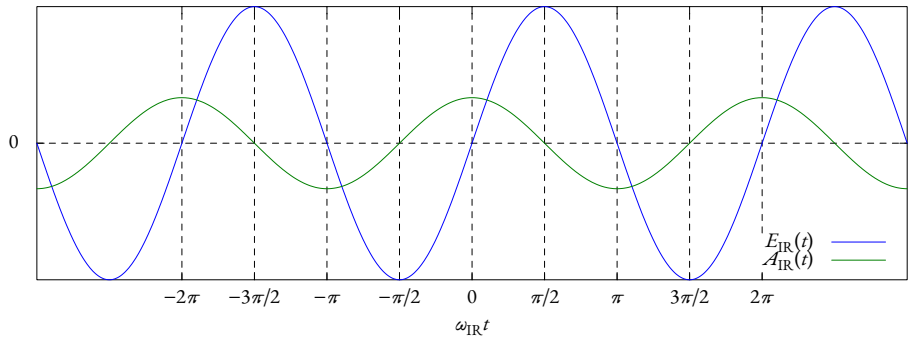


Figure 3.10: *An infinitely oscillating IR field along with its vector potential. Every electron ejected at a time t_0 such that $E_{IR}(t_0)$=0 has a possibility of returning to the core and scatter.*

In figure 3.10, an IR pulse of considerably longer duration than that in figure 3.2 is depicted. An electron ejected at any time $t = n\pi$, $n \in \mathbb{Z}$ (*i.e.* $E(t) = 0$), has a full half-cycle to transfer momentum according to $\mathbf{F} = q\mathbf{E}$. This gives the electron a chance of returning to the core. In contrast, ionization at other times will yield both acceleration and deceleration since the electric field changes sign before the electron has reached the core.

When the pulses look like those in figure 3.2 instead, there are only two ionization times where a return to the core is possible; $\omega_{\mathrm{IR}}t_0 \approx -3\pi/4$ and $\omega_{\mathrm{IR}}t_0 \approx 0$. This is in excellent agreement with the series of VMIS figures found in appendix F.1.1.

Trajectories To further reinforce this analysis, the individual trajectories of a single electron initially placed approximately a Bohr radius away from the nucleus in the positive z direction, were studied for different ionization times and some different conditions. The results can be seen in figures 3.11–3.15.

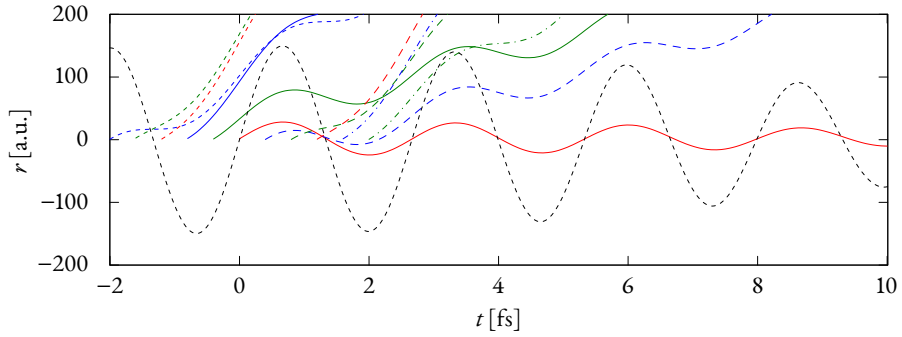


Figure 3.11: *Electron trajectories for different ionization times with $\tilde{\gamma}$=1. In this figure the IR pulse has a longer duration (in this case 20 fs,* i.e. *the pulse is not single-cycle) and it can make the electrons oscillate in the field, provided that there is no nucleus present. The streaking IR field is shown as a dashed black line.*

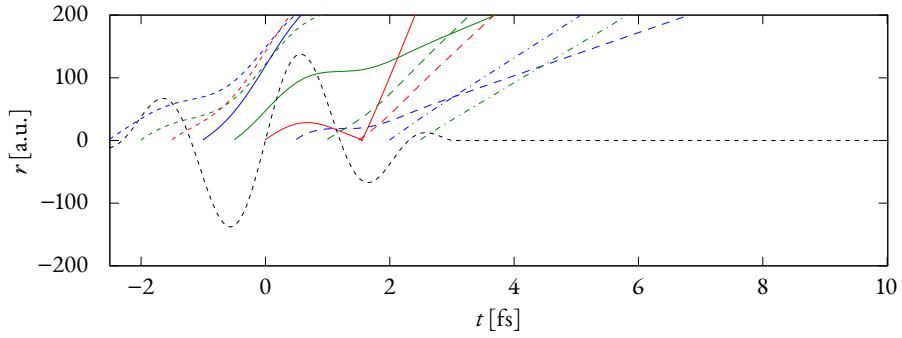


Figure 3.12: *Since the electron is initially placed some distance away from the nucleus in the positive z direction and the pulse is single-cycle ($\tilde{\gamma}$=1), only one trajectory (with t_0=0) is able to scatter from the nucleus. The electron trajectory scattering from the nucleus mentioned above with $\omega_{IR}t_0 \approx -3\pi/4$, must start some distance away from the nucleus in the* negative *z direction, since otherwise that electron is initially accelerated away from the nucleus.*

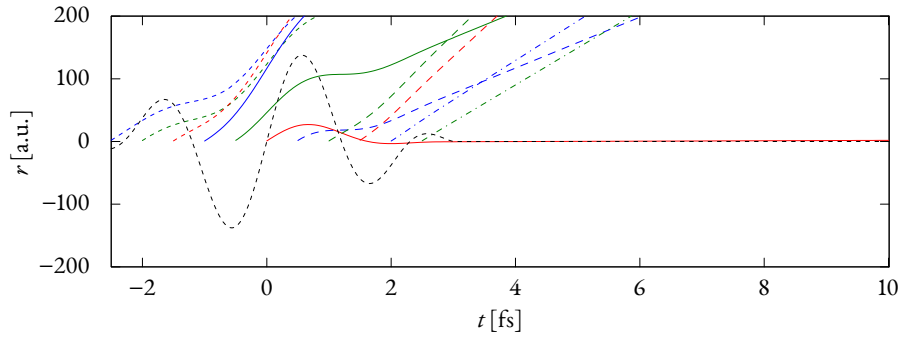


Figure 3.13: *Same as in figure 3.12, but this time without a nucleus present for the electrons to scatter from.*

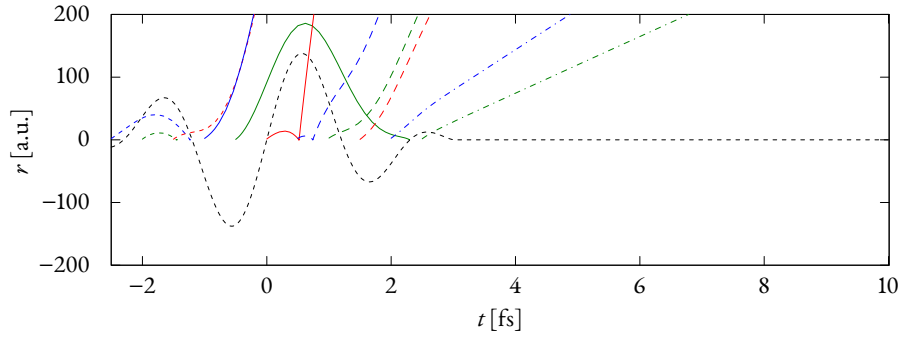


Figure 3.14: *Electron trajectories for different ionization times with $\tilde{\gamma}$<1. It is now possible for more electron trajectories to return to the core and scatter.*

