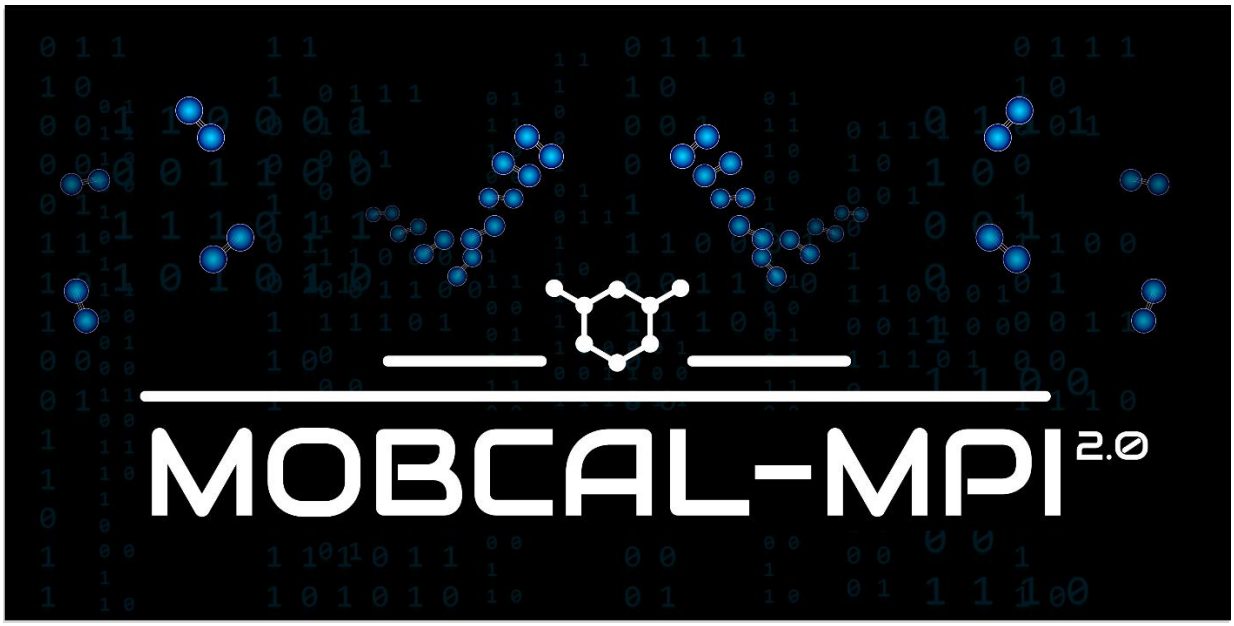
HOPKINS GROUP – UNIVERSITY OF WATERLOO

MobCal-MPI 2.0.2 User Manual



Revised January 2024

Table of Contents

1. Introduction

Welcome and thank you for choosing MobCal-MPI to enhance your ion mobility experiments. This guide offers a concise overview of the trajectory method and two-temperature theory employed by MobCal-MPI for calculating ion-neutral collision cross sections (CCSs) and ion mobilities across various field strengths. Additionally, it introduces the functionalities of the accompanying Graphical User Interface (GUI) for input file generation and post-calculation analysis.

MobCal-MPI 2.0 employs the trajectory method, as initially presented by Shvartsburg, Jarrold, et al., in their publications:

- M. F. Mesleh, J. M. Hunter, A. A. Shvartsburg, G. C. Schatz, and M. F. Jarrold, Structural Information from Ion Mobility Measurements: Effects of the Long Range Potential, *J. Phys. Chem.* **1996**, 100, 16082-16086; Erratum, *J. Phys. Chem. A* **1997**, 101, 968.

- A. Shvartsburg and M. F. Jarrold, An Exact Hard Spheres Scattering Model for the Mobilities of Polyatomic Ions, *Chem. Phys. Lett.* **1996**, 261, 86-91.

MobCal-MPI enhances this method by parallelizing gas trajectory calculations with atom-specific van der Waals (vdW) parameters, as summarized in Section 2. Please note that Section 2 is not intended to be a literature review of CCS calculations, but rather provides readers with a refresher on how the trajectory method is implemented in MobCal-MPI. For more in-depth reading, we recommend referring to publications cited in this section and those provided in the additional reading recommendations (Section 2.9).

Section 3 outlines the installation process for the MobCal-MPI Suite GUI, which is used for generating MobCal-MPI input files (.mfj) and processing data from completed MobCal-MPI output files (.mout). Section 4 details the compilation procedure for the MobCal-MPI Fortran code, provides instructions on conducting CCS calculations using MobCal-MPI, and describes the information is printed to the MobCal-MPI output file.

This manual also contains several appendices to supplement users with helpful information. Notably, Appendix A is a guide on how to calculate the CCS/mobility of a target compound starting solely from its identity. This workflow employs CREST, which is interfaced with the xTB package to generate conformers of the target analyte. These candidate structures are subsequently refined using quantum-chemical calculations via the ORCA suite of programs. The outputs of quantum-chemical calculations, which are conducted using density functional theory, can then be converted to MobCal-MPI input files.

If you incorporate MobCal-MPI into your research, we request that you cite the relevant papers listed in Appendix B. Additionally, Appendix C contains accompanying figures related to the content in Section 2, with details concerning the optimization of vdW parameters for CCS calculating in He found within Appendix D. All references cited in this manual are located in Appendix E.

1.1 Terms of use

MobCal-MPI is freely accessible to academic users. Using MobCal-MPI for commercial purposes, whether it leads to profit or not, without obtaining prior written consent from the authors, is considered a breach of the terms of use and may result in legal action against the offender(s).

1.2 Change log

V2.0.1

- *Updated calls to MPI from "mpif.h" to "use mpi", which resolves rank mismatching errors when MobCal-MPI is compiled using GCC compilers with the -Og optimization flag.*

V2.0

Version 2.0 introduces significant changes to the framework in which ion-neutral collision trajectories are sampled, enabling the rapid evaluation of CCS and mobility at various electric field strengths. These modifications are summarized below and elaborated upon in Section 2.

MobCal-MPI Fortran code changes:

- *Implementation of two-temperature theory (2TT) within the MobCal-MPI framework.*
- *Implementation of an empirical correction to 2TT, thus correcting systematic deviations between calculated and experimental high-field mobilities.*
- *Updated the methodology to evaluate uncertainty in the calculated CCS.*
- *Altered the sampling grid for ion-neutral relative velocities in collision dynamics simulations, transitioning from a weighted spacing to a linear spacing.*
- *Resolved a bug that caused N_2 rotational orientation weights to remain fixed at 500 K instead of being set to the user-defined bath gas temperature.*

The integration of 2TT into MobCal-MPI 2.0 has led to improvements to the graphical user interface (GUI). These improvements enable the creation of input files that activate the 2TT module and streamline the analysis of field-dependent mobility data:

mfj creator (see Section 3.4.1) changes

- *Updated to accommodate the implementation of 2TT within MobCal-MPI 2.0. The .mfj creator works with outputs from Gaussian (.log files) and ORCA (.out files; see 3.4.1 mfj Creator).*

Boltzmann weighter and CCS extractor changes

- *Removed and replaced with the single and multiple .mout analyzers.*

Analyze single .mout analyzer (see Section 3.4.2) changes

- *Plots and exports CCS integrands (single-temp) or CCS evolution with T_{eff} (multi-temp)*
- *Plots and exports momentum transfer integrals (single and multi-temp)*
- *Calculates alpha coefficients and corresponding alpha functions from field-dependent mobility data (multi-temp)*
- *Exports comprehensive summaries of CCS and mobility, including their evolution with field strength.*

Analyze multiple .mout (see Section 3.4.3) changes

- *Exports field-dependent CCS and mobility data from multiple .mout files (single and multi-temp calculations).*

2. Methods within MobCal-MPI

In this section, we present an overview of the trajectory method as implemented in MobCal-MPI. Each subsection draws from the publications that correspond to MobCal-MPI, and are included here in adapted forms with permission from the Royal Society of Chemistry:

1. The original MobCal-MPI publication:

   Ieritano, C.; Crouse, J.; Campbell, J. L.; Hopkins, W. S. A Parallelized Molecular Collision Cross Section Package with Optimized Accuracy and Efficiency. *Analyst* **2019**, *144* (5), 1660–1670. https://doi.org/10.1039/c8an02150c.

2. The publication corresponding to MobCal-MPI 2.0:

   Haack, A.,* Ieritano, C.,* Hopkins, W. S. MobCal-MPI 2.0: An Accurate and Parallelized Package for Calculating Field-Dependent Collision Cross Sections and Ion Mobilities. *Analyst.* **2023**, 148, 3257 – 3273. *Equal contribution. https://doi.org/10.1039/d3an00545c

3. A subsection of a review article that pertains to field-dependent mobility calculations:

   Ieritano, C., & Hopkins, W. S. (2022). The hitchhiker's guide to dynamic ion-solvent clustering: applications in differential ion mobility spectrometry. *Physical Chemistry Chemical Physics*, *24*(35), 20594–20615. https://doi.org/10.1039/d2cp02540j

We note that the information provided in Section 2 is not exhaustive by any means. Should users desire a more in-depth exploration of ion mobility calculations, Section 2.9 provides additional reading recommendations.

2.1 Ion Mobility Theory

The use of ion mobility spectrometry (IMS) as a standalone technique or when coupled to mass spectrometry (MS) continues to gain traction within the analytical and biophysical communities. This trend stems from the ability of IMS to separate analytes and probe their geometric structure. Several groups have shown that IMS-MS, which encompasses several variants,[1–3] can solve challenging analytical problems as either the sole separation dimension or when coupled to liquid chromatography (LC).[4,5] Each IMS technique uses electric fields to accelerate ions through the mobility region, with instrument variations being predominantly associated with the nature of the field (*i.e.,* oscillating or static) and its magnitude. When analytes enter the mobility region, which is filled with an inert gas (typically N_2), ion-neutral collisions create drag, countering field-induced acceleration. These opposing effects determines the velocity with which the analyte travels through the mobility cell and ultimately facilitates analyte separation. For each unique analyte, these interactions lead to a constant drift velocity for the analyte's ensemble (v_D) that is proportional to the applied electric field (*E;* **Eq. 1**):

$$v_D = KE \qquad \textbf{Eq. 1}$$

The proportionality factor, K, is colloquially known as the ion mobility coefficient, although it is far more common for practitioners to report the reduced mobility coefficient (K_0; **Eq. 2**),

$$K_0 = K\left(\frac{N}{N_0}\right) \qquad \textbf{Eq. 2}$$

where N is the particle density (molecules m^{-3}) within the IMS device, and N_0 is the particle density at 273.15 K and 1 atm (2.6868×10^{25} molecules m^{-3}), which is also called the Loschmidt constant. Rewriting **Eq. 1** using the reduced mobility coefficient yields **Eq. 3**, showing definitively that collision dynamics are influenced by the field strength and particle density, as represented by the reduced field strength term E/N. The E/N term is typically expressed in Townsend (Td), where 1 Td = 10^{-21} V m^2. Simply put, **Eq. 3** indicates that increasing the field strength induces greater acceleration of the ion, whereas increasing the particle density (*i.e.*, the pressure) increases the collision frequency such that the time for acceleration becomes shorter. Ultimately, this interdependency indicates that the reduced field strength is directly proportional to the mean collision energy of any ion-neutral collision.[6]

$$v_D = K_0 N_0 \frac{E}{N} \qquad \textbf{Eq. 3}$$

Within the low-field regime, an ion's velocity responds linearly to changes in the reduced field strength, and thus, enables relation of an ion's low-field mobility coefficient, $K(0)$, to its collision cross section (CCS) via the Mason-Schamp equation (**Eq. 4**),[6] whose derivation dates back to the early work on ion mobility by Langevin.[7]

$$K(0) = \frac{3}{16}\left(\frac{2\pi}{\mu k_B T}\right)^{1/2} \frac{q}{N\Omega(T)} \qquad \textbf{Eq. 4}$$

Here, μ is the reduced mass of the ion-neutral pair, q is the absolute charge of the ion, k_B is the Boltzmann constant, T is the absolute temperature of the bath gas, and Ω is the ion-neutral CCS, which is temperature dependent. The CCS represents the orientally-averaged collision area of the analyte with the buffer gas that fills the mobility cell, and as such, is intrinsically related to the 3D structure of the ion. The ability of IMS to discern molecular geometric structure from an ion's CCS depends on two crucial factors: 1) the precise measurement of ion mobility under strictly controlled experimental conditions (*i.e.,* temperature and pressure within the mobility region),[8,9] and 2) the meticulous modeling of CCS/ion mobility from the computed geometry of the analyte. Owing to the significant role CCSs play in chemical separations, modelling CCS has become an integral component of IMS research. Calculating an ion's CCS *in silico* often mitigates ambiguity in IMS-based structural assignments and provides valuable insight into the nature of the IMS-based separation mechanism. When modeling ion mobility, practitioners are primarily concerned with capturing the dynamics of a collision event between an ion and a gas particle, as collisions strongly depend on the electric field, pressure, and temperature. To express this relationship, the ion mobility is adjusted by the alpha function, which is usually expressed as a Taylor series composed of even-ordered alpha coefficients (**Eq. 5** and **Eq. 6**, respectively).[10]

$$K\left(\frac{E}{N}\right) = K(0)\left[1 + \alpha\left(\frac{E}{N}\right)\right] \qquad \textbf{Eq. 5}$$

$$\alpha\left(\frac{E}{N}\right) = \alpha_2\left(\frac{E}{N}\right)^2 + \alpha_4\left(\frac{E}{N}\right)^4 + \alpha_6\left(\frac{E}{N}\right)^6 + \cdots \quad \textbf{Eq. 6}$$

The alpha function is approximately zero if the reduced field strength is within the so-called low-field limit (*ca.* 0 – 10 Td), which is why low-field IMS techniques such as drift-tube IMS (DTIMS) can be used as a tool for both structural elucidation and analytical separations. Structural elucidation is typically performed within the low-field limit, where an ion's velocity distribution is characterized by a Maxwell-Boltzmann distribution at the temperature of the bath gas (T), and hence, renders the Mason-Schamp relation valid. While advances in instrumentation have enabled high-throughput measurements of CCS on many low-field IMS platforms,[11–15] development of computational frameworks for relating an ion's structure to its CCS has progressed at a much slower pace. Several methods (and software that implements these methods) have been published over the years to calculate CCSs, a summary of which is available in reference [16] and in Section C1. However, most of the common CCS packages are limited to calculating thermal CCSs and/or low-field mobilities, and unfortunately are falling behind the surge in popularity of IMS techniques that operate at high field strengths.[17,18] Both MobCal-MPI[19] and IMoS[20,21] are notable exceptions, as they have been modified to allow calculations of ion mobility at arbitrary field strengths. The calculation of mobilities above the low-field limit is based on the work of Kihara,[22] Mason and Schamp,[23] and Viehland and Mason,[24,25] the latter of whom devised two-temperature theory (2TT) to account for the non-negligible acceleration of ions in electric fields that surpass the low-field limit. Viehland and Mason define an effective temperature (T_{eff}) that is composed of thermal and field-induced contributions that describe the relative velocity distribution of the ion-neutral pair, which, when incorporated into **Eq. 5** and **Eq. 6**, generates **Eq. 7**.[24,25]

$$K = \frac{3}{16}\left(\sqrt{\frac{2\pi}{\mu k_B T_{eff}}}\right)\left(\frac{q(1+\alpha)}{N\left(\Omega\left(T_{eff}\right)\right)}\right) \quad \textbf{Eq. 7}$$

Note that T_{eff} can be expressed in terms of the drift velocity (**Eq. 8**), mobility coefficient (**Eq. 9**), or reduced mobility coefficient (**Eq. 10**), where M represents the molecular mass of the bath gas particle (*i.e.,* 28 Da for N_2).

$$T_{eff} = T + \left(\frac{M}{3k_B}\right)(v_D)^2(1+\beta) \quad \textbf{Eq. 8}$$

$$T_{eff} = T + \left(\frac{M}{3k_B}\right)(KE)^2(1+\beta) \quad \textbf{Eq. 9}$$

$$T_{eff} = T + \left(\frac{M}{3k_B}\right)\left(K_0 N_0 \frac{E}{N}\right)^2(1+\beta) \quad \textbf{Eq. 10}$$

Eq. 7 – Eq. 10 feature two correction factors, α and β, the former of which is distinct from the alpha function shown in **Eq. 5** and **Eq. 6**. These correction factors are non-zero when higher order approximations of 2TT are implemented but always result in an ion's T_{eff} being greater than T for any field strength greater than 0 Td. However, for low-field conditions (*ca.* < 10 Td), the contribution

of the electric field to an ion's drift velocity is negligible, resulting in a relative velocity distribution of the ion-neutral pair that resembles its thermal velocity. Consequently, under low-field conditions T_{eff} and T are equal, and **Eq. 7** simplifies to **Eq. 4**.

Despite our group's prior success in implementing the 2TT approach in modelling mobilities above the low-field limit,[19] we have yet to formalize our approach within the IMS literature. This manual covers the updates implemented to the first edition of the parallelized CCS calculation suite MobCal-MPI, which is now capable of accurate ion mobility calculations at arbitrary field strengths. This update also optimizes the treatment of ion-neutral collisions with N_2, where evaluating the ion-neutral interaction potential from both nitrogen atoms of N_2 instead of its centre of mass results in more accurate CCS calculations. By coupling the updated treatment of ion-neutral collisions with an empirically corrected 2TT implementation, MobCal-MPI 2.0 enables precise and efficient calculations of high-field mobilities that accurately reproduce ion behaviour on high-field IMS platforms.

2.2 How Ion Mobility is Evaluated *in silico*

Depending on the CCS package used, ion-neutral collisions are modelled using one of three methods (see **Figure 1A**). The simplest of the three is the projection approximation (PA), which ignores all interactions between the ion and collision gas and does not require any evaluation of the incident gas trajectories.[9,26] The PA calculates the CCS by defining the rotationally averaged area from buffer gas molecules that "strike" the ion. Although this method can evaluate an ion's CCS exceptionally fast, its utility is limited to large, globular species – the PA is not suitable for small molecules or microsolvated clusters because CCSs are determined exclusively from the ion's size, and not by ion-neutral interactions.

The elastic hard-sphere scattering (EHSS) improves on the PA by evaluating CCSs from the deflection angle of a gas molecule following its collision with the static analyte.[27] However, the EHSS method does not consider long-range interactions and treats all collisions as hard-sphere, specular scattering events. CCSs evaluated using the EHSS method tend to be inaccurate for small molecules where long-range interactions (*e.g.,* charge/induced-dipole) are important.[26,28,29] For example, the CCS of acetic acid in its neutral (CH_3COOH) and deprotonated (CH_3COO^-) forms are 41.82 Å^2 and 41.39 Å^2, respectively, when calculated by the PA method and 42.56 Å^2 and 41.11 Å^2, respectively, when calculated by the EHSS method. In contrast, the TM method computes values of 54.71 Å^2 and 105.34 Å^2, respectively. For systems where the long-range interactions become less crucial (*e.g.*, proteins), PA and EHSS methods produce relatively accurate CCSs in a fraction of the time required by the trajectory method (TM).[30,31] The increased calculation time of the TM stems from modelling collision events with a "soft" ion-neutral interaction potential that includes short- and long-range components. TM calculations require van der Waals (vdW), ion-induced dipole (IID), and ion-quadrupole (IQ) components to describe what happens to a polarizable gas molecule upon interaction with a charged analyte. The vdW component accounts for Pauli repulsion at short distances and attractive dispersion forces at longer distances, whereas the IID and IQ terms account for the polarizability of the collision partner, which becomes more important at distances close to the minimum on the vdW potential. The functional forms of the vdW, IID, and IQ components are discussed in Section 2.4, which, when parameterized, yields CCSs of 55.5 Å^2 and 116.4 Å^2 for the CCSs of acetic acid and the acetate anion, respectively.

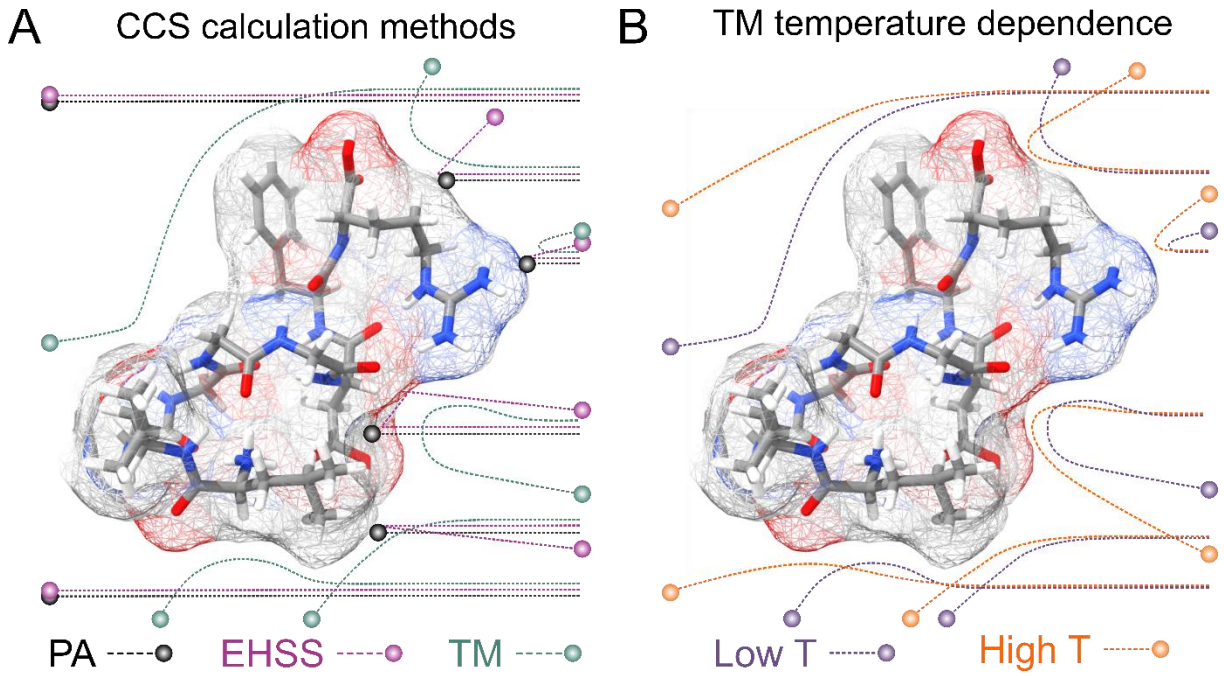


Figure 1. (**A**) Summary of the main CCS calculation routines represented through collision trajectories. The projection approximation (PA; black trace) and elastic hard sphere scattering (EHSS; light purple trace) only consider direct contact between the ion and collision gas. The trajectory method (TM; green trace) evaluates the neutral trajectory as per the ion-neutral interaction potential, which is influenced by the charge density of the analyte. Areas of high partial positive charge (blue) and high partial negative charge (red) are mapped onto the ion total electron density cloud. (**B**) The buffer gas trajectories depend on their incident velocity, which depends on the temperature of the TM simulation. Higher temperature simulations (orange) decrease the momentum transfer between the ion and buffer gas relative to lower temperature simulations (purple).

The TM determines an ion's CCS from the momentum transfer integrals that describe a collision event; these are averaged over several thousand iterations that sample multiple velocities and geometries of the ion and collision partner (see **Eq. 11**).[27] In **Eq. 11**, θ, φ, and γ define the relative orientation of the ion with respect to its collision partner, g is the relative velocity of the ion-neutral pair, b is the impact parameter, and χ is the scattering angle at which buffer gas is deflected upon interaction with the ion. Except for χ, all parameters are determined at the beginning of the CCS calculation. Evaluation of χ requires simulation of the trajectory that the buffer gas takes upon colliding with the analyte, which is governed by the interaction potential. After remission of the gas molecule, χ is calculated as the angle between the initial and final vectors that define the trajectory of the buffer gas.

Of the available CCS calculation methods, the trajectory method (TM) provides the most physically realistic approach to modelling CCS without performing large scale stochastic simulations of the ion travelling through the mobility device.[32–34] This accuracy stems from the explicit consideration of the long-range component of the interaction potential, which will influence buffer gas trajectories. The velocity of the incoming buffer depends on the collision temperature (*i.e.,* T_{eff}), meaning that an

increase in T_{eff} corresponds to an increase in the average velocity of the ion and the buffer gas. Since g describes the velocity of the buffer gas *relative to* the ion, one can simplify the evaluation of the collision event by assuming that the ion is static relative to the buffer gas. This indicates that increasing T_{eff} shifts the Maxwell-Boltzmann distribution that defines the incident buffer gas velocities to higher values. Increased gas velocities will result in the collision partner feeling less of the long-range component of the ion-neutral interaction potential. Consequently, the momentum transfer between the buffer gas and analyte will decrease, which decreases the CCS.

$$\Omega_{avg}(T_{eff}) = \left(\frac{1}{8\pi^2}\right)\int_0^{2\pi} \mathrm{d}\theta \int_0^{\pi} \sin\varphi \,\mathrm{d}\varphi \int_0^{2\pi} \mathrm{d}\gamma \frac{\pi}{8}\left(\frac{\mu}{k_b T_{eff}}\right)^3 \int_0^{\infty} (g^5)\exp\left(-\frac{\mu g^2}{2k_B T_{eff}}\right)\mathrm{d}g \int_0^{\infty} 2b(1-\cos\chi(\theta,\varphi,\gamma,g,b))\,\mathrm{d}b \qquad \textbf{Eq. 11}$$

The diminishing momentum transfer as T_{eff} increases is somewhat counterintuitive – one might expect that increasingly energetic collisions between the ion and buffer gas should yield a greater degree of momentum transfer. However, this is only true for *striking* collisions. Most collisions sampled during a TM calculation are *glancing* collisions, which deflect the collision partner to a lesser extent at high T_{eff}. This is best visualized by comparing collision events at low and high T_{eff} (see **Figure 1B**), where the scattering angle χ is *higher* for striking collisions compared to glancing collisions at high T_{eff}. Since momentum transfer is proportional to $(1-\cos\chi)$,[6] glancing collisions bring this term closer to zero [*i.e.*, for $\chi < 90°$; $(1-\cos\chi) < 1$] whereas striking collisions increase this term because head-on collisions reflect the buffer gas molecule (*i.e.*, $\chi > 90°$; $(1-\cos\chi) > 1$). The likelihood of a striking collision occurring between N_2 and a molecule with a CCS of 160 Å^2 is about 25% at 298 K. We can determine this quantity by assuming x% of collisions will be striking and $(1-x)$% of collisions will be glancing. As the ion's T_{eff} approaches infinity, its CCS approaches the hard-sphere limit, which means that the only feasible collision events are those in which N_2 strikes the ion. Thus, the probability that any collision will be striking can be estimated as the ratio between an ion's CCS at the hard-sphere limit (*i.e.,* as T_{eff} approaches infinity) and at the temperature of interest. By association, the probability that any collision will be glancing diminishes with increasing T_{eff} because the effective area in which an ion can deflect an incoming molecule of N_2 decreases. Moreover, momentum transfer becomes less efficient as incident N_2 velocities increase, meaning that any glancing collisions will be deflected to a lesser extent.

So, what does this mean in terms of calculating an ion's CCS accurately? Given that the PA and EHSS methods do not incorporate a long-range component in their treatment of a collision event, they cannot accurately evaluate CCSs at different temperatures. Consequently, the PA and EHSS methods will not be useful when modelling an ion's mobility wherein its effective temperature changes dynamically with the electric field. To evaluate an ion's CCS (and hence its mobility) at arbitrary field strengths, the temperature dependent TM must be used.

2.3 Evaluating Ion Mobility via the Trajectory Method

Figure 2 shows a schematic depicting the methodology of the trajectory method to compute collision integrals. Importantly, the collision dynamics must be sampled to evaluate χ, which depends on g, b, and the orientation of the ion relative to the starting position of the collision partner (N_2). Evaluation of χ thus requires simulation of the trajectory that the buffer gas takes upon colliding with the analyte. Depending on the initial conditions (*i.e.,* g, b, and ion-neutral orientation), substantially different collision behaviour can be observed. For example, N_2 will be scattered to a much greater extent for collisions defined by a small impact parameter (striking collision) compared to a large impact parameter (glancing collision). In terms of the relative velocity of the ion-neutral pair, which is determined by the effective temperature, the degree of scattering diminishes for glancing collisions with increasing g.

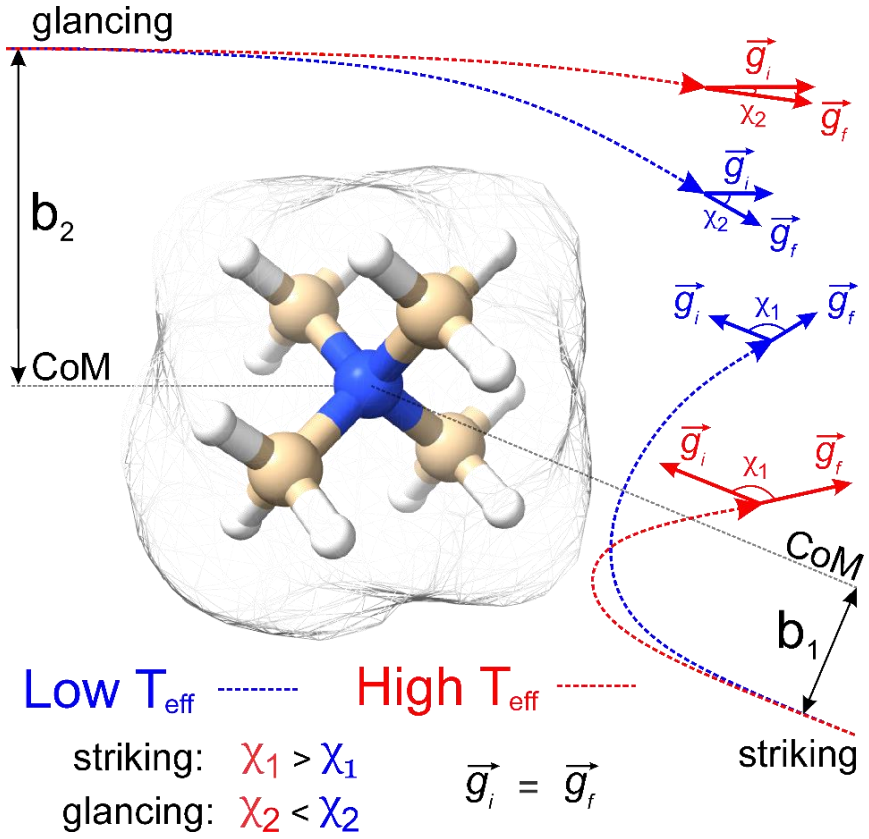


Figure 2. Schematic representation of how the trajectory method is employed for CCS calculations, using NMe_4^+ as an example. Two collision events are shown (striking vs. glancing) that exhibit different impact parameters ($b_{striking} < b_{glancing}$) and relative velocities ($g_{blue} < g_{red}$), the latter of which corresponds to differing effective temperatures (higher T_{eff} is associated with higher g). The scattering angle, χ, is calculated from the angle between the initial and final velocity vectors. Note that ion-neutral collisions are treated elastically, so the remission velocity (g_f) is equivalent to the incident velocity (g_i).

In contrast, the relative velocity has a significantly smaller impact on χ for striking collisions (see Section C2). After remission of the gas molecule, χ is calculated as the angle between the initial and final vectors that define the trajectory of the buffer gas, the magnitude of which are equal because MobCal-MPI 2.0 treats collisions elastically. With χ in hand for all starting conditions sampled, the CCS can be obtained via integration. For simplicity, we report the form of the $\Omega^{(l,s)}$ only for spherically symmetric ion-neutral interaction potentials in **Eq. 12** – **Eq. 14**, where the l and s indices denote the order of the collision integral. For further details on evaluating higher order collision integrals, please see Section C3 and C4.

$$\Omega^{(l,s)}(T_{eff}) = \int_0^\infty Q^{(l)}(g)\omega^{(s)}(g, T_{eff})\,\mathrm{d}g \qquad \textbf{Eq. 12}$$

$$\omega^{(s)}(g, T_{eff}) = \frac{1}{2^{s+1}(s+1)!}\left(\frac{\mu}{k_B T_{eff}}\right)^{s+2}\left(g^{2s+3}\exp\left(-\frac{\mu g^2}{2k_B T_{eff}}\right)\right) \qquad \textbf{Eq. 13}$$

$$Q^{(l)}(g) = 2\pi\left[1 - \frac{1+(-1)^l}{2(1+l)}\right]^{-1}\int_0^\infty [1-\cos^l(\chi)]b\,\mathrm{d}b \qquad \textbf{Eq. 14}$$

Within **Eq. 14**, integration over b and the orientation of the ion is accomplished using Monte-Carlo (MC) sampling. In other words, the ion is randomly rotated in space and an impact parameter is randomly chosen between 0 and an initially derived b_{max} (see Section C2 for details). The sample size of this MC integration is termed imp in MobCal-MPI. In contrast, the integration over the relative velocity of the ion-neutral pair is sampled on fixed grid points, which are denoted inp. In the previous implementation of MobCal-MPI, a weighted grid was used to efficiently integrate over velocity space.[27,29,35] However, changing the temperature that defines the velocity distribution requires the weighted grid to be modified for every collision integral because the weighting functions ($\omega^{(s)}$) depend on s and T_{eff} (as per **Eq. 13**). Since our goal is to efficiently calculate multiple collision integrals (up to $\Omega^{(2,4)}$) over the range defined by the temperature of the bath gas (T) and the highest effective temperature desired by the user $(T_{eff,max})$, we implemented a linear grid of velocity points defined by the temperature range. Because the linear grid of velocities is shared between all CCS integrations, only one set of N_2 scattering trajectories is required to evaluate all collision integrals. In other words, using a linear grid to sample collision events between T and $T_{eff,max}$ enables the simultaneous evaluation of the ion's mobility and CCS at any temperature within this range.

2.4 Optimizing the accuracy of CCSs calculated using MobCal-MPI

Similar to its predecessor, MobCal-MPI 2.0 computes CCSs based on the ion's geometric structure (given as xyz-coordinates), partial charges, and atom classes defined by the MMFF94 force field.[36] In order to evaluate accuracy and calculation efficiency, we employ four different test sets: the *calibration set* (N = 162), *validation set* (N = 50), *high-field validation set* (N = 132), and *peptide set* (N = 12). The

calibration and validation sets are composed of analytes with known CCSs (obtained from references [12,13,37–44]), which encompass each atom type defined by the MMFF94 forcefield. Multiple conformers are considered for each analyte, which were obtained using the protocol outlined in the original MobCal-MPI publication.[35] The *high-field validation set* is composed of small, rigid analytes (with a maximum of five conformers), for which our group has previously measured high-field mobility data.[19,45,46] Finally, structures in the *peptide set* were generated from the lowest energy structure determined from their amino acid sequence using the I-TASSER suite,[47,48] and are used exclusively for benchmarking. The *peptide set* contains 12 species ranging in length from 9 to 22 amino acid residues. To remove ambiguity in charge site assignment, the N-terminus and all basic residues (*i.e.,* His, Lys, and Arg) of each peptide were protonated, leading to charge states ranging from +2 to +5.

