Contents

Molflow 2.6 algorithm

Marton Ady*

March 3, 2016

Extract from the PhD thesis *Monte Carlo simulations of ultra high vacuum and synchrotron radiation for particle accelerators[1]*

1 Introduction

Molflow, dating back to the 1990s, is a Monte Carlo simulator intended to calculate pressure profiles and conductances in ultra-high vacuum. It uses the test particle Monte Carlo method where as introduced above - a limited number of virtual, test particles represent a larger amount of physical molecules and the derived quantities of the test particles are scaled up to match the physical numbers.

In Molflow the geometry is described by polygons called facets. All facets have user-defined physical properties (temperature, opacity, sticking, etc.). To perform a vacuum simulation, we insert molecules in the system and trace them until they are pumped. In between, collisions with the wall (referred to as hits) are registered, and where necessary, stored in memory. We do not account for collisions between molecules as the mean free path of molecules in the accelerator pressure ranges is shown before to be above the characteristic length of the vacuum chambers.

This is in contrast with direct simulation Monte Carlo (DSMC) methods, where the vacuum chamber volume is divided into 3D cells, and a set of virtual particles is kept in memory and their positions are evolved as time advances.

Historically the DSMC method [2] supersedes the TPMC, and is generally considered more advanced as treatment of inter-molecular collisions allows simulating pressures in the viscous regime (where the Knudsen number is below 1). However, in particle accelerator applications, where pressures are, with few exceptions, well within the free molecular flow regime, I argue that TPMC methods are still a good choice because of the following advantages:

- As intermolecular collisions are omitted, only one test particle has to be kept in memory. Its trajectory, assumed to be straight between the chamber walls, can be determined by ray tracing, from the point of insertion (outgassing point) to the pump location. This is in contrast with DSMC methods where each 3D cell has to store a statistically significant number of particles, increasing the memory requirement substantially.
- The TPMC method allows simulations to be event-driven: Molflow+ first looks for collisions with the walls, and once they happen, based on the position and the particle velocity, it calculates the time of the hits. This is in contrast with time-driven simulations, where time is advanced periodically by a given step and the new molecule locations are iterated, checking at each step if the new position is within the chambers volume. The problem with that method is that the time step must be scaled to the smallest feature in the system, therefore in case of a straight flight path between two walls, possibly hundreds of time steps are required in time-driven algorithms vs. only one in event-driven ones.
- Since test particles are independent, the vacuum systems behavior is linear: the algorithm is particularly suitable for parallelization: in Molflows case each CPU core traces one test particle, and hits are summed when results are evaluated. For larger parallelization (cluster, GPU), only the geometry with the physical properties have to be passed for each thread, and simulation state synchronization, usually a bottleneck on GPUs, is not necessary.
- While 3D meshing is straightforward in simple geometries, for complex ones (grids, screws, probes, RF fingers) it becomes more error-prone (see Fig.2): small geometry details can increase the number