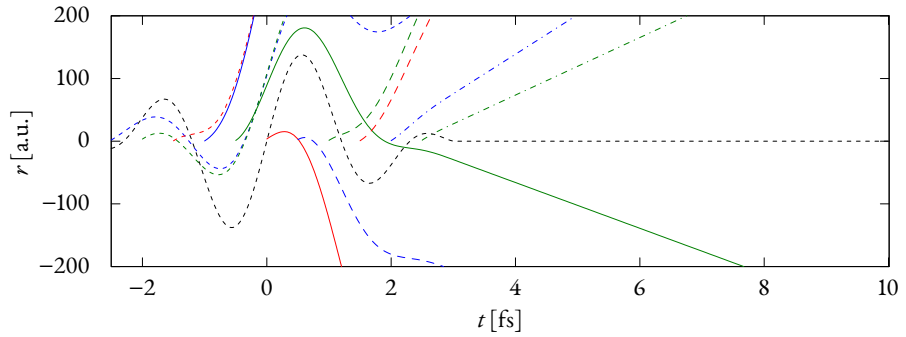


Figure 3.15: *Same as in figure 3.14, but this time without a nucleus present for the electrons to scatter from.*

3.3.2 Looking from a different angle

As a final test of the VMIS calculations, the polarization between the XUV pulse and the streaking IR field was gradually changed:

$$\mathrm{E}_{\mathrm{XUV}}(t) = E_{\mathrm{XUV}}(t)\mathrm{e}_z, \quad \mathrm{E}_{\mathrm{IR}}(t) = E_{\mathrm{IR}}(t)\mathrm{e}_{E_{\mathrm{IR}}}$$

$$\text{with } \left(\mathrm{e}_z \middle| \mathrm{e}_{E_{\mathrm{IR}}}\right) = \cos\alpha, \ \alpha = \{0^\circ, 10^\circ, 20^\circ, \dots\}.$$

This is something that is very difficult to do when solving the TDSE numerically, since it adds another dimension to the computations and the memory requirements are drastically increased.

The result can be seen in appendix F.1.2. As is expected, no scattering takes places when $\alpha = 90^\circ$, as the p distributed electron has no velocity distribution in the e_x direction (see figure 3.16).

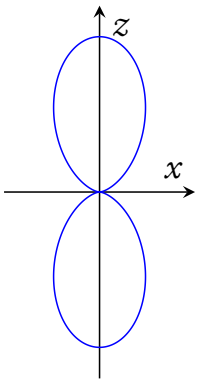


Figure 3.16: *The* p *distribution, which for a free electron corresponds to its velocity distribution. There is a cylindrical symmetry about the z axis.*

3.3.3 Quantum mechanics comparison

One could argue that the scattering seen above could be attributed to numerical instability. However, very similar scattering effects were found by [Mauritsson et al. 2008] when solving the TDSE numerically. They compared with experimental results, and while they could not discern typical scattering patterns, some features could not be explained without post-ionization electron-atom interaction.

A comparison was made for the VMIS calculations with and without an atomic potential present, and the difference between the two was calculated. The result can be seen in figure 3.17. The same comparison, but made using numerical calculations of the TDSE is depicted in figure 3.18. The agreement is very good.

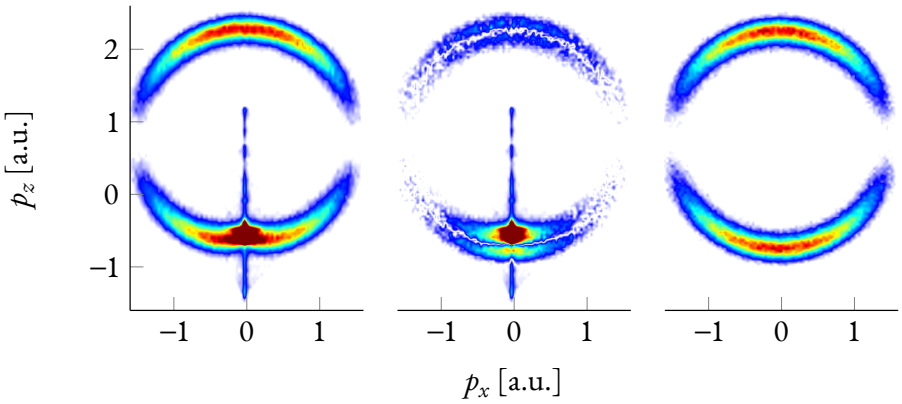


Figure 3.17: *On the left is the result from a calculation with a nucleus present, with the ionization time t_0=0,* i.e. *one of the ionization times with electron trajectories that can scatter from the nucleus. The IR field is so strong that a collision is possible. On the right is the result of a calculation with exactly the same conditions as the calculations on the left, with the exception that there is no nucleus. In the middle the difference between the two is plotted. The ratio between the upper and lower lobe is larger than one order of magnitude.*

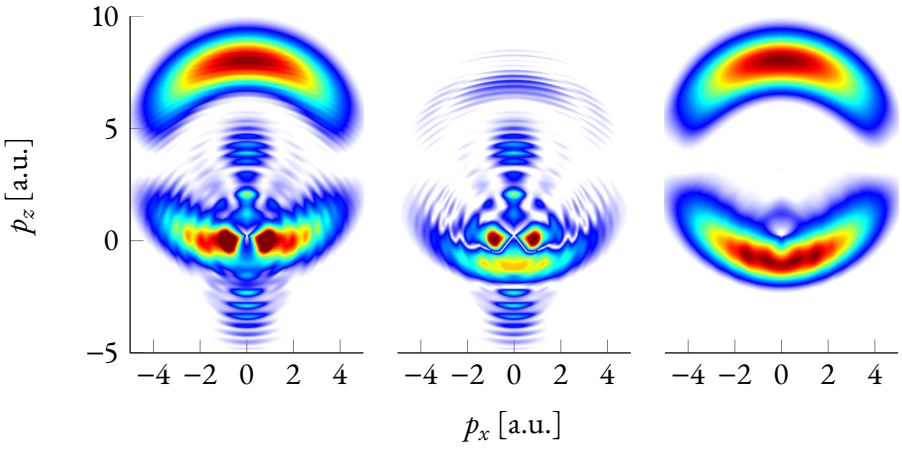


Figure 3.18: *These calculations correspond to those in figure 3.17, but they are performed by solving the TDSE instead. That is why there are such non-classical phenomena as interference present. In the right picture there are some artifacts visible, since the nucleus had to be artificially removed after ionization. Data courtesy of J. Mauritsson.*

4 Conclusions/outlook

STRIKINGLY GOOD RESULTS were achievable using the CTMC method (see figures 3.6 and 3.17–3.18), despite the sampling errors made (see appendix E). If one could overcome some of the difficulties of programming on a graphics card (some of the calculations lasted more than 48 hours), the classical methods have a promising future, since it is possible to perform calculations that are difficult to do using quantum mechanics (*e.g.* comparing what happens after photoionization with and without potential present, since the potential is necessary for the wavefunction of the free electron to be correct). QM calculations also need a lot of memory (on the order of GB for one wavefunction), whereas the corresponding classical calculations only need 32 B of storage per electron.[9]

[9] For good statistics, on the order of 10^7 electrons are needed $\Longrightarrow$ ~320 MB memory is required.