All structures in the *calibration*, *validation* and *high-field validation sets* were reoptimized using density functional theory (DFT) at the ωB97X-D3/def2-TZVPP level of theory.[49–53] Subsequent calculation of partition functions (via computation of vibrational frequencies and rotational constants) at the same level of theory enables Boltzmann weighting of different conformers for each ion. Atomic partial charges were computed via the CHELPG partition scheme using a grid composed of points spaced apart by 0.1 Å, where each grid point is at most 3.0 Å away from any atom in the molecule.[54] Due to the size of the molecules within the *peptide set*, calculations were conducted using the faster TPSS-D3/def2-TZVPP level of theory[55] and a coarser grid for partial charges (0.3 Å spacing, 2.8 Å cut-off). DFT optimization of the structures obtained from I-TASSER was not necessary as the *peptide set* merely served for benchmarking. All DFT calculations were conducted with the ORCA 5.0.3 program suite[56,57] and were uploaded to the ioChem-BD database[58] (see https://iochem-bd.bsc.es/browse/handle/100/285126). Subsequent determination of the MMFF94 force field atom types was undertaken using OpenBabel.[59]

In MobCal-MPI 2.0, we retain the form of the ion-neutral interaction potential (V_{total}; **Eq. 15**) from its predecessor, which consists of a van der Waals (vdW) term composed of a modified Buckingham potential (Exp-6) derived from the MM3 forcefield (V_{vdW}; **Eq. 16**),[60] an ion-induced dipole term (V_{IID}; **Eq. 17**),[28] and an ion-quadrupole term (V_{IQ}; **Eq. 18**).[28] With increasing buffer gas polarizability, the V_{IID} term plays an increasingly significant role in the ion-neutral potential. If the buffer gas possesses a quadrupole moment (*e.g.*, N_2), the addition of an ion-quadrupole potential is crucial for the evaluation of accurate scattering trajectories.[28,61] To effectively mimic the quadrupole moment of molecular nitrogen ($4.65 \pm 0.08 \times 10^{-40}$ C cm^2),[62] partial charges of −0.4825 *e* are assigned to each nitrogen atom and balanced by a point charge of +0.965 *e* at its centre of mass. This simplifies the calculation of the ion-quadrupole potential, where the atomic partial charges on the analyte (q_i) are iterated over their distance to each pseudo-charge site of N_2.

$$V_{total} = V_{vdw} + V_{IID} + V_{IQ} \qquad \textbf{Eq. 15}$$

$$V_{vdW} = \sum_{i=1}^{\mathcal{N}} \varepsilon_i \left(1.84 \times 10^5 \exp\left(\frac{12 r_i}{r_i^*}\right) - 2.25 \left(\frac{r_i^*}{r_i}\right)^6 \right) \qquad \textbf{Eq. 16}$$

$$V_{IID} = -\frac{\alpha e^2}{2} \left[\left(\sum_{i=1}^{\mathcal{N}} \frac{q_i x_i}{r_i^3} \right)^2 + \left(\sum_{i=1}^{\mathcal{N}} \frac{q_i y_i}{r_i^3} \right)^2 + \left(\sum_{i=1}^{\mathcal{N}} \frac{q_i z_i}{r_i^3} \right)^2 \right] \qquad \textbf{Eq. 17}$$

$$V_{IQ} = \frac{1}{4\pi\epsilon_0} \sum_{i=1}^{\mathcal{N}} \sum_{j=1}^{3} \frac{q_i q_j e^2}{r_{ij}} \qquad \textbf{Eq. 18}$$

The ε_i and r_i^* parameters were based on the implementation from the Kim group in the original MobCal-MPI publication,[3537] who combined atomic parameters from the MMFF94 force field for the ion with parameters for molecular N_2.[36] The enhanced accuracy of this approach can be attributed to distinction of atom types within the MMFF94 framework (*e.g., sp*^{3} *versus sp*^{2} hybridized carbon centres have different ε_i and r_i^*), allowing for the accurate evaluations of the vdW component of the ion-neutral interaction potential. Because the MMFF94 parameters derived for N_2 describe a molecular entity (*i.e.,* diatomic nitrogen), the distance term in the V_{vdW} pairwise interactions (r_i) can be evaluated in two ways (**Figure 3**). In the first case, N_2 can be treated as pseudo-atomic entity, whereby pairwise interactions are calculated with respect to its centre of mass (CoM). Alternatively, ion-neutral interaction potentials can be evaluated by considering the pairwise interaction between each atom in the analyte and each nitrogen atom in N_2, and then averaging the result (Avg-N_2). The latter case seems to be more reasonable, especially at short ion-neutral distances where the orientation of the N_2 molecule significantly impacts the repulsive portion of the interaction potential (see Section C5).

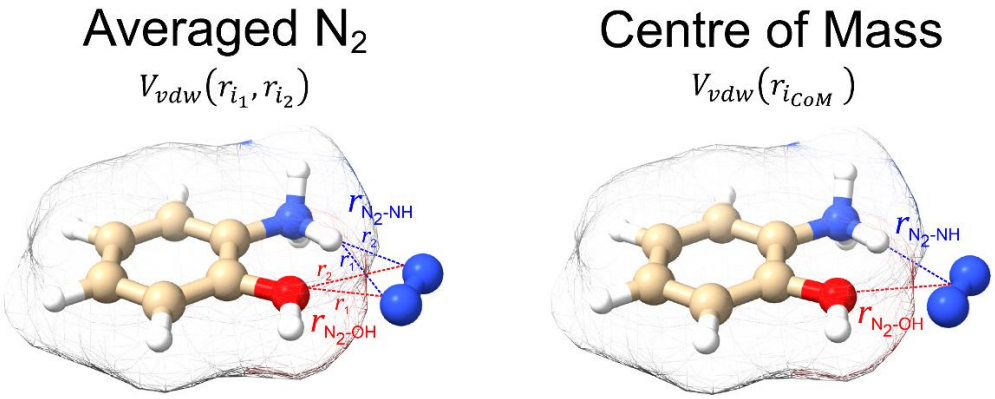


Figure 3. The two variations of calculating interaction potentials with N_2 depicted for *N*-protonated 2-aminophenol. In the averaged N_2 version, the potential is calculated from each atom of the ion to both atoms of N_2, and subsequently averaged. In the centre of mass version, the potential is calculated from each atom of the ion to the centre of mass of the N_2.

Owing to the inherent parameterization of vdW terms within the MMFF94 forcefield, linear scaling parameters (ρ_{dist} and ρ_{ener}) could be applied uniformly to the ε_i and r_i^* corresponding to each atom type, greatly simplifying their optimization (**Eq. 19** and **Eq. 20**, respectively). This suggests that the same optimization methodology for ρ_{dist} and ρ_{ener} can be applied to either the CoM or Avg-N_2 version of the potential.

$$\varepsilon_i' = \rho_{ener} \cdot \varepsilon_i \qquad \textbf{Eq. 19}$$

$$r_i^{*\prime} = \rho_{dist} \cdot r_i^* \qquad \textbf{Eq. 20}$$

To assess the accuracy of both the CoM and Avg-N_2 approaches, we calculated the CCS of the 162 compounds in the *calibration set* using various combinations of ρ_{dist} and ρ_{ener}, and compared the results with their experimentally measured CCSs. **Figure 4A** and **Figure 4B** show the root mean square errors (RMSE) between the calculated and measured CCSs as contour plots for the Avg-N_2 and CoM approaches, respectively. Similar to the previous version of MobCal-MPI, there is no single set of scaling parameters that performs best for MobCal-MPI 2.0, but rather a range of values that yield RMSEs of < 2.5 %. Interestingly, the CoM and Avg-N_2 methods exhibit comparable accuracies despite the Avg-N_2 approach being more physically realistic. It is possible that the similar accuracies observed for both approaches stem from the flexibility of the optimization of ρ_{dist} and ρ_{ener} when only considering a finite set of test molecules. To validate the accuracy of both approaches, we selected a set of scaling parameters that performed well for the CoM (ρ_{dist} = 1.000 and ρ_{ener} = 0.580) and Avg-N_2 approaches ($\rho_{dist} = 0.825$ and $\rho_{ener} = 1.275$), and assessed their accuracy using a separate set of 50 molecules (validation set; **Figures 4C** and **4D**, respectively). It is worth mentioning that a similar procedure was undertaken to optimize ρ_{dist} and ρ_{ener} for CCS calculations in helium, and you can find further details on this in Appendix D.

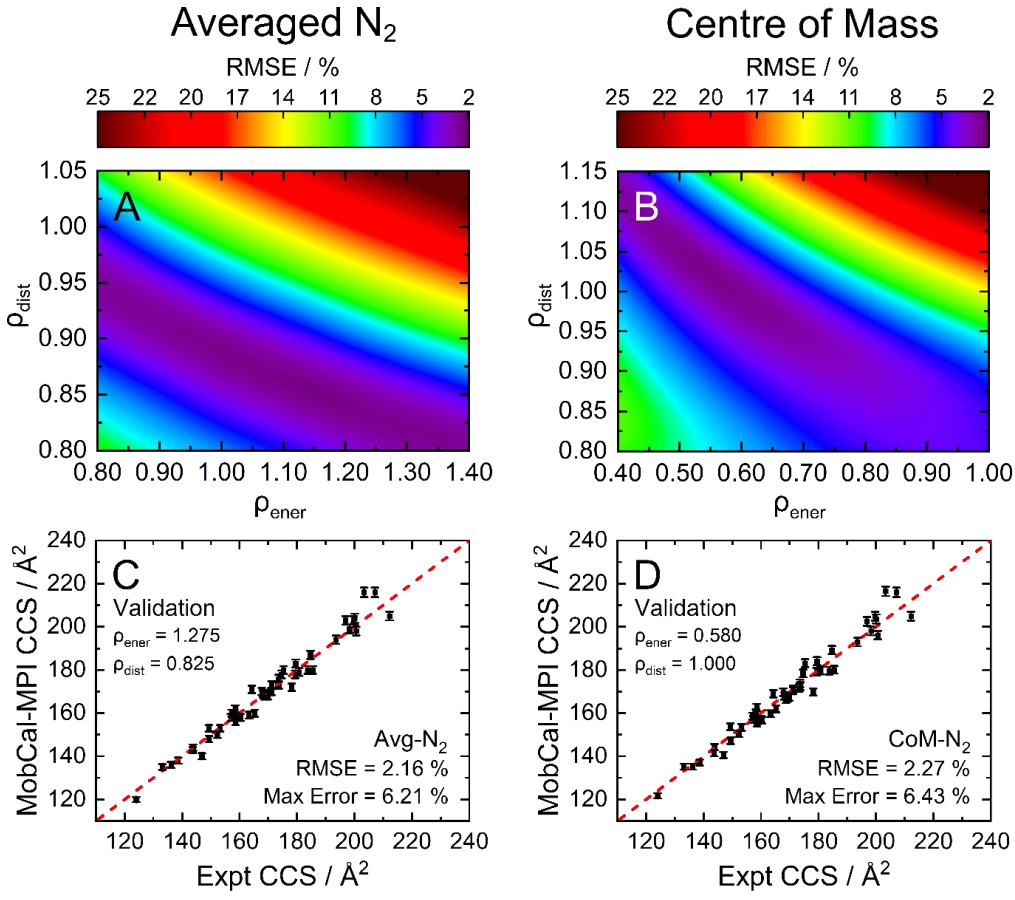


Figure 4. The RMSE, shown as a contour plot, for the optimization of ρ_{dist} and ρ_{ener} using the (**A**) Avg-N_2 or (**B**) CoM approach for the *calibration set* (162 molecules). To validate the accuracy of (**C**) Avg-N_2 or (**D**) CoM approaches, we employed the values of ρ_{dist} and ρ_{ener} yielding the lowest RMSE to calculate the CCS of 50 distinct molecules that were not present in the *calibration set* (*validation set*). All CCS calculations were performed at 298 K with itn = 5, inp = 104, and imp = 512.

By examining the differences between the experimentally determined and calculated CCSs of the *validation* set, it can be concluded that the Avg-N_2 and CoM methods both produce accurate results. We were surprised to find that the RMSE of the CoM version (2.27 %) is only slightly larger than the RMSE of the Avg-N_2 approach (2.16 %). The reason for their similarity is likely due to the prevalence of glancing collisions at 298 K, which account for approximately 75 % of all collisions for a molecule with a CCS of 160 Å^2.[19,63] The CoM approach, which does not accurately capture the repulsive portion of the interaction potential, still produces accurate CCSs at room temperature because the repulsive portion is only important when evaluating striking collisions. However, usage of the CoM rather than the Avg-N_2 approach is not justified at high reduced field strengths, because an inaccurate treatment of ion-neutral repulsion will result in erroneous CCSs under conditions where striking collisions dominate (*i.e.,* at high T_{eff}). For this reason, we implemented the Avg-N_2 approach in MobCal-MPI 2.0. We expect the Avg-N_2 approach will ensure greater internal consistency of the code when adding collision gases, for which an explicit treatment of all atoms will also be important (*e.g.*, CO_2, SF_6).

2.5 Assessing CCS calculation uncertainty via the statistical analysis of collision events

After optimizing the scaling parameters for the Avg-N_2 approach, we developed a more thorough method to propagate statistical uncertainties that influence the uncertainty of CCS calculations. In the previous version of MobCal-MPI, the uncertainty in a CCS calculation was determined by the standard deviation in the CCS measured during each itn mobility cycle. While this methodology does assess uncertainty in a reasonable manner, it does not explicitly consider the effects of *all* integration parameters on the final uncertainty (*i.e.,* itn, imp, and inp). Incorporating imp and inp within the uncertainty evaluation is thus necessary if physically realistic uncertainties are desired. Knowing that the integration over the MC sampled trajectories for a fixed relative velocity (g) yields the momentum transfer integrals ($Q^{(l)}$) for that particular velocity, imp and inp can be incorporated within the uncertainty evaluation during the itn repetitions (yielding $Q_i^{(l)}(g)$ for i = 1, 2, …, itn), which in turn produces an average (**Eq. 21**), standard deviation (**Eq. 22**), and confidence interval (CI; **Eq. 23**).

$$\overline{Q^{(l)}}(g) = \sum_{i=1}^{itn} Q_i^{(l)}(g) \qquad \textbf{Eq. 21}$$

$$\sigma\left(Q^{(l)}(g)\right) = \sqrt{\sum_{i=1}^{itn} \frac{\left(\overline{Q^{(l)}}(g) - Q_i^{(l)}(g)\right)^2}{itn - 1}} \qquad \textbf{Eq. 22}$$

$$\sigma_{CI}\left(Q^{(l)}(g)\right) = \frac{t(99\%)}{\sqrt{itn}}\,\sigma\left(Q^{(l)}(g)\right) \qquad \textbf{Eq. 23}$$

Here, $t(99\%) = 2.57$ is the two-sided, 99 % quantile of the standard normal distribution. Increasing the amount of MC sampling points (imp) will lower the variances of the $Q_i^{(l)}(g)$, and increasing the

number of repetitions (itn) will increase confidence in the average. The above analysis is performed for each point within the velocity grid, which has a size of inp. The momentum transfer integrals are then integrated over these velocity grid points to yield the collision integrals (*cf.* **Eq. 12**). This integration can also be viewed as taking a weighted average of the $Q^{(l)}(g)$, whereby $\omega^{(s)}(g)$ reflects the weighting function. Thus, the uncertainties associated with each $Q^{(l)}(g)$ can be propagated according to **Eq. 24**, where Δg is the spacing of the velocity grid. Increasing inp decreases Δg, and hence, decreases the uncertainty of the calculated CCS. Given that $\Omega^{(1,1)}$ is the primary contributor to the mobility for all field strengths, and that other $\Omega^{(l,s)}$ provide only small corrections, we consider only $\sigma_{CI}\left(\Omega^{(1,1)}\right)$ when evaluating the uncertainty of the mobility coefficient K. Note that **Eq. 24** does not require an additional normalization factor, as $\sum \omega^{(s)}(g_i)\Delta g$ equates to unity.

$$\sigma_{CI}\left(\Omega^{(l,s)}\right) = \sqrt{\sum_{i=1}^{inp}\left[\sigma_{CI}\left(Q^{(l)}(g_i)\right)\omega^{(s)}(g_i)\Delta g\right]^2} \qquad \textbf{Eq. 24}$$

The revised methodology for treating uncertainty is depicted in **Figure 5** using protonated caffeine as example. Here, the standard deviation in the calculated momentum transfer cross sections ($Q^{(l)}$; panel A) and the integrand for the CCS ($Q^{(l)}\omega^{(s)}$; panel B) are shown as a function of the relative velocity (g). Because we only sample a finite number (*imp*) of randomized trajectories per velocity point, the individual $Q_i^{(l)}(g)$ for each itn will be slightly different, the standard deviation of which is shown by the shaded areas in **Figure 5A**. The confidence interval describes the value of $Q^{(l)}(g)$ upon infinite sampling of trajectories (*i.e.,* the statistically "true" value) lies within $\overline{Q^{(l)}}(g) \pm \sigma_{CI}\left(Q^{(l)}(g)\right)$ with 99 % confidence (*cf.* **Eq. 23**); this value is reported in the output of MobCal-MPI 2.0. Thus, increasing the number of trajectories sampled (imp) reduces the standard deviation, while increasing the number of iteration cycles (itn) reduces the uncertainty with respect to the "true value". Further, increasing the number of velocity points (inp) yields a finer grid along the x-axis, which in turn increases the accuracy of the numerical integration, and leads to a decrease in the final uncertainty of the collision integrals, $\sigma_{CI}\left(\Omega^{(l,s)}\right)$.

In this statistical analysis, we observe a general trend where collision integrals of higher order or at higher temperatures always exhibit smaller uncertainties. For example, **Figure 5A** shows that the degree of momentum transfer decreases as the relative velocity between collision gas and ion increases. This decrease occurs because the contribution of glancing collisions to the total momentum transfer continually decreases with g until all collision events are striking.[64] Consequently, the standard deviations $\sigma\left(Q^{(l)}\right)$ also decrease with g, leading to differing uncertainties for the two collision integrals shown in **Figure 5B** (2.0 Å^2 *vs.* 0.7 Å^2). This disparity is a consequence of the respective weight functions ($\omega^{(s)}$), which are centred at different relative velocities owing to their evaluation at distinct values of s and T_{eff}. Note that the functional dependency of $Q^{(l)}$ on g is influenced by the various terms in the interaction potential.[65]

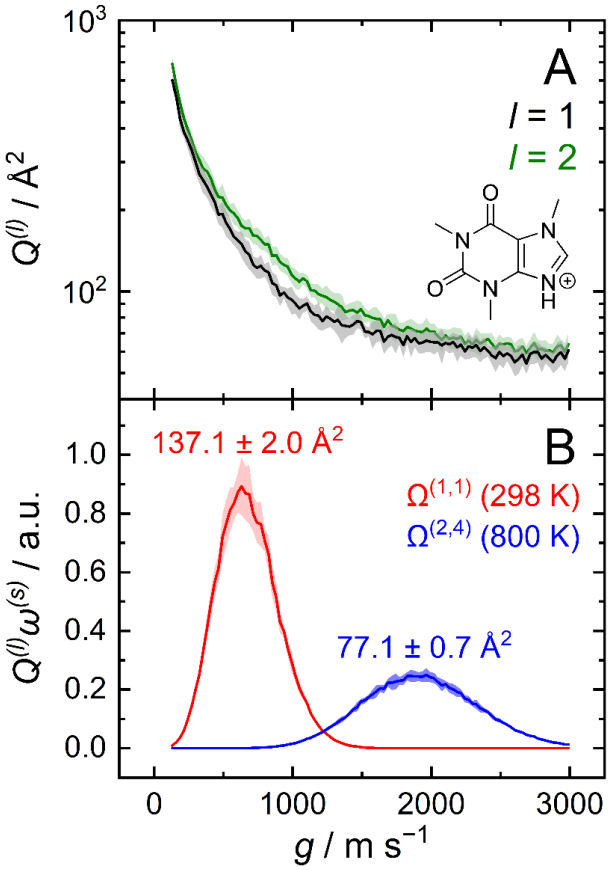


Figure 5. (**A**) Momentum transfer integrals and (**B**) CCS integrands of protonated caffeine. The standard deviations from **Eq. 22** are shown as shaded areas, and the final uncertainty for $\Omega^{(l,s)}$ corresponds to that from **Eq. 24**. CCS calculations were performed using $itn = 10$, $inp = 104$, $imp = 512$ and $T = 298$ K (panel B; red) for $\Omega^{(1,1)}$, and $T_{eff,max} = 800$ K for $\Omega^{(2,4)}$ (panel B; blue).

Although the decrease in CCS uncertainty with increasing temperature cannot be avoided unless a weighted grid for g is used, our implementation of the linear grid enables the near-instantaneous evaluation of an ion's CCS for any temperature within a user-specified range once the $Q^{(l)}$ are determined. This occurs because an ion's CCS is determined from the area under the curve of $Q^{(l)}\omega^{(s)}$, the calculation time for which is practically instantaneous compared to the time required to evaluate N_2 scattering trajectories from $\sim 10^6$ collision events. However, accurate numerical integration necessitates that $Q^{(l)}\omega^{(s)}$ goes to zero as it approaches the integration limits. To ensure this condition is met, we establish integration boundaries encompassing the relative velocities that contribute to collision integrals at a given T_{eff}. **Figure 5B** illustrates this process, where the lowest order collision integral ($\Omega^{(1,1)}$) evaluated at the lowest temperature specified by the user (*i.e.,* T) determines the lower velocity integration limit. Similarly, the highest-order collision integral ($\Omega^{(2,4)}$) evaluated at the highest user-specified effective temperature (*i.e.,* $T_{eff,max}$) establishes the upper velocity integration limit. The exact method for determining integration limits using $Q^{(1)}\omega^{(1)}$ and $Q^{(2)}\omega^{(4)}$ is provided in Section C4. Since the upper and lower limits of the effective temperature determine the range of relative velocities sampled, assessing trajectories on a common velocity grid produces internally consistent $Q^{(l)}$. This enables the use of the same trajectories to evaluate $Q^{(l)}\omega^{(s)}$

at any temperature between T and $T_{eff,max}$, thereby enabling straightforward determination of any CCS within the temperature range via numerical integration. By implementing the linear grid, MobCal-MPI 2.0 offers significantly faster CCS calculations for multiple temperatures compared to its predecessor, where collision integrals had to be recalculated for each user-specified temperature. In other words, if a user wants to compute CCSs at *n* effective temperatures, calculation via the original MobCal-MPI code takes *n*-times longer than MobCal-MPI 2.0.

2.6 An In-Depth Analysis of the Uncertainty in CCS Calculations

Owing to the impact of the size of inp and imp on the final uncertainty of calculated CCSs, a comprehensive analysis is required to understand their exact effect. This was accomplished by calculating the CCSs ($T = 298\ K$) of the *validation set* using discrete sampling sizes for the impact parameter (imp; 256, 512, 768, and 1024) and the relative velocity of the ion-neutral pair (inp; 56, 104, 152, and 200). For optimal CCS calculation efficiency within the parallelized framework of MobCal-MPI 2.0, sampling sizes for inp and imp were chosen to be divisible by the number of cores used for the CCS calculation (here, N_{cores} = 8). The distribution of final relative uncertainties ($\sigma_{CI}\left(\Omega^{(1,1)}\right)/\Omega^{(1,1)}$) is shown as a violin plot for each (imp,inp) combination, with the mean relative uncertainty noted in blue (**Figure 6**). As expected, the relative uncertainty decreases as the number of sampling points in either the imp or inp dimension increases. The average computing time also increases with increasing imp and inp because the number of trajectories sampled during one itn cycle is given by the product of imp and inp.

Our statistical analysis of uncertainty indicates the presence of a partial invariance along the diagonals of **Figure 6**. For example, the mean relative uncertainty and calculation time for (inp,imp) = (104,768) is like that observed from the (inp,imp) = (152,512). Using these relatively large inp grid sizes was necessary because of the changes implemented to MobCal-MPI 2.0, whereby velocities are sampled using a linear grid. Compared to its predecessors,[27,29] which used a weighted velocity grid that confined the velocity distribution to values populated at the temperature of the CCS calculation, only 48 inp points were required to achieve a mean relative uncertainty of 0.95 %.[35] However, since the linear grid does not confine the velocity distribution to values populated at the effective temperature of the CCS calculation, additional velocity points are necessary to achieve an equivalent level of precision. Consequently, users should choose a set of (imp,inp) such that: 1) the statistical uncertainty of MobCal-MPI 2.0 parallels that of its predecessor, and 2) the statistical uncertainty does not exceed the RMSE between calculated and experimental CCS (2.16 %; *cf.*
Figure 4. The RMSE, shown as a contour plot, for the optimization of ρ_{dist} and ρ_{ener} using the (**A**) Avg-N_2 or (**B**) CoM approach for the *calibration set* (162 molecules). To validate the accuracy of (**C**) Avg-N_2 or (**D**) CoM approaches, we employed the values of ρ_{dist} and ρ_{ener} yielding the lowest RMSE to calculate the CCS of 50 distinct molecules that were not present in the *calibration set* (*validation set*). All CCS calculations were performed at 298 K with itn = 5, inp = 104, and imp = 512.**C**). We recommend settings of imp = 512 and inp = 104 for routine CCS evaluations, as this offers a balance between high precision and calculation time. If higher precision is desired, we recommend settings of imp = 768 or imp = 1024 in tandem with an inp setting of 200 (**Table 1**).

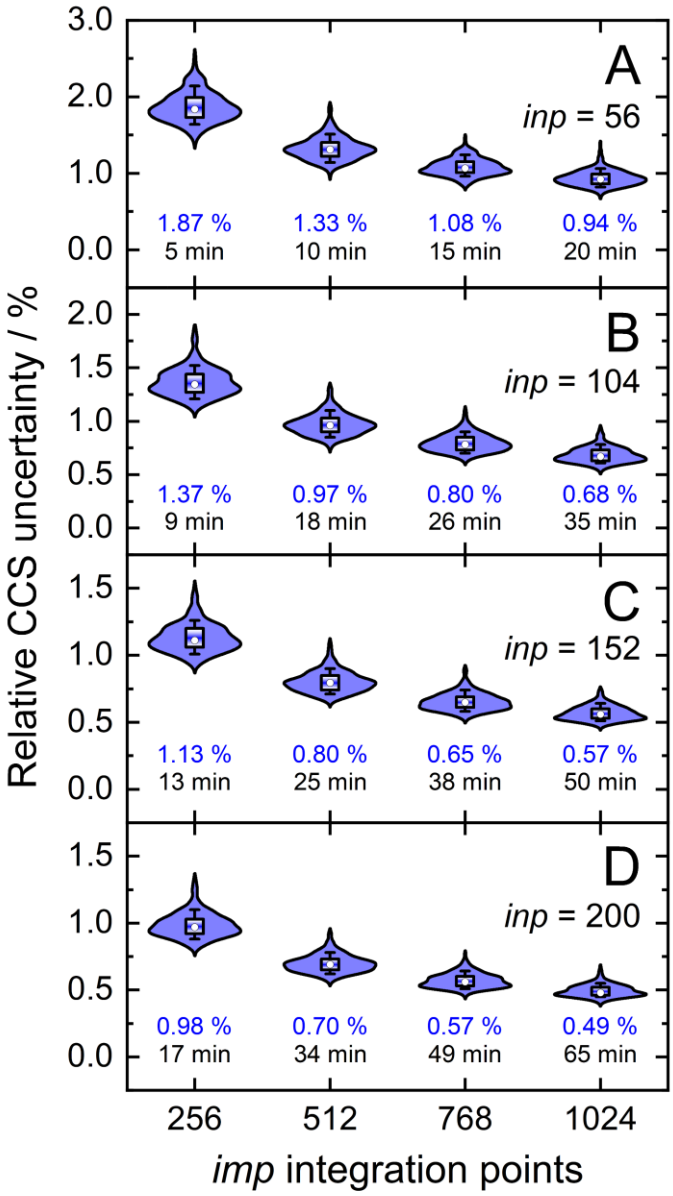


Figure 6. Distributions of relative CCS uncertainties, $\sigma_{CI}\left(\Omega^{(1,1)}\right)/\Omega^{(1,1)}$, for the validation set ($N = 50$) for different combinations of velocity sample points (inp) and orientation/impact parameter sample points (imp). Blue numbers below each distribution correspond to mean relative CCS uncertainties and black numbers to average computing time.

Table 1. Recommended settings for CCS calculations in MobCal-MPI 2.0 under low-field conditions.

Calculation Type	itn setting	inp setting	imp setting
Routine	10	104	512
High-precision	10	200	768 or 1024

We would like to remind readers that the uncertainties shown in **Figure 6** are not reflective of the deviation between an experimentally measured and calculated CCS, but rather pertain to the uncertainties in the statistical sampling of collisions and how this uncertainty is propagated within the computational workflow. Moreover, the statistical uncertainties of the CCSs depend on the range of effective temperatures used in the calculation. **Figure 5B** illustrates that, for a considerable fraction of the velocity grid points, $Q^{(l)}\omega^{(s)}$ values become negligible when the upper end of the velocity grid is defined by a relatively high effective temperature (*i.e.,* $T_{eff,max} \gg T$), thereby negating their contribution to the CCS integrand at high T_{eff}. To evaluate the effect of grid sizes on CCS calculations performed for a range of effective temperatures, we undertook an analysis akin to that of **Figure 6**. In this case, the relative uncertainty for CCSs when evaluated between T = 298 K and $T_{eff,max}$ = 800 K are approximately 30 % larger than those for a CCS calculation performed solely at T = 298 K (1.26 % *vs.* 0.97 %; see Section C6). Because this comparison was made for the suggested sampling sizes of inp = 104 and imp = 512, we recommend that users seeking CCSs at various effective temperatures employ the high-precision sampling sizes. For T = 298 K and $T_{eff,max}$ = 800 K, inp = 200 and imp = 1024 yields a mean relative uncertainty of 0.64 % for $\Omega^{(1,1)}$ at T = 298 K, which decreases further for the higher temperatures. Despite the high-precision settings being more computationally expensive, the linear grid significantly decreases the computing time because the trajectories determining the $Q^{(l)}$ need be evaluated only once. Thus, the added computational expense is offset, making this drawback relatively minor in terms of the efficiency of MobCal-MPI 2.0.

2.7 Implementing 2TT for mobility calculations at arbitrary field strength

In the MobCal-MPI 2.0 interface, users are prompted to input the bath gas temperature and an upper limit for the ion effective temperature (*i.e.,* T and $T_{eff,max}$, respectively). Specifying a temperature range calls the 2TT module, which initiates calculation of the ion's CCS and reduced mobility at reduced field strengths that fall within the given temperature range. Users can also specify a grid size for the temperature range such that CCSs and reduced mobilities are printed at desired increments of T_{eff}.

Notably, the CCS and its uncertainty decreases at higher effective temperatures, which is a consequence of the decreased efficiency in momentum transfer with increasing relative velocity (*cf.* **Figure 5**).[19,63–65] The inverse relationship between CCS and mobility leads one to expect that K_0 should increase with T_{eff}, but this is not the case. The decrease of K_0 with T_{eff} is a consequence of the kinetic theory of gases, which states that the apparent viscosity of the collision gas increases with T_{eff}.[66] This underscores the fact that the ion mobility is not a constant, but rather a function of the field strength. Since the reduced field strength computed by MobCal-MPI 2.0 is determined by the effective temperature of the ion, the alpha function can be readily obtained from the calculated mobility data using **Eq. 5Error**! Reference source not found. and may be subsequently fit to an even order polynomial of the form given by **Eq. 6** to determine the alpha coefficients. Employing this workflow for protonated amoxapine results in $\alpha_2 = -2.935 \times 10^{-6}$ Td^{-2}, $\alpha_4 = 3.080 \times 10^{-11}$ Td^{-4}, and $\alpha_6 = -2.553 \times 10^{-16}$ Td^{-6}.

$$\alpha\left(\frac{E}{N}\right) = \frac{K_0(E/N)}{K_0(0)} - 1 \qquad \textbf{Eq. 25}$$

Table 2

Table 2 shows an example output from MobCal-MPI 2.0, where the CCS and reduced mobility of protonated amoxapine in N_2 was calculated between T = 373 K and an arbitrary choice of $T_{eff,max}$ = 700 K on a temperature grid composed of 8 points. T = 373 K corresponds to the bath gas temperature, which yields a CCS and reduced mobility in the zero-field limit (*i.e.,* E/N = 0). Because all collision integrals are evaluated on the same linear velocity grid, users can increase the number of points sampled within the temperature range up to $T_{eff,max}$ without incurring an increase in calculation time.

Notably, the CCS and its uncertainty decreases at higher effective temperatures, which is a consequence of the decreased efficiency in momentum transfer with increasing relative velocity (*cf.* **Figure 5**).[19,63–65] The inverse relationship between CCS and mobility leads one to expect that K_0 should increase with T_{eff}, but this is not the case. The decrease of K_0 with T_{eff} is a consequence of the kinetic theory of gases, which states that the apparent viscosity of the collision gas increases with T_{eff}.[66] This underscores the fact that the ion mobility is not a constant, but rather a function of the field strength. Since the reduced field strength computed by MobCal-MPI 2.0 is determined by the effective temperature of the ion, the alpha function can be readily obtained from the calculated mobility data using **Eq. 5**Error! Reference source not found. and may be subsequently fit to an even order polynomial of the form given by **Eq. 6** to determine the alpha coefficients. Employing this workflow for protonated amoxapine results in $\alpha_2 = -2.935 \times 10^{-6}$ Td^{-2}, $\alpha_4 = 3.080 \times 10^{-11}$ Td^{-4}, and $\alpha_6 = -2.553 \times 10^{-16}$ Td^{-6}.

$$\alpha\left(\frac{E}{N}\right) = \frac{K_0(E/N)}{K_0(0)} - 1 \qquad \textbf{Eq. 25}$$

Table 2. Example output data from MobCal-MPI 2.0 for protonated amoxapine, showing relevant mobility data in N_2 for a range of effective temperatures (T = 373 K and $T_{eff,max}$ = 700 K).

T_{eff} / K	E/N / Td	K_0 / cm^2 V^{-1} s^{-1}	CCS / Å^2	CCS *uncertainty* / %
373.0	0.0	1.2250	154.27	0.64%
402.7	50.0	1.2162	150.08	0.63%
449.0	81.3	1.2028	144.50	0.62%
501.8	107.9	1.1878	139.19	0.61%
551.4	129.1	1.1742	135.02	0.61%
600.9	148.2	1.1609	131.44	0.60%
650.5	166.1	1.1479	128.34	0.59%
700.0	183.0	1.1353	125.63	0.58%

To assess the accuracy of 2TT in calculating high field mobilities, one can compare the alpha curves obtained through experimentation and computation. This comparison is shown in **Figure 7A** for protonated amoxapine (blue trace), whose experimental alpha curve (black trace) was determined using the protocol outlined in Section C7. MobCal-MPI 2.0 captures the general trend of amoxapine's alpha curve, which adopts increasingly negative values as the reduced field strength increases.

However, the computed alpha curve exhibits more negative values than the experimental curve, indicating that calculated mobility coefficient decreases faster with field strength compared to the actual measurement. This finding aligns with the observations made by Siems *et al.*,[67] who reported that the 2TT approach yields accurate mobilities at low and medium field strengths, but underestimates mobilities at high field strengths. Consequently, alpha curves evaluated using 2TT will almost always exhibit a systematic, negative deviation, especially at high field strengths.

To address the systematic deviation between the computed and measured alpha functions, we introduced an empirical correction to 2TT, inspired by the form of the deviations found by Siems *et al.* (see Section C8 for details).[67] Briefly, a field-dependent correction factor (f_{corr}) is used to adjust the mobilities calculated from 2TT (K_{2TT}) using **Eq. 26** and **Eq. 27**.

$$K_{corr} = f_{corr} \cdot K_{2TT} \qquad \textbf{Eq. 26}$$

$$f_{corr}\left(\frac{E}{N}\right) = 1 + A\exp\left(-\frac{B}{E/N}\right) \qquad \textbf{Eq. 27}$$

The parameters *A* and *B*, which represent the maximum deviation at infinitely high field strengths and the field strength at which deviations become significant, respectively, can be tuned for a specific ion such that the deviation between the computed and measured alpha curves is minimized. For protonated amoxapine, optimizing *A* and *B* leads to almost perfect alignment of the computed and experimental alpha curves (**Figure 7A**; red trace). Moreover, the value obtained for *A* (4.4 %) is consistent with the range reported by Siems *et al.* (5 – 7 %), and the value of *B* = 132 Td indicates that the correction becomes most significant at high field strengths.

To investigate the general performance of the empirical correction, we assessed its ability to correct high-field mobilities for a large set of compounds. An efficient means of obtaining a dataset of high-field mobilities is via DMS, which is also known as field-asymmetric ion mobility spectrometry (FAIMS).[10,18,68] In DMS, the ions are subjected to an asymmetric waveform (separation voltage; SV) that consists of high-field and low-field components. The SV induces the off-axis displacement of the ion by an amount proportional to its alpha function. The application of a species-specific compensation voltage (CV) to the SV enables ions to traverse the DMS device; at a given SV, each analyte is transmitted at a distinct CV that is intrinsically linked to the ion's alpha function. By monitoring the dependence of the CV required for analyte transmission at different strengths of the SV field, the corresponding SV *vs.* CV relationship (*i.e.,* a dispersion plot) can be employed to derive the alpha function using the method described in Section C7.

Dispersion plots for 132 compounds were procured from previously published datasets from our group to generate the *high-field validation* set.[19,45,46] Each analyte's dispersion plot was recorded on the SelexION DMS platform (SCIEX; 1 mm gap and T = 373 K) in a pure N_2 environment between SV = 0 – 4000 V. In theory, each analyte's dispersion curve can be converted to the corresponding alpha curve. However, because our goal is to reproduce experimentally measured quantities, Haack and Hopkins employed an iterative method previously published by our group to calculate dispersion plots from the mobility data generated by the 2TT implementation within MobCal-MPI 2.0.[69] In general, the calculated dispersion plots displayed positive compensation voltage (CV) shifts relative to those determined experimentally (**Figure 8A**). Positive CV shifts indicate that the systematic

underestimation observed for protonated amoxapine is also observed for most compounds in the *high-field validation set*.

Since the empirical correction to 2TT could be employed to replicate the alpha curve of amoxapine, it is reasonable to assume that the same approach can be employed to address the systematic CV shift observed in the dispersion plots. To this end, A and B were optimized such that the deviation between the calculated and experimental dispersion curves were minimized (for select examples, see Section C8). **Figure 7B** shows the distributions of the A and B parameters obtained for each compound in the *high-field validation set*, the majority of which align with the results found by Siems *et al.* (5 – 7 % for A and 100 – 200 Td for B).[67] The parameters exhibit unimodal distributions when considered independently or together, the latter of which is best visualized in the contour plot. The data also suggests little to no correlation between the two parameters, although it is worth noting that B becomes undefined as A approaches zero.