*CERN, marton.ady@cern.ch

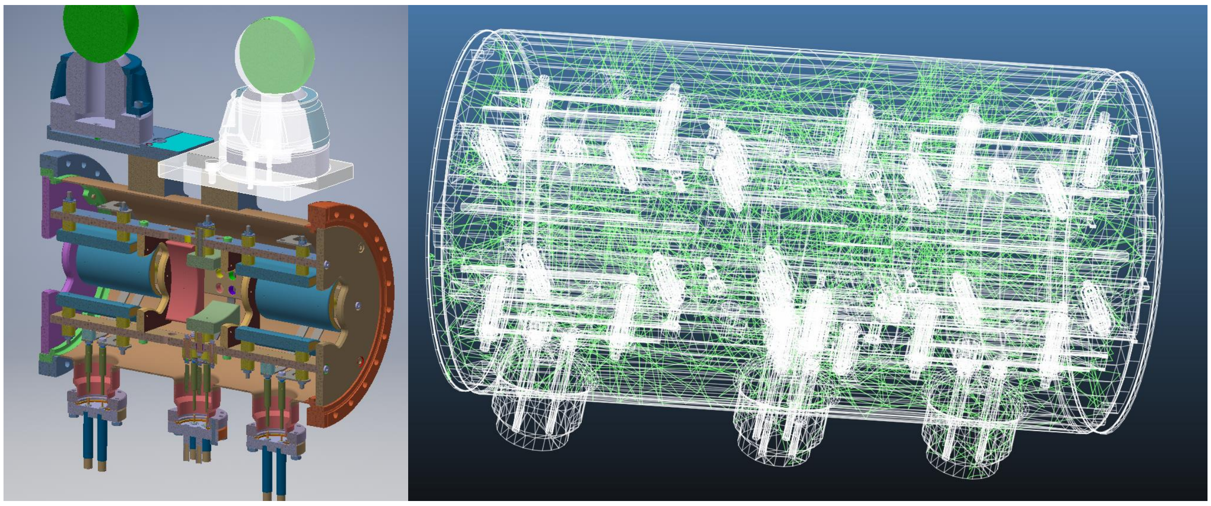


Figure 1: Complex geometry and its Molflow+ equivalent

of cells used, and the mesh size, if not chosen correctly, can introduce non-converging solutions. In TMPC simulations only surfaces are described, imported from CAD programs in the STL format.

Due to the lower memory requirement and the faster algorithm explained above, most vacuum chambers can be simulated with the TPMC method on an average desktop computer. As an example of a complex geometry, the tank with electrodes and correctors in Fig.1 can be described by 37.000 facets and a pressure solution with acceptably low statistical scattering is reached within an hour (corresponding to 20 million hits) in Molflow+. Volumetric meshing of the geometry can take over a day and the consequent DSMC simulation of the system can typically be achieved only with high-performance computation clusters due to the large memory and computational requirement.

The original Molflow code developed outside of this PhD project calculated results as numbers of Monte Carlo hits which could be converted into pressures or density in case of a steady-state simulation where the whole system was isothermal. Based on the ray-tracing engine and the custom GUI framework[1], I have extended the algorithm to include the distribution of the molecule velocities (so pressure can be calculated in non-equilibrium scenarios), and once this is achieved, make simulations time-dependent. This new code is presented below.

2 Steady-state simulations

First Molflow's steady-state algorithm is presented: in this case, there is a continuous influx of gas particles to the system, with constant pumping speeds, and our goal is to determine the (constant) pressure, density and other physical quantities on the facets. It is worth mentioning that unless the user explicitly defines time-dependent parameters, this steady-state algorithm is used. The algorithms main steps, with an overview on Fig.3, are identical to almost all TPMC vacuum calculators. Details for each step are explained below.

2.1 Particle generation

In the steady-state case the outgassing rate $Q = d(pV)/dt$ is constant. Gas may come from the walls through thermal desorption, or from injection points. In both cases the source points will be represented by facets where for each facet f the user defines the outgassing Q_f. From this quantity, the influx rate of physical particles, $dN_{f,real}/dt$ can be calculated: using the ideal gas equation:

$$pV = N_{real} k_B T \tag{1a}$$

$$\frac{dN_{f,real}}{dt} = \frac{pV}{dt}\frac{1}{k_B T_f} = \frac{Q_f}{k_B T_f}, \tag{1b}$$