Further work has to be invested to be able to perform the classical calculations for more general systems. At this point, the program implemented only works for hydrogen-like systems, and is only tested for H and He^+. A first step would be to try ionizing from *e.g.* highly excited CO molecules (using an effective potential), to study the trajectories taken by the electrons attracted by two nuclei. Something that also would be interesting to try is the dynamics of double photoionization, *i.e.* the trajectories of two photoelectrons, but it is not entirely clear how one would treat both of them as classical particles at the same time (thus neglecting eventual interference phenomena).

The classical methods could also benefit from more storage space and processing power; performing the calculations on a conventional cluster, it would be possible to use stiff ODE solvers which do not fit on the GPU, and that have highly divergent execution paths. Furthermore, as the classical methods allow studies of single electron trajectories, it would be possible to study these more closely and gain a better understanding of systems that are too complex to understand using normal QM methods.

References

Abrines, R. and I. Percival (1966). "Classical theory of charge transfer and ionization of hydrogen atoms by protons". In: *Proceedings of the Physical Society* 88, p. 861.

Alerstam, E., T. Svensson, and S. Andersson-Engels (2009). *CUDAMCML User manual and implementation notes*.

Bonnans, J., J. Gilbert, C. Lemaréchal, and C. Sagastizábal (2003). *Numerical optimization: Theoretical and practical aspects*. New York: Springer.

Box, G. and M. Müller (1958). "A note on the generation of random normal deviates". In: *The Annals of Mathematical Statistics* 29.2, pp. 610–611.

Crockford, D. (2002). *Introducing JSON*.

Dahlström, J. M. (2011). "Light-Matter Interaction on the Attosecond Timescale". PhD thesis. Lund University. ISBN: 978-91-7473-140-8.

Dasch, C. (1992). "One-dimensional tomography: a comparison of Abel, onion-peeling, and filtered backprojection methods". In: *Applied Optics* 31.8, pp. 1146–1152.

Devroye, L. (1986). *Non-Uniform Random Variate Generation*. Springer.

Einstein, A. (1905). "Über einen die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt". In: *Annalen der Physik* 322.6, pp. 132–148.

Eppink, A. and D. Parker (1997). "Velocity map imaging of ions and electrons using electrostatic lenses: Application in photoelectron and photofragment ion imaging of molecular oxygen". In: *Review of scientific instruments* 68, p. 3477.

Göddeke, D., R. Strzodka, and S. Turek (2005). *Accelerating double precision FEM simulations with GPUs*.

Häßler, S. (2009). "Generation of Attosecond Pulses in Atoms and Molecules". PhD thesis. Université Paris-Sud XI and Commissariat à l'Energie Atomique, Saclay.

Itatani, J., F. Quéré, G. Yudin, M. Ivanov, F. Krausz, and P. Corkum (2002). "Attosecond streak camera". In: *Physical Review Letters* 88.17, p. 173903.

Ivanov, M. and O. Smirnova (2011). "How Accurate Is the Attosecond Streak Camera?" In: *Physical Review Letters* 107.21, p. 213605.

Klünder, K., J. M. Dahlström, M. Gisselbrecht, T. Fordell, M. Swoboda, D. Guénot, P. Johnsson, J. Caillat, J. Mauritsson, A. Maquet, R. Taïeb, and A. L'Huillier (2011). "Probing single-photon ionization on the attosecond time scale". In: *Physical Review Letters* 106.14, p. 143002.

Marsaglia, G. and A. Zaman (1991). "A new class of random number generators". In: *The Annals of Applied Probability*, pp. 462–480.

Mauritsson, J., P. Johnsson, E. Gustafsson, M. Swoboda, T. Ruchon, A. LHuillier, and K. J. Schafer (2008). "Coherent Electron Scattering Captured by an Attosecond Quantum Stroboscope". In: *Physical Review Letters* 100, p. 073003.

Nagele, S., R. Pazourek, J. Feist, K. Doblhoff-Dier, C. Lemell, K. Tőkési, and J. Burgdörfer (2011). "Time-resolved photoemission by attosecond streaking: extraction of time information". In: *Journal of Physics B: Atomic, Molecular and Optical Physics* 44, p. 081001.

NVIDIA (2011). *CUDA C Programming Guide, version 4.0.*

O'Brien, B. (1936). "A High Speed Running Film Camera for Photographic Photometry". In: *Physical Review* 50.4, pp. 400–401.

Palmer, G. (2003). *Technical Java: developing scientific and engineering applications*. Prentice Hall. ISBN: 0131018159.

Schultze, M., M. Fieß, N. Karpowicz, J. Gagnon, M. Korbman, M. Hofstetter, S. Neppl, A. Cavalieri, Y. Komninos, T. Mercouris, et al. (2010). "Delay in photoemission". In: *Science* 328.5986, p. 1658.

Seber, G. and C. Wild (2003). *Nonlinear regression*. Wiley series in probability and statistics. Hoboken, NJ: Wiley-Interscience. XX, 768. ISBN: 978-0-471-47135-6.

Vrakking, M. (2001). "An iterative procedure for the inversion of two-dimensional ion/photoelectron imaging experiments". In: *Review of Scientific Instruments* 72, p. 4084.

Wigner, E. (1955). "Lower limit for the energy derivative of the scattering phase shift". In: *Physical Review* 98.1, pp. 145–147.

Appendices

Atomic units are used to simplify calculations in atomic physics via a coordinate transform in which important quantities are set to unity. In the Hartree atomic units system, the following identities hold:

$$m_\mathrm{e} = e = \hbar = 4\pi\varepsilon_0 = 1$$

All other quantities are derived from these, giving the following values in SI units for one unit in atomic units (table taken from [Häßler 2009]):