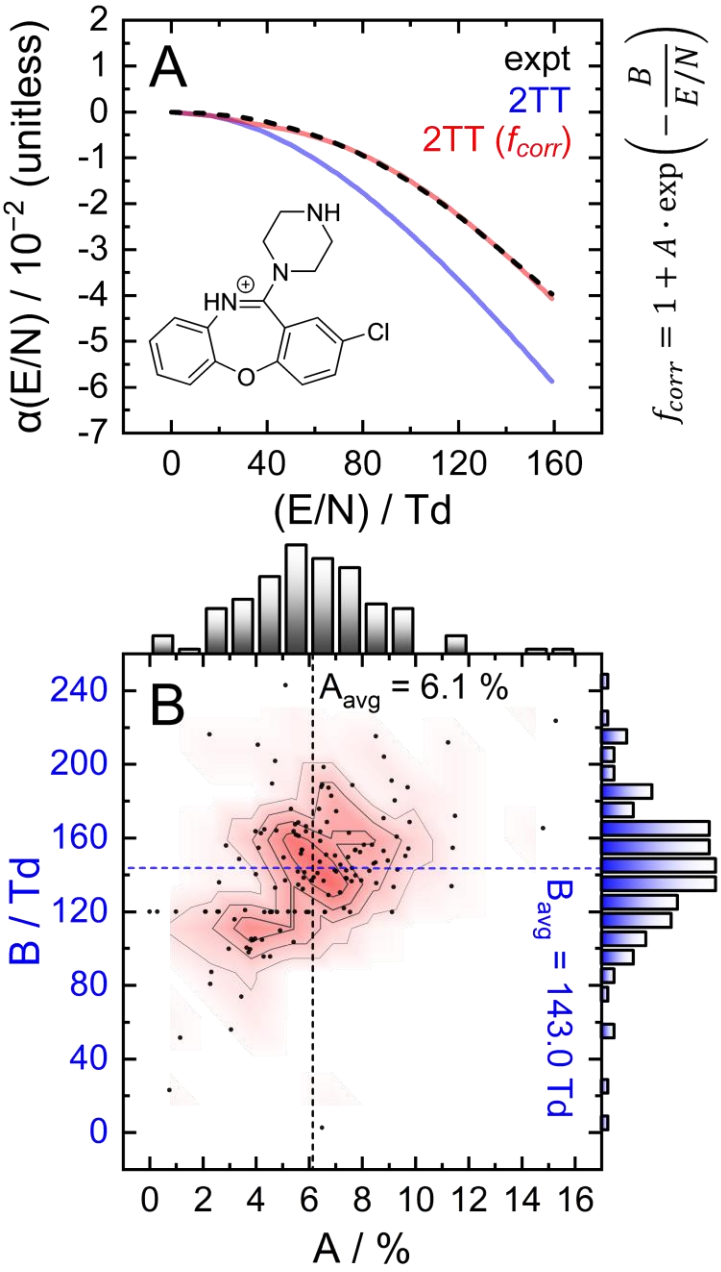


Figure 7. Investigation of the empirical correction to 2TT. (**A**) Comparison of experimental and calculated alpha functions for protonated amoxapine. Optimized parameters are A = 4.4 % and B = 132 Td. (**B**) 2D distribution of optimized A and B parameters of the empirical correction, f_{corr}, for each of the 132 compounds is the *high-field validation set*.

Expanding the applicability of the empirical correction to various chemical systems necessitates knowledge of values for A and B *a priori* such that they can be applied to a broad range high-field mobilities predicted by 2TT. Because these parameters are uniformly distributed about their respective means and are seemingly uncorrelated with properties relevant to collision theory (see Fig S8), we chose to use the average values of A (6.1 %) and B (143.0 Td) to test the accuracy of this approach. A_{avg} and B_{avg} were hard-coded into MobCal-MPI 2.0, enabling the calculation of high-field mobility data using a uniform set of correction factors. The deviations between calculated and experimental

dispersion plots that either employ or exclude the empirical correction to 2TT are shown in **Figure 8B**. As previously mentioned, exclusion of the empirical correction results in a systematic overestimation of the CV values for all 132 species, especially at high field strength. Using the empirical correction with A_{avg} and B_{avg} removes the systematic deviation, shifting the average error in CV towards zero for all SVs sampled in the dispersion plot. **Figure 8C** shows the distribution of CV deviations at SV_{pp} = 4000 V. When applying the empirical correction, over 88 % of the data falls within a ± 4 V range of the experimental value. The mean deviation between calculated and experimental CV is – 0.4 V at SV = 4000 V, equivalent to a relative deviation of approximately 4 %. The full width at half maximum (FWHM) of this distribution is 5.7 V, which is comparable to the experimental FWHM at this separation field amplitude (≈ 3 V).[18] It is worth mentioning that the variances are virtually unaffected by the empirical correction, meaning that this approach strictly corrects the general deviations of 2TT in determining high-field mobilities but does not impact uncertainties specific to individual analytes.

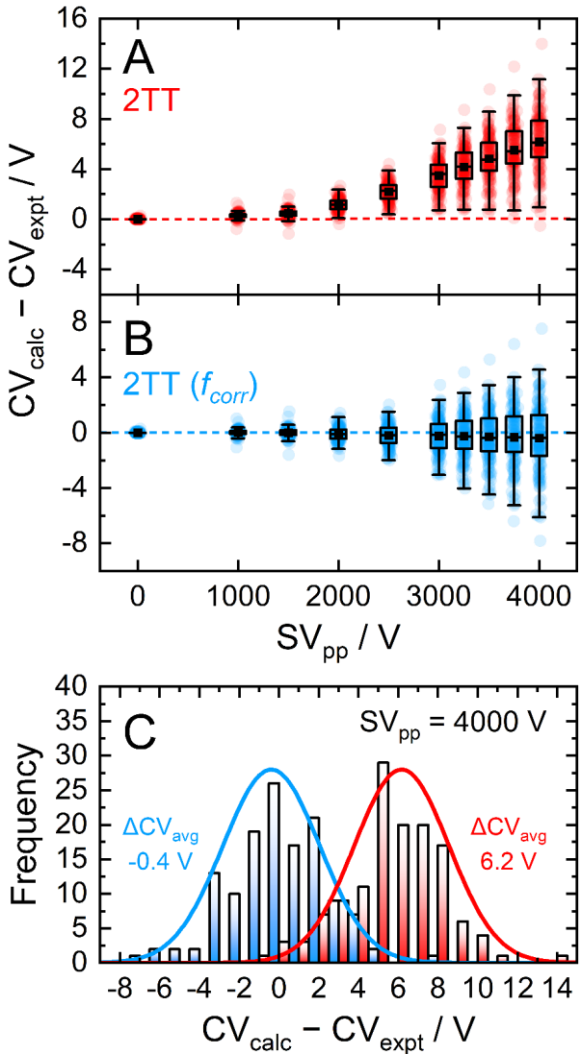


Figure 8. Comparison of measured vs. calculated dispersion plots over a range of separation field amplitudes as box plots for (**A**) uncorrected 2TT and (**B**) 2TT including the empirical correction employing A_{avg} and B_{avg}. Panel (**C**) shows the distribution of CV deviations using the uncorrected and empirically correct 2TT at the highest SV_{pp} (4000 V).

Although we derived the empirical correction based on fundamental deviations of 2TT at high fields,[67] it is not possible to deconvolute the error incurred from breakdown of 2TT at high-field strengths from other sources of error. For example, MobCal-MPI 2.0 treats all collisions elastically,[27,29,35] and as such, ignores the inherent inelasticity of ion-neutral collisions with molecular entities. Specifically, collision events do not consider the deposition of energy into the ion's rotational and vibrational degrees of freedom, both of which will influence the ion mobility, especially at high field strengths due to species-to-species differences in momentum transfer,[70] vibrational broadening,[71] and increased rotation of the ion during the collision event.[72] The complexity of this problem currently precludes the development of a computational framework that can model these effects, as simulating these effects requires exact knowledge of how internal energy is exchanged between the collision gas and the ion's rovibrational states. As advances are made in treating the inelasticity problem, we expect that the width of the distributions shown in **Figure 8C** will decrease.

2.8 Benchmarking the performance of MobCal-MPI 2.0

Having demonstrated the accuracy and precision with which MobCal-MPI 2.0 can model mobility data at arbitrary field strengths, we sought to benchmark code performance on a parallelized high-performance computing (HPC) architecture. Proper benchmarking requires the explicit consideration of parameters that effect CCS calculation times, which include the number of atoms within an analyte molecule and the number of HPC cores used. This was accomplished by monitoring the time taken to calculate the CCS for each conformer of the 50 species in the *validation set* ($N = 238$ unique conformers). Since trajectory method calculations are often used for short and medium length peptides, we opted to supplement the *validation set* with the *peptide set*, which comprises 12 peptides ranging in length from 9 to 22 amino acid residues that can adopt charge states between +2 to +5. Combining the *validation* and *peptide* sets yields 250 conformers that range in size from 16 to 374 atoms. The time required to calculate the CCS for each conformer at $T = 298$ K was measured using 4, 8, 16, and 32 cores in parallel with $itn = 10$, $inp = 96$, and $imp = 512$. The benchmarking process uses a different value for the inp parameter than the recommended setting of 104, as MobCal-MPI 2.0 requires inp (and imp) to be divisible by the number of cores utilized for parallel computing. Because 104 is not evenly divisible by 16 or 32, we chose the nearest integer that is evenly divisible by 4, 8, 16, and 32, which is 96. Additionally, it is important to consider that the random seed number used to determine the starting orientation of N_2 relative to the ion influences the runtimes. Because certain starting conditions may lead to "lost" trajectories (*i.e.,* those where the N_2 gets captured by the ion) calculation times can be artificially inflated. To account for this effect during benchmarking, we conduct CCS calculations using three random seeds and averaged the respective runtimes, ensuring that any increase in calculation time caused by lost trajectories from one seed will be averaged out.

The averaged runtimes for the 250 conformers as a function of the number of atoms are presented in **Figure 9A**. As expected, runtimes increase with an increasing number of atoms owing to the iterative nature with which the ion-neutral interaction potential is evaluated. Calculation times increase linearly with the number of atoms in the ion when the *validation set* and the *peptide set* are considered independently ($\mathcal{O}(N)$; see Section C9). However, extrapolation of the line of best fit determined from linear regression of the *validation set* indicates that the peptides do not conform to the same trend line, requiring slightly longer runtimes to complete. The increased runtimes for the peptides are likely a

consequence of their greater charge, which can induce the capture of N_2 molecules at low relative velocities. In other words, N_2 trajectories with small relative velocities are more susceptible to becoming "lost," and thus artificially increase the runtime. It is hard to gauge the exact behaviour of how runtimes might evolve for systems larger than those studied here, but we expect that the scaling of these times will always outperform $\mathcal{O}(N^2)$, where N is the number of atoms.

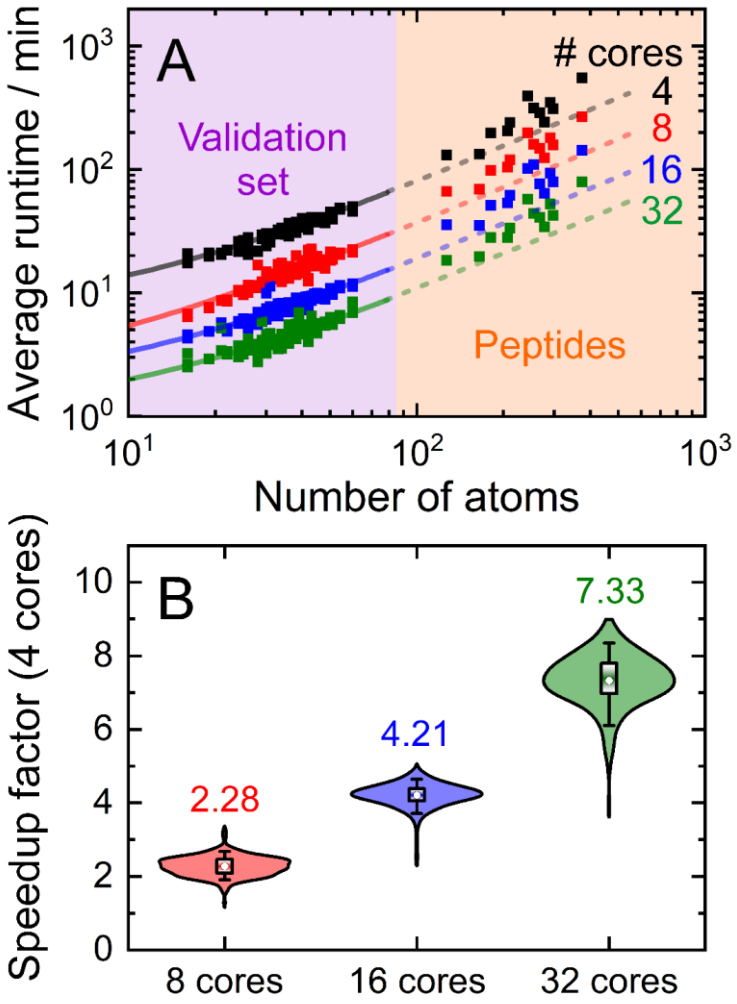


Figure 9. (**A**) MobCal-MPI 2.0 runtimes for CCS calculations (T = 298 K) of the validation set and 12 peptides as a function of the number of atoms. Calculations used 4 (black), 8 (red), 16 (blue), or 32 (green) cores in parallel. Linear regression is performed on the validation set, the lines for which are extrapolated to the peptides. (**B**) Violin plot showing the distribution of speedup factors from using 8, 16, or 32 cores compared to 4 cores. All runtimes are reported as the average from three CCS evaluations that employ different seed numbers on the same grid size (itn = 10, inp = 96, and imp = *512*).

Linear regression of the averaged runtimes for the validation set with respect to the number of atoms yields slopes that follow an inverse relationship with the number of cores. This indicates efficient code parallelization, where doubling the number of cores approximately halves the runtime. This trend can also be observed in **Figure 9B**, where the distributions of the individual speedup factors for all

conformers in the *validation* and *peptide set* are shown as violin plots. The averages of these distributions reflect the aforementioned speed increase, which is to be expected as the total number of collision events sampled is evenly distributed among all cores (*i.e.,* $itn \times inp \times imp$). Only for 32 cores do we observe a slightly lower average speedup than the expected value of 8 (as compared to runtime for 4 cores), suggesting the presence of a small parallel execution overhead. Naturally, the absolute runtimes heavily depend on the computing system used, so the runtimes reported here, which were assessed using 2nd generation AMD EPYC processers (AMD Rome 7532 @ 2.40 GHz), will vary from other systems. Nevertheless, calculation of "routine" small molecules (*i.e.,* those containing less than 70 atoms) complete in less than 30 minutes on 8 cores, highlighting the efficiency with which MobCal-MPI 2.0 can compute accurate CCSs to complement experimental results.

2.9 Additional Reading

1. An excellent review on prior methods and future directions for ion mobility calculations by Prof. Carlos Larriba-Andaluz and Prof. James S. Prell

   Larriba-Andaluz, Carlos., Prell, James S. Fundamentals of ion mobility in the free molecular regime. Interlacing the past, present and future of ion mobility calculations. *Int. Rev. Phys. Chem.* **2020**, 39 (4), 569-623. https://doi.org/10.1080/0144235X.2020.1826708

2. A benchmark performance of MobCal-MPI using several DFT hybrid functionals in conjunction with double or triple zeta basis sets augmented with polarization basis functions.

   Ieritano, C., & Hopkins, W. S. (2021). Assessing collision cross section calculations using MobCal-MPI with a variety of commonly used computational methods. *Materials Today Communications*, *27*, 102226. https://doi.org/10.1016/j.mtcomm.2021.102226

3. Another excellent review by Prof. James S. Prell on the use of molecular dynamics simulations in tandem with ion mobility calculations to compliment native IMS experiments.

   Rolland, A. D.; Prell, J. S. Computational Insights into Compaction of Gas-Phase Protein and Protein Complex Ions in Native Ion Mobility-Mass Spectrometry. *TrAC - Trends Anal. Chem.* **2019**, *116*, 282–291. https://doi.org/10.1016/j.trac.2019.04.023.

4. Two excellent papers demonstrating the importance of considering ion rotation when evaluating ion mobility *in silico*.

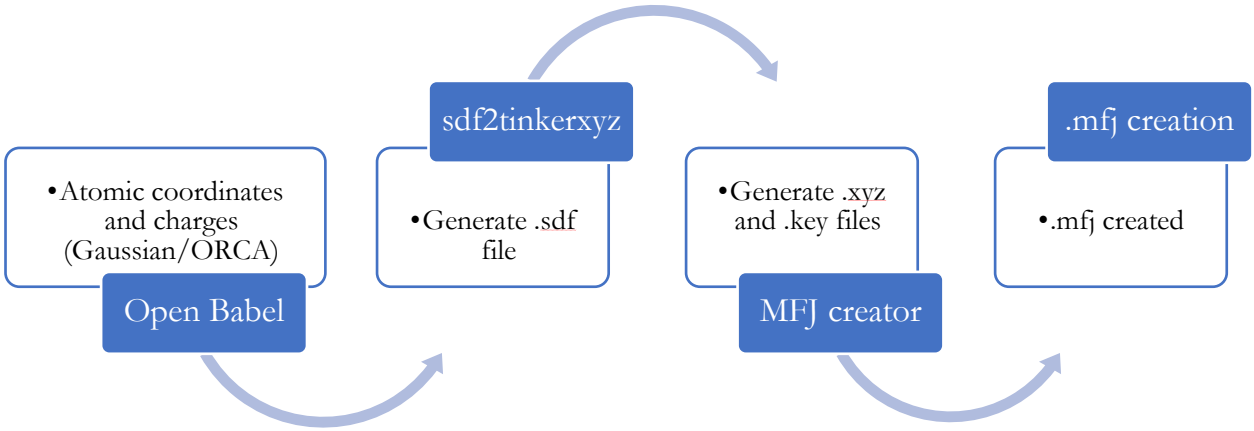
   Harrilal, C. P., Garimella, S. V. B., Chun, J., Devanathan, N., Zheng, X., Ibrahim, Y. M., Larriba-Andaluz, C., Schenter, G., & Smith, R. D. (2023). The Role of Ion Rotation in Ion Mobility: Ultrahigh-Precision Prediction of Ion Mobility Dependence on Ion Mass Distribution and Translational to Rotational Energy Transfer. *The Journal of Physical Chemistry A*. **2023**, *127* (25), 5458–5469. https://doi.org/10.1021/ACS.JPCA.3C01264

   Shvartsburg, A. A.; Mashkevich, S. V.; Siu, K. W. M. Incorporation of Thermal Rotation of Drifting Ions into Mobility Calculations: Drastic Effect for Heavier Buffer Gases. *J. Phys. Chem. A* **2000**, *104* (42), 9448–9453. https://doi.org/10.1021/jp001753a.

3. Installing the MobCal-MPI Suite Graphical User Interface (GUI)

MobCal-MPI includes a Graphical User Interface (GUI) to extend its functionality without requiring the use of the command prompt or Python environments like IDLE. This GUI incorporates subroutines that facilitate the following tasks:

1. Creating MobCal-MPI 2.0 input files (.mfj) from Gaussian output files (.log) or ORCA output files (.out), which follows this workflow:



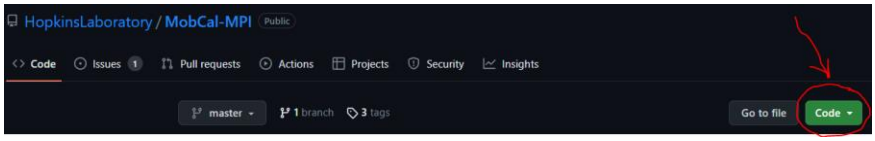
2. Analyzing MobCal-MPI 2.0 output files (.mout) with built-in plotting capabilities.
3. Batch extraction of CCSs and mobility data from multiple .mout files.

Because the GUI requires external programs link OpenBabel, Python, and SDF2XYZ2SDF, these must be configured on your machine before the GUI can be used. The installation procedure for each package is provided in the sections below.

3.1 Downloading the Code and Installing Required Packages

3.1.1 MobCal-MPI and GUI

Retrieve the MobCal-MPI 2.0 code and its accompanying GUI by accessing the GitHub repository. To do this, click on the "Code" button, and then download the ZIP version from the following link. Extract the .zip file to a convenient location on your computer. Choose somewhere that is accessible, as the GUI will be launch from this directory.



3.1.2 OpenBabel 2.4.1

The GUI is designed to work with OpenBabel version 2.4.1, which is a program that converts between file formats that are commonly by computational chemistry programs.[73] You can download OpenBabel 2.4.1 from the following link.

We strongly recommend using OpenBabel v2.4.1, as there have been syntax changes in later releases that affect the conversion of ORCA .out files. To install OpenBabel 2.4.1, simply download the OpenBabel-2.4.1.exe from the link above, follow the on-screen instructions, and install it in the **default directory**.

3.1.3 SDF2XYZ2SDF

The MobCal-MPI 2.0 GUI requires the SDF2XYZ2SDF program to convert .sdf files into formatted .xyz files. This conversion process involves taking .sdf files created by OpenBabel, which include optimized molecular structures obtained from Gaussian or ORCA outputs. These .sdf files are transformed into tinker XYZ files, which include atom type information according to the MM3 forcefield. You can find additional information regarding the functionality of this program and the tinker XYZ format in the following publication:

P. Tosco and T. Balle, Journal of Molecular Modeling, 2011, 17, 3021-3023.

To obtain the SDF2XYZ2SDF program, please visit the following link and download sdf2xyz2sdf-1.05_windows_setup.exe. To install SDF2XYZ2SDF, simply follow the on-screen instructions after running the executable, ensuring it gets installed it in the **default directory**.

3.1.4 Adding Babel and SDF2XYZ2SDF directories to PATH

To enable the GUI to detect OpenBabel and SDF2XYZ2SDF, they should be accessible within your system's PATH. To add the programs to your systems PATH, follow these instructions:

1. In the Windows Search bar, type "Environment Variables."
2. Select the "Edit the system environment variables" option from the search results.
3. At the bottom of the System Properties window, click the "Environment Variables..." button.
4. In the "Environment Variables" window, locate the "Path" variable in the top box, then click the "Edit..." button.
5. To add the necessary entries, click the "New" button and include the following two entries:
    a. C:\Program Files\OpenBabel-2.4.1
    b. C:\open3dtools\bin
6. Ensure that you can find "obabel.exe" and "babel.exe" in the directory C:\Program Files\OpenBabel-2.4.1.
7. Verify that "sdf2tinkerxyz.exe" is located in the C:\open3dtools\bin directory.

3.2 Installing Python and the Necessary Python Libraries on Your Computer

3.2.1 Python 3.12

The GUI is fully compatible with Python 3.12+ releases; compatibility with prior Python versions is not guaranteed. You can obtain your preferred version from https://www.python.org/downloads/. During the installation process, please make sure to select the radio button labeled 'Add Python 3.X to PATH'; you will find it at the bottom of the installer window. Additionally, opt for a custom installation of Python, which allows you to specify the installation directory. A convenient location for this purpose could be C:/Python312 (for Python 3.12).

3.2.2 GitHub Desktop and Git

The MobCal-MPI 2.0 GUI features an automatic update functionality that necessitates the installation of GitHub Desktop on your computer. To get started, simply download GitHub Desktop and Git from the links provided. During the installation process, follow to the on-screen instructions and ensure that you install the application in the default directory suggested by the installer.

Please be aware that having a GitHub account is not necessary for installing GitHub Desktop or Git. However, completing these installations must be done before moving on to Section 3.2.3; failing to do so will result in an unsuccessful attempt to import git.

3.2.3 Python site packages

The MobCal-MPI Suite GUI relies on the numpy, scipy, matplotlib, PyQt6, and git libraries. To install them, follow these steps:

1. Launch the command prompt (or the equivalent terminal for Mac/Linux). Windows users can do this by typing "cmd" into the search bar.
2. Type the following command:

   pip install numpy pyqt6 scipy matplotlib gitpython

Please be aware that running either of these commands may trigger a prompt to update pip. This is normal.

If users receive a message that "pip" is not recognized, it means that pip (Python's package installer) is not in the system's PATH, and the command prompt cannot find it. To resolve this issue, you can follow these steps:

1. **Locate the Path to Python Scripts:**
   - First, identify the location of the Python Scripts directory. It should be something like C:\Users\YourUsername\AppData\Local\Programs\Python\Python3X\Scripts

where YourUsername is your Windows username and Python3X is the Python version you have installed).

2. **Add Python Scripts to PATH:**
    - Open the Windows search bar and type "Environment Variables."
    - Click on "Edit the system environment variables."
    - In the "System Properties" window, click the "Environment Variables" button.
    - Under "System Variables," scroll down to find and select "Path," then click "Edit."
    - Click "New" and add the path to the Python Scripts directory (e.g., C:\Users\YourUsername\AppData\Local\Programs\Python\Python3X\Scripts).
    - Click "OK" to save the changes.
3. **Restart the Command Prompt:**
    - Close and reopen the command prompt (cmd) or terminal window.

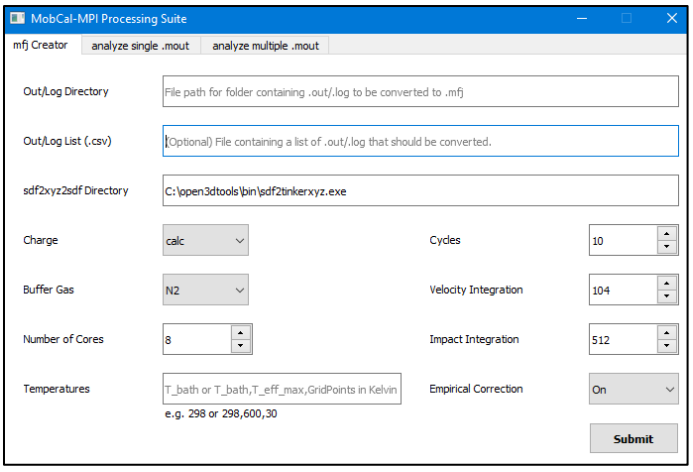
4. **Retry the Installation:**
    - Run the installation commands for numpy and PyQt6 again:

After following these steps, the "pip" command should be recognized, and you should be able to install the required libraries without any issues.

3.3 Verifying installation

To initiate the MobCal-MPI Suite GUI, navigate to the \GUI folder found within the files downloaded from the GitHub repository, and launch the Launcher.py module through your preferred Python environment. In case any necessary libraries are absent during execution, the user will receive prompts to install them. Any identified bugs or suggested improvements for the GUI should be reported in the "Issues" section of the GitHub repository.

Upon successful installation of all the required packages, users will encounter the following interface. Note that on the first instance of running the GUI, a popup will appear prompting the user to update the code (even if it was just cloned from the GitHub repository). This is normal; please click yes and allow the update to run.



3.4 Using the MobCal-MPI Suite GUI

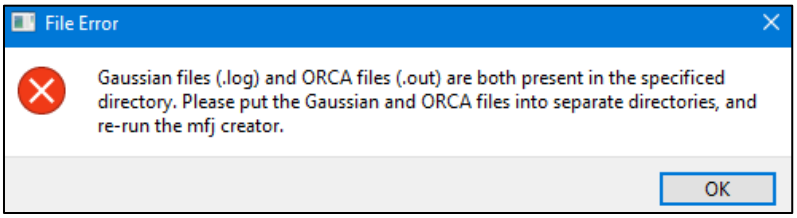
3.4.1 mfj Creator

The "mfj Creator" tool streamlines the process of generating MobCal-MPI 2.0 input files (.mfj) from Gaussian (.log) or ORCA (.out) files. For specific instructions on configuring Gaussian or ORCA calculations to produce output files that can be converted into MobCal-MPI 2.0 inputs, please refer to Appendix A.

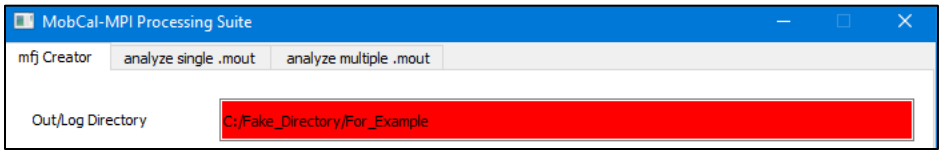
In the "mfj Creator," every field can be tailored to your preferences. Here's an overview of what each field's purpose is:

Out/Log Directory: This field corresponds to the directory where the .log files or .out files for conversion are located. It's important to note that the directory should exclusively contain outputs from one program, meaning it should contain either only Gaussian .log files or only ORCA .out files.

If both .log and .out files are present within this directory, users will encounter the following error message:



If the specified directory does not exist on the user's machine, the dialog box will be highlighted in red. For instance:



Out/Log List (.csv): This is an optional field where you can input the name of a .csv file that contains a list of .log files to be converted into .mfj files. The .csv file should be present in the *Out/Log Directory* specified in the previous field.

Including the Out/Log List can be particularly helpful for excluding high-energy isomers from CCS calculations without the need to relocate files. A sample .csv file is accessible in the "GUI_Samples_Files" directory of the MobCal-MPI 2.0 release, named "Test_CSV_List.csv."

sdf2xyz2sdf Directory: In case sdf2xyz2sdf was not installed in the default directory, you can indicate its specific location here. By default, the installation directory is pre-filled. However, if the sdf2tinkerxyz.exe file is not located in the default directory, you should specify its location in this field.

If you installed sdf2xyz2sdf in the default directory, there's no need to make any alterations in this section, but you will need to ensure that this directory is found in your system's PATH.

Charge: This variable allows you to specify the partial charge assignment scheme. There are three available options: "Calc," "Equal," and "None."

charge = 'calc' (Recommended)
This selection indicates that a unique partial charge scheme will be utilized. Currently, the MFJ creator can extract charges predicted by electrostatic potential mapping methods directly from the Gaussian and ORCA output files. For more details, refer to Appendix A.

charge = 'equal' (Not recommended)
With this choice, equivalent partial charges are assigned to each atom based on the relationship:

$$partial\ charge\ =\ (ion\ charge)\ /\ (number\ of\ atoms)$$

charge = 'none' (Not recommended)
Selecting this option assigns zero charges to all atoms, effectively eliminating the ion-induced dipole and ion-quadrupole terms for the ion-gas potential.

It's important to note that regardless of the charge scheme used, this script is designed to work with Gaussian .log/ORCA .out files that contain partial charges calculated using the MK or ChelpG schemes. If a .log/.out file lacks partial charges, the .mfj creator will not execute, regardless of the charge scheme selected. Furthermore, forcefields used to evaluate ion-neutral collision trajectories were parameterized using the 'calc' option, meaning that **using either the 'equal' or 'none' options will result in incorrect CCSs/mobilities.**

Buffer Gas: MobCal-MPI is parameterized for CCS calculations in N_2 and He. Here, you can specify the buffer gas that you want to perform your CCS calculations in.

Temperature List: In MobCal-MPI version 2.0, it's possible to compute CCS values at arbitrary field strengths. The field strength is determined based on the ion's effective temperature, which is specified within the temperature list.

1) For zero-field CCS calculations, which are analogous to typical CCS calculations in MobCal-MPI 1.2, users should enter a single temperature value representing the bath gas temperature of their experiment as an integer in Kelvin. For example:
   - 298

2) For CCS calculations at arbitrary field strengths, users should provide three comma-separated values in this dialog box. Each value corresponds to (in order of appearance):
   - The bath gas temperature of the mobility experiment (T_{bath})
   - The maximum effective ion temperature the user wishes to evaluate to ($T_{eff,max}$)
   - The number of "grid points" between T_{bath} and $T_{eff,max}$. **Note that a maximum value of 99 grid points can be specified.**

For instance, if a user wants to calculate the field-dependent mobility of an ion at T_{bath} = 298 K in increments of 50 K up to 798 K, the temperature list would be as follows (no spaces between entries):

- 298,798,10

Completing this setup will generate the following output for Amifostine_3.mfj, a conformer of Amifostine available in the GUI sample files (Gaussian outputs). It's important to highlight that the CCS value at 298 K corresponds to the zero-field CCS, while the CCS values at higher effective temperatures are calculated using the second-order correction within the framework of two-temperature theory.

**********Mobility Summary**********
**********at T_bath= 298.00K**********

Teff [K]	E/N [Td]	K0 [cm**2/Vs]	CCS [A**2]	uncertainty
298.00	0.00	1.5470	139.22	1.32%
348.00	51.06	1.5472	129.51	1.28%
398.00	72.67	1.5491	121.78	1.25%
448.00	89.65	1.5491	115.51	1.23%
498.00	104.36	1.5475	110.33	1.20%
548.00	117.68	1.5446	105.98	1.18%
598.00	130.06	1.5405	102.29	1.16%
648.00	141.78	1.5353	99.13	1.14%
698.00	152.98	1.5293	96.38	1.13%
748.00	163.78	1.5226	93.97	1.11%
798.00	174.26	1.5153	91.85	1.10%

Number of cores: This setting determines the number of CPU cores to be utilized for MobCal-MPI calculations. It's worth noting that this variable does not impact the generation of .mfj input files; instead, it's intended for error handling purposes. The number of cores requested should be divisible by the product of the trajectory parameters (itn *x* inp *x* imp). This ensures that the workload is evenly distributed across the specified number of cores.

The recommended setting for the number of cores is 8 or 16, both of which are compatible with the recommended values of *itn*, *inp*, and *imp* below. For a more in-depth analysis of how CCS calculation times scale with the number of cores employed, we refer you to Section 2.8.

Cycles (*itn*): This parameter defines the number of cycles that MobCal-MPI will execute for CCS calculations. The resulting CCS value will be computed from all trajectories sampled in each *itn* cycle.

The recommended number of cycles for CCS calculations, whether at zero-field or arbitrary field strengths, is **10**. For details on how the number of *itn* cycles influences the uncertainty of a calculated CCS, please refer to Sections 2.5 and 2.6.

Velocity Integration (*inp*): This setting specifies the number of relative velocities of the buffer gas to be sampled during each *itn* cycle. For zero-field CCS calculations (*i.e.,* those that specify T_{bath} only), the recommended value is **104**. For CCS calculations at arbitrary field strengths (*i.e.,* those that specify T_{bath}, $T_{eff,max}$, and a grid size), the recommended value is **200**. For details on how the number of *inp* integration points influences the uncertainty of a calculated CCS, please refer to Sections 2.5 and 2.6.

Impact Integration (*imp*): This setting determines the number of collisions to sample for each relative velocity of the colliding neutral gas (*inp*). For molecules of moderate size (< 400 Da), **512** points are recommended. However, for larger molecules, it is recommended to increase the number of collisions sampled to minimize the variance of the CCS. In such cases, we suggest using either **768 or 1024** points. For details on how the number of *imp* integration points influences the uncertainty of a calculated CCS, please refer to Sections 2.5 and 2.6.

Empirical Correction: This setting determines whether the empirical correction to two-temperature theory should be applied to mobility calculations (**On**), or if the uncorrected version of two-temperature theory should be used (**Off**). For details on the empirical correction, see Section 2.7.