[1]Programmed by of J.L.Pons, ESRF in 2007

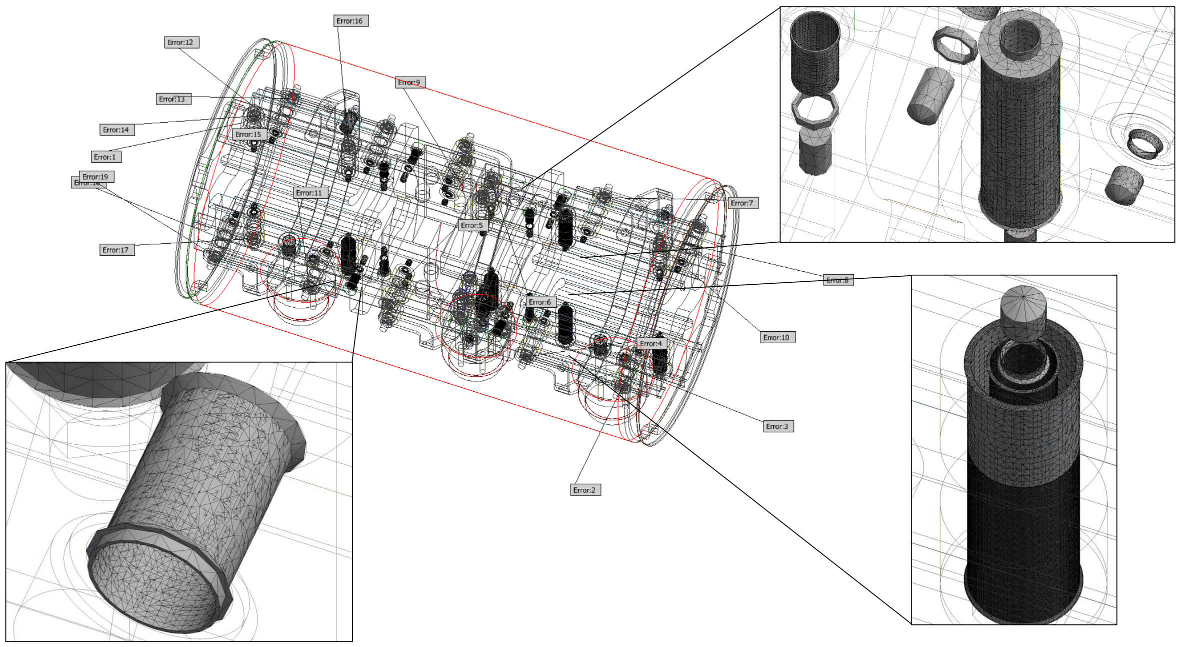


Figure 2: 3D tetrahedral meshing failures with Autodesk Inventor 2016 a 3D mesh can take days to generate and the memory requirement is well outside the capability of current desktop PCs

where T_f is the temperature of the gas source that facet f represents. [2]

For each generated virtual particle, Molflow+ chooses a starting location. The probability of choosing a certain facet as source is proportional to the local flux dN_{real}/dt of molecules on the facet, derived above. Once the source facet is decided, the start position of the test particle is chosen randomly with uniform distribution on its surface.

The total number of particles entering the system every second, $\sum \frac{dN_{real}}{dt}$ is the sum of the particle fluxes on the source facets. Even for a low outgassing rate, for example 1×10^{-8} mbar l/s, it is in the order of 1×10^{12} particles/second, and our computing power is insufficient to simulate all of them. This is why we represent a large number of real particles with a smaller number of virtual (test) particles. In this case, if we generate a total N_{virtual} test particles, each one will represent a number of real particles

[2]For clarity, I would emphasize that in the steady-state mode a test particle doesn't represent an absolute number of physical molecules, but instead as the system is in equilibrium a certain influx or rate of particles (incoming particles per second). Consequently, a virtual hit on the wall will represent a certain number of real hits per second, contributing to the (constant) pressure.

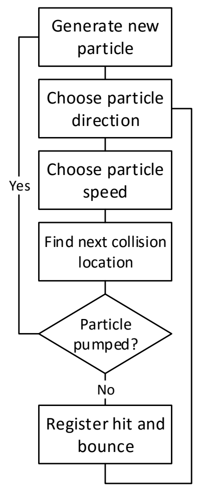


Figure 3: Main steps of Molflow's algorithm

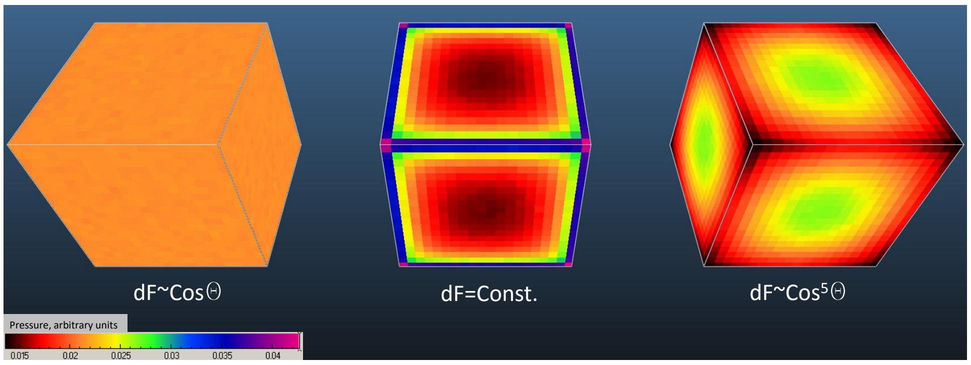


Figure 4: Pressure profiles in a cube with uniform desorption, based on the directional distribution of emitted and reflected particles

per second equal to:

$$K_{\frac{real}{virtual}} = \frac{\sum dN_{\text{real}}/dt}{N_{\text{virtual}}}$$

2.2 Choosing particle direction

For particle generation and surface rebounds, Knudsens cosine law is applied, which states [3] that the probability ds that a molecule leaves a surface in solid angle $d\omega$ is

$$ds = d\omega\pi \cdot \cos\theta$$

where θ is the angle between the particles direction vector and the surface normal[3].