Quantity			Value
Angular momentum		$\hbar =$	$1.054571726 \cdot 10^{-34}$ Js
Mass		$m_\mathrm{e} =$	$9.109383 \cdot 10^{-31}$ kg
Charge		$e =$	$1.60217653 \cdot 10^{-19}$ C
Length	$a_0 =$	$\dfrac{4\pi\varepsilon_0\hbar^2}{m_\mathrm{e}e^2} =$	$5.2917720859 \cdot 10^{-11}$ m
Velocity	$v_\mathrm{B} =$	$\dfrac{e^2}{4\pi\varepsilon_0\hbar} =$	$2.1876912633 \cdot 10^{6}$ m/s
Momentum		$m_\mathrm{e}v_\mathrm{B} =$	$1.99285166 \cdot 10^{-24}$ kg m/s
Time	$\tau_0 =$	$\dfrac{a_0}{v_\mathrm{B}} =$	$2.41888430 \cdot 10^{-17}$ s
Frequency		$\tau_0^{-1} =$	$4.13413738 \cdot 10^{16}$ Hz
Energy	$E_\mathrm{h} =$	$\dfrac{m_\mathrm{e}e^4}{(4\pi\varepsilon_0)^2\hbar^2} =$	$4.35974417 \cdot 10^{-18}$ J $= 27.211$ eV
Electric field	$\mathscr{E}_0 =$	$\dfrac{e}{4\pi\varepsilon_0 a_0^2} =$	$5.14220651 \cdot 10^{11}$ V/m
Intensity		$\dfrac{1}{2}\varepsilon_0 c_0 \mathscr{E}_0^{\,2} =$	$3.5094452 \cdot 10^{16}$ W/cm^2

C Code

In this appendix, an overview of the code used in the calculations is given. In the following subsections, it is described how the different building blocks are implemented.

When the program is started, a specified file in the JSON format (see [Crockford 2002]), containing the parameters for the calculations, is loaded. All parameters specified in SI units are converted to atomic units, as per appendix A. Then the calculations commence. They consist of propagating electrons of different positions and velocities according to eq. (3.2). For all electrons four quantities are stored: time of ionization, final speed (length of momentum vector), final position and final velocity. The individual trajectories of the electrons are in general discarded; a single trajectory could easily occupy 32 kB (two Cartesian vectors, one for position and one for velocity, consisting of three components plus one padding,[10] every component requires 4 B of storage and a trajectory may require up to a thousand steps), which means that 32 GB of storage would be needed to store the trajectories of a million electrons.

[10] To make use of optimized SIMD methods, data fetches has to be in sizes that are powers of two, *i.e.* 2^n. Therefore, the three Cartesian components are padded with one, meaningless, extra component.

Also worth noting is that single-precision floating-point values (named `floats` in C/C++) are used (hence 4 B per component) instead of double-precision floating-point values (`doubles`), because the GPU only has hardware support for `floats`. The execution times for calculations with `doubles` are drastically increased (by more than one order, [Göddeke et al. 2005]).

C.1 Building blocks

C.1.1 Sampling

The basis for successful sampling of probability density functions is a reliable random number generator as mentioned in section 2.3. Each executing thread is assigned its own generator with a unique multiplier (see [Alerstam et al. 2009] for details on the implementation).

Position The initial position is decided by three coordinates: the radius r and the two angles $\vartheta \in [0, \pi)$ and $\phi \in [0, 2\pi)$. The two angles are sampled isotropically, while the radius is sampled from the radial function of the s state, such that $\int_0^\infty \mathrm{d}r'\, r'^2 |R_{1,0}(r')|^2 = 1$. The inverse transformation method is used, *i.e.* a stochastic variable U is sampled uniformly on the interval, and then equation (3.7) is solved for r with Newton–Raphson's method:

```
inline void NewtonFind(float (*f)(float, float&),
                       float& x, float y,
                       float& maxError,
                       float maxI=30){
   float error=INFINITY, derivative=0.0f;
   int i=0;
```

```
    while(abs(error)>maxError && i<maxI){
        error=f(x,derivative)-y; //Calculate distance
              //between current point (f(x)) and wanted
              //value (y). The value of f'(x) is
              //retrieved, since derivative is passed
              //as a reference, not by value.
        x=x-error/derivative; //Refine guess
        i++;
    }
    maxError = error;
}

float sRadialFunction(float r, float& derivative){
  int Z = deviceStreakingParams.Z;
  float exp2Zr = exp(-2*Z*r);
  float r2 = r*r;
  derivative = 4*Z*Z*Z*r2*exp2Zr;
  return 1-(2*Z*(Z*r2+r)+1)*exp2Zr;
}

float sampleRadius(SimData& data, int tid){
  float r=1.0f; //Initial guess (=a_0)
  //Maximum absolute error accepted
  float error = 1e-10f;

  NewtonFind(&sRadialFunction, //Pointer to function
             r,
             rand_MWC_co(data.rng, tid), //Sampled U
             error);
  return r;
}
```

Newton–Raphson's method is an iterative method for finding a root of an equation. One first has to make an initial guess of the root, and a new root is then taken as

$$x^{(1)} = x^{(0)} - \frac{f(x^{(0)})}{f'(x^{(0)})}.$$

This procedure is repeated until the error is sufficiently small.

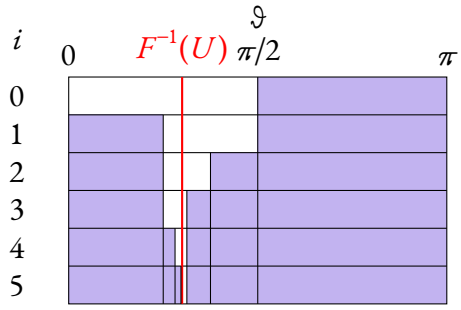


Figure C.1: *An illustration of the binary search algorithm. The initial guess is placed in the middle of the domain. The algorithm then tests if the sought value is in the left or right subdomain. The next guess is then chosen as the middle of that subdomain and the procedure is repeated.*

Velocity Newtons–Raphson's method has quadratic convergence, as long as the derivative $f'(x) \neq 0$, it is Lipschitz continuous around the root and the initial guess is sufficiently close to the root (see [Bonnans et al. 2003]). It can not, therefore, be used when sampling the p distribution

$$Y_1^0(\vartheta, \phi) = \sqrt{\frac{3}{4\pi}} \cos\vartheta,$$

as the derivative of its CDF

$$\begin{aligned} F(\vartheta) &= \int_0^{2\pi} \mathrm{d}\phi \int_0^{\vartheta} \mathrm{d}\vartheta' \, |Y_1^0(\vartheta', \phi)|^2 \sin\vartheta' \\ &= 2\pi \frac{3}{4\pi} \left[-\frac{\cos^3\vartheta'}{3} \right]_0^{\vartheta} \\ &= \frac{1}{2}(1 - \cos^3\vartheta) \end{aligned}$$

has a root in the interval $(0, \pi)$. This is instead sampled using a simple *binary search algorithm*, where the root is found by first guessing that $\vartheta = \pi/2$ and then stepping in the direction that minimizes the error $|F(\vartheta) - U|$, while simultaneously halving the step length, until the error is small enough. See figure C.1 for an illustration.

```
inline void SearchFind(float (*f)(float, float&),
                       float& x, float y,
                       float& maxError,
                       float maxI=40){
   float derivative=0.0f;
   float step=0.5*x;
   x=step;
   float error=f(x,derivative)-y;
   int i=0;
   while(abs(error)>maxError && i<maxI){
      step *= 0.5;
      x -= step*(error>0?1:-1);
      error = f(x,derivative)-y;
      i++;
   }
   maxError = error;
}

float SphericalHarmonic1_0(float theta,
                           float& derivative){
   float cosTheta = cos(theta);
   //Cannot use Newton-Raphson anyway
   derivative = 0.0f;
   return 0.5f*(1-cosTheta*cosTheta*cosTheta);
}

float sampleTheta(SimData& data, int tid){
   float theta=PI;
   float error = 1e-10f;
   theta=PI;
   SearchFind(&SphericalHarmonic1_0,
              theta, rand_MWC_co(data.rng, tid),
              error);
   return theta;
}
```