After entering all the parameters, click the 'Submit' button to initiate the conversion process. The code will systematically process files in alphabetical order, converting Gaussian/ORCA output files into MobCal-MPI input files. Upon successful execution of the MFJ creator workflow, a series of .mfj input files will be generated with the following format:

```
AMIFOSTINE_3 <-- Title
1    <-- Number of coordinate sets (always 1)
28   <-- Number of atoms
ang  <-- Units for atomic coordinates (ang = Ångstrom; a.u = atomic units)
calc <-- Charge distribution (calc = charges from ESP; equal = equal charges for all atoms; none = no atomic partial charges)
1    <-- Empirical correction to mobility (0 = off; 1 = on)
10 104 512 2 -447950396 298 798 10 <-- (10: itn) (104: inp) (512: imp) (gas; 1 = He, 2 = N2) (seed number) (298 = Tbath) (798 = Teff,max) (10 = no. of temp grid points)
```

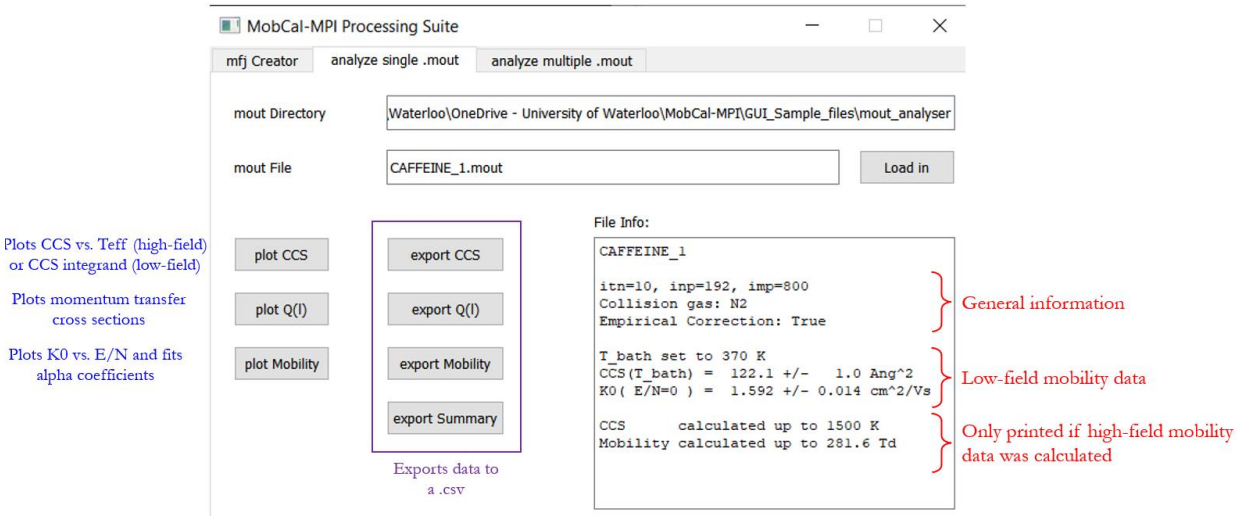
xyz coordinates			Atomic mass	Partial charges	MMFF94 vdW parameters			
1.290700	0.833300	-1.273600	31.972	-0.193666	3.00	4.800	3.320	1.345
1.835900	-0.567900	0.215300	30.974	0.928334	1.600	4.500	3.320	1.345
1.601500	0.228400	1.605900	15.995	-0.608579	0.70	3.150	3.890	1.282
3.441700	-0.700200	0.177400	15.995	-0.621605	0.70	3.150	3.890	1.282
1.123800	-1.872800	0.078400	15.995	-0.623803	0.75	3.150	3.890	1.282
-1.323000	0.767800	0.692000	14.003	-0.039906	1.15	2.820	3.890	1.282
-1.458700	-1.483100	-0.811300	14.003	-0.302638	1.00	2.820	3.890	1.282
-2.757700	0.544900	1.014800	12.000	-0.136973	1.050	2.490	3.890	1.282
-3.480200	-0.274000	-0.071900	12.000	-0.019077	1.050	2.490	3.890	1.282
-1.054200	1.971400	-0.111000	12.000	-0.040526	1.050	2.490	3.890	1.282
0.439900	2.194000	-0.338400	12.000	-0.014244	1.050	2.490	3.890	1.282
-2.860900	-1.659300	-0.288600	12.000	0.095374	1.050	2.490	3.890	1.282
-2.799000	-0.005900	1.960400	1.008	0.091435	0.250	0.800	4.200	1.209
-3.284100	1.496100	1.171900	1.008	0.080214	0.250	0.800	4.200	1.209
-3.500200	0.275800	-1.022000	1.008	0.046225	0.250	0.800	4.200	1.209
-4.525400	-0.406500	0.225100	1.008	0.065223	0.250	0.800	4.200	1.209
-1.458100	2.875500	0.371900	1.008	0.042858	0.250	0.800	4.200	1.209
-1.565300	1.866600	-1.074800	1.008	0.082288	0.250	0.800	4.200	1.209
0.974400	2.358200	0.598800	1.008	0.076472	0.250	0.800	4.200	1.209
0.589100	3.075000	-0.968200	1.008	0.131005	0.250	0.800	4.200	1.209
-3.432300	-2.270100	-0.990200	1.008	0.053687	0.250	0.800	4.200	1.209
-2.779200	-2.204400	0.655600	1.008	0.072372	0.250	0.800	4.200	1.209
-1.429400	-1.306500	-1.816500	1.008	0.278131	0.150	0.800	4.200	1.209
-0.795400	-2.237000	-0.596000	1.008	0.326918	0.150	0.800	4.200	1.209
2.343200	0.155300	2.228600	1.008	0.491602	0.150	0.800	4.200	1.209
3.745900	-1.567900	-0.135000	1.008	0.495074	0.150	0.800	4.200	1.209
-0.776100	0.804000	1.549800	1.008	0.178081	0.150	0.800	4.200	1.209
-1.083700	-0.620300	-0.299900	1.008	0.065725	0.150	0.800	4.200	1.209

3.4.2 Analyze single .mout

The "*Analyze single .mout*" module contains several features that are useful for a detailed analysis of any CCS calculation. To run this module, specify the directory that contains the .mout file(s) you wish to analyze in the "mout Directory" dialog box, then specify the name of the .mout file within the "mout File" dialog box. Finally, click "Load in" to see a summary of the file. Note that if the "mout Directory" does not exist, the respective dialog box will be highlighted in red.

Once the "Load in" option has been selected, users will see a printout containing relevant information for the respective MobCal-MPI 2.0 output. For example, this is the printout for

AMIFOSTINE_3_SingleTemp.mout, which is provided in the GUI sample files in the Single_mout_analyzer subdirectory. Upon the successful read-in of a .mout file, the *File Info* dialog box will become populated with relevant information from the output.

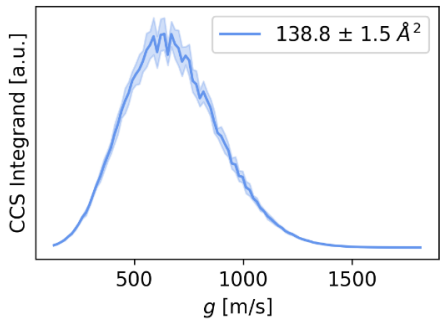


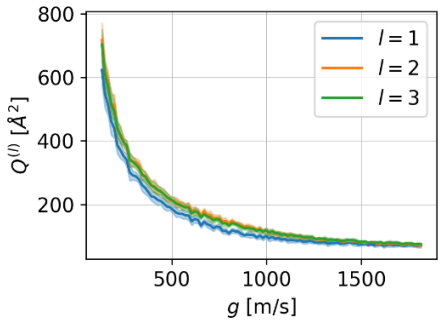
The *Analyze single .mout* module is capable of processing jobs performed at a single bath gas temperature, as well as when a $T_{eff,max}$ and temperature grid is specified. Depending on the type of job being analyzed, the printouts and options available to the users are slightly different, so this manual will cover the options available for both cases.

3.4.2.1 MobCal-MPI jobs conducted at a single bath gas temperature

In the case that a single temperature is specified (*i.e.,* T_{bath} only), the two-temperature theory module is not called, and hence, no high-field mobility data will be calculated. Consequently, only the zero-field CCS and mobility are calculated, thus negating the need to export the field-dependent CCS and mobility data.

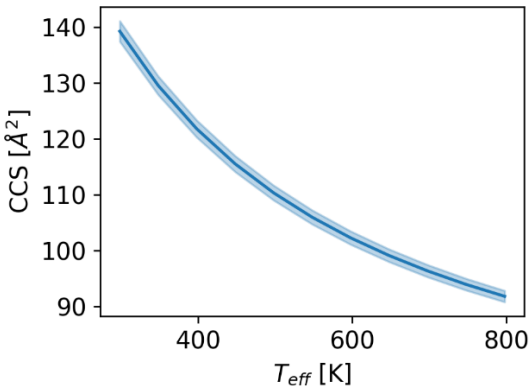
The CCS and momentum transfer integrals that determine the zero-field CCS (see Sections 2.2 and 2.3) can be visualized by the *plot CCS* and *plot Q(l)* options, respectively. For Amifostine_3_SingleTemp.mout, the plots generated are shown below. The raw data corresponding to the momentum transfer data for $l = 1$, 2, and 3 can be saved as a .csv by selecting the *export Q(l)* option. Finally, a .csv containing a summary of the zero-field CCS and mobility can be obtained using the *export summary* option.

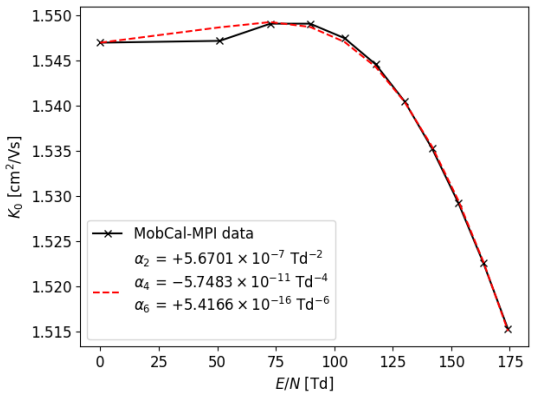




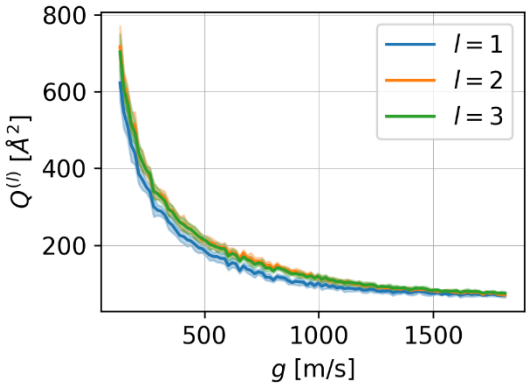
3.4.2.1 MobCal-MPI jobs that specify multiple temperatures (i.e., T_{bath} and $T_{eff,max}$)

In the case that high-field mobility data is available (*i.e.,* the MobCal-MPI job was provided a T_{bath}, $T_{eff,max}$, and temperature grid size), then all options within the *Analyze single .mout* module will become available. In this case, selecting the *plot CCS* option yields the evolution of CCS with T_{eff} instead of the CCS integrand. The *plot mobility* option also becomes available, producing a plot of mobility as a function of field strength and fitting the raw data to the first three alpha coefficient (a_2, a_4, and a_6; see **Eq. 6**). Sample plots for Amifostine_3_MultiTemp.mout are shown below.





As was the case with MobCal-MPI jobs specifying a single temperature, users can also visualize the momentum transfer cross sections $\Omega^{(l)}$ for $l = 1$, 2, and 3.



The data corresponding to each figure can be exported as a .csv using the respective option. A .csv containing a summary of the field-dependent CCS and mobility can be obtained using the *export summary* option. Note that the field strength is determined by the effective temperature of the ion via two-temperature theory.

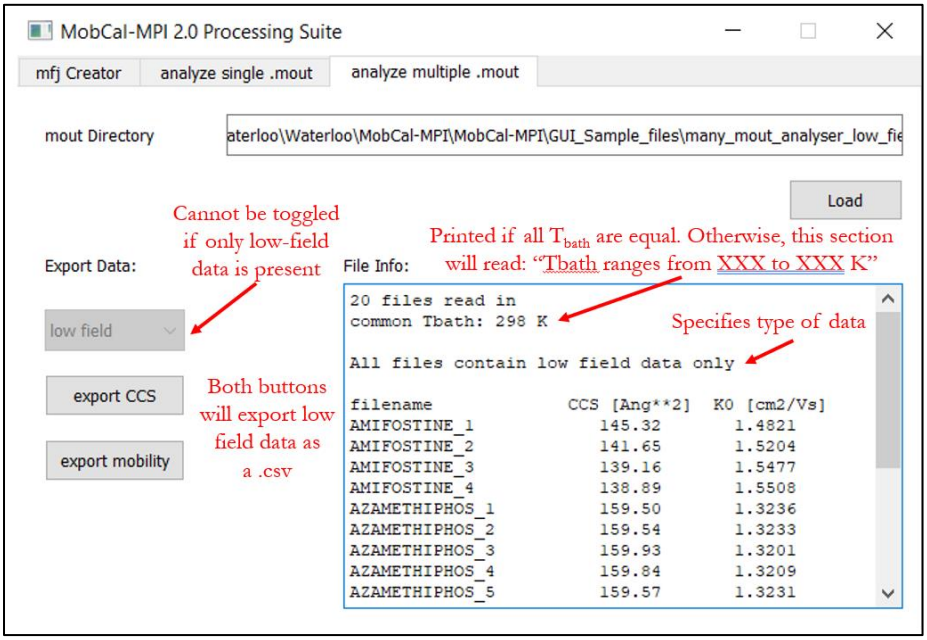
3.4.3 Analyze multiple .mout

The "*Analyze multiple .mout*" module contains several useful features for analyzing groups of CCS calculations (for example, a conformational ensemble of a single molecule or extracting the CCSs of several different molecules). To run this module, specify the directory that contains the .mout files

you wish to analyze in the *mout Directory* dialog box, then select "Load" to see a summary of the files. Note that if the *mout Directory* does not exist, the respective dialog box will be highlighted in red. Once a directory has been successfully loaded in, the *File Info* dialog box will become populated. The dialog box will print one of three options depending on the types of jobs that were conducted:

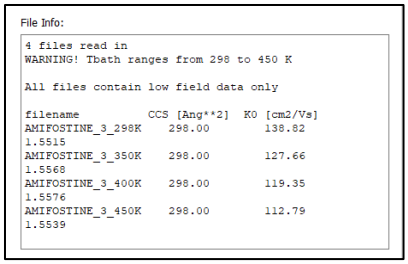
1) All .mout files within the *mout directory* correspond to low-field jobs only.

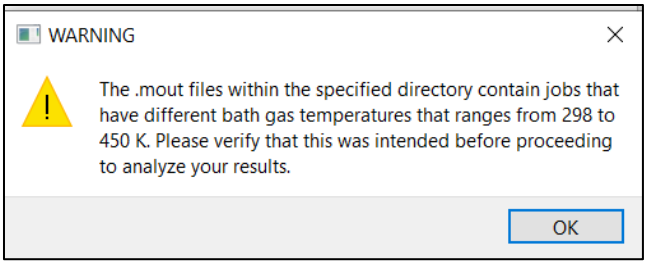
If all MobCal-MPI input files were provided a single temperature (*i.e.,* T_{bath}), then all .mout files will contain information pertaining to the ion's low-field mobility. The sample files provided in the *many_mout_analyzer_low_field* subdirectory fit this criterion; the successful read-in of the files will be met with the following interface:



From here, users can export .csv files that contain the calculated CCS (and its uncertainty) of each ion within the *mout directory* via the *export CCS* option, or the calculated mobility (and its uncertainty) via the *export mobility* option. The T_{bath} in which the MobCal-MPI job was conducted at will also be printed to the .csv.

We note that users may desire to evaluate the low-field CCS of an analyte at several bath gas temperatures. However, this may also be an erroneous combination of files, so to ensure users to not unintentionally mix up MobCal-MPI jobs conducted at different T_{bath}, the following warning message will appear if the *mout directory* contains .mout files that contain different T_{bath}. The file info dialog box will also contain a similar warning. Despite the warnings, the *export CCS* and *export mobility* features are fully functional.

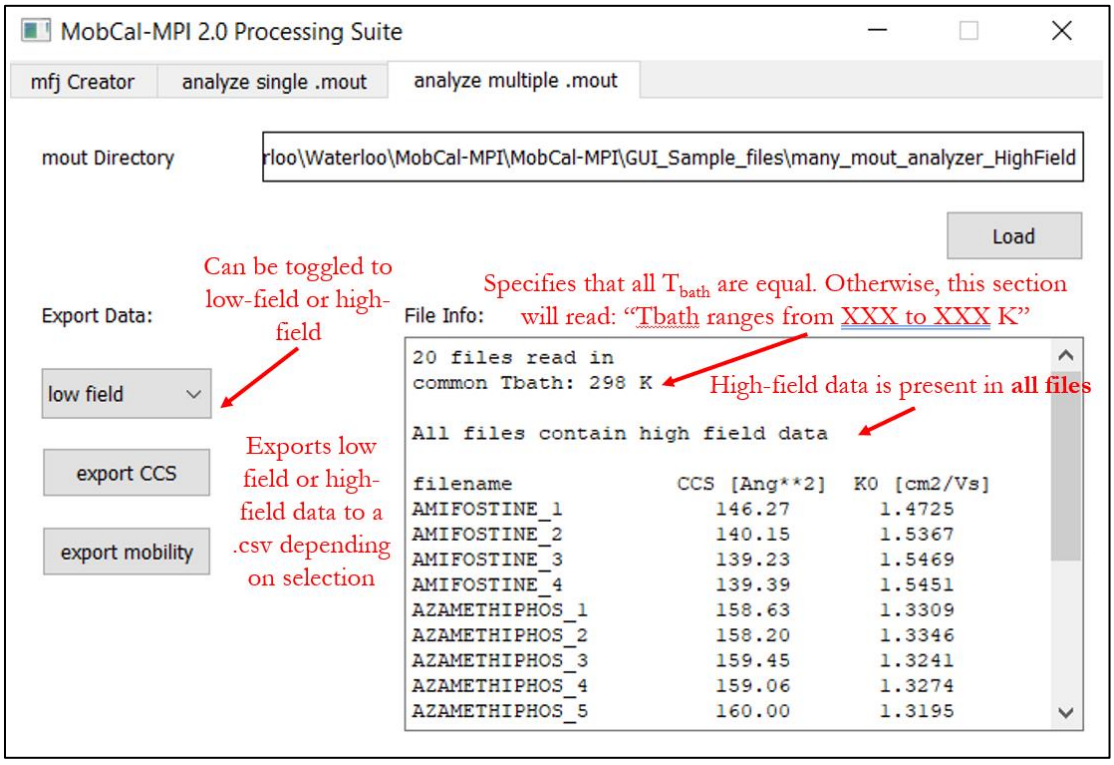




Once the .mout files are successfully read in, users can export the low-field CCSs and mobilities using the respective buttons. Clicking the button will produce a .csv file. For examples of these outputs, please see the files found within the *many_mout_analyzer_LowField* (common T_{bath}) and *many_mout_analyzer_LowField_multiple_Tbath* (different T_{bath}) subdirectories.

2) All .mout files within the *mout directory* correspond to high-field jobs only.

If all MobCal-MPI input files were provided with a T_{bath}, $T_{eff,max}$, and temperature grid, then all .mout files will contain information pertaining to how the ion's mobility evolves when exposed to an external field. In other words, the .mout files contain both low-field and high-field data. As before, if all files contain a common T_{bath}, the second line of the *File Info* dialog box will state the common T_{bath}. Otherwise, the second line will state the range of T_{bath} in which calculations were performed at.



From here, users can export the CCSs and mobilities found within each .mout file in the *mout directory*. Note that users can opt to export **only** the low-field data (*i.e.,* CCS and mobility at the

specified T_{bath}) by selecting the *low field* option under *Export Data*, or they can export the evolution of CCS and mobility with field strength by selecting the *high field* option under *Export Data*. Example printouts and .mout files are provided in the *many_mout_analyzer_HighField* subdirectory.

For users conducting high-field mobility calculations, we recommend that they set up the T_{eff} grid to be the same across all analytes. In other words, all .mfj input files should specify the same T_{bath}, $T_{eff,max}$, and temperature grid size. However, we recognize that situations may arise when unequal temperature grids may be desired, or users may inadvertently provide files with unequal temperature grids and/or bath gas temperatures. In either case, the user will be prompted with a message stating this as both a pop-up and within the *file info* dialog box. This is expected behaviour and does not impact the functionality of any export options.

Should the *mout directory* contain files that were either calculated at different bath gas temperatures or exhibit different temperature grids, an interpolation between the lowest and highest **common** T_{eff} (and the associated field strength) will be done. Subsequently, CCSs and mobilities will be reevaluated on the interpolated grid for consistency amongst files. Note that we provide this option for functionality only – if a user wishes to compute high-field mobilities of analytes, we recommend that a consistent temperature grid is employed. For an example on how the output .csv files look for inconsistent temperature grids, please see the sample files provided in the *many_mout_analyzer_HighField_multiTbath* subdirectory.

3) .mout files within the *mout directory* correspond to both low-field and high-field jobs.

Finally, the *analyze_multiple_mout* module will also work on directories that contain MobCal-MPI jobs that contain both low-field and high-field jobs (*i.e.,* some jobs provide only a T_{bath}, and others provide T_{bath}, $T_{eff,max}$, and a temperature grid). In this case, the user is only able to extract the low-field data from all .mout files found within the *mout directory*. We advise the user to separate the low-field and high-field jobs into separate directories so that the high-field data can be analyzed accordingly.

Note that specifying a *mout directory* that contains "mixed" high-field and low-field jobs will prompt the user with a warning message. This error message is purely for user awareness; the user will still be able to proceed to analyze the low-field data. For an example of this behaviour and to see what the output looks like, please use the files found within the *many_mout_analyser_mixed* subdirectory.

4. Running CCS Calculations in MobCal-MPI

4.1 Compiling MobCal-MPI

4.1.1 Compiler version

MobCal-MPI is coded in Fortran and necessitates the use of Fortran compilers. Further, as MobCal-MPI uses the Message-Passing Interface (MPI) for parallel computing, an MPI environment is needed. Publicly funded high-performance computing (HPC) systems are usually pre-equipped with Fortran compilers, MPI environments, and their associated documentation. We recommend using MobCal-MPI on such systems, although we understand that not all users may have access to HPC infrastructure. Hence, we've decided to offer users a brief introduction and guidelines on configuring Fortran compilers on local UNIX systems.

Firstly, there are two common Fortran compilers, *ifort* developed by Intel, and *gfortran* from the freely available GNU project. To compile code usable within an MPI framework, wrapper tools have been developed that use the underlying Fortran compilers to interface them with the local MPI environment. Depending on the MPI structure, different wrappers exist. IntelMPI provides the mpiifort wrapper, whereas the freely available OpenMPI framework comes with the mpifort wrapper. Note, however, that both can be linked to any underlying Fortran compiler (for some examples, see here or here). The Fortran complier that is currently linked to the wrapper can be checked using the following command:

```
$ mpifort --version
ifort (IFORT) 19.1.1.217 20200306          # linked Fortran compiler and version
```

Users with access to high performance computing (HPC) systems will likely have the Fortran compilers pre-configured, although depending on your system's configuration, the compilers and wrappers may need to be loaded in as modules. We refer users with access to HPC systems to Section 4.1.2. For users without access to externally managed HPC architecture, they are still able to perform MobCal-MPI calculations using a local UNIX machine. We recommend the Intel OneAPI Toolkit, which is equipped with the *mpiifort* compiler. If OpenAPI does not work for some users, Section 4.1.4 details the process of installing *gfortran* Fortran compilers, setting up OpenMPI, and using these packages to compile MobCal-MPI.

We do note that the Intel Fortran compiler offers a much greater functionality but is only available commercially. Importantly, this allows for speedups of a factor of 2-3 as compared to the GNU compiler as different optimization flags can be used during compilation (see below and this issue on the GitHub page). We further note that users have successfully used the Intel OneAPI Toolkit to compile and run MobCal-MPI as well.

For user reference, our testing and validation of MobCal-MPI 2.0 was conducted on Digital Research Alliance of Canada's clusters (https://docs.alliancecan.ca/wiki/MPI) using mpifort linked to the Intel *ifort* compiler.

4.1.2 Compilation of MobCal-MPI on HPC systems (recommended over standalone systems)

If you have access to a high-performance computing (HPC) system, Fortran compilers and MPI environments should be pre-configured. If you find that they are not, please reach out to your system's administrator for assistance.

Additionally, when working within the HPC environment, we recommend that users check for the availability of *ifort* due to its notable performance advantages over *gfortran*. Note that the process of calling *mpifort*/*mpiifort* can vary depending on the HPC infrastructure, but typically involves loading in the *intel* or *gcc* modules. For instructions on how to do this, consult your system's documentation. Once you have the respective modules loaded, you should be able to check the version of linked Fortran compiler by using the commands:

```
$ mpifort --version
$ mpiifort --version
```

If *ifort* is installed, the commands above will provide the version number of *mpifort*/*mpiifort*. The printout should look something like this, although your version number will likely be different from the one shown below:

```
ifort (IFORT) 19.1.1.217 20200306
```

A successful call to *mpifort*/*mpiifort* means that MobCal-MPI can be compiled using the appropriate command:

For *mpifort*:

```
$ mpifort -o MobCal_MPI_201.exe MobCal_MPI_201.f -Ofast
```

For *mpiifort*:

```
$ mpiifort -o MobCal_MPI_201.exe MobCal_MPI_201.f -Ofast
```

If *gfortran* (*i.e.,* GNU Fortran (GCC)) is installed, the *mpifort* --version command will produce something like:

```
$ mpifort –version
GNU Fortran (GCC) 12.2.0
```

If GNU Fortran (GCC) fortran is installed, you **must** compile MobCal-MPI using the following command, as the -Ofast optimization flag is not compatible with GCC compilers.

```
$ mpifort -o MobCal_MPI_201.exe MobCal_MPI_201.f -Og
```

Successful compilation of MobCal-MPI should result in the creation of a file named MobCal_MPI_201.exe with no error messages appearing in the terminal. If error messages are found, please report them to the issues section of the GitHub repository along with the details of how you compiled MobCal-MPI and your compiler version number. Please ensure that you place the MobCal_MPI_201.exe either in your system's *bin* or in the directory that contains the .mfj files for which you intend to calculate CCSs.

4.1.3 Compilation of MobCal-MPI on local UNIX systems using Intel oneAPI and *mpiifort* (recommended over OpenMPI)

If you're beginning from scratch on a local UNIX machine, we recommend installing the Intel oneAPI toolkit, which includes a free version of *mpiifort*. Specific instructions for configuring oneAPI on Debian (and Debian-like) Linux systems, as well as Arch Linux, can be found in the Quick Guide for installing Intel OneAPI to compile Fortran code. The .pdf of this guide can be found in the /Manual subdirectory on the primary MobCal-MPI GitHub page.

4.1.4 Compilation of MobCal-MPI on local UNIX systems using OpenMPI and gfortran (only use this if 4.1.2 and 4.1.3 are not available)

If the workflows described in 4.1.2 and 4.1.3 are not applicable and you are using a standalone UNIX system, you can compile MobCal-MPI using OpenMPI. You'll first need to install some essentials. *Sudo* privileges are needed for this, which can be granted by your system administrator. If you have *sudo* privileges, you can install the necessary packages using the following commands:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install build-essential
```

Setting up the essentials installs several useful packages, including the GNU compilers needed for the next step: installing OpenMPI. While there is a quickstart guide available in the OpenMPI documentation (https://docs.open-mpi.org/en/v5.0.x/installing-open-mpi/quickstart.html), we have found it to be somewhat cumbersome. Instead, we recommend following the tutorial provided by the developers of the ORCA computational chemistry package (https://www.youtube.com/watch?v=61mDG1q7z44). We also recommend that users download and install the version of OpenMPI that is compatible with the latest ORCA release. This compatibility will be important if you also intend to install the ORCA suite of programs on your system, which can be useful for generating input data for MobCal-MPI.

Once the steps above are completed, run the following command:

```
$ sudo apt install libopenmpi-dev
```

Then, run the command

```
$ sudo apt install openmpi-bin
```

OpenMPI should now be ready to use. You should be able to check the version using the command:

```
$ mpifort --version
GNU Fortran (GCC) 12.2.0
```

Once the compiler and wrapper are installed, MobCal-MPI can be compiled using the following command:

```
$ mpifort -o MobCal_MPI_201.exe MobCal_MPI_201.f -Og
```

Note that the -Og flag must be specified for proper performance of MobCal-MPI. **Erroneous CCSs will be produced if this optimization flag is not specified during compilation.**

Proper compilation of MobCal-MPI should result in the creation of MobCal_MPI_201.exe with no error messages appearing in the terminal. If error messages are printed when compiling, please report them to the issues section of the GitHub repository along with the details of your mpifort version.

Please ensure that you place the MobCal_MPI_201.exe either in your system's *bin* or in the directory that contains the .mfj files for which you intend to calculate CCSs.

4.2 Running MobCal-MPI

For improved management of multiple job executions, MobCal-MPI requires a submission file. This additional file can have any name, doesn't require a specific extension, and adopts a simple format: the first line should contain the name of the .mfj input file located in the same directory, and the second line should specify the name of the output file for the results with a .mout extension. For example, if your input file is named AMIFOSTINE_3.mfj, and you want to calculate its CCS, the contents of the job submission file would look like this:

```
AMIFOSTINE_3.mfj
AMIFOSTINE_3.mout
```

You can create this file using a text editor like Notepad and save it with a descriptive name, such as AMIFOSTINE_3.sub. Keep in mind that each CCS calculation requires its own job submission file. Depending on your system environment, the actual execution of the calculation will differ:

Local Unix machine without a job scheduler
If you have a local Unix machine without a scheduler like SLURM configured, you can run MobCal-MPI via the following command, which requests 8 cores (-np 8):

```
$ mpirun -np 8 ./MobCal_MPI_201.exe AMIFOSTINE_3.sub
```

Additionally, we offer a user-friendly bash script with this MobCal-MPI release (see example, titled *runmob_direct.sh*, designed to streamline the file submission process. This script scans the current directory for all *.mfj* files, generates the necessary submission files automatically, and initiates the MobCal-MPI calculation using the provided command. Please keep in mind that you might need to modify the core allocation within the script to match your system's capabilities and to avoid

overloading your system with more calculations than it can handle, as this will slow down the computations.

HPC or Unix machine with SLURM
HPC systems or Unix machines that are shared between users usually employ a scheduler like SLURM to organize the user demands. To run MobCal-MPI on those systems, we also provide a bash script, named *runmob_SLURM.sh*, that will look for all **.mfj* files in the current folder, create the submission files and submit them to the SLURM scheduler. The calculations might not start right away but that is exactly what the SLURM scheduler is for. The positive effect of this is that you can submit as many jobs as you want, and the scheduler will run all of them successively. Please note that the content of the runmob_SLURM.sh submission script is tailored to the clusters maintained by the Digital Research Alliance of Canada; you will need to modify this script to suit your system. Further, SLURM settings like number of cores or time limit can be changed as well.

Running the submission scripts
Note that all submission scripts need to be made executable before they can be used. This can be done via:

```
$ chmod a+x runmob_<version>.sh
```

where <version> is either "direct" or "SLURM" (from the provided sample scripts). Execution of the bash script is then done via:

```
$./runmob_<version>.sh
```

4.3 Contents of a MobCal-MPI output (.mout)

For each MobCal-MPI calculation, a .mout output file is created that contains important information pertaining to the CCS evaluation. This section of the manual section explains the data contained within each section of the output and describes its significance.

4.3.1 – File Header

The initial segment of the .mout file provides an overview of the MobCal-MPI job. It includes details such as the filename, atom count, charge assignments, temperature grid, and other pertinent information as depicted in the figure below:

```
input file name = AMIFOSTINE_3.mfj  Label
input file label = AMIFOSTINE_3
number of coordinate sets =     1  Number of coordinate sets defined in the input file (always 1)
number of atoms =  28  Number of atoms within the .mfj input file
coordinates in angstroms  Units for the coordinates provided in the .mfj input (ang=Ångstoms, a.u.=atomic units)
using a calculated (non-uniform) charge distribution  Charge distribution (unchanged)
total charge = 1.0000E+00
total absolute charge = 6.2020E+00  Charge information
mass of ion = 2.1507D+02 amu  Tbath, Teff_max, # Tgrid
Temperature range from input read:
 T_bath     =  298.00K
 T_eff max =  798.00K  Information about the temperature(s) specified in the .mfj input
 # of T_eff grid points =  10
van der Waals scaling parameters: eo= 1.3400E-03eV, ro= 3.0430E+00A
 rho_e= 1.275 rho_s= 0.825  van der Waals scaling parameters that depend on the collision gas (fixed)
Empirical correction: 1+A*exp(-B/(E/N))
 A =    0.0611  Empirical correction to mobility (can be toggled
 B = 143.00 Td  on/off when creating the .mfj input via the GUI
dipole constant = 2.0072E-58
```

4.3.2 – Setting up ion-neutral collision trajectory simulations

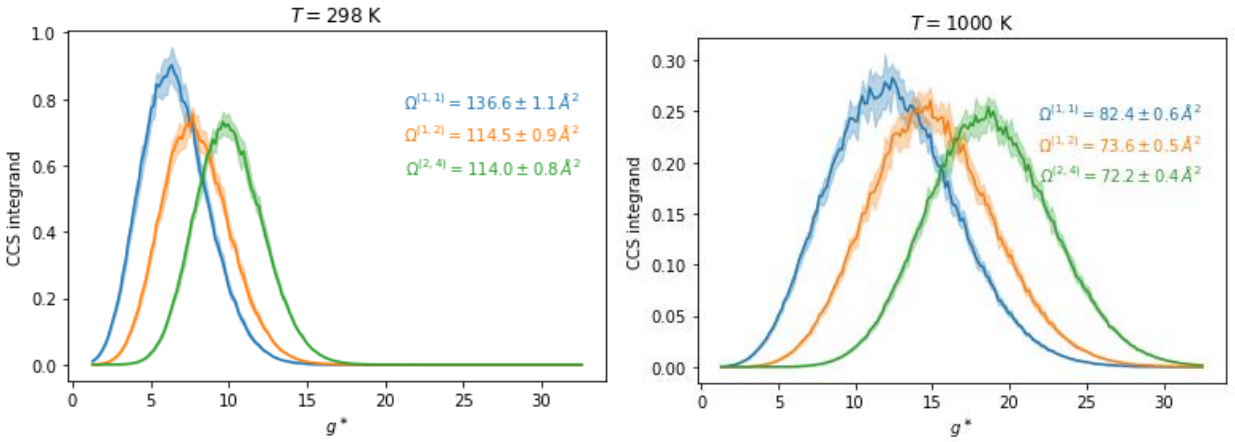
In the next section of the .mout file, information that is pertinent to the evaluation of the momentum transfer integrals, $Q^{(l)}(g)$ for $l = 1,2,3$, is printed. This includes some general thresholds that define when a collision trajectory should be terminated, as well as integration limits for both the impact parameter and relative collision velocity.

As this section of the output file contains integration limits, it is important to describe how they are derived. Beginning with the velocity grid limits, the ideal scenario would be to sample relative velocities from zero to infinity. However, considering the weighting functions $\omega^{(s)}$ shown in **Eq. 13**, the likelihood of very small ($g \to 0$) and very large ($g \to \infty$) velocities occurring for finite values of T_{eff} is zero. Therefore, using finite velocity limits should suffice so long as the integrand in **Eq. 13** is sufficiently close to zero at these limits.