Using results from [5], the ϕ azimuth and θ angles can be generated as follows:

$$\phi = rnd \cdot 2\pi$$

$$\theta = \sin^{-1}\sqrt{rnd}$$

where rnd is a pseudo-random number generated uniformly between 0 and 1. We generate these random numbers using the Mersenne-Twister algorithm[6].

2.3 Choosing particle speed

For simplicity, several Monte Carlo vacuum simulator implementations calculate either with the average molecule speed [7], or a constant speed that remains the same through the collisions [8]. In a real system the speed distribution of molecules in a given volume of ideal gas at equilibrium is described by the Maxwell-Boltzmann distribution[9], and after each collision the particle spends a certain sojourn time on the wall during which it thermalizes hence obtaining a new speed. Implementing this accurately can simulate effects where after a gas injection some faster particles can arrive to distant locations faster than that predicted by the average gas speed. To include these effects, I choose therefore a new molecule speed at every collision, as it happens in a real vacuum system.

In my event-driven algorithm, test particles aren't generated volume-wise, but - as explained above - representing the flux of incoming particles during a given time through source facets. This is important as the speed distribution is different in the latter case: fast molecules of the gas cross or hit a surface more frequently than slow ones.

[3]If we have Monte Carlo tools available, Knudsen's cosine law can be "proven" by a simple thought experiment. If we have a cube with uniform desorption and uniform sticking on all sides, we expect its pressure to be uniform at every point. In Fig.4 I model three cubes, one with cosine desorption obeying Knudsen's law, an other one with "uniform" desorption, meaning that each outgoing direction is chosen with equal probability, and one with a "collimated" desorption, where the probability of choosing a direction is proportional to $cos^5(\theta)$, therefore preferring directions close to the facets normal vector.

On the pressure values, visualized by textures, we can see that any deviation from the cosine law results in particles tending to get stuck in corners or on the contrary, tending to avoid them. Only a random walk following the cosine law will result in uniform pressure regardless of the geometry shape.

For a more formal proof that the cosine distribution must be chosen see [4]

If the probability density function (p.d.f.) of molecular speeds in a volume is $f(v)_{gas}$, then it is possible[4]to calculate the p.d.f. of molecules colliding with a wall during a period of time, $f(v)_{coll}$:

$$f(v)_{coll} = v^3 \exp\left(-\frac{v^2}{2a^2}\right) \frac{1}{2a^4} \tag{2}$$

where

$$a = \sqrt{\frac{k_B T}{m}}$$

Now that we have the speed distribution of molecules crossing or colliding on a facet, we can generate molecules according to it by the numerical inversion method [10]: we calculate the cumulative distribution function (C.D.F) for the speed distribution, which will assign for every given speed v a probability (between 0 and 1) that a particles velocity is lower than v. Then we generate pseudo-random numbers uniformly between 0 and 1 (using again the Mersenne-Twister algorithm), then interpolate in the C.D.F. the speed v corresponding to the generated probability. For this, the C.D.F for our modified speed distribution must be calculated, which - unlike for the Maxwell Boltzmann distribution - can be expressed in closed form:

$$F(v)_{coll} = \int_{-\infty}^{v} f(v')_{coll} dv' = 1 - \exp\left(-\frac{v^2}{2a^2}\right)\left[1 + \frac{v^2}{2a^2}\right]$$

For speedup of the simulation, at the beginning, this C.D.F. functions values are calculated for 100 equidistant speed values between 0 and $4 \cdot v_{\text{most\_probable}} = 4 \cdot \sqrt{\frac{2RT}{M_{\text{kg/mole}}}}$ for every facet temperature present in the system (this interval contains most of the distribution), and the algorithm - instead of interpreting the analytic expression at each collision - performs an inverse lookup: by reverse interpolation it looks up the speed that belongs to a generated random number.