Sampling Y_1^0 as above yields the initial direction of the velocity, but an initial speed, *i.e.* length of the velocity vector, is needed as well. This is sampled from the spectral distribution (which is Gaußian) of the XUV pulse, using the Box–Müller transform:

If U_1 and U_2 are independent, stochastic variables uniformly distributed on $(0, 1]$, then

$$Z_1 = \sigma\sqrt{-2\ln U_1}\cos(2\pi U_2) + \mathbb{E} \qquad Z_2 = \sigma\sqrt{-2\ln U_1}\sin(2\pi U_2) + \mathbb{E}$$

are independent, stochastic variables normally distributed around $\mathbb{E}$ with a standard deviation of σ.

Only one value is needed to sample the energy:

```
float sampleE(SimData& data, int tid){
   //Box-Müller normal distribution
   float E=sqrt(-2*log(rand_MWC_oc(data.rng, tid)))*
      cos(2*PI*rand_MWC_co(data.rng, tid));

   E *= deviceStreakingParams.XUV.stdDevE;
   E += deviceStreakingParams.XUV.centerE;

   return E;
}
```

C.1.2 Propagation

The propagation is done by a RK45 ODE solver, in an implementation adapted and generalized from [Palmer 2003]. The code makes heavy use of optimization techniques such as those mentioned in section 2 and *templating*[11] to run as efficiently as possible on a GPU.

[11] A programming language construct used to offload the processor by deciding parts of the execution path at compile time.

C.1.3 Detection

The detection process works like this: a mask vector of *booleans*,[12] as long as the number of electrons used in the calculations, is allocated. A kernel that is executed for each and every electron, stores in the detection vector a `true` if the electron has a momentum vector inside a cone (1D) or if the electron is spatially located inside a slice of thickness Δy (2D), `false` otherwise. The number of elements with the value `true` is then counted, and new, shorter, vectors are allocated, to store the ionization times, final momenta and final positions of the detected electrons.

[12] An integral value type, that can only take two values: 0, corresponding to `false` and 1, corresponding to `true`.

C.1.4 Least Squares fitting

The idea of *Least Squares* fitting is to minimize the squared residuals between some measurement data and a model by finding the optimum parameters for that model. This can also be viewed as minimizing the distance between the measurement vector and its projection on the subspace spanned by the parameters. The solution to such a system is $\widehat{\boldsymbol{\beta}} = (\mathrm{X}^\top \mathrm{X})^{-1}\mathrm{X}^\top \mathrm{y}$, where $(\mathrm{X}^\top \mathrm{X})^{-1}$ is called the pseudo-inverse of X, the measurement positions and y is the measurement values at those positions. However, if the model is non-linear, *i.e.* not on the form $\mathrm{y} = \mathrm{X}\boldsymbol{\beta}$, a non-linear least squares fit has to made where the model is linearized around some $\boldsymbol{\beta}$ and a step $\Delta\boldsymbol{\beta}$ is taken until the gradient

$$\nabla S = \nabla \sum_i^N \left(y_i - y\left(x_i, \boldsymbol{\beta}\right)\right)^2 = \nabla \sum_i^N \Delta y_i^2$$

is zero. The step is found by $\Delta\boldsymbol{\beta} = (\mathrm{J}^\top \mathrm{J})^{-1}\mathrm{J}^\top \Delta\mathrm{y}$, where $\mathrm{J} = \left(J_{ij}\right)$ is the Jacobian of the model y.

In the case of the streaking spectrograms, the parameter vector $\boldsymbol{\beta} = \{t_{\mathrm{CTMC}}\ \alpha\}$ (the time shift and the amplitude modulation), and the model is equation (3.6). At each step, the Jacobian is calculated:

$$\mathrm{J}^{(k+1)} = \left(\frac{\partial \mathrm{p}\left(\mathrm{t}, t_{\mathrm{CTMC}}^{(k)}, \alpha^{(k)}\right)}{\partial t_{\mathrm{CTMC}}^{(k)}} \quad \frac{\partial \mathrm{p}\left(\mathrm{t}, t_{\mathrm{CTMC}}^{(k)}, \alpha^{(k)}\right)}{\partial \alpha^{(k)}} \right),$$

where t is a vector containing the ionization times. The elements of $\mathrm{J}^{(k+1)}$ are calculated on the GPU, and then $\mathrm{J}^{(k+1)\top}\mathrm{J}^{(k+1)}$ and $\mathrm{J}^\top \Delta\mathrm{p}$ are calculated using parallel reduction as described in section 2. Lastly, the parameter step is calculated by solving

$$\mathrm{J}^{(k+1)\top}\mathrm{J}^{(k+1)}\Delta\boldsymbol{\beta}^{(k+1)} = \mathrm{J}^{(k+1)\top}\Delta\mathrm{p}^{(k)}$$

using LU decomposition.

E Errors

HERE FOLLOWS A SUMMARY of errors made in the calculations. They are all related to how the initial positions and momenta of the electrons are sampled.

- Radial functions for heavier elements than hydrogen still sampled with $Z = 1$ as the program was first implemented for hydrogen. When testing with He^+ this was forgotten, and as the results seemed plausible (compare with figure 3.6), it was not discovered until fairly late in the project. When testing with correct wavefunction, the calculation times increased immensely, and as the error did not seem to have large impact, it was decided to leave it as is.

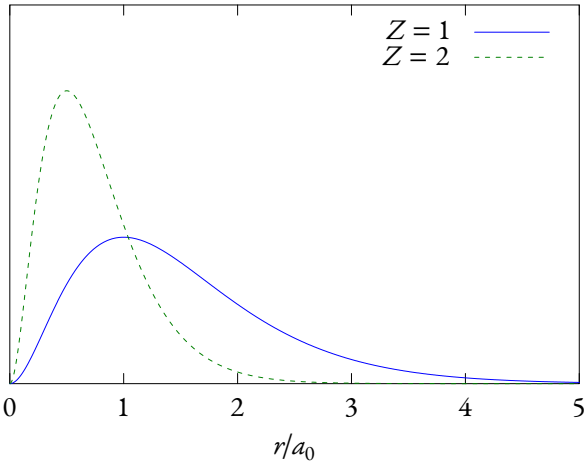


Figure E.1: *The radial wavefunctions for hydrogen-like systems with a nucleus of Z=1 and 2. As is natural, the electrons are bound more tightly to a larger nucleus; this also causes trouble for the calculations, since the electron trajectories more often start in a stiff region* $\Longrightarrow$ *longer calculation times.*

- Initial velocity is sampled isotropically (as in [Nagele et al. 2011]), instead of according to a p distribution. However, this is only done in the one-dimensional case. Sampling according to a p distribution seems to yield larger variance in t_{CTMC}.
- Initial energy sampled from a pulse of too small spectral width, which cannot really be said to be as short as needed for the approximation that ionization is instantaneous upon the arrival of the XUV photon. Sampling from the correct pulse spectrum yielded too large variance in t_{CTMC}.