```
-------------------------------------------------------------------------------
Calculating trajectories to obtain momentum transfer cross sections Q*(l), l=1,2,3
-------------------------------------------------------------------------------

Set-ups
-------

global trajectory parameters                  general parameters to determine
sw1 = 5.0000E-05       sw2 = 5.0000E-03       the termination of a trajectory
dtsf1 = 5.0000E-01     dtsf2 = 1.0000E-01
inwr =  1              ifail =  100

maximum extent orientated along x axis        orient molecule for b_max determination
along x axis emax =-3.2908E-02eV rmax = 7.4634E+00A r00 = 6.7522E+00A

Set up velocity integration:
inp=104 gst points on regular grid between    define lower and upper limit of the
gst_min = 0.1285E+01 and gst_max = 0.2906E+02 velocity integration grid

Set up Monte-Carlo integration:
imp= 512 random points over impact parameter b and orientation
determine b2max value for each gst grid point using
minimum value of (1-cosX) = 5.0000E-04   threshold for b_max

    gst             b2max/ro2         b/A                 Printout of b²_max/ρ²_0
 1.285278E+00      7.940000E+01      2.711516E+01          (dimensionless and b/Å
 1.554929E+00      6.660000E+01      2.483356E+01          for every velocity point,
 1.824580E+00      5.760000E+01      2.309475E+01          g* (dimensionless)
 2.094231E+00      5.090000E+01      2.171005E+01
```

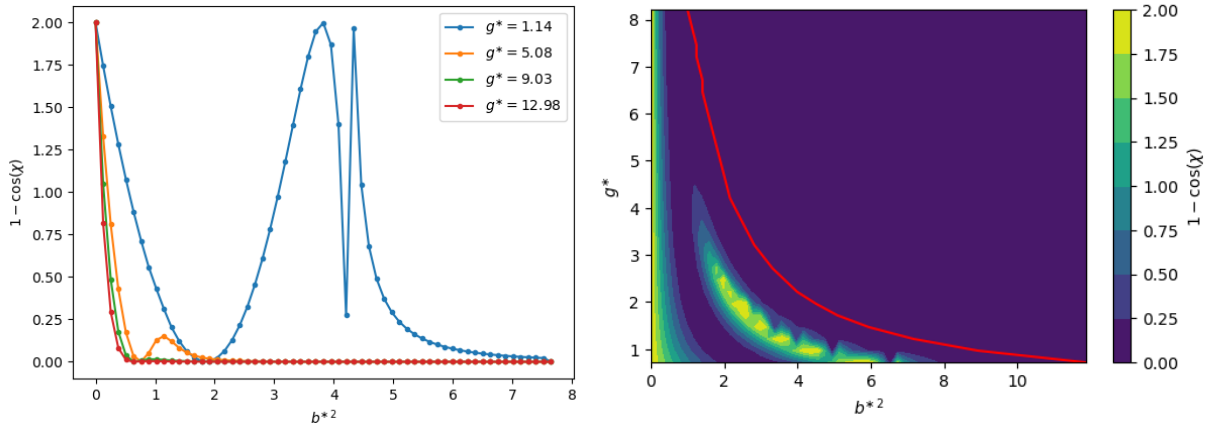
Because we want to use a fixed grid for the calculation of all collision integrals, it is essential for the condition that $\omega^{(s)} \to 0$ as $g \to 0$ and $g \to \infty$ to be fulfilled for every value of s (1 – 4) and for all T_{eff} values sampled between T_{bath} and $T_{eff,max}$. Details concerning the determination of velocity grid limits are explained in Section C4, which also considers the fact that slow trajectories (*i.e.,* as $g \to 0$) tend to be captured by the ion and, as a result, do not get reemitted.

The integration limits are printed to the .mout file as *gst_min* and *gst_max* in dimensionless units. Below, you can see the CCS integrands for various combinations of (l, s) such as including $(1,1)$, $(1,2)$ and $(2,4)$ displayed over the velocity grid defined by T_{bath} = 298 K to $T_{eff,max}$ = 1000 K. Evidently, the chosen limits of the velocity grid ensure that all CCS values are accurately computed for all possible values of s and T_{eff} (the area under the curves shown below).



Similarly, integration limits need to be set for the impact parameter, b. In this case, the lower integration limit will always be zero, as $b = 0$ constitutes a head-on collision. However, the upper integration limit, in theory, can extend to infinity (see **Eq. 14**).

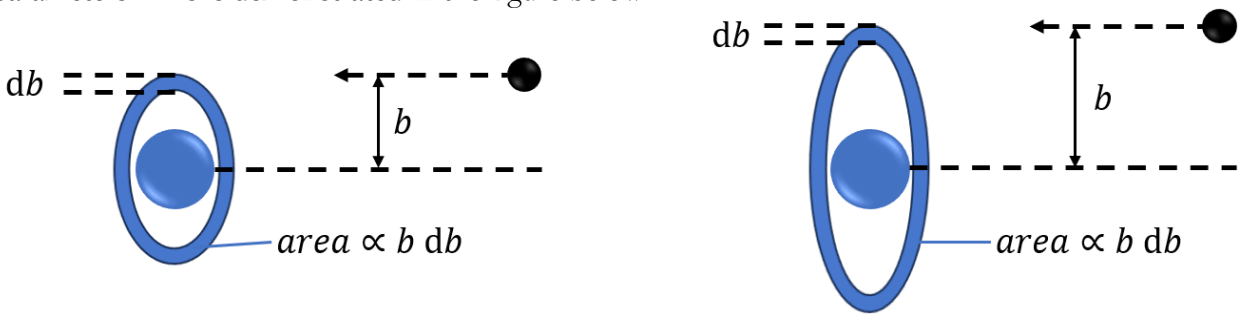
Recognizing the impracticality of sampling between $b = 0$ to $b = \infty$, we need to find a finite value that captures the important parts of the *imp* space, much like the approach used to determine integration limits for the velocity grid. In this context, we can look at the integrand of interest, $2b(1 - \cos\chi)$ from **Eq. 14**. This integrand represents the degree of momentum transfer for a specific combination of collision velocity (g) and impact parameter (b). It is worth noting that this integrand approaches zero as $b \to \infty$, and does so even more rapidly for larger values of g. In simpler terms, as collision velocity increases, collision events with large impact parameters contribute progressively less to the total momentum transfer. Therefore, it is efficient to establish an upper limit for the impact parameter, which we denote as b_{max}, for each point on the chosen velocity grid. The process for determining this limit is demonstrated by the following figures:



Here, the red line shows the upper limit of the impact parameter integration (b_{max}^2). This approach avoids calculating collision trajectories that have a negligible contribution to the overall momentum transfer (indicated by the large dark-blue are to the right of the red b_{max}^2 line). MobCal-MPI prints b_{max}^{*2}/ρ_0^2 (dimensionless) as well as b_{max} (in Ångstrøm) for each velocity grid point, g^* (dimensionless). Note that the molecule is initially orientated in a way that produces the largest b_{max} is largest for all possible orientations. For more details on the selection of b_{max}^2, see Section C2.

4.3.3 – Evaluating collision integrals

Now that everything is set up, the momentum transfer integrals, $Q^{(l)}$, are calculated for $l = 1,2,3$ at each velocity point (*gst*) in the velocity grid. At every velocity point, the impact parameter sample points (*imp*) are evenly divided across the number of cores. Then, imp/N_{core} trajectories are started on each core, with each core selecting a random orientation and an impact parameter between 0 and b_{max} (depending on the current velocity, *inp*) and returning the scattering angle, χ, for each combination of (*inp,imp*). Note that by picking a random value of b^2 between 0 and b_{max}^2 instead of a random value of b between 0 and b_{max} ensures that collision events are not equally likely across all impact parameters sampled. In other words, the likelihood of a collision occurring at a certain impact parameter b is proportional to the area of the annulus $b\ db$, where small impact parameters form a small circle around the axis of approach, and thus, are less likely to occur compared to larger impact parameters. This is demonstrated in the figure below.



Each core then sums the different momentum transfers integrands, $2b(1-\cos^l\chi)$, which are then averaged over all cores. After this is completed for all velocity grid points within a single itn cycle, the averaged $Q^{(l)}$ are saved. This process of computing the averaged $Q^{(l)}$ repeats as per the number of itn cycles defined by the user. The distribution of the different $Q^{(l)}$ determined from each itn cycle

is then used to calculate the final mean and error of the momentum transfer integrals according to **Eq. 15 – Eq. 17.** These averages and associated standard errors are then printed to the .mout file in dimensionless units. The data shown in this portion of the .mout file is plotted when the "plot Q(l)" function is selected in the "single .mout" tab of the GUI.

```
Final averaged values of Q*(l):

    gst                q1st                    q2st                    q3st
 1.2853E+00  2.1411E+01 +/- 1.63E+00  2.4647E+01 +/- 1.87E+00  2.4171E+01 +/- 1.64E+00
 1.5549E+00  1.7676E+01 +/- 1.29E+00  2.0191E+01 +/- 1.06E+00  1.9588E+01 +/- 1.14E+00
 1.8246E+00  1.5356E+01 +/- 9.94E-01  1.7266E+01 +/- 9.94E-01  1.7313E+01 +/- 8.98E-01
 2.0942E+00  1.3362E+01 +/- 1.24E+00  1.5202E+01 +/- 1.01E+00  1.5149E+01 +/- 1.18E+00
 2.3639E+00  1.2682E+01 +/- 8.79E-01  1.4396E+01 +/- 9.63E-01  1.4309E+01 +/- 9.09E-01
 2.6335E+00  1.0817E+01 +/- 8.94E-01  1.2434E+01 +/- 6.84E-01  1.2370E+01 +/- 8.24E-01
 2.9032E+00  1.0140E+01 +/- 9.02E-01  1.1258E+01 +/- 5.77E-01  1.1438E+01 +/- 8.30E-01
 3.1728E+00  9.3387E+00 +/- 4.67E-01  1.0626E+01 +/- 4.41E-01  1.0541E+01 +/- 3.60E-01
 3.4425E+00  9.0847E+00 +/- 5.78E-01  1.0215E+01 +/- 5.15E-01  1.0314E+01 +/- 5.22E-01
 3.7121E+00  7.8688E+00 +/- 4.34E-01  8.7330E+00 +/- 5.69E-01  8.9232E+00 +/- 5.29E-01
 3.9818E+00  7.4997E+00 +/- 3.93E-01  9.0032E+00 +/- 3.92E-01  8.6473E+00 +/- 3.82E-01
 4.2514E+00  7.2607E+00 +/- 4.27E-01  8.3038E+00 +/- 4.03E-01  8.2291E+00 +/- 3.53E-01
 4.5211E+00  7.1051E+00 +/- 4.40E-01  8.1534E+00 +/- 4.52E-01  8.1077E+00 +/- 4.61E-01
```

After evaluating all collision trajectories, it is only necessary to compute the integral presented in **Eq. 12** for every T_{eff} within the designated temperature grid to obtain the CCSs. This calculation is straightforward, and its computation time is negligible compared to the trajectory calculations. For each point on the temperature grid, the collision integral, denoted as $\Omega^{(l,s)}$, is printed to a table in dimensionless units (note that $s \geq l$). The conversion factor to convert them to Å² is determined internally and printed for convenience. As an example, here is the relevant output for T_{eff} = 298 K (showing the effective temperature in dimensionless units in parenthesis; Tst).

```
Collision integrals OM*(l,s) at Teff= 298.00K (Tst= 1.9164E+01)
---------------------------------------------------------------
OM*(l,s)     s=1         s=2         s=3         s=4
   l=1    4.7859E+00  4.0372E+00  3.5738E+00  3.2669E+00
   l=2                4.8523E+00  4.2792E+00  3.8660E+00
   l=3                            4.2109E+00  3.8325E+00
(multiply by  2.9091E+01 for values in Angstrom**2)

red. field stregth E/N   =     0.000 Td
drift velocity vD        =     0.0   m/s
red. mobility (1st ord.) = 0.1547E-03 +/- 0.20E-05 m2/Vs
red. mobility (2nd ord.) = 0.1547E-03 +/- 0.20E-05 m2/Vs
red. mobility (emp corr) = 0.1547E-03 +/- 0.20E-05 m2/Vs
av. coll cross section   = 0.1392E+03 +/- 0.18E+01 A**2
```

After the collision integrals are printed, the output includes the reduced field strength, E/N, expressed in Td, along with the respective drift velocity, v_D, given in m/s, for the specific T_{eff} value in the temperature grid sampled. Subsequently, the reduced mobility, K_0, is printed in both the 1st and 2nd order corrections of 2TT, followed by its final value when employing the empirical correction described in Section 2.7 (if it is activated). The reported errors are determined from the uncertainties of the first collision integral, $\Omega^{(1,1)}$, as this contributes the most to the overall uncertainty even in the 2nd order 2TT correction (see Section 2.5). The differences between the ions reduced mobility in the

1st, 2nd, and empirically corrected implementations of 2TT are apparent when $T_{eff} > T_{bath}$ (see below).

```
Collision integrals OM*(l,s) at Teff= 498.00K (Tst= 3.2026E+01)
---------------------------------------------------------------
OM*(l,s)     s=1         s=2         s=3         s=4
   l=1    3.7925E+00  3.2529E+00  2.9497E+00  2.7640E+00
   l=2                3.8098E+00  3.3740E+00  3.0846E+00
   l=3                            3.3924E+00  3.1321E+00
(multiply by  2.9091E+01 for values in Angstrom**2)

red. field stregth E/N   =   104.355 Td
drift velocity vD        =   427.3   m/s
red. mobility (1st ord.) = 0.1510E-03 +/- 0.18E-05 m2/Vs
red. mobility (2nd ord.) = 0.1524E-03 +/- 0.18E-05 m2/Vs
red. mobility (emp corr) = 0.1548E-03 +/- 0.19E-05 m2/Vs
av. coll cross section   = 0.1103E+03 +/- 0.13E+01 A**2
```

4.3.4 – Summary table

The final section of the .mout file provides a comprehensive summary of the MobCal-MPI calculation. This includes the number of calculation cycles (*itn*), velocity points (*inp*), impact parameters sampled (*imp*), the total number of collisions evaluated (*itn* $\times$ *inp* $\times$ *imp*), and the buffer gas used for the simulation (He or N_2). Most notably, this section consolidates the quantities of interest in a table, which include the reduced field strength, the reduced mobility, the CCS and its associated uncertainty for each T_{eff} point defined by the temperature grid. This table shows the dependencies of K_0 on E/N , as well as Ω on T_{eff}, which can be visualized via plotting the data using the "plot Mobility" and "plot CCS" options, respectively, within the “analyze single .mout” tab of the MobCal-MPI GUI.

```
 SUMMARY

 program version = MobCal_MPI_2_beta.f
 input file name = AMIFOSTINE_3.mfj
 input file label = AMIFOSTINE_3


 number of complete cycles (itn) =      10
 number of velocity points (inp) =     104
 number of random points   (imp) =     512
 total number of points          =  532480
 number of failed trajectories   =       0
 Mobility Calculated under N2 gas
 Job Completed in   596.327301482000      s

**********************Mobility Summary**********************
*********************at T_bath= 298.00K*********************
 Teff [K]   E/N [Td]  K0 [cm**2/Vs]  CCS [A**2]  uncertainty
------------------------------------------------------------
  298.00       0.00       1.5470       139.22       1.32%
  348.00      51.06       1.5472       129.51       1.28%
  398.00      72.67       1.5491       121.78       1.25%
  448.00      89.65       1.5491       115.51       1.23%
  498.00     104.36       1.5475       110.33       1.20%
  548.00     117.68       1.5446       105.98       1.18%
  598.00     130.06       1.5405       102.29       1.16%
  648.00     141.78       1.5353        99.13       1.14%
  698.00     152.98       1.5293        96.38       1.13%
  748.00     163.78       1.5226        93.97       1.11%
  798.00     174.26       1.5153        91.85       1.10%
************************************************************
```

The ion’s alpha function can be directly obtained from the summary table via **Eq. 5**, and the drift velocity via **Eq. 3**. Additionally, users can gauge the extent of heating experienced by the ion due to

its acceleration in the electric field by tracking the evolution of T_{eff} with E/N. It is worth noting that the ion's internal temperature is distinct from its T_{eff} determined from 2TT, as the latter reflects the ion's velocity distribution. Because of the evidence suggesting that these temperatures are equal when the collision gas is atomic,[74–76] we expect that a small, rigid molecule like N_2 with little heat capacity in their vibrational degrees of freedom to behave similarly.

4.4 Verifying proper performance of MobCal-MPI 2.0

It has been reported that different combinations of Fortran compilers and optimization flags can cause erroneous behaviour. For example, we found that using OpenMPI with *gfortran* and -Ofast caused the code to not work properly. To ensure MobCal-MPI is functioning properly, we recommend performing CCS calculations using the supplied test files and validating that the CCS results match those in the reference file.

Sample input files (and their corresponding output files) are included with MobCal-MPI in the GUI_Samples_files folder (found under mfj_creator). To validate the correct operation of MobCal-MPI, please conduct CCS calculations using these inputs and review the contents of each section of the corresponding .mout file to confirm their consistency with those presented in the sample files. If these entries and CCS values do not match those in the test files, we request that you report this in the "Issues" section and/or contact us via email at a2haack@uwaterloo.ca, cieritan@uwaterloo.ca, and shopkins@uwaterloo.ca. When reaching out to us, please ensure that you attach both the .mfj input and .mout output files and include information regarding the compiler type and version number employed.

Appendix A – Generating a series of conformers of a target analyte for subsequent CCS evaluations

Appendix A outlines the basic procedure for exploring the potential energy surface (PES) of a molecule to generate a series of local minima. The geometries of these local minima and their electrostatic potentials are refined at higher levels of computational theory, namely using Density Functional Theory (DFT), to generate candidate structures that represent those populated within the gas-phase ensemble. These DFT optimized structures serve as the basis for generation of MobCal-MPI input files and the subsequent evaluation of the Boltzmann-weighted CCS.

This appendix uses the Conformer-Rotamer Ensemble Sampling Tool (CREST; see https://crest-lab.github.io/crest-docs/), a free to use package that is interfaced with the xTB package (https://xtb-docs.readthedocs.io/en/latest/) to identify structures that best describe the investigated system under given conditions. Local minima identified by CREST are ported to the ORCA computational chemistry suite (see https://orcaforum.kofo.mpg.de/app.php/portal) for refinement at a higher level of theory.

A1. The workflow for identifying candidate geometries by mapping potential energy surfaces

This tutorial uses supplementary files and python codes, all of which can be found in the Manual/Appendix_A subdirectory on the MobCal-MPI GitHub repository.

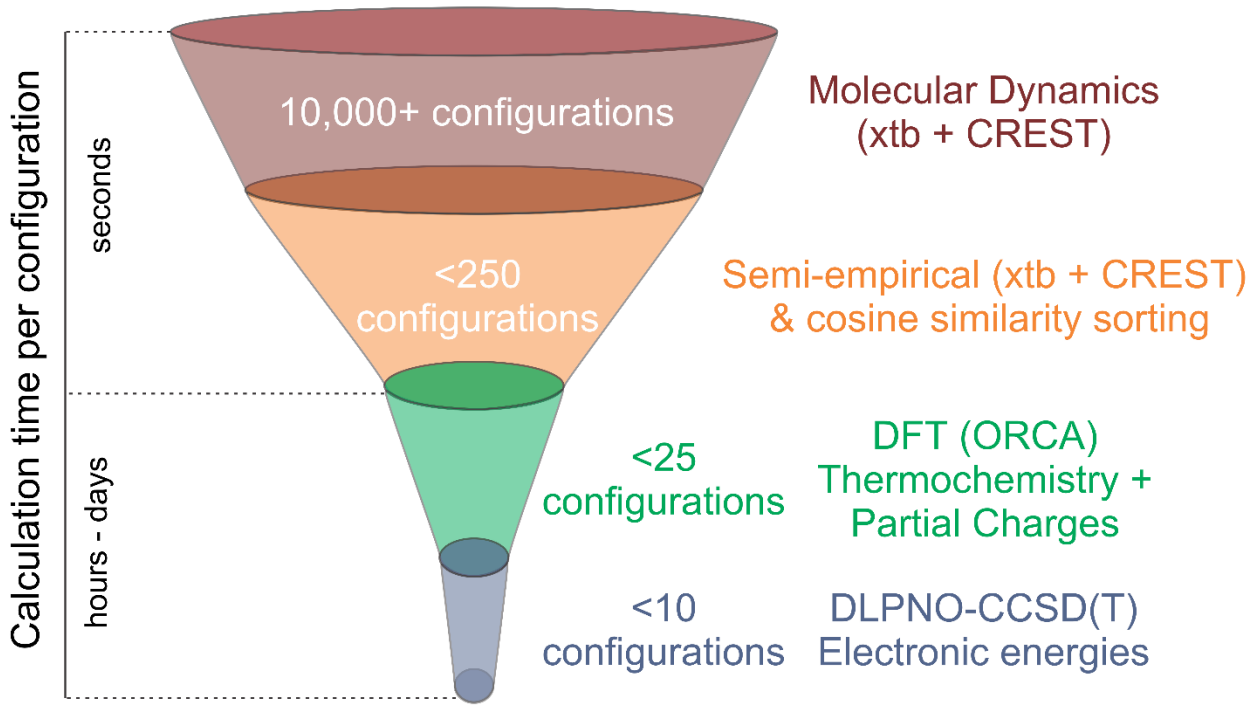
When an analyte contains flexible chemical bonds, it is improbable for it to maintain a single configuration as it travels through any mobility device (e.g., a drift tube). This is because collisions between the analyte and buffer gas facilitate transitions between ion configurations, with each configuration possessing a distinct CCS. Thus, any attempt to model an ion's CCS using a single, static structure will not capture the evolution of an ion's structure though time and will not be accurate. Although a molecular dynamics approach that accounts for structural changes induced by ion-neutral collisions would be the most appropriate, its computational expense does not justify its use. Frozen-core methods, which omit considerations of atomic motion, can offer a reasonably accurate alternative if enough structures are sampled. The frozen-core approach involves calculating CCS values for representative configurations of the ion, and then weighting those configurations based on their population in the gas-phase ensemble using a Boltzmann-weighting scheme. This approach is illustrated as a hierarchical workflow in the figure below and relies on specific information for implementation:

1) The electronic energy of a series of optimized, low-energy configurations of the ion.

2) The translational, rotational, and vibrational partition functions of each optimized configuration, which will enable the calculation of its population in the gas-phase ensemble via calculation of its Gibbs corrected energy.

3) Atomic partial charges for each optimized configuration.

4) The CCS of each optimized configuration.



The Boltzmann-weighting workflow begins with the CREST package (Conformer–Rotamer Ensemble Sampling Tool), which works in conjunction with xtb (GFN2-xTB, a quantum chemistry method) to explore and map the low-lying regions of the potential energy surface:

1. **Initialization**: CREST begins with an initial molecular structure, typically in a specific conformation or geometry.

2. **Classical Molecular Dynamics (MD)**: CREST then conducts MD simulations, where the initial structure is allowed to move and explore different conformations over time.

3. **QM Energy Evaluation with xtb**: At specified intervals during the MD simulation, CREST performs quantum mechanical energy calculations using xTB. These calculations provide accurate electronic energy information for the current molecular configuration.

4. **Ensemble Sampling**: The combination of MD and periodic QM calculations results in the sampling of a diverse ensemble of low-energy molecular conformations and rotamers. This ensemble represents the potential energy landscape of the molecule.

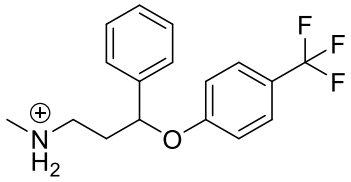
5. **Analysis**: CREST analyzes the data to identify low-lying regions of the potential energy surface, which correspond to energetically favorable conformations, and ultimately results in a series of unique configurations that can be further sorted based on cosine similarities.[77] For more detailed information on the workflows that CREST employs, please refer to the CREST documentation.

Although the conformers predicted by GFN2-xTB are like those predicted by higher level *ab initio* theory, the electronic energies and thermochemical corrections require further refinement. Density Functional Theory (DFT)[78] calculations provide the necessary improvement to molecular geometries, electronic energies, and thermochemical data such that ion configurations can be determined with sufficient accuracy.[19,71,79] When appropriate DFT functionals are used (*i.e.,* hybrid/long-range corrected functionals or those that contain empirical dispersion corrections), benchmarking suggests that the resulting DFT energies and thermochemical data can be used to accurately reproduce experimentally measured binding energies, vibrational spectra, and reaction barriers.[80–84] We direct readers to reference 84 for guidance on selecting a DFT functional for a specific chemical purpose. We typically employ the ωB97X-D3/Def2-TZVPP or ωB97X-D3BJ/Def2-TZVPP level of theory when performing DFT calculations to determine vibrational frequencies and/or relative isomer populations. Using this level of DFT theory, one can also accurately map the cluster electrostatic potential to determine atomic partial charges, which are required for CCS calculations.

It is worth noting that the thermochemistry computed using DFT methods relies on harmonic vibrational frequencies, which can fail to accurately reproduce thermochemical corrections for highly anharmonic systems (*e.g.*, hydrogen-bonded clusters).[85–87] In principle, one can circumvent this issue by computing anharmonic vibrational partition functions, but this comes at a high computational cost. Since most hybrid DFT functionals are parametrized to yield good thermochemistry with the harmonic approximation,[84] this considerable computational cost is not justified. Calculation accuracy can be improved by combining DFT-computed harmonic thermochemistry with electronic energies calculated at the coupled cluster level of theory, which is the "gold-standard" for computational accuracy. However, the N^7 scaling of coupled cluster methods also imposes a substantial computational cost and, depending on available computational resources, may be impractical for some chemical systems (*e.g.*, those with more than *ca.* 150 electrons). Much of the computational cost for CCSD(T) calculations (*i.e.*, coupled cluster singles, doubles, and perturbative triples) stems from computing electron-electron correlations for *every* electron-electron pairing in the molecule. One can therefore reduce calculation times (and achieve relatively accurate results) by considering only local electron correlations. The domain-based local pair natural orbital (DLPNO) coupled cluster methods implemented in the ORCA software package achieves comparable accuracy to CCSD(T) (< 5 kJ mol^{-1}),[56,57,88,89] but scales linearly with the number of electrons in the system.[90–92] Thus, DLPNO-CCSD(T) energies combined with DFT thermochemistry yield accurate and efficient calculations of ion-solvent cluster thermochemistry.

To showcase this workflow to users, we've prepared a tutorial illustrating the process of mapping the potential energy surface of the molecule fluoxetine. We selected fluoxetine due to its conformational flexibility and the fact that it has a single protonation site at the secondary amine, although we

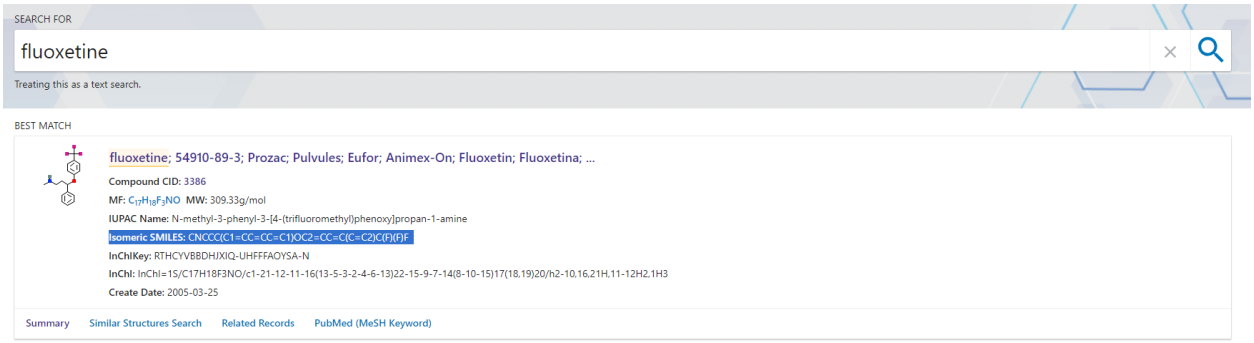
recognize that some analytes may contain multiple protonation sites. If a molecule that you are interested in has multiple protonation sites, we want to draw attention to a feature in CREST that can assess the most probable protonation sites (see https://crest-lab.github.io/crest-docs/page/examples/example_5.html).



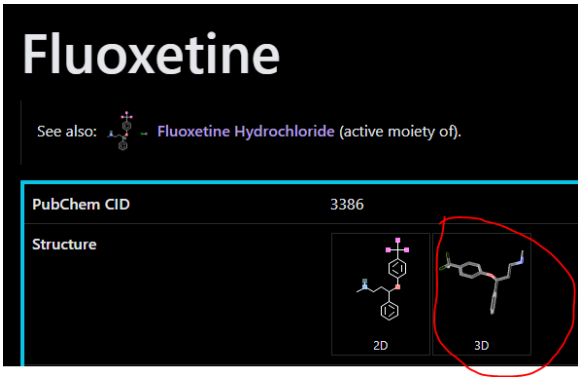
However, please note that the screening of protonation sites is not within the scope of this demonstration. Users who are keen to explore this functionality are encouraged to consult the comprehensive CREST documentation, available at https://crest-lab.github.io/crest-docs/. You can also find instructions on how to install CREST by visiting this link.

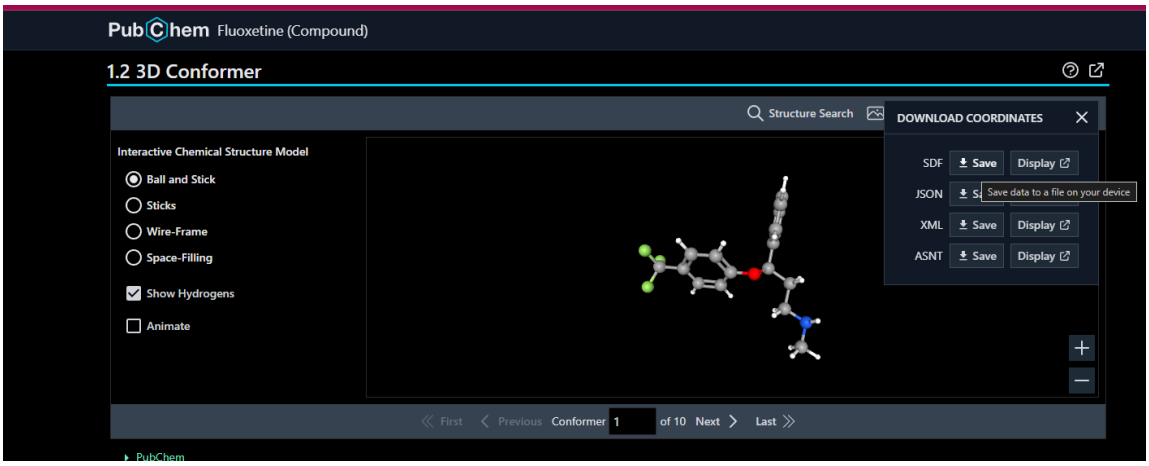
A2. Using CREST to generate local minima

It's important to recognize that multiple methods exist for generating a molecule's 3D conformation. If users have a preferred workflow for generating 3D coordinates of molecules, they are encouraged to use it. However, since the purpose of this Appendix is to provide a complete workflow to conduct conformational mapping for the purpose of CCS modelling, we are being as thorough as possible. For this demonstration, we will introduce users to a slow, but fail-safe version of generating 3D conformers by downloading 3D coordinate files from PubChem (https://pubchem.ncbi.nlm.nih.gov/). Searching for fluoxetine on PubChem will yield the following window.



Click on the entry for fluoxetine, go to its structural information by clicking on the 3D structure window (as shown in the screenshot), then select "download coordinates" and save the file as a .sdf format to a convenient location on your local machine. Feel free to rename the file to "Fluoxetine.sdf".

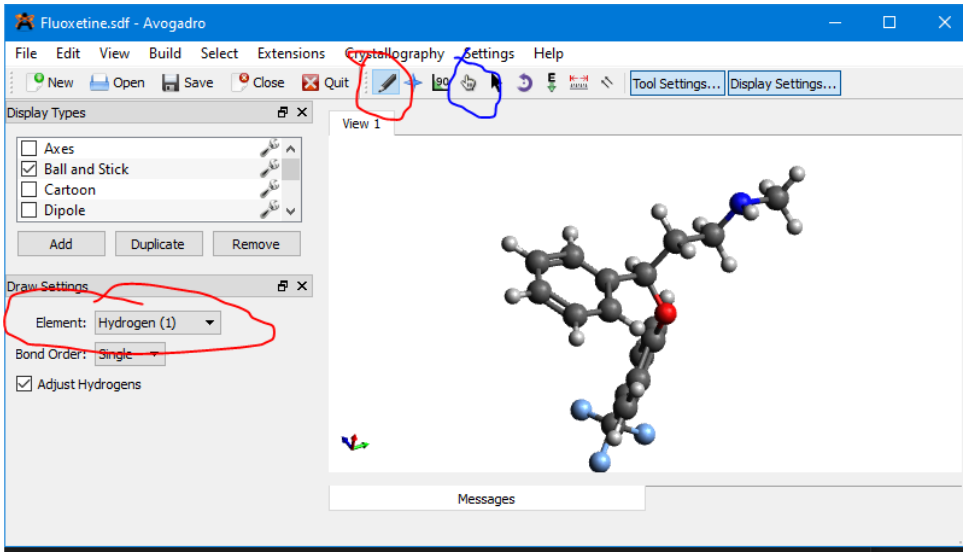




Having downloaded the file, the next step involves manually adding a proton to the secondary amine. To accomplish this, open the .sdf file in a visualization tool. We recommend using the Avogadro implementation that is compatible with ORCA outputs, available on the ORCA forum. ORCA forum.

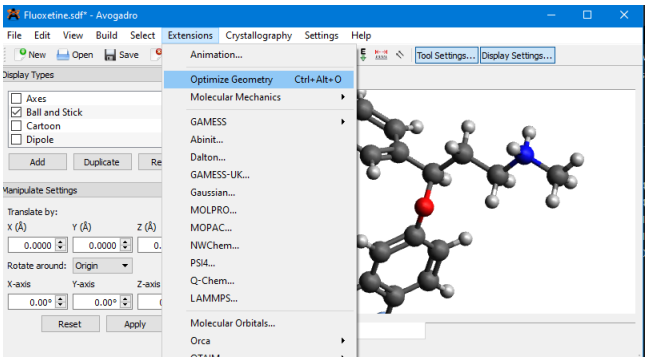
Once in the visualization tool, access the pen tool from the top toolbar (indicated by a red circle in the screenshot). In the Draw Settings on the left toolbar (also marked with a red circle), select "Hydrogen (1)" as the Element. Click on the secondary amine, then, without releasing the click, drag to add a proton to the amine. If you make an error, simply use CTRL+Z or the Edit menu to undo and then attempt adding the proton again. The precise orientation of the proton on the amine is not critical at this stage, as the geometry will be optimized shortly.

If desired, you can adjust the proton's position using the Manipulation tool (highlighted by a blue circle). To do this, select the manipulation tool, then click and drag the proton to the desired location. If you make any mistakes, don't worry; you can always use CTRL+Z to undo them. Keep in mind that using Avogadro can be frustrating, even for us!



After adding the proton, you can perform a basic geometry optimization by using the CTRL+ALT+O shortcut or through the menu option (Extensions > Optimize Geometry). Once the optimization is

finished, save the protonated version of Fluoxetine as a .xyz file by going to File > Save as (for example, name it "Fluoxetine.xyz").



The .xyz file should resemble the following (possibly with minor variations in the xyz coordinates):

Fluoxetine.xyz

```
41
    Energy:      74.3102073
F         -4.94835       -0.26648        1.53671
F         -5.30434       -0.81632       -0.53810
F         -5.35987        1.27470        0.05942
O          0.86360        0.67920       -0.79980
N          4.02567        3.36731       -0.05567
C          1.75761        0.28918        0.25764
C          3.04572        1.11269        0.07827
C          2.09153       -1.19066        0.19334
C          2.74533        2.60480        0.10843
C          2.23945       -1.85762       -1.03341
C          2.29409       -1.91262        1.38041
C         -0.45967        0.52870       -0.48416
C          2.56756       -3.21381       -1.07064
C          2.62724       -3.26777        1.33970
C         -1.21277       -0.34196       -1.26841
C         -1.07365        1.28904        0.50976
C         -3.20990        0.24574       -0.01554
C          2.76234       -3.91754        0.11523
C          3.85522        4.84621       -0.03301
C         -2.58048       -0.49165       -1.02921
C         -2.44220        1.14072        0.74841
C         -4.69070        0.10495        0.25279
H          1.33192        0.52637        1.24179
H          3.76935        0.85618        0.86251
H          3.51597        0.87010       -0.88428
H          2.08859        2.91185       -0.71139
H          2.31048        2.90556        1.06717
H          2.08664       -1.33108       -1.97317
H          2.18700       -1.43350        2.35111
H          4.46644        3.10142       -0.94792
H          2.66540       -3.72587       -2.02521
H          2.77493       -3.82212        2.26390
H         -0.74012       -0.91421       -2.06269
H         -0.51346        2.00871        1.09650
H          3.01302       -4.97552        0.08533
H          4.84407        5.29103       -0.16329
H          3.43135        5.12766        0.93381
H          3.19459        5.12872       -0.85597
H         -3.15477       -1.18570       -1.64254
```

With Fluoxetine now protonated, possessing xyz coordinates, and saved as an .xyz file, we can initiate the CREST routine. If CREST has not been installed yet, install it by following these instructions. Once installed, go to the system where CREST is installed and put Fluoxetine.xyz file into a new, empty directory. Access this directory using your shell client and input the following command. Keep in mind that the --T flag specifies the use of 16 cores; you should modify this number according to your system's capabilities.