An additional speedup could be achieved by calculating the inverse of the C.D.F. and doing forward interpolation (interpolate the Y value at a known X value) instead of reverse, but due to the long tail, i.e. non-zero probability for high speeds in the Maxwell-Boltzmann distribution this would introduce sampling artifacts at high speeds.

A summary of the Maxwell-Boltzmann and the modified speed distributions are presented in Table 1, and the two p.d.f. functions are shown in Fig.8.

2.4 Finding next collision location

Once the particles direction and velocity are calculated, Molflow+ uses a ray-tracing algorithm [11] to find the next collision point with a facet. Then, depending on the sticking coefficient (see below) of the hit facet, the particle is either pumped and the algorithm proceeds to generate the next particle from source, or it rebounds from the facet:

[4]We start from the fact that fast particles will collide more often, or more precisely, the collision frequency will scale linearly with the particle speed v:

$$f(v)_{coll} = f(v)_{gas} \cdot v \cdot C \tag{3}$$

where C is a normalizing factor, that can be obtained since the integral of the collision p.d.f. must by definition be 1:

$$\int_0^\infty f(v)_{coll} \cdot dv = 1$$

$$\int_0^\infty f(v)_{gas} \cdot v \cdot C \cdot dv = 1$$

$$C \cdot \int_0^\infty f(v)_{gas} \cdot v \cdot dv = 1$$

$$C \cdot \langle v \rangle_{gas} = 1$$

$$C = \frac{1}{\langle v \rangle_{gas}} \tag{4}$$

where $\langle v \rangle_{gas}$ is the average molecule speed in the gas. In case of the Maxwell-Boltzmann distribution, $\langle v \rangle_{gas} = \sqrt{\frac{8RT}{\pi m}}$ with T the gas temperature, R the ideal gas constant and m the molar mass (in kg). Assuming that we have a gas with Maxwell-Boltzmann distribution, where the speed p.d.f is known[9]:

$$f(v)_{gas(M.B.)} = v^2 \exp\left(-\frac{v^2}{2a^2}\right) \frac{1}{a^3}\sqrt{\frac{2}{\pi}}$$

and substituting the value of the normalizing factor, we can express $f(v)_{coll}$, as eq.2 in the main text.

Table 1: The Maxwell-Boltzmann and the modified speed distribution

	Maxwell-Boltzmann	**Surface collisions**
P.D.F.	$f(v)_{gas} = v^2 \exp\left(-\frac{v^2}{2a^2}\right) \frac{1}{a^3}\sqrt{\frac{2}{\pi}}$	$f(v)_{coll} = v^3 \exp\left(-\frac{v^2}{2a^2}\right) \frac{1}{2a^4}$
	$a = \sqrt{\frac{k_B T}{m_{[\text{kg/particle}]}}} = \sqrt{\frac{RT}{M_{[\text{kg/mole}]}}}$	
C.D.F.	$CDF_{gas} = erf(\frac{x}{\sqrt{2}a}) - \sqrt{\frac{2}{\pi}}\frac{x \cdot \exp(\frac{-x^2}{2a^2})}{a}$	$CDF_{coll} = -\exp(\frac{-x^2}{2a^2})[1 + \frac{-x^2}{2a^2}] + 1$
avg.	$< v>_{gas} = \int\limits_0^\infty v \cdot f(v)_{gas} dv = \sqrt{\frac{8}{\pi}}a$	$< v>_{coll} = \int\limits_0^\infty v \cdot f(v)_{coll} dv = 3\sqrt{\frac{\pi}{8}}a$
speed	$< v>_{gas,\text{300K,28g/mole}} = 476.2\text{m/s}$	$< v>_{coll,\text{300K,28g/mole}} = 561\text{m/s}$
$< \frac{1}{v} >$	$< \frac{1}{v}>_{gas} = \int\limits_0^\infty \frac{1}{v} \cdot f(v)_{gas} dv = \frac{\sqrt{\frac{2}{\pi}}}{a}$	$< \frac{1}{v}>_{coll} = \int\limits_0^\infty \frac{1}{v} \cdot f(v)_{coll} dv = \frac{\sqrt{\frac{\pi}{8}}}{a}$
	$< \frac{1}{v}>_{gas,\text{300K,28g/mole}} = 0.0027 = \frac{1}{374}\text{m/s}$	$< \frac{1}{v}>_{coll,\text{300K,28g/mole}} = 0.0021 = \frac{1}{476.2}\text{m/s}$
Com-parison	$< v>_{coll}/ < v>_{gas} = \frac{3\pi}{8} \approx 1.178$	
	$< \frac{1}{v}>_{gas}/ < \frac{1}{v}>_{coll} = \frac{4}{\pi} \approx 1.273$	