It is likely that by sampling from Kepler orbits as is traditionally done in CTMC calculations, these errors could have been avoided.

F.1 VMIS

F.1.1 Electron scattering

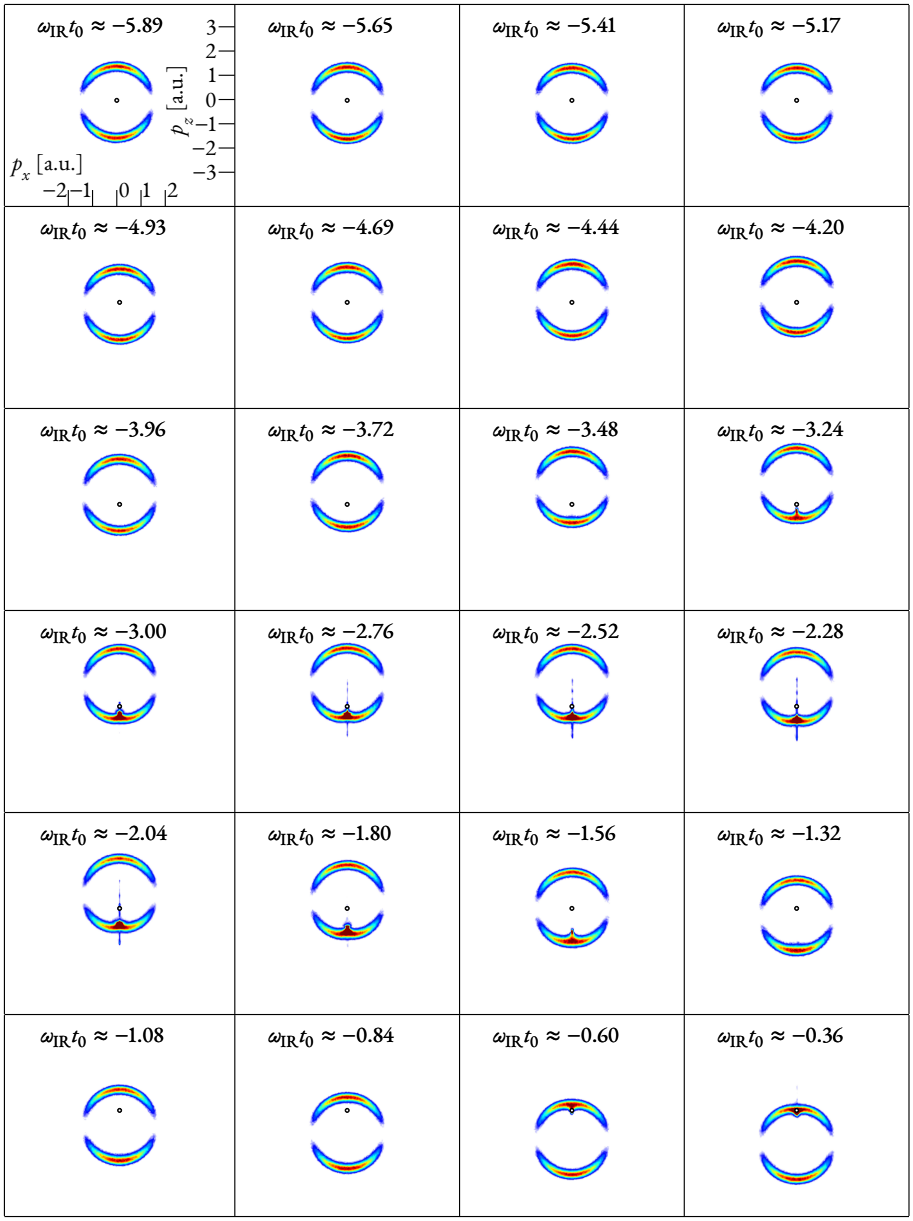


Figure F.1: *VMIS images of the electron momentum distribution when ionizing in a strong IR field ($\tilde{\gamma} < 1$) for different ionization times t_0.*