```
$ nohup crest Fluoxetine.xyz --gfn2 --chrg 1 --T 16 > crest.out &
```

If everything goes as planned, CREST will begin its conformational search on your molecule. It will add all the output text to the "crest.out" file, which should now be visible in your working directory, alongside various other files relevant to the CREST process. In case any errors occur during the CREST routine, they will be written to the "crest.out" file. For troubleshooting errors with CREST, please refer to their extensive documentation (https://crest-lab.github.io/crest-docs/). Once the CREST job is finished, a message will be written to "crest.out" confirming normal termination. Below this message, you'll find a summary table of the results. **Please be aware that if users follow the workflow outlined in this manual, they may obtain a different total number of conformers from CREST than our initial run. This variance is due to the inherent randomness in the conformer generation process, which involves sampling from molecular dynamics (MD) simulations.**

A3. Post-processing of CREST results and DFT optimization

Following the completion of CREST, the unique minima will be saved to the crest_conformers.xyz file in order of increasing energy. Because hundreds of conformations are often identified as unique by CREST, optimizing all of them via DFT is impractical, necessitating an additional sorting routine. Josh Featherstone has developed a script as part of the Featherstone suite GUI for performing cosine similarity searches on the energy-sorted conformers. You can download this suite from GitHub at https://github.com/jrjfeath/Featherstone_Labs_Suite.

After downloading and installing the Featherstone Suite per the ReadMe instructions, you can extract the contents of "crest_conformers.xyz" into separate .gjf files for subsequent sorting, utilizing the "xyz_file_splitter.py" script located in the "Appendix_A" folder. Move the "crest_conformers.xyz" file to your local machine's directory, then open xyz_file_splitter.py in your preferred Python environment (e.g., IDLE or VS Code), and edit the first three lines as follows:

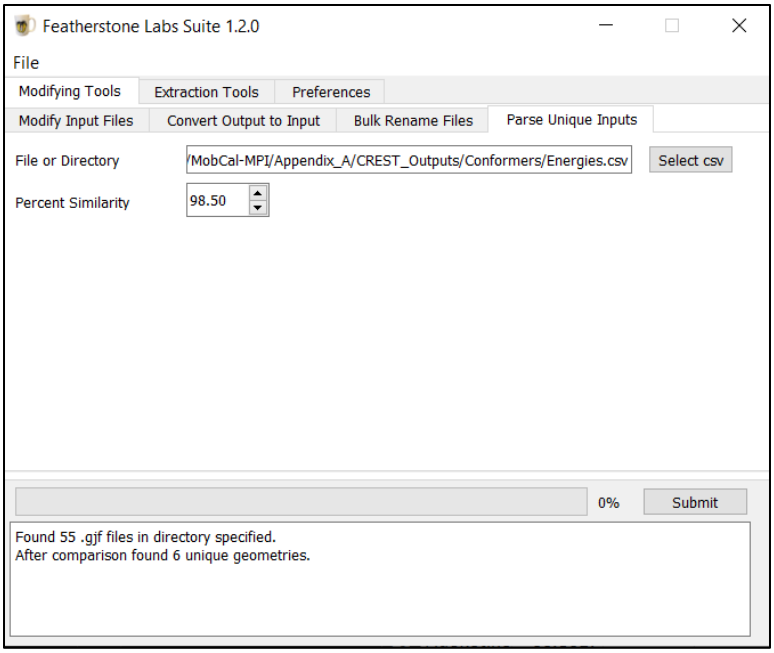
- **Directory:** Specify the directory containing the "crest_conformers.xyz" file. This must be a raw string.
- **Filename:** Keep this as "crest_conformers.xyz"; there's no need to change it.
- **Basename:** This should be the base name for naming the .gjf files. Ensure this entry is a string enclosed in quotations.

Running this Python code will create a new subdirectory named "Conformers" where each unique conformer from the "crest_conformers.xyz" file will be saved as an individual .gjf file, labeled numerically to reflect its energetic order. Additionally, an "Energies.csv" file is generated to summarize the energy of each conformer in this directory.

After extracting files with "xyz_file_splitter.py," open the Featherstone Suite GUI, go to "Modifying Tools," and select "Parse Unique Inputs." Then, click "select csv" and load in the "Energies.csv" file that was just generated.

Now, let's set the similarity threshold for isomers, which is a user-dependent parameter and directly affects the number of subsequent DFT calculations. It is essential to use chemical intuition when determining how many conformers are needed to adequately describe the potential energy surface of your analyte. For moderately flexible molecules like fluoxetine, 5 to 10 representative conformers should suffice. Adjust the similarity percentage to achieve your desired number of conformers, but keep in mind that higher energy conformers identified by CREST may not be populated in the gas-phase ensemble, and that keeping more conformations will increase computational resource requirements during DFT optimization. Additionally, be cautious about overly strict criteria, as they might overlook vital conformers. We recommend users manually review results from the parsing routine, assess relative energies, and tailor the similarity threshold to their specific needs, as there's no one-size-fits-all approach. For fluoxetine, these similarity percentages correspond to various unique geometries, all within 25 kJ/mol of the lowest energy structure. This range of 25 kJ/mol works fairly well for capturing the important conformations without being overly restrictive.

99.5 – 11 unique geometries
99.0 – 7 unique geometries
98.5 – 6 unique geometries
98.0 – 4 unique geometries



Assigning a 99% or 98.5% similarity level both yield a manageable number of low-energy conformations (usually < 10, but this is not a hard cutoff), so we will choose the unique structures from the 99% similarity run. **Note that the similarity thresholds sampled here may not yield the same number of unique conformers as your CREST run due to the randomness in the conformer generation process.**

Keep in mind that the unique conformers that are written to the "uniques" subdirectory will persist through multiple runs of "Parse Unique Inputs." To avoid retaining duplicate structures in each iteration of the similarity threshold screening, clear the "uniques" directory after every run. Then, once you have found the optimal similarity threshold, run Parse Unique Inputs for a final time. Failure to

adhere to this workflow will result in the accumulation of geometrically similar conformers within the "uniques" subdirectory.

Having reduced the number of CREST-identified conformers, these representative local minima are now ready for refinement using Density Functional Theory (DFT). While choosing a DFT functional and basis set might initially feel as frightening as reading Fortran code, there is indeed a method to the madness. Martin Head-Gordon wrote an excellent review that assesses the performance of 200 density functionals and scored their accuracy based on the agreement between calculated and experimentally measured physicochemical properties.[84] For predicting thermochemical parameters like Gibbs energies, ωB97X-D3 paired with basis sets from Ahlrichs and Weigand (e.g., def2-TZVPP) performed optimally. Consequently, **we recommend the ωB97X-D3/def2-TZVPP functional and basis set, as well as computing partial charges via the ChelpG partition scheme on a fine grid size**. These calculations are possible using the ORCA suite of programs, which is accessible at no cost for academic users, is open source, and exhibits exceptional performance in terms of its accuracy and speed.

To execute DFT computations using ORCA, you must prepare input files. Here's how you can create an ORCA input file:

1. Utilize the "gaussian_gjf_to_ORCA_input.py" script, which is available in the "Appendix_A" subdirectory of the MobCal-MPI GitHub repository.

2. Open this script within your preferred Python environment.

3. Supply the script with the following information:

   Directory: This should be the path to the directory where the ".gjf" files, previously parsed for cosine similarities, are located. Be sure to provide it as a raw string, enclosed in double quotes and prefixed with "r." For example, use **r"C:/SampleDirectory/ParsedInputs"**.

   Mpp (Memory per Processor): Specify the amount of memory required per core for the DFT calculation. This quantity will depend on the resources available on your system, and will need to be adjusted. For example, on our clusters, we request that jobs run with 4096 MB, but set the mpp in this code to 3200 MB.

   Ncores (Number of Cores): Indicate the number of cores to be used in parallel for running the DFT calculation. Optimal performance is typically achieved with 8 or 16 cores, balancing performance and overhead.

   Charge: Set the charge of the ion.

   Multiplicity: Set the multiplicity of the ion (1 = singlet; 2 = doublet; 3 = triplet; etc.)

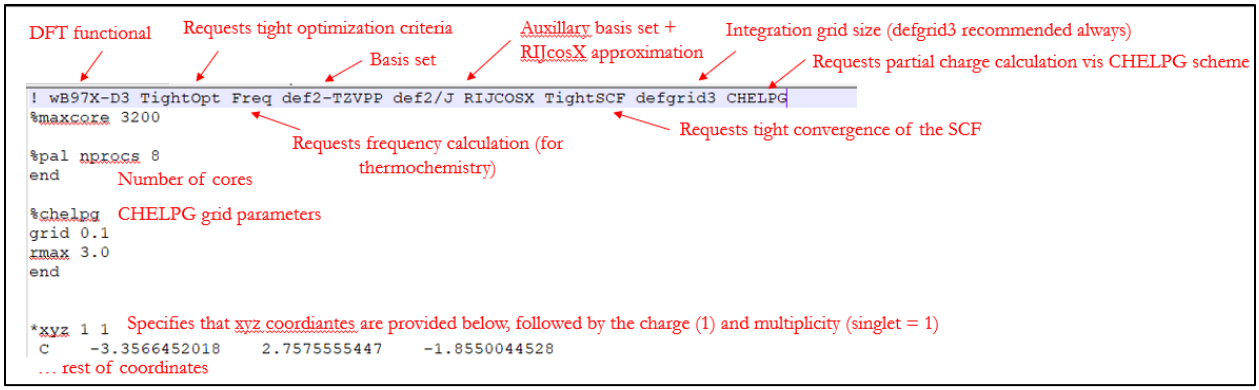
   Calc_line (Calculation Line): This line specifies the operations that ORCA will perform. For accurate geometry generation, thermochemistry, and partial charge calculations, we recommend the following. For users experienced with quantum-chemical calculations, they may choose variants other than ωB97X-D3 functional and Def2-TZVPP basis set. The impact of different model chemistries on the calculated CCS is investigated in this following study.

```
calc_line = '! wB97X-D3 TightOpt Freq def2-TZVPP def2/J RIJCOSX TightSCF
defgrid3 CHELPG' #Must be a string
```

ESP_Charges: Set this option to "True" to enable the calculation of partial charges. Without this option, the subsequent generation of MobCal-MPI 2.0 inputs will not work correctly.

Write_xyz: Use this option to specify whether the xyz coordinates should be written directly to the .inp file (set to "False") or if they should be written to an external .xyz file referenced within the .inp (set to "True").

After filling in all the required fields, execute the code. It will transform the unique fluoxetine geometries identified by the CREST routine into input files for DFT calculations in ORCA. A sample input for "Fluoxetine_1.inp" is provided in the figure below, along with an explanation of each section within the input file.



Each of the generated input files should be submitted to a suitable high-performance computing system for DFT calculations. Once the DFT computations are finished and the files contain the optimized geometry, vibrational frequencies, and partial charges, you can then proceed to the next step, which involves further processing to create MobCal-MPI input files.

A4. Post-processing of DFT results, DLPNO-CCSD(T) calculations, and .mfj file creation

A4.1 – Extracting thermochemistry from DFT outputs

After completing your DFT calculations, you'll need to follow these steps:

1. Move the .out files to a directory on your Desktop.

2. You can visualize the optimized geometry, vibrational frequency animations, and other valuable data by opening the .out file in Avogadro (for download instructions, visit the ORCA forum).

3. Locate and open the "ORCA_Thermochem_extractor.py" script, which can be found in the "Appendix_A" subdirectory. Using your preferred Python development environment, open the "ORCA_Thermochem_extractor.py" script.

4. In the script, modify the directory line to point to the folder where all the .out files are located. Ensure that you format the directory line as a "raw string" by adding the 'r' prefix before the quotations, like this:

```
'''
Created by CI 2021-04-15

Extractes thermochemical qunatities directly from ORCA output files
'''
#Where are your ORCA outputs located?
directory = r'D:\OneDrive\OneDrive - University of Waterloo\Waterloo\MobCal-MPI\MobCal-MPI\Appendix_A\DFT'
```

5. Once you have designated the directory as a raw string, execute the code to extract data from each .out file. This will output the relevant data to the "Thermo_data.csv" file.
6. If any files did not finish correctly or contain imaginary frequencies, the script will print the filenames to the terminal. All absent values will be substituted with -12345.0 in the corresponding .csv output.

7. When the code finishes, a .csv will be generated that contains several columns. Each column corresponds to the following:

   Filename: The name of the file ORCA .out file.

   Imaginary Frequencies: The number of imaginary frequencies calculated. Note that zero imaginary frequencies denote a true minimum on the potential energy surface, whereas any number greater than zero indicates an n^{th} order saddle point. If users encounter imaginary frequencies in their jobs, these will need to be reoptimized to remove the imaginary frequency because any imaginary frequencies will result in inaccurate corresponding thermochemistry. To do this, open the file in Avogadro, and animate the imaginary frequency. Quite often, imaginary frequencies correspond to torsional modes whereby internal dihedral angles are equal to or very close to 180 degrees. Distort this region of the molecule by modifying the dihedral angle away from 180 degrees (e.g., to 178 degrees), re-save the geometry as a new ,inp file, and resubmit the job to ORCA with the same calculation criteria as the original job. After this step, the imaginary frequency should be removed, and the corresponding thermochemistry will be accurate.

   In the case of flexible molecules, encountering imaginary frequencies in the range of 0 to -10 cm^{-1} may be inevitable. Should users come across such small-magnitude imaginary frequencies that persist even after altering the molecular geometry and reoptimizing it, they can safely disregard them. These frequencies are of negligible magnitude and do not significantly affect the accuracy of thermochemical corrections beyond the inherent errors associated with DFT.

   Electronic energy (E): The total electronic energy obtained after DFT optimization (reported in Hartree).

ZPE Correction (E_{zpe}): The zero-point energy correction

Thermal correction ($E_{thermal}$): The thermal correction to energy, which is defined as the zero-point energy correction plus the vibrational, rotational, and translational thermal energies.

Enthalpy Correction (E_H): The thermal enthalpy correction, which is defined as the $E_{thermal} + k_bT$.

Gibbs Correction (E_{Gibbs}): The correction for Gibbs energy, where G = H – T*S

Total ZPE: The zero-point corrected energy: $E + E_{ZPE}$

Total Thermal Energy: The total thermal energy: $E + E_{thermal}$

Total Enthalpy: The total enthalpy: $E + E_H$

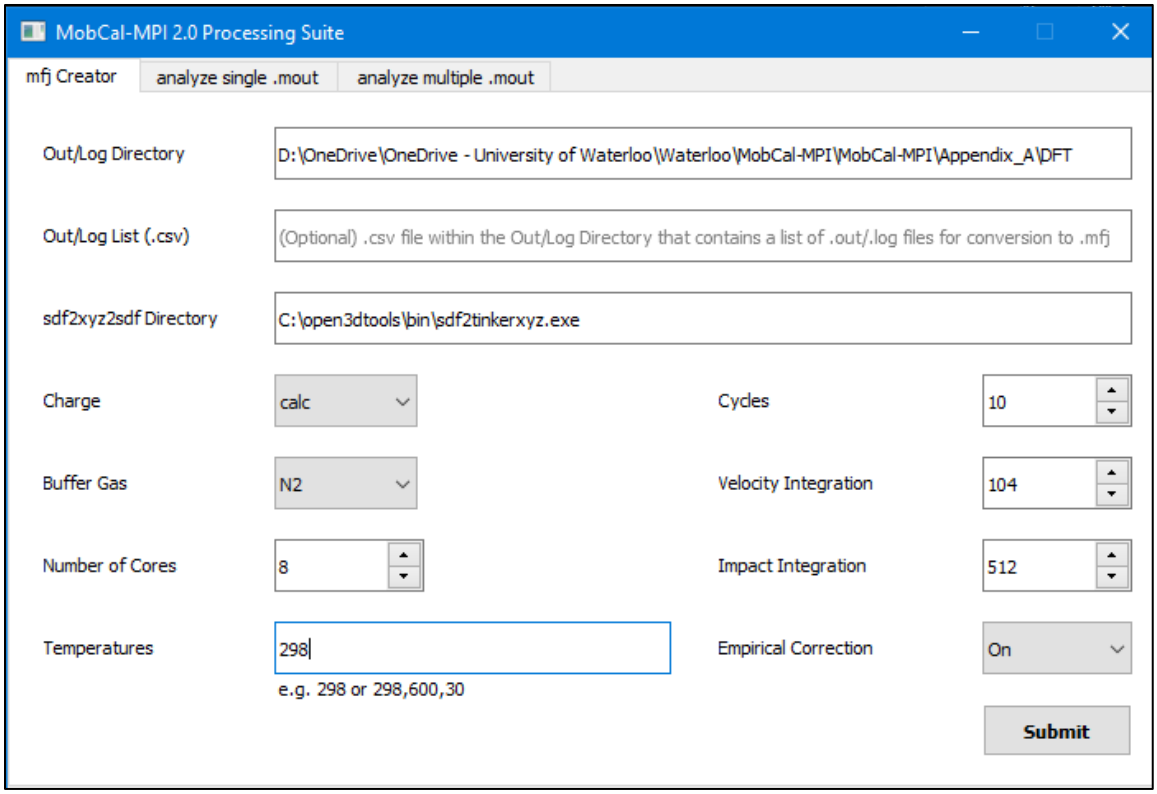
Total Entropy: The total entropy determined from electronic, vibrational, rotational, and translational contributions [$T*S = T*(S_{elec} + S_{vib} + S_{rot} + S_{trans})$].

Total Gibbs Energy: The total Gibbs energy: $E + E_{Gibbs}$

Keep note of the Gibbs Corrections, as we will be using these values in conjunction with DLPNO-CCSD(T) electronic energies to Boltzmann-weight the associated CCSs.

A4.2 – Creating MobCal-MPI input files from the optimized DFT .out files

Once you have extracted the thermochemistry from the .out files, MobCal-MPI input files can be made. To do this, open the MobCal-MPI GUI, enter the mfj Creator tab, input the directory that contains all of the .out files in the Out/Log directory field, and populate the rest of the fields using the values shown in the figure below. For a description of what each field within the GUI represents, see Section 3.4.1. Once the .mfj files are created, submit them to MobCal-MPI according to the instructions in Section 4.2 and the values shown in the screenshot below.



A4.3 – Creating DLPNO-CCSD(T) input files for calculating electronic energies from DFT outputs (optional)

This section details the refinement of DFT electronic energies with those obtained from DLPNO-CCSD(T) calculations. While this step is optional, we recommend performing DLPNO-CCSD(T) calculations due to their superior accuracy over DFT and modest computational cost. If users choose not to conduct DLPNO-CCSD(T) calculations, they can still follow the workflow outlined in this appendix, but substitute references to DLPNO-CCSD(T) electronic energies with DFT electronic energies.

Creating the input files for DLPNO-CCSD(T) calculations from the DFT output files is done using the ORCA_out_to_ORCA_inp.py script, which can be found in the "Appendix_A" subdirectory on the main GitHub repository. This script extracts the optimized geometry from each .out file and writes it to a new ORCA .inp file along with the DLPNO-CCSD(T) calculation line, which is requested via:

! DLPNO-CCSD(T) def2-TZVPP def2-TZVPP/C VeryTightSCF

The ORCA_out_to_ORCA_inp.py script is set to generate DLPNO-CCSD(T) inputs by default. You only need to modify the directory variable to point to the location of the DFT .out files. Here's how the script's header should appear (excluding the system-specific directory location):

```
#Where are your xyz files located?
directory = r'C:\Users\Chris\OneDrive - University of Waterloo\Waterloo\MobCal-MPI\MobCal-MPI\Appendix_A\DFT'

#Important things for the input block
mpp = 3400          #Memory per processor in MB. Must be integer
ncores = 8          #Number of cores to use in the calculation
charge = 1          #What is the charge?
multiplicity = 1    #What is the multiplicity?