2.5 Pump or bounce

The user can define pumps in the system by assigning sticking factors s (probability of absorption for the impinging particle) or pumping speeds S to certain facets. Assuming equilibrium, the two quantities can be converted to each other:

$$S[m^3/s] = s \cdot \frac{1}{4} v_{avg}[m/s] \cdot A[m^2]$$

where v_{avg} is the average molecule speed and A is the facet area. When a collision happens with a pump facet, a uniformly distributed pseudo-random number is generated, and if it is smaller than s, the test particle is removed from the system and a new one is generated at a source facet. Otherwise the test particle rebounds.

2.6 Registering hits

Pressure on a surface comes from the momentum change rate of particles colliding with it. Every facet (or texture or profile slice, see later) has three counters in memory that we increment by the following quantities:

- We increment N_{hit}, the "number of Monte Carlo hits" counter by 1. Knowing the number of MC hits on a facet will allow us to calculate the impingement rate, moreover it provides important information on the statistical accuracy of our results.
- To $\sum I_\perp$, the "total orthogonal momentum change" counter we add the orthogonal momentum change, $mv_\perp = mv\cos\theta$ of the incoming and outgoing particle (θ is the test particles incident angle, v is its speed and m is its mass) [5]. We will need this counter to calculate the pressure.
- We also use a third counter, $\sum \frac{1}{v_\perp}$ to store the sum of the reciprocals of the orthogonal speed components, $\frac{1}{v_\perp} = \frac{1}{v\cos\theta}$. We need to store this quantity for the calculation of particle density near the facet, as explained later.

2.7 Rebound

Once we have registered the hit by incrementing the counters, unless the particle is pumped, the algorithm proceeds to the next iteration: it assigns a new velocity and a new direction for the outgoing particle. The direction is chosen according to the user setting: by default, a facet is diffuse so a new random direction is generated following Knudsens cosine law, but the user can also choose specular reflection.

A particle can interact with the wall which results in energy and momentum exchange. Its new speed thus depends on the thermal accommodation coefficient, A_{acc}. By default, total thermalization

[5] in case of desorption, the outgoing component is added, whereas the incoming for absorption and both for reflection

($A_{acc} = 1$) is assumed, in which case - regardless of the incident velocity - a new velocity v_{wall} is generated according to the frequency-modified speed distribution, which takes into account the facets temperature.

The accomodation coefficient is defined differently across literature (for example, [12] uses my notation comparing the incident and outbound energies as opposed to [13] comparing the incident and outgoing velocites whereas [14] is distinguishing an orthogonal and tangential accomodation factor). I consider the thermalization coefficient as the ratio of kinetic energy change compared to that of total thermalization (definition above). In this case, the new velocity is calculated as:

$$v_{new}^2 = v_{old}^2 + A_{acc}(v_{wall}^2 - v_{old}^2)$$

where v_{old} is the incident velocity and the thermal accommodation coefficient, A_{acc} must be between 0 and 1.

Once the new speed and direction are chosen, then the orthogonal momentum change and speed reciprocal counters of the facet are incremented again to take into account the outgoing particles impulse and reciprocal speed.

It is worth mentioning that this algorithm is executed parallel on every CPU core present in the system (one core taking care of one test particle at a time), and taking advantage of the linearity of UHV systems, the counters are summed in the end.

2.8 Calculating physical quantities

Using the counters above, the following physical quantities can be calculated:

2.8.0.1 Impingement rate The impingement rate on a wall is the number of collisions, N_{real} per second per unit area. To calculate it, we need the following quantities:

- The number of incoming physical molecules per second each test particle represents, $K_{\frac{real}{virtual}}$
- The number of test particle hits on the given facet, N_{hit}
- The area of the facet, A

With all the above, the impingement rate can be expressed:

$$z = \frac{N_{real}}{dt \cdot A} = \frac{N_{hit} \cdot K_{\frac{real}{virtual}}}{A}$$

Molflow+ applies a few more coefficients to correct for double-sided and half-transparent facets [6], but for simplicity they are omitted here.