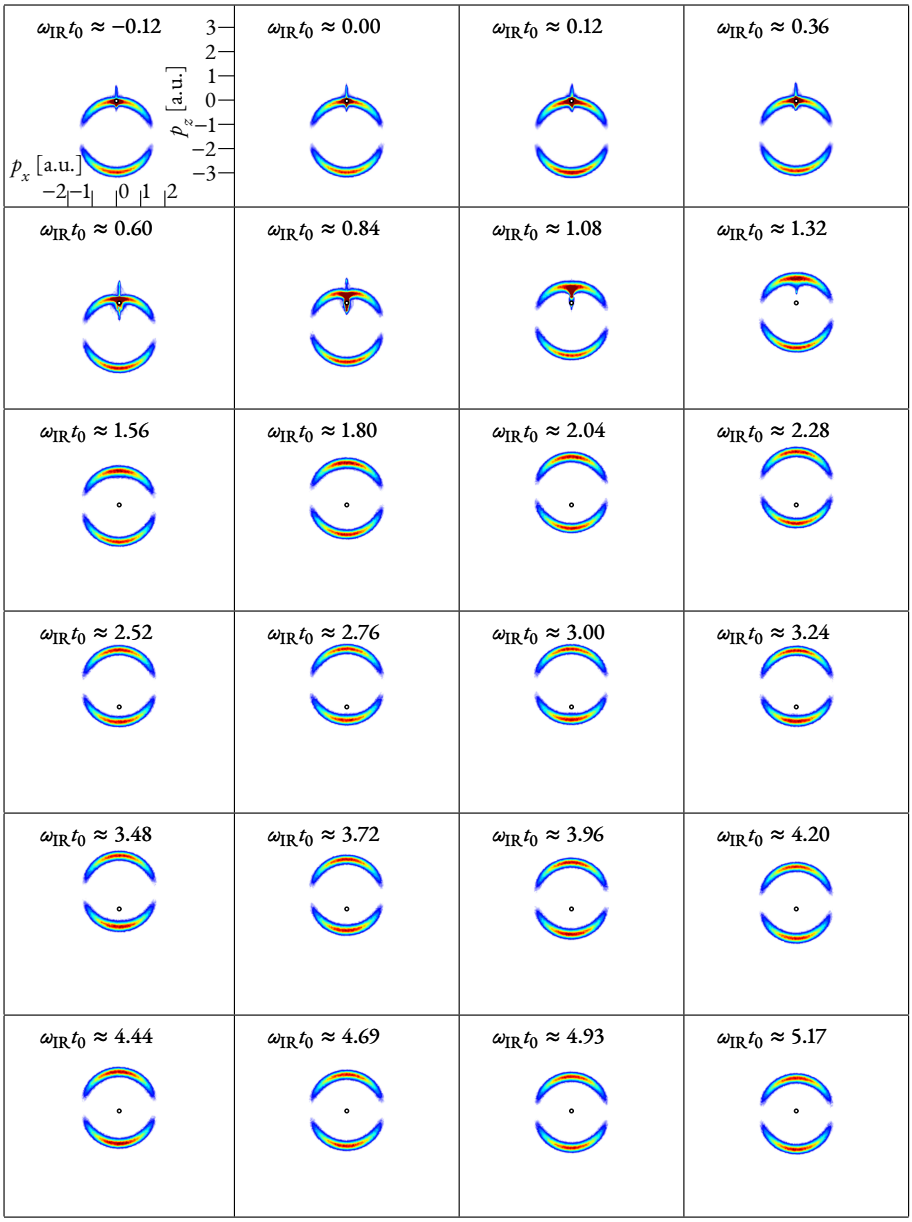


Figure F.2: *Continuation of the series from the previous figure.*

F.1.2 Polarization dependence

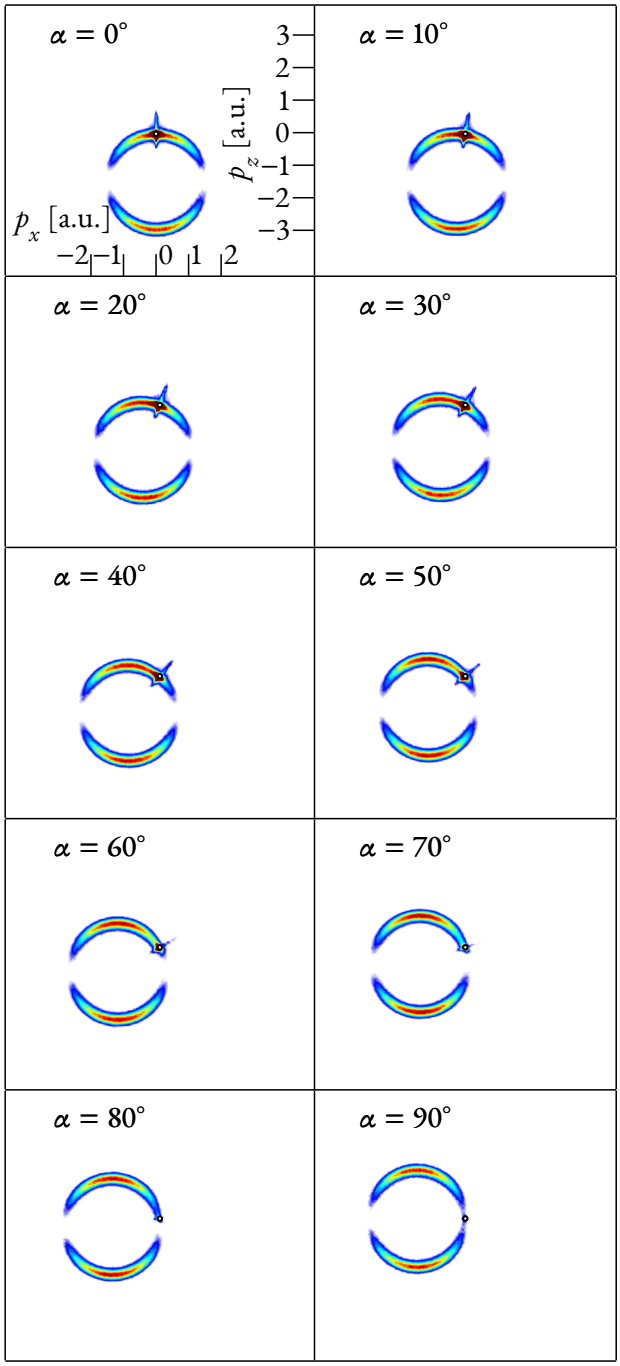


Figure F.3: *VMIS images of the electron momentum distribution for different IR field polarizations.*

R Rederivation of time shift approximation

In this appendix, a rederivation of the analytical approximation of the measurement induced time delay, as derived by [Ivanov and Smirnova 2011], is done, but for a Coulomb potential of $V(r) = Z/r$. The derivation is very brief, with little more explanation than necessary, since it is only provided for completeness. The equation numbering follows that of the original article.

The traditional streak camera formula

$$\mathrm{v}_f(t_i) \approx \mathrm{v}_0 - \mathrm{A}(t_i) \tag{R.1}$$

is only an approximation. A more accurate expression is

$$\mathrm{v}_f(t_i) = \mathrm{v}_0 - \mathrm{A}(t_i) + \Delta\mathrm{v}(t_i),$$

with the additional velocity shift

$$\Delta\mathrm{v}(t_i) = \int_{t_i}^{\infty} \mathrm{d}t \,\{\mathrm{F}_C[\mathrm{r}_{\mathrm{IR}}(t)] - \mathrm{F}_C[\mathrm{r}_{\mathrm{FF}}(t)]\}. \tag{R.2}$$

In the eikonal-Volkov approximation, a single characteristic trajectory is obtained:

$$\mathrm{r}_{\mathrm{IR}}(t) \approx \underbrace{\mathrm{r}_0 + \mathrm{v}_0(t - t_i)}_{\equiv \mathrm{r}_{\mathrm{FF}}(t)} + \underbrace{\int_{t_i}^{t} \mathrm{d}t' \,[\mathrm{A}(t') - \mathrm{A}(t_i)]}_{\equiv \Delta\mathrm{r}(t,t_i)} . \tag{R.3}$$

r_0 is denoted $r_0 = 1/a v_0$. a is found by matching the free particle wave function of the eikonal approximation with the exact outgoing Coulomb wave:

$$\begin{aligned} \psi_{\mathrm{v}_0} \propto \exp\left[i v_0 r + \frac{i}{v_0}\int_{r_0}^{r} \mathrm{d}r' \,\frac{Z}{r'}\right] &= \exp\left[i v_0 r + i\frac{Z}{v_0}\ln\left(\frac{r}{r_0}\right)\right] \\ &\equiv \exp\left[i v_0 r + i\frac{Z}{v_0}\ln(2v_0 r) - i2\delta\right], \\ \delta &= \arg\left[\Gamma(\ell + 1 - iZ/v_0)\right] \end{aligned} \tag{R.4}$$

$$\begin{aligned} &\iff i\frac{Z}{v_0}\ln\left(\frac{r}{r_0}\right) = i\frac{Z}{v_0}\ln(2v_0 r) - i2\delta \\ &\iff \frac{r}{r_0} = 2v_0 r \exp(-2v_0\delta/Z) \\ &\iff a(v_0) = 2\exp(-2v_0\delta/Z). \end{aligned} \tag{R.5}$$

(The exact outgoing Coulomb wave was found by backpropagating the plane wavefront at the detector (positioned at $+\infty$). The phase difference near the atom between

this wave and a plane wave without the scattering potential present was found to be -2δ. This may not be seen as a rigorous derivation, but nonetheless this definition of r_0 proved to yield good results.)