'''Typical workflow'''
#calc_line = '! wB97X-D3 TightOpt Freq def2-TZVPP def2/J RIJCOSX TightSCF defgrid3 CHELPG'    #DFT calcs, thermochemistry, and partial charges
calc_line = '! DLPNO-CCSD(T) def2-TZVPP def2-TZVPP/C VeryTightSCF'                           #DLPNO-CCSD(T) calcs
```

Running the code will generate a series on .inp files with the same filename as the corresponding .out file in a new directory called "New_Inputs." Each .inp file that is generated should be submitted for calculation via ORCA.

A4.4 – Calculating the Boltzmann weights using DLPNO-CCSD(T) electronic energies and DFT thermochemistry

After the DLPNO-CCSD(T) jobs are complete, move the .out files to a new directory on your system. We can extract the electronic energies from each .out file using the ORCA_CCSDT.py script. To extract electronic energy from each .out file, you can utilize the ORCA_CCSDT.py script. Simply specify the directory containing the DLPNO-CCSD(T) .out files using the directory variable (which must be a raw string), and then execute the script. The electronic energies will be displayed in a .csv file named DLPNO_CCSDT_Energies.csv, and when opened in Excel, it will appear as follows:

A	B
Filename	DLPNO-CCSD(T) energy
Fluoxetine_1.out	-1086.829471
Fluoxetine_16.out	-1086.829428
Fluoxetine_18.out	-1086.825801
Fluoxetine_35.out	-1086.821054
Fluoxetine_41.out	-1086.821537
Fluoxetine_55.out	-1086.82793
Fluoxetine_9.out	-1086.825741

Following the extraction of electronic energies using the ORCA_CCSDT.py script, you should copy the filename and CCSDT columns into a new Excel sheet. For guidance, please refer to the Fluoxetine_Analysis.xlsx example provided in the Appendix_A subdirectory. Additionally, transfer the Gibbs corr column from the Thermo_data.csv file created during the extraction of information from the DFT outputs into this new Excel sheet.

Now, to calculate the Gibbs energy for each Fluoxetine configuration, add the Gibbs correction to the DLPNO-CCSD(T) electronic energy. Subsequently, calculate the Gibbs energy of each configuration relative to the lowest energy configuration using the equation below. Keep in mind that

the values are initially in Hartree, so you will need to convert them to the more common unit of kJ/mol, where 1 Hartree equals 2625.5 kJ/mol.

$$E_{rel}(i) = (Gibbs_i - \min(Gibbs)) \cdot 2625.5$$

To summarize what we have done so far, the Excel sheet should look like this:

A	B	C	D	E
Filename	DLPNO-CCSD(T) energy	Gibbs corr	Gibbs Energy	Erel
Fluoxetine_1.out	-1086.829471	0.292979	-1086.536492	0
Fluoxetine_16.out	-1086.829428	0.292971	-1086.536457	0.091893
Fluoxetine_18.out	-1086.825801	0.29339	-1086.532411	10.71467
Fluoxetine_35.out	-1086.821054	0.292655	-1086.528399	21.24817
Fluoxetine_41.out	-1086.821537	0.293568	-1086.527969	22.37714
Fluoxetine_55.out	-1086.82793	0.292865	-1086.535065	3.746588
Fluoxetine_9.out	-1086.825741	0.292645	-1086.533096	8.916198

With each column containing the following information:

Filename: The name of each file, as taken from the Filename column in the ORCA_CCSDT.csv file.

DLPNO-CCSD(T) energy: The DLPNO-CCSD(T) electronic energy taken from the CCSDT column in the ORCA_CCSDT.csv file.

Gibbs corr: The thermochemical correction for Gibbs energies, taken from the G corr column in the Thermo_data.csv column (DFT calculation).

Gibbs Energy: The sum of the entries in the DLPNO-CCSD(T) and Gibbs corr columns in this excel sheet.

Erel: The energy of the ith configuration relative to the lowest energy configuration (in kJ/mol). $E_{rel}(i) = (Gibbs_i - \min(Gibbs)) \cdot 2625.5$

With the relative Gibbs energy (E_{rel}) in hand, we can proceed to calculate their respective Boltzmann weights. To accomplish this, we must determine the population (n) of each fluoxetine conformer in the gas-phase ensemble using the following equation:

$$Rel\ Pop(n) = \frac{\exp\left(\frac{E_{rel}}{RT}\right)}{\sum_n \exp\left(\frac{E_{rel}}{RT}\right)}$$

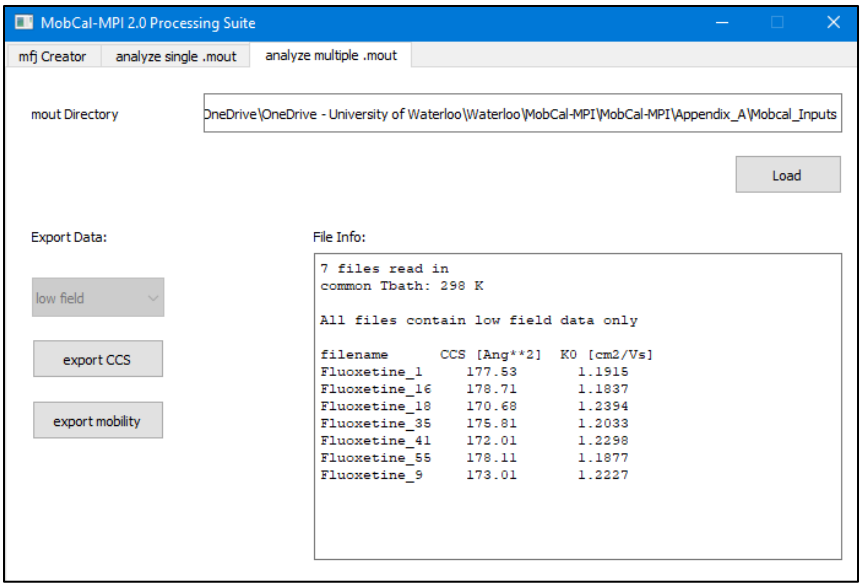
To simplify this calculation in Excel, begin by computing the numerator of the equation. This corresponds to the "Pop" column in the provided figure. In this calculation, use the values R = 8.3145E-3 kJ/mol·K and T = 298.15 K. Subsequently, you can calculate the relative population

by dividing each of these computed values by the sum of all the populations; this is the "Rel pop" column.

F	G
Pop	Rel pop
1	0.449395
0.96361	0.433041
0.01327	0.005964
0.000189	8.51E-05
0.00012	5.4E-05
0.220612	0.099142
0.027413	0.012319

A4.5 – Calculating the Boltzmann-weighted CCS

Subsequently, you can load the CCSs (Collision Cross Sections) of each fluoxetine configuration into the Excel sheet. To do this, begin by transferring the .mout files to a directory on your computer. Then, navigate to the "analyze multiple .mout" tab of the MobCal-MPI GUI and input the directory containing the .mout files. Once the directory is specified, choose the "Load" option, followed by the "Export CCS" operation. The CCS values will be exported to a .csv file named "Export_mout_CCS_lowfield.csv."



Copy the CCSs from Export_mout_CCS_lowfield.csv into the Excel sheet. The Boltzmann weighted CCS is then calculated by weighted each conformers CCS by its population in the gas-phase ensemble:

$$CCS = \sum_{n} CCS(n) \cdot Rel\ Pop(n)$$

Following this method yields a Boltzmann-weighted CCS of 178.0 $Å^2$ in N_2, which is a deviation of 2.0% from its experimentally measured CCS of 174.5 by Hines *et al.*

Note that the CCS you calculate will be slightly different from 178.0 $Å^2$ due to differences in the conformers generated from CREST and due to differences in the random seed number that determines the starting position of N_2 when sampling collision trajectories. However, the CCS you calculate should be within the uncertainty of the CCS calculation (*ca.* 1.8 %).

Appendix B – Citing MobCal-MPI

Any usage of MobCal-MPI should cite the following papers:

(1) Haack, A., Ieritano, C., & Hopkins, W. S. MobCal-MPI 2.0: an accurate and parallelized package for calculating field-dependent collision cross sections and ion mobilities. *Analyst* **2023**, *148*, 3257–3273. https://doi.org/10.1039/d3an00545c.

(2) Ieritano, C.; Crouse, J.; Campbell, J. L.; Hopkins, W. S. A Parallelized Molecular Collision Cross Section Package with Optimized Accuracy and Efficiency. *Analyst* **2019**, *144* (5), 1660–1670. https://doi.org/10.1039/c8an02150c.

(3) Ieritano, C.; Hopkins, W. S. Assessing Collision Cross Section Calculations Using MobCal-MPI with a Variety of Commonly Used Computational Methods. *Mater. Today Commun.* **2021**, *27*, 102226. https://doi.org/10.1016/j.mtcomm.2021.102226.

Appendix C – Supporting Content

C1. A brief review of other CCS calculation tools

1. In the mid-1990s, the MobCal program emerged for evaluating CCSs in helium.[93,94] Mobcal gained substantial recognition for its accurate CCS prediction capabilities, which it accomplished using three methods:

    - the Projection Approximation (PA), in which the "shadow" of the ion is averaged over all orientations,
    - the Exact Hard Sphere Scattering (EHSS) model, where ion-neutral collision trajectories are evaluated using a purple hard-sphere scattering interaction potential.
    - the Trajectory Method (TM), which is like the EHSS model, but employs a more realistic Lennard-Jones (LJ) 12-6 potential.

    As the precision of these techniques improves (progressing from PA to TM), so does their computational overhead. In 2012, the MobCal code was extended to support N_2 as a collision gas,[95,96] leading to additional modifications of the interaction potential to consider N_2's polarizability and quadrupole moment.[37,97] Currently, our latest enhancements encompass parallelization, atom-specific ion-neutral interaction potentials, and the inclusion of 2TT, all of which constitute the most recent update known as MobCal-MPI.[98,99]

2. A lesser-known variant of the MobCal code is the HPCCS program,[100] which also utilizes the TM and can compute CCSs in He and N_2.

3. The IMoS code has become quite popular in recent years.[101–103] IMoS takes a distinct approach by explicitly computing the drag experienced by the ion from the surrounding gas. It achieves this through a trajectory method that incorporates realistic interaction potentials and considers the velocities of both the ion and the gas molecules. This differs from MobCal, which focuses on the relative velocity of the ion-neutral pair. Despite these mathematical differences compared to MobCal, research has demonstrated that both methods yield comparable results.[104] IMoS then obtains the ion mobility coefficient, K, from the calculated drag and the CCS through rearranging Eq. (4) of the main manuscript.

    It is worth highlighting that in 2023, IMoS underwent an update to include 2TT to enable the calculation of ion mobility at arbitrary field strengths.[105] However, this update involved implementing a TM methodology akin to MobCal instead of calculating the drag experienced by the ion.

4. Collidoscope stands out as another CCS predictor that employs the TM approach and incorporates a 12-6 Lennard-Jones interaction potential for both helium (He) and nitrogen (N2) as collision gases.[106] Benchmarking indicates that its accuracy is similar to IMoS. Furthermore, the code's parallelization ensures it offers a similar level of speed as MobCal-MPI, HPCCS, and IMoS.
5. When dealing with large ions such as proteins or supramolecular complexes, employing the Trajectory Method (TM) becomes both computationally impractical and unnecessary. This is primarily due to the

significant size of these ions, where intricacies in scattering trajectories become less important when evaluating CCS. Consequently, projection approximation algorithms can yield sufficiently accurate results in a fraction of the time required to compute the CCS via the TM. A notable extension of the PA is the Projection Superposition Approximation (PSA).[107–110] By meticulously calibrating collision probabilities, the PSA can replicate temperature-dependent variations in the CCS, a feat that is not possible with the traditional PA approach.

C2. Collision dynamics

The amount of momentum transferred upon a collision between two particles (at relative velocity g, impact parameter b and assuming a spherically symmetric interaction potential), can be calculated from the scattering angle, χ, and is proportional to $(1 - \cos\chi)$. In **Figure C1**, some example trajectories are shown for two different relative velocities and a set of impact parameters (ranging from $b = 0$ to a cut-off value b_{max}). It further shows the amount of momentum being transferred as a function of the impact parameter. The integral under these curves is proportional to the momentum transfer cross section, $Q^{(1)}(g)$.

We can separate collision events into two categories: glancing collisions (large b, only probing the attractive part of the potential) and striking collisions (small b, probing both the attractive and repulsive part of the potential). Both kinds contribute to the overall momentum transfer for small velocities but as can be seen in **Figure C1**, for higher relative velocities, only the striking collisions transfer momentum. At higher velocities, trajectories with large b do not significantly contribute to the total momentum transfer.

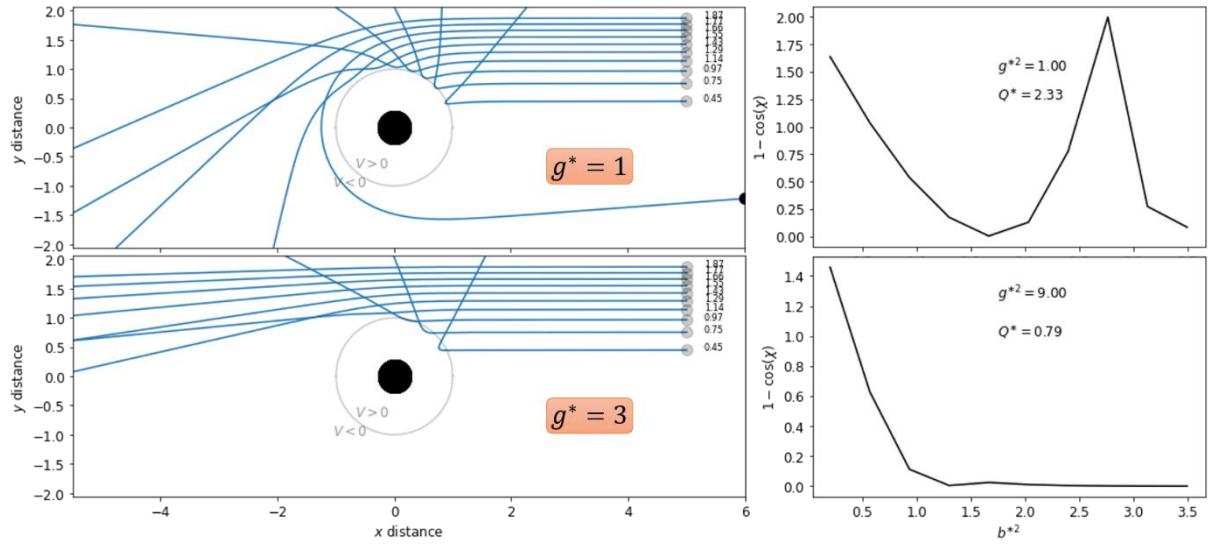


Figure C1. Left: Collision trajectories for two different relative speeds, g^*, and for a set of impact parameters, b^* (in dimensionless units). **Right**: Corresponding momentum transfer as a function of b^{*2}. The interaction potential is assumed to be spherically symmetric, and the grey circle denotes the distance at which the potential becomes repulsive.

Thus, when integrating over both the velocity as well as impact parameter space, using a common upper limit, b_{max}, will result in a lot of trajectories that do not contribute to the overall momentum transfer integrals. This can be seen in **Figure C2**, which shows the momentum transfer as contour plot over both

velocity and impact parameter space. Most trajectories (started at pairs of (b, g)) would yield close to zero momentum transfer and can thus be neglected to save computing time. Consequently, it is beneficial to calculate a b_{max} value for each velocity grid point. This was already done in the original MobCal implementation, the strategy for which is retained in MobCal-MPI 2.0. This is depicted with the red line in **Figure C2**. This way, at every of the *inp* velocity points, *imp* impact parameter (and orientation) samples are taken, but only for $b \leq b_{max}(g)$, where there is significant momentum transfer.

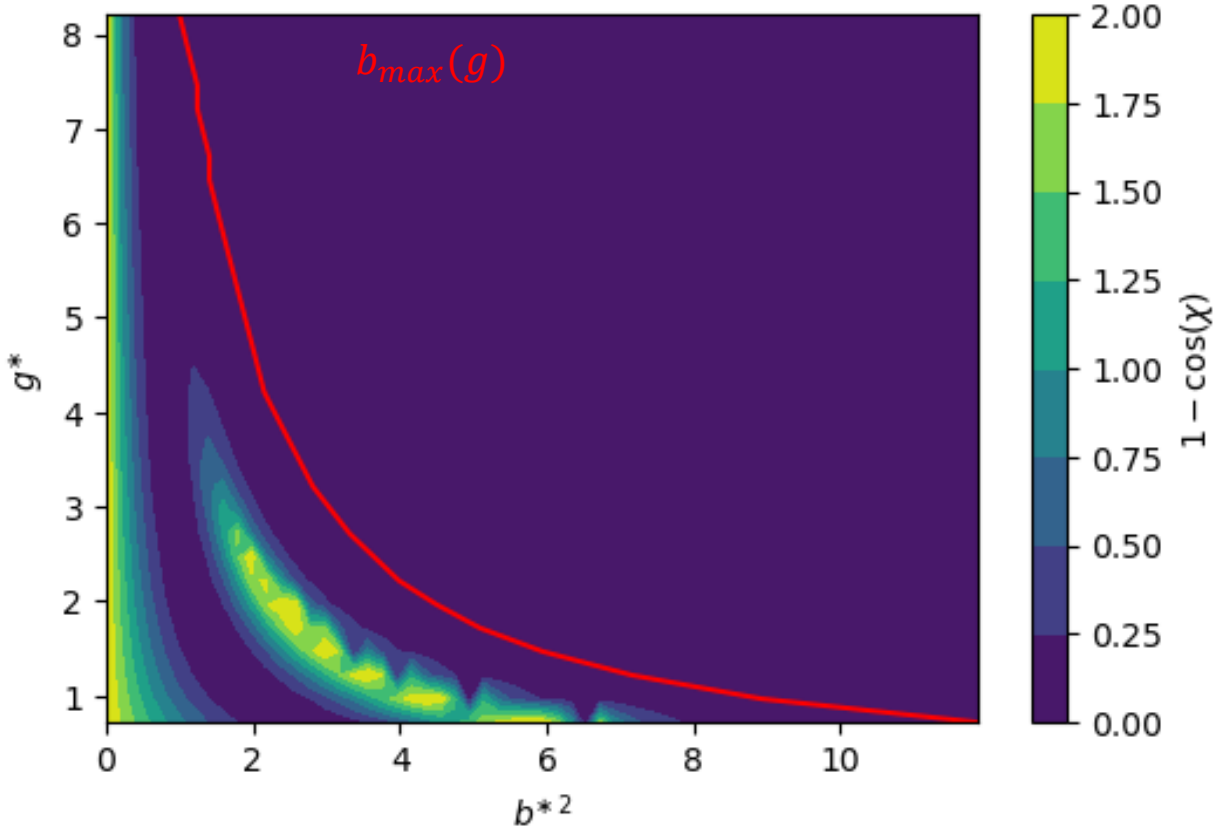


Figure C2. Contour plot of $1 - cos\,\chi$ (proportional to the momentum transfer) as a function of the impact parameter, b, and the relative velocity, g, of the colliding particles (both in dimensionless units). The red line denotes the threshold values $b_{max}(g)$ for every velocity point.

C3. Evaluating Collision Integrals using the Chapman-Enskog Formalism

The motion of ions through a gas is governed by the Boltzmann transport equation.[111] The quantity of interest in this framework is the ion velocity distribution, f_{ion}. One can calculate the drift velocity (v_D) as a moment of the ion velocity distribution (f_{ion}; Eq. C1), for which we use the convention that the electric field points along the z-axis. Note that n is a normalization factor.[6]

$$v_D = \langle v_z \rangle = \frac{1}{n} \int v_z f_{ion}\, dv \qquad \text{(C1)}$$

Although the velocity distribution of the bath gas particles (f_{bg}) can be described by a Maxwell-Boltzmann distribution at temperature T, f_{ion} will be distorted compared to f_{bg} due to the acceleration caused by the electric field. Within the framework of 2TT, field-induced acceleration of the ion is accommodated by introducing a second temperature (T_{ion}) that accounts for the skewed velocity distribution. The effective temperature reflects the distribution of *relative* velocities between bath gas particles and ions. In general, $T_{ion} \geq T_{eff} \geq T$. To solve for f_{ion}, one can apply the Chapman-Enskog formalism,[64,112] which expresses f_{ion} as a Taylor series of basis functions. 2TT expands on this formalism by using Burnett-like basis functions, which are defined in Eqs. C2 and C3:[24,25]

$$\psi_{lm}^{(r)}(w_{ion}) = w_{ion}^{l} S_{l+1/2}^{(r)}(w_{ion}^2)(Y_l^m(\theta,\phi)) \tag{C2}$$

$$w_{ion}^2 = \frac{mv^2}{2k_B(T_{ion})} \tag{C3}$$

Here, $S_{l+1/2}^{(r)}$ are Laguerre polynomials, Y_l^m are the spherical harmonics, m is the molecular mass of the ion, and v is its velocity. w_{ion} acts as dimensionless velocity to simplify the equations. Rather than solving for the full ion velocity distribution, 2TT attempts to find solutions only to the moments of these basis functions. This is sufficient for ion mobility because the drift velocity can be expressed as moment of $\psi_{10}^{(0)}$, as per Eq. C4.

$$v_D = \sqrt{\frac{2k_B(T_{ion})}{m}}\langle\psi_{10}^{(0)}\rangle \tag{C4}$$

Iterative computation is used to solve for these moments, with higher-order approximations requiring additional iterations. The specifics of the iterative procedure are beyond the scope of this manuscript, although interested readers are directed to references [24] and [25] for further information.

C4. Velocity grid limits

To ensure that the integration over the relative velocity is accurate, limits for the velocity grid must be chosen such that the integrand, $Q^{(l)}\omega^{(s)}$, is significantly close to zero at those limits. Since it is not possible to predict the magnitude and functional dependency of $Q^{(l)}(g)$, we focused on the weight functions $\omega^{(s)}$. *I.e.*, we want to find velocity limits such that upon integration, the error caused by the cutoff becomes negligible. To simplify the algebra, we work with unitless velocity and temperature:

$$g^{*2} = \frac{\mu g^2}{2\epsilon'} \tag{C5a}$$

$$T^* = \frac{k_B T}{\epsilon'} \tag{C5b}$$

Here, μ is the reduced mass of the collision pair, k_B is the Boltzmann constant, and $\epsilon' = 1.34$ meV is a fixed energy constant.[93] Writing the weight functions now in dimensionless units:

$$\omega^{(s)}(g^*, T^*) = \left[\frac{(s+1)!}{2} T^{*s+2}\right]^{-1} \times g^{*2s+3} \exp\left(-\frac{g^{*2}}{T^*}\right) \tag{C6}$$

we can express their integrals as:

$$\int_0^{G_{upper}} \omega^{(s)}(g^*, T^*)\, \mathrm{d}g^* = 1 - \frac{1}{(s+1)!}\, \Gamma\left(s+2, \frac{G_{upper}^2}{T^*}\right) \tag{C7a}$$

$$\int_{G_{lower}}^{\infty} \omega^{(s)}(g^*, T^*)\, \mathrm{d}g^* = \frac{1}{(s+1)!}\, \Gamma\left(s+2, \frac{G_{lower}^2}{T^*}\right) \tag{C7b}$$

where Γ is the upper incomplete gamma function and G_{lower} and G_{upper} define the lower and upper limit for the velocity integration, respectively. From this, we can obtain and G_{lower} and G_{upper} by defining acceptable errors, *i.e.*, asking how close the integrals of Eq. (S7) should be to unity:

$$\frac{1}{(s+1)!}\, \Gamma\left(s+2, \frac{G_{upper}^2}{T^*}\right) = \xi_{upper} \tag{C8a}$$

$$\frac{1}{(s+1)!}\, \Gamma\left(s+2, \frac{G_{lower}^2}{T^*}\right) = 1 - \xi_{lower} \tag{C8b}$$

Because $Q^{(l)}(G_{lower}) \gg Q^{(l)}(G_{upper})$, we pick a tighter value for ξ_{lower}. However, a too low G_{lower} will result in lost trajectories because at very low relative velocities collision particles will stick to/orbit the ion. In the end, we chose:

$$\xi_{upper} = 10^{-3} \overset{s=4}{\rightleftarrows} G_{upper} = \sqrt{16.455 \cdot T_{upper}^*} \tag{C9a}$$

$$\xi_{lower} = 10^{-4} \overset{s=1}{\rightleftarrows} G_{lower} = \sqrt{0.0862 \cdot T_{lower}^*} \tag{C9b}$$

Here, T_{lower}^* is calculated from T_{bath} and T_{upper}^* is calculated from $T_{eff,max}$ according to Eq. (C1b).

C5. Avg-N2 versus CoM-N2 Potentials

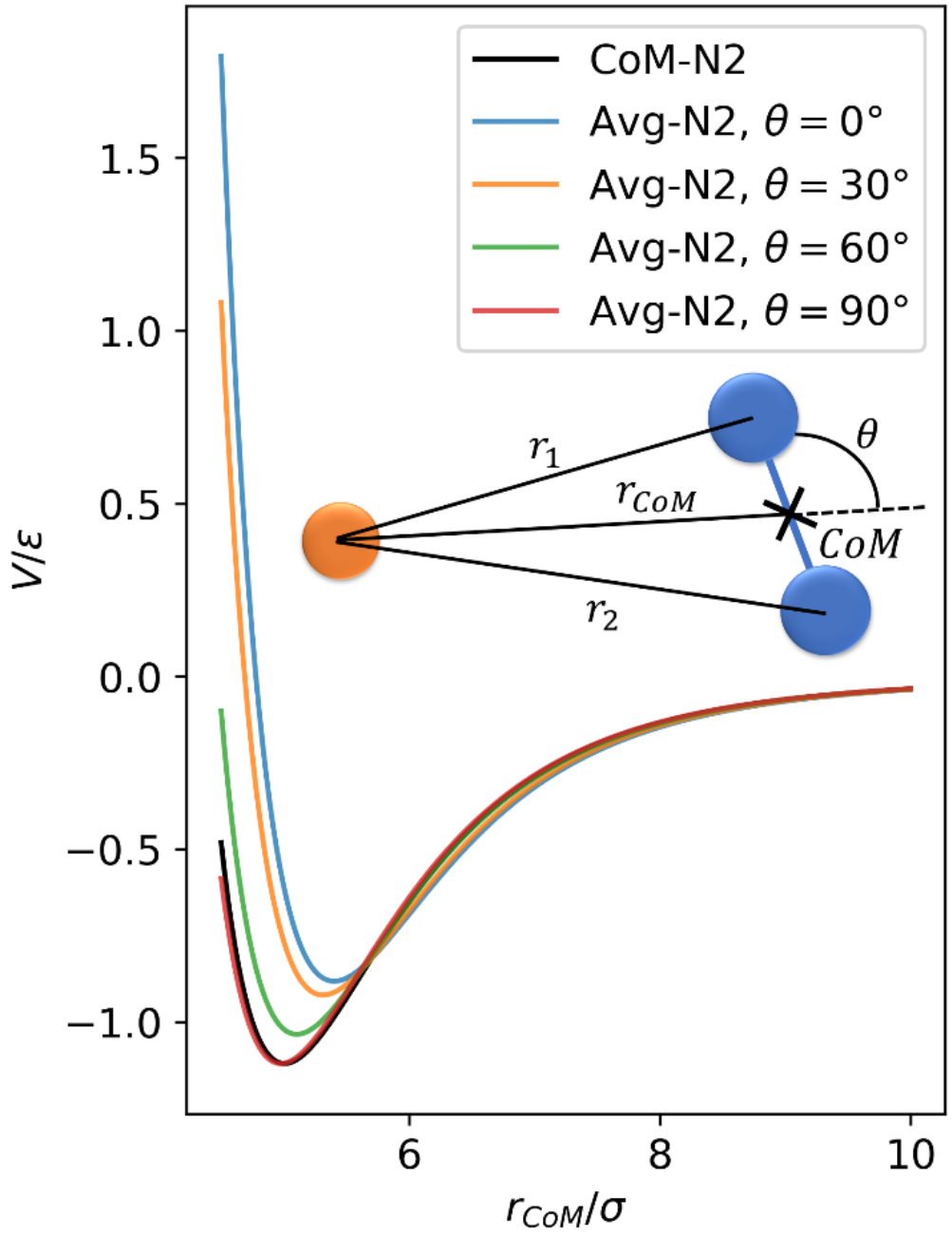


Figure C3. Van-der-Waals interaction potentials (Exp-6) are used for both the CoM-N_2 and Avg-N_2 versions. These potentials vary depending on the orientation of the N_2 molecule relative to the ion, which is described by the angle θ. For the CoM-N_2 scheme, the interaction potential is independent of θ. Conversely, the Avg-N_2 potential exhibits a pronounced dependence on θ such as when $\theta = 0°$ (head-on collision), where the potential becomes rapidly repulsive with decreasing r_{CoM}. This behavior aligns with a more realistic scenario, as during a head-on collision, one end of the N_2 molecule approaches the collision partner much earlier than its center of mass (CoM).

C6. Precision for high-field calculations

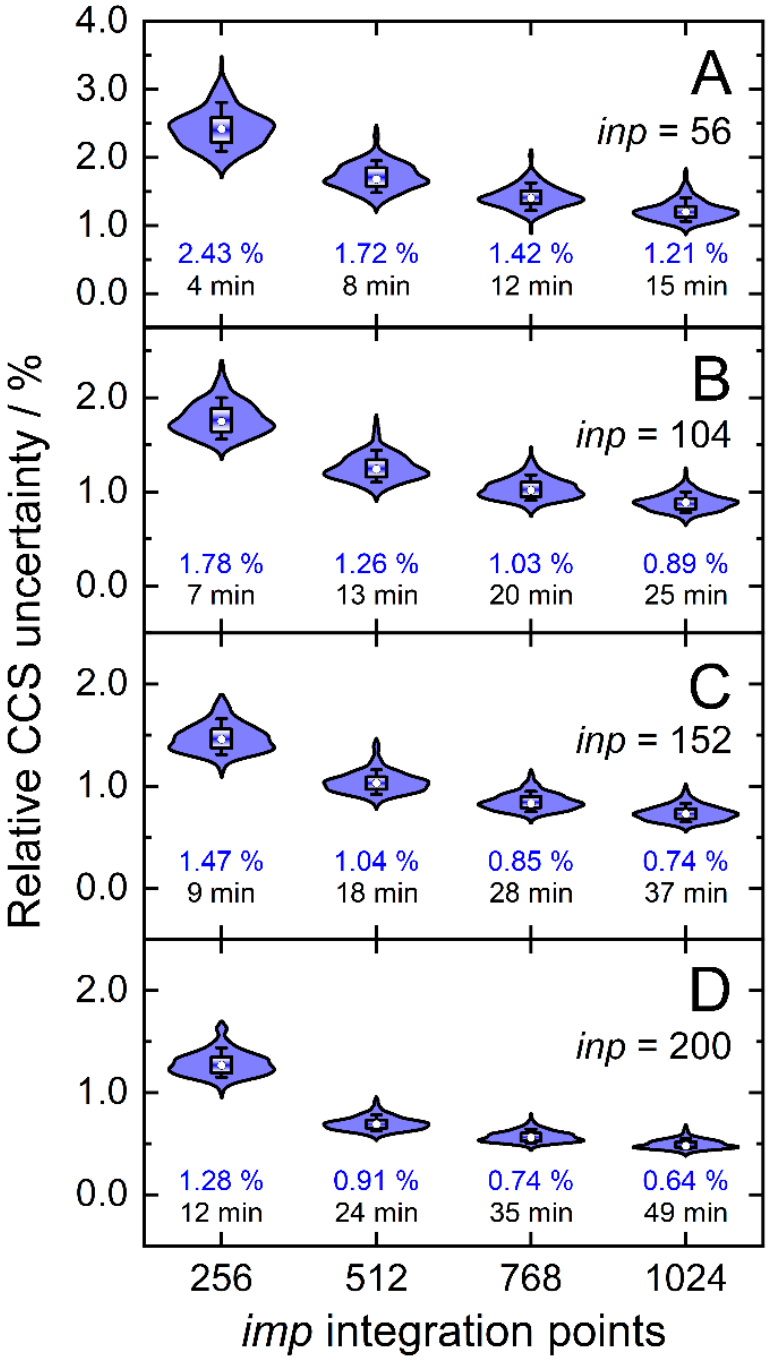


Figure C4. Distributions of relative CCS uncertainties $(\sigma_{CI}(\Omega^{(1,1)})/\Omega^{(1,1)})$ for the validation set (N = 50) for different combinations of velocity sample points (inp) and orientation/impact parameter sample points (imp) when calculating mobilities/CCS between 298 K and 800 K. The blue numbers below each distribution correspond to mean relative CCS uncertainties and the black numbers to average computing time.

C7. The connection between alpha functions and dispersion plots

The alpha function of an ion describes how the mobility at higher field strengths $\left(K\left(\frac{E}{N}\right)\right)$ deviates from its value within the low-field limit $K(0)$:

$$\alpha\left(\frac{E}{N}\right) = \frac{\left(K\left(\frac{E}{N}\right)\right)}{(K(0))} - 1 \tag{C10}$$

In the realm of differential mobility spectrometry, an asymmetric, oscillating field is employed to investigate this alpha function. This field encompasses both high- and low-field strengths and is denoted as the separation field, E_S(t). Various forms of E_S(t) are possible, including square wave and double sine waveforms, among others.[113] However, a key requirement is that the integral over a single duty-cycle is zero:

$$\frac{1}{\tau}\int_0^{\tau} E_S(t)\,\mathrm{d}t = \langle E_S(t)\rangle = E_S\langle f(t)\rangle = 0 \tag{C11a}$$

$$\langle f^{2n+1}(t)\rangle \neq 0 \tag{C11b}$$

where $E_S(t) = E_S f(t)$ with $f(t)$ being the normalized form of the waveform and E_S its amplitude (maximum value). τ denotes the time for one wave cycle.

In addition to the separation field, a constant field, denoted as E_C, is applied to ensure the transmission of ions. Therefore, the total field is given by:

$$E(t) = E_S f(t) + E_C \tag{C12}$$

In the present context, a dispersion plot is utilized to assess the compensation field, E_C, required for optimal ion transmission based on the amplitude of the separation field, E_S. To derive the alpha function from a dispersion plot,[10] Buryakov proposed the following procedure: suggested the following method:

Initially, the dispersion plot data is fitted to an uneven polynomial of the (2N+1)th order:

$$E_C = \sum_{n=1}^{N} B_{2n+1} E_S^{2n+1} \tag{C13}$$

The obtained coefficients B_{2n+1} can then be used to determine the alpha function, which is expressed as an even-ordered polynomial:[10,68]

$$\alpha(E) = \sum_{n=1}^{N} \alpha_{2n} E^{2n} \tag{C14a}$$

$$\alpha_{2n} = \frac{1}{\langle f^{2n+1}(t)\rangle}\left\{B_{2n+1} + \sum_{k=1}^{n-1}(2(n-k)+1)B_{2k+1}\alpha_{2(n-k)}\langle f^{2(n-k)}\rangle\right\} \tag{C14b}$$

t's important to acknowledge that this workflow may exhibit discrepancies at higher field strengths, a common occurrence with polynomial fittings. However, the lower end ($E \to 0$) poses no concerns because the requirement of $B_0 = \alpha_0 = 0$ ensures the correct behaviour. Specifically, it ensures that $E_C(E_S = 0) = 0$, and that $\alpha(E = 0) = 0$. Conversely, it's also possible to predict the dispersion plot when the alpha function is known. If E_C is anticipated to be significantly smaller than E_S, Buryakov's first-order approximation can be used:[10]

$$E_C = \frac{\langle \alpha f(t)\rangle E_S}{1 + \langle \alpha\rangle + \langle \alpha' f(t)\rangle E_S} \tag{C15}$$

Please note that α is indeed a function of $\big(E_S(t)\big)$, but for the sake of clarity, we have omitted the field dependency in our notation. Additionally, α' represents the derivative of α with respect to E. However, this equation breaks down if E_C becomes sufficiently large, as recently showed in the literature.[114] In such cases, the compensation value, E_C, can be determined iteratively by beginning with a guess, $E_C^{(guess)}$, and subsequently calculate the ion's net drift velocity induced by the waveform's asymmetry using the following expression:

$$\langle v_D\rangle = \langle K \cdot [E_S f(t) + E_C^{(guess)}]\rangle \tag{C16}$$

Once more, it's important to emphasize that we've excluded the explicit notation of the mobility's dependence on the total separation field: $K = K\big(E(t)\big) = K(E_S f(t) + E_C)$. The mobility can be derived from the alpha function by rearranging Eq. (S8). We then use the obtained net drift velocity to obtain a new guess for the separation field as follows:

$$E_C^{(new)} = E_C^{(old)} - \frac{\langle v_D\rangle}{K(0)}\delta_{damp} \tag{C17}$$

where δ_{damp} is a damping factor usually used in iterative procedures (typically 0.8). The new guess of E_C is used again in Eq. (S14), and this iterative process continues until $\langle v_D\rangle < v_{thresh}$, indicating that the remaining drift velocity has become negligible.

C8. Details concerning the empirical correction to 2TT

C8.1. Choice of functional form

Siems *et al.* conducted a comparison between 2nd order 2TT and a precise modeling approach using the Gram-Charlier (GC) method for idealized test systems, specifically atomic ions in atomic gases.[115] Their findings indicated that 2TT prodcues accurate results across a wide range of field strengths (0-50 Td) but exhibits deviations of 5-7% beyond 100 Td.

In cases involving "heavy ions" (*i.e.,* where the mass of the ion is significantly greater than the bath gas), deviations are absent at low fields. However, with rising field strengths, 2TT progressively underestimates ion mobilities, eventually reaching a constant level of underestimation at extremely high field strengths.

Siems *et al.* compared 2nd order 2TT with accurate modelling of an ion's mobility using the Gram-Charlier (GC) approach for idealized test systems, *i.e.*, atomic ions in atomic gases. They found that 2TT yields good results over a broad range of field strength (0-50 Td) but shows 5-7% deviations above 100 Td. For the "heavy ion" case, *i.e.*, when the ion is much heavier than the bath gas particle, the deviations are zero at low fields, then the 2TT mobilities become increasingly too small, eventually reaching a constant underestimation at very high field strengths. To address this discrepancy, we should select a formulation that mirrors this pattern: no deviations at low fields, a sustained deviation at very high fields, and a seamless transition in between. Consequently, we chose the following function:

$$K_{corr} = f_{corr} \cdot K_{2TT} \tag{S18a}$$

$$f_{corr}\left(\frac{E}{N}\right) = 1 + A \exp\left(-\frac{B}{E/N}\right) \tag{S18b}$$

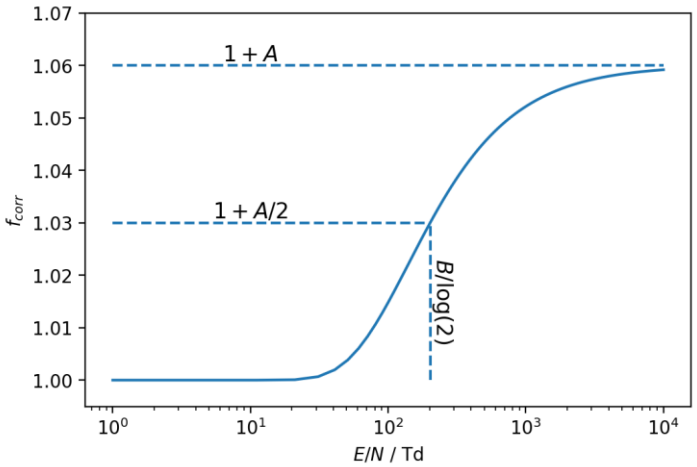


Figure C5. The functional form of the empirical correction used in MobCal-MPI (Eq. S18). The parameters A and B play key roles in determining the behaviour of this function: A dictates the limit at extremely high fields, while B governs the rate of its increase. Here, we used representative values of $A = 6\ \%$ and $B = 140$ Td.

As depicted in Figure , the shape of f_{corr} satisfies the requirements of the empirical correction. Specifically, at low fields $f_{corr} = 1$, which results in $K_{corr} = K_{2TT}$. As the field strength increases, f_{corr} progressively increases, and eventually reaches a constant value of $K_{corr} = (1 + A)K_{2TT}$ at very high E/N, representing a relative mobility increase of A (%). The rate in which the transition between $f_{corr} = 1$ and $f_{corr} = 1 + A$ occurs is determined by the parameter B.

Additional support for the chosen functional form is provided by comparing the disparities between the measured and predicted alpha functions of protonated amoxapine (see Fig. 6A in Section 2.7). Because our correction operates multiplicatively, we can derive the required correction by considering the experimental and 2TT-predicted alpha functions using the following formula:

$$f_{true} = \frac{\alpha_{expt} + 1}{\alpha_{2TT} + 1} \tag{C19}$$

C8.2. Examples of dispersion plots utilizing the empirical correction

We utilized calculated mobilities based on 2nd order 2TT (K_{2TT}) to simulate the dispersion plot behavior for the 132 compounds in the "high-field validation set" using the approach outlined in Section S7. Generally, we observed that 2TT tends to overestimate the measured dispersion plots, consistent with the deviations noted by Siems et al. To assess the extent of correction required for K_2TT, we conducted parameter fitting for all 132 compounds to achieve the most accurate prediction of the experimental dispersion plots. The average values for parameters A and B obtained from this fitting process were then employed as a common empirical correction.

The outcomes of this process are depicted in **Figure C6** for two illustrative examples. Without any correction, 2TT yields dispersion plots that always overestimate the CV required for ion transmission compared to experiment. However, when we individually fit the empirical correction parameters A and B to each ion, the resulting dispersion plots fall well within the range of experimental uncertainties. This outcome instills confidence in the suitability of the selected functional form for this purpose. **Figure C6** also shows the computed dispersion plots that employed fixed empirical correction parameters (A_{avg}, B_{avg}).

In the case of Bentazon, applying an individually optimized and universal empirical correction yields excellent agreement with the experimental data. This is primarily because the empirically fitted correction parameters closely resemble the averaged values, ensuring a strong match. In contrast, the fixed empirical correction exhibits more pronounced discrepancies for salicylic acid when compared to the individually fitted corrections and the experimental data. This discrepancy becomes apparent when considering the correction parameters, which significantly vary between the individually and fixed empirical corrections. Nevertheless, we argue that the dispersion plot incorporating the fixed empirical correction more closely resembles the experimental data or, at the very least, provides a level of alignment similar to that of the uncorrected 2TT predictions.

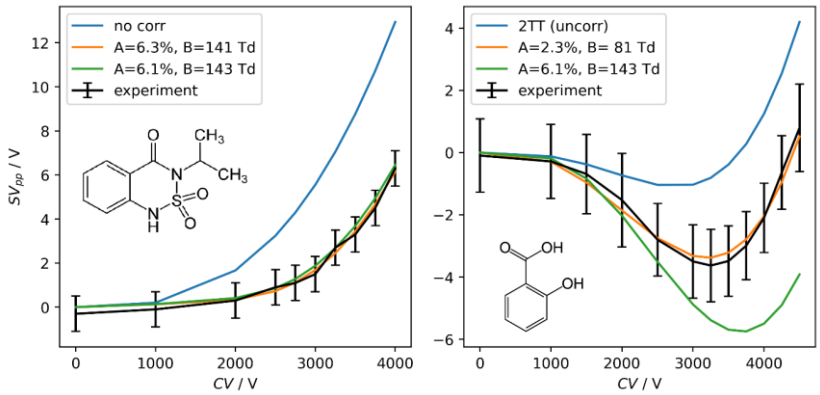


Figure C6. Measured and calculated dispersion plots of protonated Bentazon (**left**) and protonated salicylic Acid (**right**). The calculated dispersion plots are presented in three scenarios: first, for uncorrected 2nd order 2TT; second, for 2TT with individually optimized empirical correction parameters; and third, for 2TT with a consistent empirical correction, where A_{avg} = 6.1 % and B_{avg} = 143 Td.

C8.3. Correlation of *A* and *B* fit parameters to physicochemical properties

Prior to adopting the average values of the A and B parameters, i.e., A_{avg} and B_{avg}, we tried to establish correlations between the optimized A and B parameters and properties pertinent to the simulation of collision dynamics. However, our analysis did not reveal any significant correlations between these parameters and the selected properties, as illustrated in **Figure C7**.

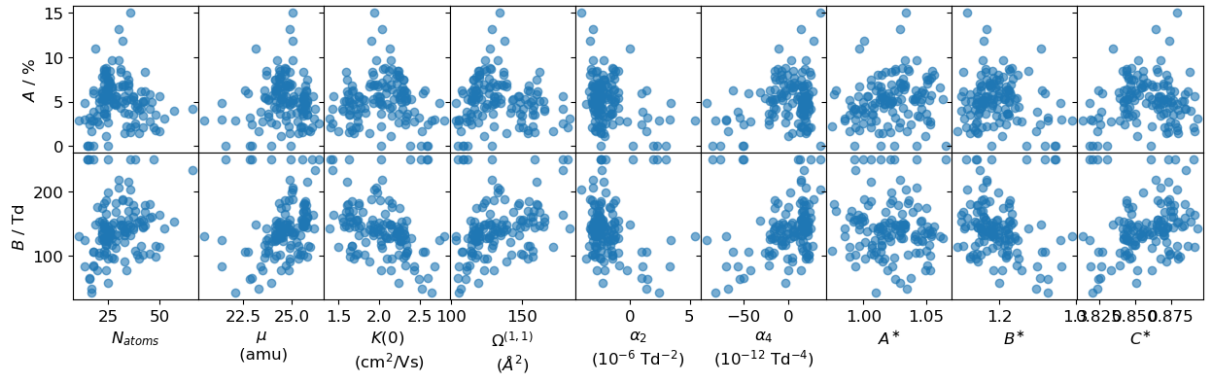


Figure C7. Correlation between the A and B parameters to typical parameters appearing in the formalism of ion mobility, namely the number of atoms (N_{atoms}), the reduced mass (μ), the low field mobility ($K(0)$), the first collision integral ($\Omega^{(1,1)}$), alpha function coefficients (α_2 and α_4) and important ratios of collision integrals ($A^* = \Omega^{(2,2)}/\Omega^{(1,1)}$, $B^* = (5\Omega^{(1,2)} - 4\Omega^{(1,3)})/\Omega^{(1,1)}$, and $C^* = \Omega^{(1,2)}/\Omega^{(1,1)}$).

C9. Additional plots from benchmarking MobCal-MPI 2.0

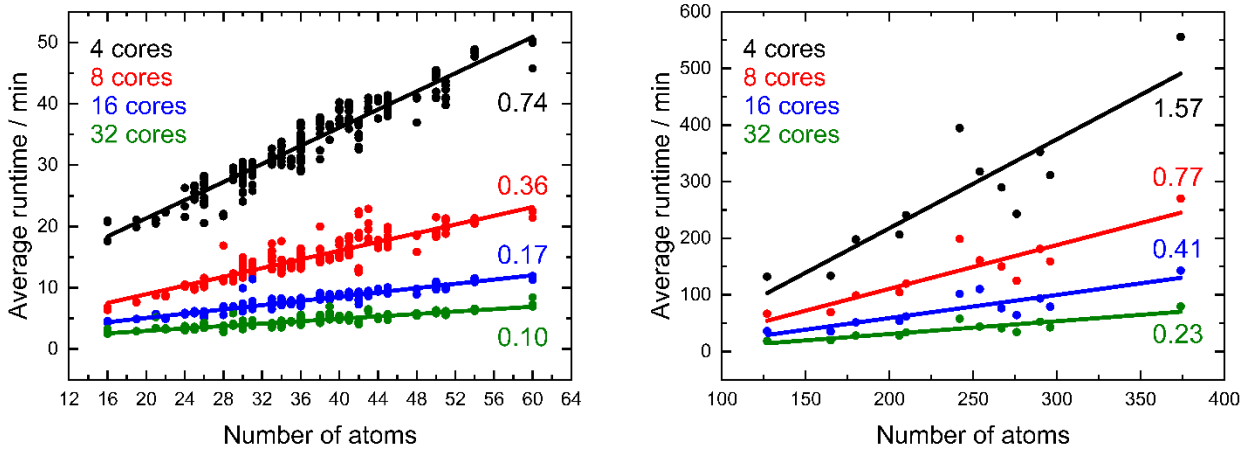
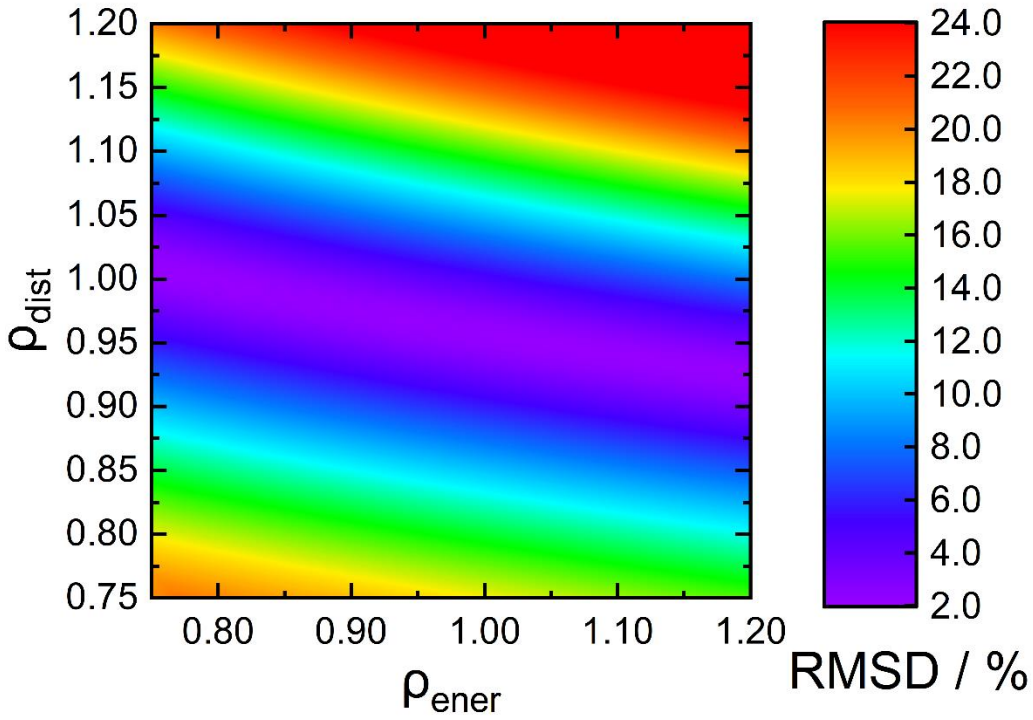


Figure C8. Benchmarking of runtimes to calculate CCSs in dependence of the number of cores used in parallel computing. Slopes corresponding to linear regression of the benchmarking data are shown on the right side of each figure. **(Left):** Test data is composed of the 238 different conformers of the validation set. **(Right):** Test data is composed of 12 peptides from the peptide set. All calculations are repeated for three different random seed numbers. The linear behavior shows the expected $\mathcal{O}(N)$ dependency with respect to the number of atoms, whereas the $1/N_{cores}$ decrease in the slopes (numbers adjacent to the regressions) indicates efficient parallelization.

Appendix D – Calibration of vdW parameters for CCS calculations in He

Before MobCal-MPI 2.0, the ρ_{ener} and ρ_{dist} parameters for evaluating CCSs in He (as defined by **Eqs. 23** and **24** in Section 2.4) were optimized by the Kim group using 23 analytes.[116] In Version 2.0, we have enhanced the optimization of ρ_{ener} and ρ_{dist} by supplementing the original set with 12 additional analytes commonly used for calibrating CCSs on the TWIMS platform.[44] You can find information on the identities, CCS values, and their respective sources in the table provided on the following page. For *N*-protonated Verapamil, the Boltzmann-weighting scheme includes the R,R and R,S protonation-induced diastereomers,[117] as these stereoisomers cannot be resolved via low-field IMS techniques that employ He as the collision gas.