2.8.0.2 Pressure When particles desorb, absorb or bounce on a facet, their rate of momentum change exerts a force on the wall, which - divided by the area - is the pressure. Since we have previously stored the sum of momentum change rates in a dedicated counter, to obtain the pressure all we have to do is to scale it up by the $K_{\frac{real}{virtual}}$ ratio (which already includes the time normalization to one second) and then divide by the facet area:

$$p = \frac{\sum dI}{dt \cdot A} = \frac{\sum I_\perp \cdot K_{\frac{real}{virtual}}}{A}$$

2.8.0.3 Density It is not straightforward to calculate density directly in Molflow+: in an event-driven simulator, the main idea is to simulate events only on walls and ignore everything in between, whereas for a direct density measurement we would need to count the number of test particles in a test volume. It is, however, achievable through a simple derivation[7] that connects the density in a volume with the number of particles crossing a surface nearby:

As shown in Fig.5, a molecule traveling with velocity $v_\perp$ will hit a surface within time Δt if it is closer than $v_\perp \Delta t$ to the wall. Therefore, the number of collisions for molecules traveling with velocity $v_\perp$ during the next Δt period is:

$$\#\text{collisions}, v_\perp = \frac{N}{V} A \Delta t v_\perp$$

[6]these half-transparent facets serve to represent periodic structures, for example grids, where regardless of the hit location there is a certain, constant probability of a particle being able to cross to the other side

[7]This derivation can be found in many places in literature, one example is [15]

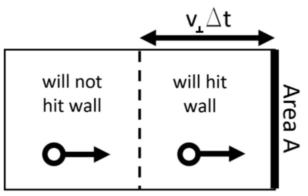


Figure 5: For a given position within the volume, only particles with high enough orthogonal velocity will hit the surface in the next Δt period. Redrawn based on [15]

where N/V is the density of molecules and A is the surface area. Since the molecular speeds are distributed, we integrate this expression over the velocity distribution function $f(v_\perp)$[8]:

$$\#\text{collisions} = \frac{N}{V} A\Delta t \int_0^\infty v_\perp f(v_\perp) dv_\perp = < n >_\text{volume} A\Delta t < v_\perp >_\text{gas}$$

And since the impingement rate is the number of collisions per unit time and area, we can write:

$$< n >_\text{volume} = \frac{\#\text{collisions}}{A\Delta t < v_\perp >_\text{gas}} = \frac{z_\text{surface}}{< v_\perp >_\text{gas}}$$

Here z_surface is the impingement rate on the surface, and $< v_\perp >_\text{gas}$ is the average of the orthogonal speed components of those particles that move toward the surface. Weve already calculated the impingement rate above. Obtaining $< v_\perp >_\text{gas}$ requires more steps, though: unless we use test volumes or other tools, we wont be able to compute it directly. What we can calculate, however, is the speed distribution on an adjacent facet. Using the definition $\int_0^\infty f(v)_{\perp\text{gas}} dv = 1$ we can write:

$$< v_\perp >_\text{gas} = \frac{< v_\perp >_\text{gas}}{\int_0^\infty f(v)_{\perp\text{gas}} dv}$$

$$< v_\perp >_\text{gas} = \frac{< v_\perp >_\text{gas}}{\int_0^\infty f(v)_{\perp\text{gas}} v \frac{1}{v} dv}$$

Then using eq.4:

$$< v_\perp >_\text{gas} = \frac{1}{\int_0^\infty f(v)_{\perp\text{gas}} v C \frac{1}{v} dv}$$

Then, using eq.3 we can write:

$$< v_\perp >_\text{gas} = \frac{1}{\int_0^\infty f(v)_{\perp\text{coll}} \frac{1}{v} dv} = \frac{1}{< \frac{1}{v_\perp} >_\text{coll}}$$