With an electric field of $E_{\mathrm{IR}} = E_0 \cos(\omega_{\mathrm{IR}} t)$ and accompanying magnetic vector potential of $A_{\mathrm{IR}}(t) = -(E_0/\omega_{\mathrm{IR}}) \sin(\omega_{\mathrm{IR}} t)$, the trajectory in (R.3) becomes

$$r_{\mathrm{IR}}(t) \approx r_0 + v_0(t - t_i) + \frac{E_0}{\omega_{\mathrm{IR}}^2} \left[\sin(\omega_{\mathrm{IR}} t_i)\omega_{\mathrm{IR}}(t - t_i) + \cos(\omega_{\mathrm{IR}} t) - \cos(\omega_{\mathrm{IR}} t_i)\right]. \quad \text{(R.6)}$$

With a weak IR field ($\Delta r(t, t_i) \ll r_{\mathrm{FF}}(t) \implies v_F \equiv E_0/\omega_{\mathrm{IR}} \ll v_0$), (R.8) becomes

$$\left.\begin{aligned} \Delta v(t_i) &= \frac{2}{v_0^3} \int_{t_i}^{\infty} \mathrm{d}t \, \frac{Z \Delta r(t, t_i)}{\left[r_0 + v_0(t - t_i)\right]^3} \\ \Delta r(t, t_i) &= \frac{E_0}{\omega_{\mathrm{IR}}^2} \left[\sin(\omega_{\mathrm{IR}} t_i)\omega_{\mathrm{IR}}(t - t_i) + \cos(\omega_{\mathrm{IR}} t) - \cos(\omega_{\mathrm{IR}} t_i)\right] \end{aligned}\right\} \quad \text{(R.7)}$$

$$\implies \Delta v(t_i) = -Z \frac{E_0}{v_0^3} \left[\cos(\omega_{\mathrm{IR}} t_i) f_c\left(\frac{r_0 \omega_{\mathrm{IR}}}{v_0}\right) - \sin(\omega_{\mathrm{IR}} t_i) f_s\left(\frac{r_0 \omega_{\mathrm{IR}}}{v_0}\right)\right] \quad \text{(R.8)}$$

with

$$\left.\begin{aligned} f_c(x) = c(x)\cos(x) + s(x)\sin(x), \quad f_s(x) = s(x)\cos(x) - c(x)\sin(x) \\ c(x) = \int_x^{\infty} \mathrm{d}x' \, \frac{\cos x'}{x'}, \quad s(x) = \int_x^{\infty} \mathrm{d}x' \, \frac{\sin x'}{x'}. \end{aligned}\right\} \quad \text{(R.9)}$$

$x = r_0 \omega_{\mathrm{IR}} \ll 1 \implies$

$$\left.\begin{aligned} f_c\left(\frac{r_0 \omega_{\mathrm{IR}}}{v_0}\right) &\approx \ln\left(\frac{v_0}{\omega_{\mathrm{IR}} r_0}\right) - \gamma_{\mathrm{Euler}} + \frac{\pi}{2} \frac{r_0 \omega_{\mathrm{IR}}}{v_0} \\ f_s\left(\frac{r_0 \omega_{\mathrm{IR}}}{v_0}\right) &\approx \frac{\pi}{2} - \frac{r_0 \omega_{\mathrm{IR}}}{v_0} \left[\ln\left(\frac{v_0}{\omega_{\mathrm{IR}} r_0}\right) - \gamma_{\mathrm{Euler}}\right] \end{aligned}\right\} \quad \text{(R.10)}$$

Substituting (R.8) into the expression for the final energy of the electron yields

$$v_f = v_0 - A(t_i)\left[1 + \frac{Z \omega_{\mathrm{IR}}}{v_0^3} f_s\right] - E(t_i) \frac{Z}{v_0^3} f_c, \quad \text{(R.11)}$$

which can be rewritten as

$$v_f = v_0 - \alpha A\left(t_i + \Delta t_{\mathrm{IR}}^{(\mathrm{LR})}\right) \quad \text{(R.12)}$$

with

$$\begin{aligned}
\Delta t_{\mathrm{IR}}^{(\mathrm{LR})} &= -\frac{Z}{v_0^3} f_c \frac{1}{1+\omega_{\mathrm{IR}}\dfrac{Z}{v_0^3} f_s} \\
&\approx -\frac{Z}{v_0^3}\left[\ln\left(\frac{av_0^2}{\omega_{\mathrm{IR}}}\right) - \gamma_{\mathrm{Euler}} + \frac{\pi}{2}\frac{\omega_{\mathrm{IR}}}{av_0^2}\right] \frac{1}{1+\omega_{\mathrm{IR}}\dfrac{Z}{v_0^3}\left\{\dfrac{\pi}{2} - \dfrac{\omega_{\mathrm{IR}}}{av_0^2}\left[\ln\left(\dfrac{av_0^2}{\omega_{\mathrm{IR}}}\right) - \gamma_{\mathrm{Euler}}\right]\right\}} \\
&= -\left[\ln\left(\frac{av_0^2}{\omega_{\mathrm{IR}}}\right) - \gamma_{\mathrm{Euler}} + \frac{\pi}{2}\frac{\omega_{\mathrm{IR}}}{av_0^2}\right] \frac{Z}{v_0^3 + \omega_{\mathrm{IR}} Z \underbrace{\left\{\dfrac{\pi}{2} - \dfrac{\omega_{\mathrm{IR}}}{av_0^2}\left[\ln\left(\dfrac{av_0^2}{\omega_{\mathrm{IR}}}\right) - \gamma_{\mathrm{Euler}}\right]\right\}}_{\approx \frac{\pi}{2},\ \frac{\omega_{\mathrm{IR}}}{av_0^2} \ll 1}} \\
&\approx -\left[\ln\left(\frac{av_0^2}{\omega_{\mathrm{IR}}}\right) - \gamma_{\mathrm{Euler}} + \frac{\pi}{2}\frac{\omega_{\mathrm{IR}}}{av_0^2}\right] \frac{Z}{v_0^3 + \omega_{\mathrm{IR}} Z\pi/2}.
\end{aligned} \tag{R.13}$$

Thanks

I would like to thank the following people, for being of help during my project:

Johan Mauritsson for excellent supervision and giving me the opportunity to do this project which fit me perfectly; my assistant supervisor Erik Alerstam for helping me when my mind was boggled by the GPU, and always being positive and encouraging. Marcus Dahlström was the first to help me understand the subject thoroughly and get an idea what I was supposed to do. I have also had a lot of help from Diego Guénot, answering my multitude of questions.

Rasmus Henningsson gave me help early on in the mathematical/programming phase of the project (I am especially thankful for his finding those two missing bytes).

When I visited Aarhus University in Denmark, I met Adam Etches. Thank you for listening intently, giving advice and your hospitality.

I had a very interesting and enlightening discussion with professor Misha Ivanov of Imperial College London, regarding the extension of the time delay formula in the paper recently published by him and Olga Smirnova; thank you for your friendliness!

These researchers have also helped me with different aspect of the project: Professor Claus Führer, Mathieu Gisselbrecht and Tomas Persson. Furthermore, Johannes Feist kindly provided numerical reference data.

I would also like to thank family and friends for always being supportive; from the latter group I would especially want to thank Peter Fransson and Dorothea Scheunemann. Και Δόξα τω Θεώ.