The Figure shown below illustrates a contour plot depicting the root mean square errors (RMSE) between the calculated CCS values (Boltzmann-weighted) and the experimentally measured CCS values for each of the 36 calibrant ions. The parameters ρ_{ener} and ρ_{dist} were systematically varied in the range of 0.75 to 1.20 in increments of 0.01. Similar to the optimization surfaces seen in Section 2.4 for N_2, there is no single set of scaling parameters that performs best for MobCal-MPI 2.0, but rather a range of values that yield RMSEs of ~ 2 %. In this case, we chose the scaling parameters $\rho_{ener} = 0.93$ and $\rho_{dist} = 0.97$ given their lowest RMSE between calculation and experiment and their proximity to unity.



Chemical Entity	Expt CCS / Å^2	Calc CCS / Å^2 $\rho_{ener} = 0.93$ $\rho_{dist} = 0.97$	%diff from expt CCS	Ref
1,2,4,5-Benzenetetramine	59.2	61.1	3.2	Lee 2017
3,4,5-Tribromopyridine	65.1	64.1	1.5	Lee 2017
3,5-DibromoanthranilicAcid	72.5	71.3	1.7	Lee 2017
4-Aminosalicylic Acid	62.5	62.2	0.5	Lee 2017
5,7-Dichloroquinolin-8-ol	71.2	70.2	1.4	Lee 2017
5-Aminosalicylic Acid	63.5	61.9	2.6	Lee 2017
9(10H)-Acridone	74.5	83.1	0.4	Lee 2017
9,9-Dimethyl-9,10-Dihydroacridine	82.8	74.6	0.1	Lee 2017
Acetaminophen	67	66.8	0.3	Campuzano 2012
Alprenolol	96.9	102.3	5.6	Campuzano 2012
Amantadine	67.2	64.2	4.5	Lee 2017
Betamethasone	115.1	116.1	0.9	Lee 2017
Carbamazepine	86.2	86.2	0.1	Lee 2017
Carbazole	69.6	70.4	1.2	Lee 2017
Clozapine-N-oxide	112.5	113.5	0.9	Lee 2017
Colchicine	128.6	127.8	0.6	Campuzano 2012
Dexamethasone	116.2	115.3	0.8	Lee 2017
Epineprine	75.5	76.1	0.8	Lee 2017
flufenamic acid	92.2	91.9	0.3	Lee 2017
Guanine	59.7	58.2	2.5	Lee 2017
Leflunomide	90.8	91.2	0.4	Lee 2017
Mefanamic Acid	90.9	91.2	0.3	Lee 2017
Methyl-2,6-dichloronicotinate	71.5	70.2	1.9	Lee 2017
Methylene blue	101.7	104.5	2.7	Lee 2017
Tetraethylamine	65.9	66.1	0.3	Campuzano 2012
N-ethylanaline	63	63.6	0.9	Campuzano 2012
Nicotine	72.4	72.7	0.5	Lee 2017
Tetramethylamine	48.5	47.9	1.2	Campuzano 2012
Tetrapropylamine	88.9	91.4	2.8	Campuzano 2012
Ondansetron	105.9	103.1	2.6	Campuzano 2012
Paraquat	84.7	85.7	1.2	Lee 2017
Reserpine	178.8	188.3	5.3	Campuzano 2012
Saccharin	65.4	64.8	0.9	Lee 2017
Thymine	55.7	54.4	2.4	Lee 2017
Verapamil	142.6	143.9	0.9	Campuzano 2012

Appendix E – References

(1) May, J. C.; McLean, J. A. Ion Mobility-Mass Spectrometry: Time-Dispersive Instrumentation. *Anal Chem* **2015**, *87* (3), 1422–1436. https://doi.org/10.1021/ac504720m.

(2) Dodds, J. N.; Baker, E. S. Ion Mobility Spectrometry: Fundamental Concepts, Instrumentation, Applications, and the Road Ahead. *J Am Soc Mass Spectrom* **2019**, *30* (11), 2185–2195. https://doi.org/10.1007/s13361-019-02288-2.

(3) Lanucara, F.; Holman, S. W.; Gray, C. J.; Eyers, C. E. The Power of Ion Mobility-Mass Spectrometry for Structural Characterization and the Study of Conformational Dynamics. *Nat Chem* **2014**, *6* (4), 281–294. https://doi.org/10.1038/nchem.1889.

(4) Baker, E. S.; Livesay, E. A.; Orton, D. J.; Moore, R. J.; Danielson, W. F.; Prior, D. C.; Ibrahim, Y. M.; LaMarche, B. L.; Mayampurath, A. M.; Schepmoes, A. A.; Hopkins, D. F.; Tang, K.; Smith, R. D.; Belov, M. E. An LC-IMS-MS Platform Providing Increased Dynamic Range for High-Throughput Proteomic Studies. *J Proteome Res* **2010**, *9* (2), 997–1006. https://doi.org/10.1021/pr900888b.

(5) Rister, A. L.; Dodds, E. D. Liquid Chromatography-Ion Mobility Spectrometry-Mass Spectrometry Analysis of Multiple Classes of Steroid Hormone Isomers in a Mixture. *Journal of Chromatography B* **2020**, *1137*, 121941. https://doi.org/10.1016/j.jchromb.2019.121941.

(6) Mason, E. A.; McDaniel, E. W. *Transport Properties of Ions in Gases*; John Wiley and Sons: New York, 1988.

(7) Langevin, P. Une Formule Fondamentale de Théorie Cinétique. *Annales de Chimie et de Physique* **1905**, *5*, 245–288.

(8) Gabelica, V.; Shvartsburg, A. A.; Afonso, C.; Barran, P.; Benesch, J. L. P.; Bleiholder, C.; Bowers, M. T.; Bilbao, A.; Bush, M. F.; Campbell, J. L.; Campuzano, I. D. G.; Causon, T.; Clowers, B. H.; Creaser, C. S.; De Pauw, E.; Far, J.; Fernandez-Lima, F.; Fjeldsted, J. C.; Giles, K.; Groessl, M.; Hogan, C. J.; Hann, S.; Kim, H. I.; Kurulugama, R. T.; May, J. C.; McLean, J. A.; Pagel, K.; Richardson, K.; Ridgeway, M. E.; Rosu, F.; Sobott, F.; Thalassinos, K.; Valentine, S. J.; Wyttenbach, T. Recommendations for Reporting Ion Mobility Mass Spectrometry Measurements. *Mass Spectrom Rev* **2019**, *38* (3), 291–320. https://doi.org/10.1002/mas.21585.

(9) Gabelica, V.; Marklund, E. Fundamentals of Ion Mobility Spectrometry. *Curr Opin Chem Biol* **2018**, *42*, 51–59. https://doi.org/10.1016/J.CBPA.2017.10.022.

(10) Buryakov, I. A.; Krylov, E. V.; Nazarov, E. G.; Rasulev, U. K. A New Method of Separation of Multi-Atomic Ions by Mobility at Atmospheric Pressure Using a High-Frequency Amplitude-Asymmetric Strong Electric Field. *Int J Mass Spectrom Ion Process* **1993**, *128* (3), 143–148. https://doi.org/10.1016/0168-1176(93)87062-W.

(11) Chouinard, C. D.; Nagy, G.; Webb, I. K.; Shi, T.; Baker, E. S.; Prost, S. A.; Liu, T.; Ibrahim, Y. M.; Smith, R. D. Improved Sensitivity and Separations for Phosphopeptides Using Online

Liquid Chromotography Coupled with Structures for Lossless Ion Manipulations Ion Mobility–Mass Spectrometry. *Anal Chem* **2018**, *90* (18), 10889–10896. https://doi.org/10.1021/acs.analchem.8b02397.

(12) Zheng, X.; Aly, N. A.; Zhou, Y.; Dupuis, K. T.; Bilbao, A.; Paurus, V. L.; Orton, D. J.; Wilson, R.; Payne, S. H.; Smith, R. D.; Baker, E. S. A Structural Examination and Collision Cross Section Database for over 500 Metabolites and Xenobiotics Using Drift Tube Ion Mobility Spectrometry. *Chem Sci* **2017**, *8* (11), 7724–7736. https://doi.org/10.1039/c7sc03464d.

(13) Hines, K. M.; Ross, D. H.; Davidson, K. L.; Bush, M. F.; Xu, L. Large-Scale Structural Characterization of Drug and Drug-Like Compounds by High-Throughput Ion Mobility-Mass Spectrometry. *Anal Chem* **2017**, *89* (17), 9023–9030. https://doi.org/10.1021/acs.analchem.7b01709.

(14) May, J. C.; Morris, C. B.; McLean, J. A. Ion Mobility Collision Cross Section Compendium. *Anal Chem* **2017**, *89* (2), 1032–1044. https://doi.org/10.1021/acs.analchem.6b04905.

(15) Chen, T. C.; Ibrahim, Y. M.; Webb, I. K.; Garimella, S. V. B.; Zhang, X.; Hamid, A. M.; Deng, L.; Karnesky, W. E.; Prost, S. A.; Sandoval, J. A.; Norheim, R. V.; Anderson, G. A.; Tolmachev, A. V.; Baker, E. S.; Smith, R. D. Mobility-Selected Ion Trapping and Enrichment Using Structures for Lossless Ion Manipulations. *Anal Chem* **2016**, *88* (3), 1728–1733. https://doi.org/10.1021/acs.analchem.5b03910.

(16) Rolland, A. D.; Prell, J. S. Computational Insights into Compaction of Gas-Phase Protein and Protein Complex Ions in Native Ion Mobility-Mass Spectrometry. *TrAC - Trends in Analytical Chemistry* **2019**, *116*, 282–291. https://doi.org/10.1016/j.trac.2019.04.023.

(17) Kirk, A. T.; Grube, D.; Kobelt, T.; Wendt, C.; Zimmermann, S. High-Resolution High Kinetic Energy Ion Mobility Spectrometer Based on a Low-Discrimination Tristate Ion Shutter. *Anal Chem* **2018**, *90* (9), 5603–5611. https://doi.org/10.1021/acs.analchem.7b04586.

(18) Schneider, B. B.; Nazarov, E. G.; Londry, F.; Vouros, P.; Covey, T. R. Differential Mobility Spectrometry/Mass Spectrometry History, Theory, Design Optimization, Simulations, and Applications. *Mass Spectrom Rev* **2016**, *35* (6), 687–737. https://doi.org/10.1002/mas.21453.

(19) Haack, A.; Bissonnette, J. R.; Ieritano, C.; Hopkins, W. S. Improved First-Principles Model of Differential Mobility Using Higher Order Two-Temperature Theory. *J Am Soc Mass Spectrom* **2022**, *33* (3), 535–547. https://doi.org/10.1021/jasms.1c00354.

(20) Gandhi, V. D.; Larriba-Andaluz, C. Predicting Ion Mobility as a Function of the Electric Field for Small Ions in Light Gases. *Anal Chim Acta* **2021**, *1184*, 339019. https://doi.org/10.1016/j.aca.2021.339019.

(21) Gandhi, V. D.; Short, K.; Hua, L.; Rodríguez, I.; Larriba-Andaluz, C. A Numerical Tool to Calculate Ion Mobility at Arbitrary Fields from All-Atom Models. *J Aerosol Sci* **2023**, *169*, 106122. https://doi.org/10.1016/j.jaerosci.2022.106122.

(22) Kihara, T. The Mathematical Theory of Electrical Discharges in Gases. B. Velocity-Distribution of Positive Ions in a Static Field. *Rev Mod Phys* **1953**, *25* (4), 844–852. https://doi.org/10.1103/RevModPhys.25.844.

(23) Mason, E. A.; Schamp, H. W. Mobility of Gaseous Ions in Weak Electric Fields. *Ann Phys (N Y)* **1958**, *4* (3), 233–270. https://doi.org/10.1016/0003-4916(58)90049-6.

(24) Viehland, L. A.; Mason, E. A. Gaseous Lon Mobility in Electric Fields of Arbitrary Strength. *Ann Phys (N Y)* **1975**, *91* (2), 499–533. https://doi.org/10.1016/0003-4916(75)90233-X.

(25) Viehland, L. A.; Mason, E. A. Gaseous Ion Mobility and Diffusion in Electric Fields of Arbitrary Strength. *Ann Phys (N Y)* **1978**, *110* (2), 287–328. https://doi.org/10.1016/0003-4916(78)90034-9.

(26) Wyttenbach, T.; Von Helden, G.; Batka, J. J.; Carlat, D.; Bowers, M. T. Effect of the Long-Range Potential on Ion Mobility Measurements. *J Am Soc Mass Spectrom* **1997**, *8* (3), 275–282. https://doi.org/10.1016/S1044-0305(96)00236-X.

(27) Shvartsburg, A. A.; Jarrold, M. F. An Exact Hard-Spheres Scattering Model for the Mobilities of Polyatomic Ions. *Chem Phys Lett* **1996**, *261* (1–2), 86–91. https://doi.org/10.1016/0009-2614(96)00941-4.

(28) Kim, H.; Kim, H. I.; Johnson, P. V.; Beegle, L. W.; Beauchamp, J. L.; Goddard, W. A.; Kanik, I. Experimental and Theoretical Investigation into the Correlation between Mass and Ion Mobility for Choline and Other Ammonium Cations in N2. *Anal Chem* **2008**, *80* (6), 1928–1936. https://doi.org/10.1021/ac701888e.

(29) Mesleh, M. F.; Hunter, J. M.; Shvartsburg, A. A.; Schatz, G. C.; Jarrold, M. F. Structural Information from Ion Mobility Measurements: Effects of the Long-Range Potential. *Journal of Physical Chemistry* **1996**, *100* (40), 16082–16086. https://doi.org/10.1021/JP961623V.

(30) Shvartsburg, A. A.; Mashkevich, S. V.; Baker, E. S.; Smith, R. D. Optimization of Algorithms for Ion Mobility Calculations. *Journal of Physical Chemistry A* **2007**, *111* (10), 2002–2010. https://doi.org/10.1021/jp066953m.

(31) Marklund, E. G.; Degiacomi, M. T.; Robinson, C. V.; Baldwin, A. J.; Benesch, J. L. P. Collision Cross Sections for Structural Proteomics. *Structure* **2015**, *23* (4), 791–799. https://doi.org/10.1016/j.str.2015.02.010.

(32) Lai, R.; Dodds, E. D.; Li, H. Molecular Dynamics Simulation of Ion Mobility in Gases. *J Chem Phys* **2018**, *148* (6), 064109. https://doi.org/10.1063/1.4998955.

(33) Larriba-Andaluz, C. A Perspective on the Theoretical and Numerical Aspects of Ion Mobility Spectrometry. *Int J Mass Spectrom* **2021**, *470*, 116719. https://doi.org/10.1016/J.IJMS.2021.116719.

(34) Rajkovic, M.; Benter, T.; Wißdorf, W. Molecular Dynamics-Based Modeling of Ion-Neutral Collisions in an Open Ion Trajectory Simulation Framework. *J Am Soc Mass Spectrom* **2023**. https://doi.org/10.1021/JASMS.3C00139.

(35) Ieritano, C.; Crouse, J.; Campbell, J. L.; Hopkins, W. S. A Parallelized Molecular Collision Cross Section Package with Optimized Accuracy and Efficiency. *Analyst* **2019**, *144* (5), 1660–1670. https://doi.org/10.1039/c8an02150c.

(36) Halgren, T. A. Merck Molecular Force Field. II. MMFF94 van Der Waals and Electrostatic Parameters for Intermolecular Interactions. *J Comput Chem* **1996**, *17* (5–6), 520–552. https://doi.org/10.1002/(SICI)1096-987X(199604)17:5/6<520::AID-JCC2>3.0.CO;2-W.

(37) Lee, J. W.; Lee, H. H. L.; Davidson, K. L.; Bush, M. F.; Kim, H. I. Structural Characterization of Small Molecular Ions by Ion Mobility Mass Spectrometry in Nitrogen Drift Gas: Improving the Accuracy of Trajectory Method Calculations. *Analyst* **2018**, *143* (8), 1786–1796. https://doi.org/10.1039/C8AN00270C.

(38) Righetti, L.; Bergmann, A.; Galaverna, G.; Rolfsson, O.; Paglia, G.; Dall'Asta, C. Ion Mobility-Derived Collision Cross Section Database: Application to Mycotoxin Analysis. *Anal Chim Acta* **2018**, *1014*, 50–57. https://doi.org/10.1016/j.aca.2018.01.047.

(39) Bijlsma, L.; Bade, R.; Celma, A.; Mullin, L.; Cleland, G.; Stead, S.; Hernandez, F.; Sancho, J. V. Prediction of Collision Cross-Section Values for Small Molecules: Application to Pesticide Residue Analysis. *Anal Chem* **2017**, *89* (12), 6583–6589. https://doi.org/10.1021/acs.analchem.7b00741.

(40) Bauer, A.; Kuballa, J.; Rohn, S.; Jantzen, E.; Luetjohann, J. Evaluation and Validation of an Ion Mobility Quadrupole Time-of-Flight Mass Spectrometry Pesticide Screening Approach. *J Sep Sci* **2018**, *41* (10), 2178–2187. https://doi.org/10.1002/jssc.201701059.

(41) Paglia, G.; Williams, J. P.; Menikarachchi, L.; Thompson, J. W.; Tyldesley-Worster, R.; Halldórsson, S.; Rolfsson, O.; Moseley, A.; Grant, D.; Langridge, J.; Palsson, B. O.; Astarita, G. Ion Mobility Derived Collision Cross Sections to Support Metabolomics Applications. *Anal Chem* **2014**, *86* (8), 3985–3993. https://doi.org/10.1021/ac500405x.

(42) Wu, T.; Derrick, J.; Nahin, M.; Chen, X.; Larriba-Andaluz, C. Optimization of Long Range Potential Interaction Parameters in Ion Mobility Spectrometry. *Journal of Chemical Physics* **2018**, *148* (7), 074102. https://doi.org/10.1063/1.5016170.

(43) Regueiro, J.; Negreira, N.; Berntssen, M. H. G. Ion-Mobility-Derived Collision Cross Section as an Additional Identification Point for Multiresidue Screening of Pesticides in Fish Feed. *Anal Chem* **2016**, *88* (22), 11169–11177. https://doi.org/10.1021/acs.analchem.6b03381.

(44) Campuzano, I.; Bush, M. F.; Robinson, C. V.; Beaumont, C.; Richardson, K.; Kim, H. H. I.; Kim, H. H. I. Structural Characterization of Drug-like Compounds by Ion Mobility Mass Spectrometry: Comparison of Theoretical and Experimentally Derived Nitrogen Collision Cross Sections. *Anal Chem* **2012**, *84* (2), 1026–1033. https://doi.org/10.1021/ac202625t.

(45) Ieritano, C.; Lee, A.; Crouse, J.; Bowman, Z.; Mashmoushi, N.; Crossley, P. M.; Friebe, B. P.; Campbell, J. L.; Hopkins, W. S. Determining Collision Cross Sections from Differential Ion Mobility Spectrometry. *Anal Chem* **2021**, *93* (25), 8937–8944. https://doi.org/10.1021/acs.analchem.1c01420.

(46) Ieritano, C.; Campbell, J. L.; Hopkins, W. S. Predicting Differential Ion Mobility Behaviour in Silico Using Machine Learning. *Analyst* **2021**, *146* (15), 4737–4743. https://doi.org/10.1039/D1AN00557J.

(47) Yang, J.; Zhang, Y. I-TASSER Server: New Development for Protein Structure and Function Predictions. *Nucleic Acids Res* **2015**, *43* (W1), W174–W181. https://doi.org/10.1093/nar/gkv342.

(48) Zheng, W.; Zhang, C.; Li, Y.; Pearce, R.; Bell, E. W.; Zhang, Y. Folding Non-Homologous Proteins by Coupling Deep-Learning Contact Maps with I-TASSER Assembly Simulations. *Cell Reports Methods* **2021**, *1* (3), 100014. https://doi.org/10.1016/j.crmeth.2021.100014.

(49) Grimme, S.; Antony, J.; Ehrlich, S.; Krieg, H. A Consistent and Accurate Ab Initio Parametrization of Density Functional Dispersion Correction (DFT-D) for the 94 Elements H-Pu. *Journal of Chemical Physics* **2010**, *132* (15), 154104. https://doi.org/10.1063/1.3382344.

(50) Weigend, F. Accurate Coulomb-Fitting Basis Sets for H to Rn. *Physical Chemistry Chemical Physics* **2006**, *8* (9), 1057–1065. https://doi.org/10.1039/b515623h.

(51) Weigend, F.; Ahlrichs, R. Balanced Basis Sets of Split Valence, Triple Zeta Valence and Quadruple Zeta Valence Quality for H to Rn: Design and Assessment of Accuracy. *Physical Chemistry Chemical Physics* **2005**, *7* (18), 3297–3305. https://doi.org/10.1039/b508541a.

(52) Chai, J. Da; Head-Gordon, M. Long-Range Corrected Hybrid Density Functionals with Damped Atom-Atom Dispersion Corrections. *Physical Chemistry Chemical Physics* **2008**, *10* (44), 6615–6620. https://doi.org/10.1039/b810189b.

(53) Lin, Y. S.; Li, G. De; Mao, S. P.; Chai, J. Da. Long-Range Corrected Hybrid Density Functionals with Improved Dispersion Corrections. *J Chem Theory Comput* **2013**, *9* (1), 263–272. https://doi.org/10.1021/ct300715s.

(54) Breneman, C. M.; Wiberg, K. B. Determining Atom-Centered Monopoles from Molecular Electrostatic Potentials. The Need for High Sampling Density in Formamide Conformational Analysis. *J Comput Chem* **1990**, *11* (3), 361–373. https://doi.org/10.1002/jcc.540110311.

(55) Tao, J.; Perdew, J. P.; Staroverov, V. N.; Scuseria, G. E. Climbing the Density Functional Ladder: Nonempirical Meta–Generalized Gradient Approximation Designed for Molecules and Solids. *Phys Rev Lett* **2003**, *91* (14), 146401. https://doi.org/10.1103/PhysRevLett.91.146401.

(56) Neese, F. The ORCA Program System. *WIREs Computational Molecular Science* **2012**, *2* (1), 73–78. https://doi.org/10.1002/wcms.81.

(57) Neese, F. Software Update: The ORCA Program System—Version 5.0. *WIREs Computational Molecular Science* **2022**, *12* (5), e1606. https://doi.org/10.1002/WCMS.1606.

(58) Álvarez-Moreno, M.; De Graaf, C.; López, N.; Maseras, F.; Poblet, J. M.; Bo, C. Managing the Computational Chemistry Big Data Problem: The IoChem-BD Platform. *J Chem Inf Model* **2015**, *55* (1), 95–103. https://doi.org/10.1021/ci500593j.

(59) O'Boyle, N. M.; Banck, M.; James, C. A.; Morley, C.; Vandermeersch, T.; Hutchison, G. R. Open Babel: An Open Chemical Toolbox. *J Cheminform* **2011**, *3* (1), 33. https://doi.org/10.1186/1758-2946-3-33.

(60) Lii, J. H.; Allinger, N. L. Molecular Mechanics. The MM3 Force Field for Hydrocarbons. 3. The van Der Waals' Potentials and Crystal Data for Aliphatic and Aromatic Hydrocarbons. *J Am Chem Soc* **1989**, *111* (23), 8576–8582. https://doi.org/10.1021/ja00205a003.

(61) Kim, H. I.; Kim, H.; Pang, E. S.; Ryu, E. K.; Beegle, L. W.; Loo, J. A.; Goddard, W. A.; Kanik, I. Structural Characterization of Unsaturated Phosphatidylcholines Using Traveling Wave Ion Mobility Spectrometry. *Anal Chem* **2009**, *81* (20), 8289–8297. https://doi.org/10.1021/ac900672a.

(62) Graham, C.; Imrie, D. A.; Raab, R. E. Measurement of the Electric Quadrupole Moments of CO2 , CO, N2 , Cl2 and BF3. *Mol Phys* **1998**, *93* (1), 49–56. https://doi.org/10.1080/002689798169429.

(63) Ieritano, C.; Hopkins, W. S. The Hitchhiker's Guide to Dynamic Ion-Solvent Clustering: Applications in Differential Ion Mobility Spectrometry. *Physical Chemistry Chemical Physics* **2022**, *24* (35), 20594–20615. https://doi.org/10.1039/d2cp02540j.

(64) Hirschfelder, J. O.; Curtiss, C. F.; Bird, R. B. *Molecular Theory of Gases and Liquids*, Corrected.; John Wiley \& Sons, Inc.: New York, NY, USA, 1964.

(65) Hahn, H.-S.; Mason, E. A. Random-Phase Approximation for Transport Cross Sections. *Chem Phys Lett* **1971**, *9* (6), 633–635. https://doi.org/10.1016/0009-2614(71)85149-7.

(66) Alberty, R. J. *Physical Chemistry*, 2nd ed.; 1996.

(67) Siems, W. F.; Viehland, L. A.; Hill, H. H. Correcting the Fundamental Ion Mobility Equation for Field Effects. *Analyst* **2016**, *141* (23), 6396–6407. https://doi.org/10.1039/c6an01353h.

(68) Krylov, E.; Nazarov, E. G.; Miller, R. A.; Tadjikov, B.; Eiceman, G. A. Field Dependence of Mobilities for Gas-Phase-Protonated Monomers and Proton-Bound Dimers of Ketones by Planar Field Asymmetric Waveform Ion Mobility Spectrometer (PFAIMS). *J Phys Chem A* **2002**, *106* (22), 5437–5444. https://doi.org/10.1021/jp020009i.

(69) Haack, A.; Hopkins, W. S. Kinetics in DMS: Modeling Clustering and Declustering Reactions. *J Am Soc Mass Spectrom* **2022**, *33*, 2250–2262. https://doi.org/10.1021/jasms.2c00224.

(70) Viehland, L. A.; Lin, S. L.; Mason, E. A. Kinetic Theory of Drift-Tube Experiments with Polyatomic Species. *Chem Phys* **1981**, *54* (3), 341–364. https://doi.org/10.1016/0301-0104(81)85111-7.

(71) Haack, A.; Crouse, J.; Schlüter, F. J.; Benter, T.; Hopkins, W. S. A First Principle Model of Differential Ion Mobility: The Effect of Ion-Solvent Clustering. *J Am Soc Mass Spectrom* **2019**, *30* (12), 2711–2725. https://doi.org/10.1007/s13361-019-02340-1.

(72) Shvartsburg, A. A.; Mashkevich, S. V; Siu, K. W. M. Incorporation of Thermal Rotation of Drifting Ions into Mobility Calculations: Drastic Effect for Heavier Buffer Gases. *J Phys Chem A* **2000**, *104* (42), 9448–9453. https://doi.org/10.1021/jp001753a.

(73) O'Boyle, N. M.; Banck, M.; James, C. A.; Morley, C.; Vandermeersch, T.; Hutchison, G. R. Open Babel: An Open Chemical Toolbox. *J Cheminform* **2011**, *3* (10), 1–14. https://doi.org/10.1186/1758-2946-3-33.

(74) Viehland, L. A.; Lin, S. L.; Mason, E. A. Kinetic Theory of Drift-Tube Experiments with Polyatomic Species. *Chem Phys* **1981**, *54* (3), 341–364. https://doi.org/10.1016/0301-0104(81)85111-7.

(75) Ieritano, C.; Featherstone, J.; Haack, A.; Guna, M.; Campbell, J. L.; Hopkins, W. S. How Hot Are Your Ions in Differential Mobility Spectrometry? *J Am Soc Mass Spectrom* **2020**, *31* (3), 582–593. https://doi.org/10.1021/jasms.9b00043.

(76) Carpenter, J. E.; McNary, C. P.; Furin, A.; Sweeney, A. F.; Armentrout, P. B. How Hot Are Your Ions Really? A Threshold Collision-Induced Dissociation Study of Substituted Benzylpyridinium "Thermometer" Ions. *J Am Soc Mass Spectrom* **2017**, *28* (9), 1876–1888. https://doi.org/10.1007/s13361-017-1693-0.

(77) Zhou, C.; Ieritano, C.; Hopkins, W. S. Augmenting Basin-Hopping With Techniques From Unsupervised Machine Learning: Applications in Spectroscopy and Ion Mobility. *Front Chem* **2019**, *7*, 519. https://doi.org/10.3389/fchem.2019.00519.

(78) Roothaan, C. C. J. New Developments in Molecular Orbital Theory. *Rev Mod Phys* **1951**, *23* (2), 69–89. https://doi.org/10.1103/RevModPhys.23.69.

(79) Crouse, J.; Haack, A.; Benter, T.; Hopkins, W. S. Understanding Nontraditional Differential Mobility Behavior: A Case Study of the Tricarbastannatrane Cation, N(CH2CH2CH2)3Sn. *J Am Soc Mass Spectrom* **2020**, *31* (4), 796–802. https://doi.org/10.1021/jasms.9b00042.

(80) Klimeš, J.; Michaelides, A. Perspective: Advances and Challenges in Treating van Der Waals Dispersion Forces in Density Functional Theory. *Journal of Chemical Physics* **2012**, *137* (12), 1–15. https://doi.org/10.1063/1.4754130.

(81) Riley, K. E.; Pitoňčák, M.; Jureččka, P.; Hobza, P. Stabilization and Structure Calculations for Noncovalent Interactions in Extended Molecular Systems Based on Wave Function and Density Functional Theories. *Chem Rev* **2010**, *110* (9), 5023–5063. https://doi.org/10.1021/cr1000173.

(82) Grimme, S. Supramolecular Binding Thermodynamics by Dispersion-Corrected Density Functional Theory. *Chemistry - A European Journal* **2012**, *18* (32), 9955–9964. https://doi.org/10.1002/chem.201200497.

(83) Goerigk, L.; Grimme, S. A Thorough Benchmark of Density Functional Methods for General Main Group Thermochemistry, Kinetics, and Noncovalent Interactions. *Physical Chemistry Chemical Physics* **2011**, *13* (14), 6670–6688. https://doi.org/10.1039/c0cp02984j.

(84) Mardirossian, N.; Head-Gordon, M. Thirty Years of Density Functional Theory in Computational Chemistry: An Overview and Extensive Assessment of 200 Density Functionals. *Mol Phys* **2017**, *115* (19), 2315–2372. https://doi.org/10.1080/00268976.2017.1333644.

(85) Jordan, M. J. T.; Del Bene, J. E. Unraveling Environmental Effects on Hydrogen-Bonded Complexes: Matrix Effects on the Structures and Proton-Stretching Frequencies of Hydrogen-Halide Complexes with Ammonia and Trimethylamine. *J Am Chem Soc* **2000**, *122* (9), 2101–2115. https://doi.org/10.1021/ja993981s.

(86) Bende, A.; Muntean, C. M. The Influence of Anharmonic and Solvent Effects on the Theoretical Vibrational Spectra of the Guanine-Cytosine Base Pairs in Watson-Crick and Hoogsteen Configurations. *J Mol Model* **2014**, *20* (3), 1–12. https://doi.org/10.1007/s00894-014-2113-z.

(87) Ieritano, C.; Featherstone, J.; Carr, P. J. J.; Marta, R. A.; Loire, E.; McMahon, T. B.; Hopkins, W. S. The Structures and Properties of Anionic Tryptophan Complexes. *Physical Chemistry Chemical Physics* **2018**, *20* (41), 26532–26541. https://doi.org/10.1039/c8cp04533j.

(88) Liakos, D. G.; Sparta, M.; Kesharwani, M. K.; Martin, J. M. L.; Neese, F. Exploring the Accuracy Limits of Local Pair Natural Orbital Coupled-Cluster Theory. *J Chem Theory Comput* **2015**, *11* (4), 1525–1539. https://doi.org/10.1021/ct501129s.

(89) Liakos, D. G.; Guo, Y.; Neese, F. Comprehensive Benchmark Results for the Domain Based Local Pair Natural Orbital Coupled Cluster Method (DLPNO-CCSD(T)) for Closed- And Open-Shell Systems. *Journal of Physical Chemistry A* **2020**, *124* (1), 90–100. https://doi.org/10.1021/acs.jpca.9b05734.

(90) Riplinger, C.; Sandhoefer, B.; Hansen, A.; Neese, F. Natural Triple Excitations in Local Coupled Cluster Calculations with Pair Natural Orbitals. *Journal of Chemical Physics* **2013**, *139* (13), 134101. https://doi.org/10.1063/1.4821834.

(91) Riplinger, C.; Neese, F. An Efficient and near Linear Scaling Pair Natural Orbital Based Local Coupled Cluster Method. *Journal of Chemical Physics* **2013**, *138* (3), 34106. https://doi.org/10.1063/1.4773581.

(92) Riplinger, C.; Pinski, P.; Becker, U.; Valeev, E. F.; Neese, F. Sparse Maps - A Systematic Infrastructure for Reduced-Scaling Electronic Structure Methods. II. Linear Scaling Domain Based Pair Natural Orbital Coupled Cluster Theory. *Journal of Chemical Physics* **2016**, *144* (2), 24109. https://doi.org/10.1063/1.4939030.

(93) Mesleh, M. F.; Hunter, J. M.; Shvartsburg, A. A.; Schatz, G. C.; Jarrold, M. F. Structural Information from Ion Mobility Measurements: Effects of the Long-Range Potential. *J Phys Chem* **1996**, *100* (40), 16082–16086. https://doi.org/10.1021/jp961623v.

(94) Shvartsburg, A. A.; Jarrold, M. F. An Exact Hard-Spheres Scattering Model for the Mobilities of Polyatomic Ions. *Chem Phys Lett* **1996**, *261* (1–2), 86–91. https://doi.org/10.1016/0009-2614(96)00941-4.

(95) Kim, H.; Kim, H. I.; Johnson, P. V.; Beegle, L. W.; Beauchamp, J. L.; Goddard, W. A.; Kanik, I. Experimental and Theoretical Investigation into the Correlation between Mass and Ion Mobility for Choline and Other Ammonium Cations in N 2. *Anal Chem* **2008**, *80* (6), 1928–1936. https://doi.org/10.1021/ac701888e.

(96) Campuzano, I.; Bush, M. F.; Robinson, C. V.; Beaumont, C.; Richardson, K.; Kim, H.; Kim, H. I. Structural Characterization of Drug-like Compounds by Ion Mobility Mass Spectrometry: Comparison of Theoretical and Experimentally Derived Nitrogen Collision Cross Sections. *Anal Chem* **2012**, *84* (2), 1026–1033. https://doi.org/10.1021/ac202625t.

(97) Lee, J. W.; Davidson, K. L.; Bush, M. F.; Kim, H. I. Collision Cross Sections and Ion Structures: Development of a General Calculation Method via High-Quality Ion Mobility Measurements and Theoretical Modeling. *Analyst* **2017**, *142* (22), 4289–4298. https://doi.org/10.1039/C7AN01276D.

(98) Ieritano, C.; Crouse, J.; Campbell, J. L.; Hopkins, W. S. A Parallelized Molecular Collision Cross Section Package with Optimized Accuracy and Efficiency. *Analyst* **2019**, *144* (5), 1660–1670. https://doi.org/10.1039/C8AN02150C.

(99) Ieritano, C.; Hopkins, W. S. Assessing Collision Cross Section Calculations Using MobCal-MPI with a Variety of Commonly Used Computational Methods. *Mater Today Commun* **2021**, *27* (February), 102226. https://doi.org/10.1016/j.mtcomm.2021.102226.

(100) Zanotto, L.; Heerdt, G.; Souza, P. C. T.; Araujo, G.; Skaf, M. S. High Performance Collision Cross Section Calculation-HPCCS. *J Comput Chem* **2018**, *39* (21), 1675–1681. https://doi.org/10.1002/jcc.25199.

(101) Larriba, C.; Hogan, C. J. Free Molecular Collision Cross Section Calculation Methods for Nanoparticles and Complex Ions with Energy Accommodation. *J Comput Phys* **2013**, *251*, 344–363. https://doi.org/10.1016/j.jcp.2013.05.038.

(102) Larriba, C.; Hogan, C. J. Ion Mobilities in Diatomic Gases: Measurement versus Prediction with Non-Specular Scattering Models. *J Phys Chem A* **2013**, *117* (19), 3887–3901. https://doi.org/10.1021/jp312432z.

(103) Larriba-Andaluz, C.; Fernández-García, J.; Ewing, M. A.; Hogan, C. J.; Clemmer, D. E. Gas Molecule Scattering & Ion Mobility Measurements for Organic Macro-Ions in He versus N 2 Environments. *Physical Chemistry Chemical Physics* **2015**, *17* (22), 15019–15029. https://doi.org/10.1039/C5CP01017A.

(104) Shrivastav, V.; Nahin, M.; Hogan, C. J.; Larriba-Andaluz, C. Benchmark Comparison for a Multi-Processing Ion Mobility Calculator in the Free Molecular Regime. *J Am Soc Mass Spectrom* **2017**, *28* (8), 1540–1551. https://doi.org/10.1007/s13361-017-1661-8.

(105) Gandhi, V. D.; Short, K.; Hua, L.; Rodríguez, I.; Larriba-Andaluz, C. A Numerical Tool to Calculate Ion Mobility at Arbitrary Fields from All-Atom Models. *J Aerosol Sci* **2023**, *169*, 106122. https://doi.org/10.1016/J.JAEROSCI.2022.106122.

(106) Ewing, S. A.; Donor, M. T.; Wilson, J. W.; Prell, J. S. Collidoscope: An Improved Tool for Computing Collisional Cross-Sections with the Trajectory Method. *J Am Soc Mass Spectrom* **2017**, *28* (4), 587–596. https://doi.org/10.1007/s13361-017-1594-2.

(107) Bleiholder, C.; Wyttenbach, T.; Bowers, M. T. A Novel Projection Approximation Algorithm for the Fast and Accurate Computation of Molecular Collision Cross Sections (I). Method. *Int J Mass Spectrom* **2011**, *308* (1), 1–10. https://doi.org/10.1016/j.ijms.2011.06.014.

(108) Bleiholder, C.; Contreras, S.; Do, T. D.; Bowers, M. T. A Novel Projection Approximation Algorithm for the Fast and Accurate Computation of Molecular Collision Cross Sections (II). Model Parameterization and Definition of Empirical Shape Factors for Proteins. *Int J Mass Spectrom* **2013**, *345–347*, 89–96. https://doi.org/10.1016/j.ijms.2012.08.027.

(109) Anderson, S. E.; Bleiholder, C.; Brocker, E. R.; Stang, P. J.; Bowers, M. T. A Novel Projection Approximation Algorithm for the Fast and Accurate Computation of Molecular Collision Cross Sections (III): Application to Supramolecular Coordination-Driven Assemblies with Complex Shapes. *Int J Mass Spectrom* **2012**, *330–332*, 78–84. https://doi.org/10.1016/j.ijms.2012.08.024.

(110) Bleiholder, C.; Contreras, S.; Bowers, M. T. A Novel Projection Approximation Algorithm for the Fast and Accurate Computation of Molecular Collision Cross Sections (IV). Application to Polypeptides. *Int J Mass Spectrom* **2013**, *354–355*, 275–280. https://doi.org/10.1016/j.ijms.2013.06.011.

(111) Boltzmann, L. Weitere Studien Über Das Wärmegleichgewicht Unter Gasmolekülen. *Sitzungsberichte der kaiserlichen Akademie der Wissenschaften (II)* **1872**, *66*, 275–370.

(112) Chapman, S.; Cowling, T. G. *The Mathematical Theory of Non-Uniform Gases*, 3rd ed.; Cambridge University Press: London, UK, 1970.

(113) Krylov, E. V; Coy, S. L.; Vandermey, J.; Schneider, B. B.; Covey, T. R.; Nazarov, E. G. Selection and Generation of Waveforms for Differential Mobility Spectrometry. *Rev Sci Instrum* **2010**, *81* (2), 024101. https://doi.org/10.1063/1.3284507.

(114) Haack, A.; Hopkins, W. S. Kinetics in DMS: Modeling Clustering and Declustering Reactions. *J Am Soc Mass Spectrom* **2022**, *33* (12), 2250–2262. https://doi.org/10.1021/jasms.2c00224.

(115) Siems, W. F.; Viehland, L. A.; Hill, H. H. Correcting the Fundamental Ion Mobility Equation for Field Effects. *Analyst* **2016**, *141* (23), 6396–6407. https://doi.org/10.1039/C6AN01353H.

(116) Lee, J. W.; Davidson, K. L.; Bush, M. F.; Kim, H. I. Collision Cross Sections and Ion Structures: Development of a General Calculation Method via High-Quality Ion Mobility Measurements and Theoretical Modeling. *Analyst* **2017**, *142* (22), 4289–4298. https://doi.org/10.1039/c7an01276d.

(117) Ieritano, C.; Yves Le Blanc, J. C.; Schneider, B. B.; Bissonnette, J. R.; Haack, A.; Hopkins, W. S. Protonation-Induced Chirality Drives Separation by Differential Ion Mobility Spectrometry. *Angewandte Chemie - International Edition* **2022**, *61* (9), e202116794. https://doi.org/10.1002/anie.202116794.