At this point we moved from using $< v_\perp >_\text{gas}$ to $< \frac{1}{v_\perp} >_\text{coll}$ which can be sampled by Molflow+. Therefore the density on a facet, using the $\sum \frac{1}{v_\perp}$ counter, will become:

$$< n >_\text{volume} = \frac{z_\text{surface}}{< v_\perp >_\text{gas}} = z_\text{surface} \cdot < \frac{1}{v_\perp} >_\text{coll} = \frac{N_{hit} \cdot K_{\frac{real}{virtual}}}{A} \cdot \frac{\sum \frac{1}{v_\perp}}{N_{hit}} = \frac{\sum \frac{1}{v_\perp} K_{\frac{real}{virtual}}}{A}$$

2.8.0.4 Average velocity As described in the previous point and also shown in Table 1, the average speed in the gas can be calculated:

$$< v_\perp >_\text{gas} = \frac{1}{< \frac{1}{v_\perp} >_\text{coll}} = \frac{N_{hit}}{\sum \frac{1}{v_\perp}}$$

[8]For clarity, here I ignore the molecules that move away from the surface ($v_\perp < 0$). Otherwise I would integrate from ∞ to $+\infty$ and divide the result by 2

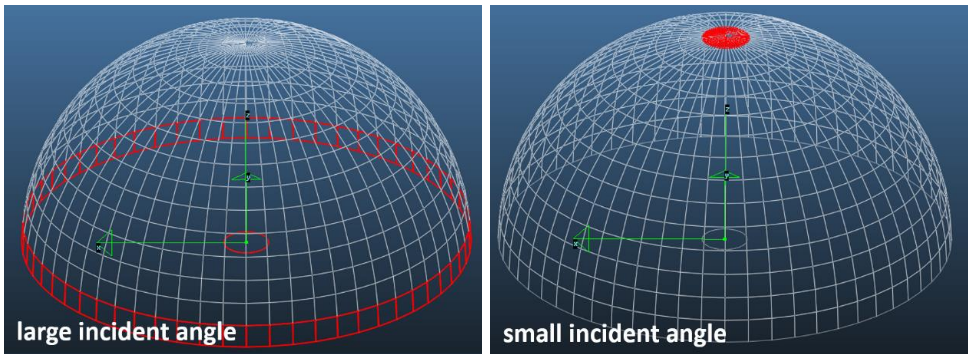


Figure 6: Solid angles for small and large incident angles: even in uniform directional distribution, more particles would reach the target facet (in the center of the hemisphere) from large incident angles

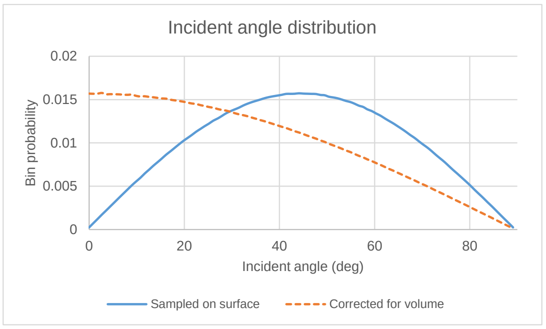


Figure 7: Incident angle distribution on a facet in equilibrium state, for a gas of 28 g molar mass and 300 K temperature

2.8.0.5 Angular profiles In Molfow+ we can also calculate the directional distribution of incident molecules on a surface. A 100-value histogram counts the absolute value of the incident angle (ie. two particles coming from opposite sides with the same incident angle will count the same, since facets dont have default directions). The incident angle range of 0..90 degrees is distributed over the 100 profile values. One important note is that - similar to velocities - there is a difference between angular distribution in a volume of gas and the angular distribution of particles that cross or hit a surface during a given time. This is because particles with large incident angles (incoming almost parallel to the surface) might originate from a larger solid angle than those that arrive perpendicularly. This is demonstrated in Fig.6.

Therefore a normalization is included (as a user option) to correct for this effect: the value for each bin of the angular distribution is normalized by the corresponding solid angle (the solid angle for incident angle θ is $d\omega(\theta,\phi) = \sin\theta d\theta d\phi$, therefore the normalization consists of a division by $\sin\theta$, then a second division to assure that the sum of the distribution is 1). In equilibrium, without this correction, the distribution would be $p(\theta) \approx \sin(\theta)\cos(\theta)$, and with the correction, it becomes $p(\theta) \approx \cos(\theta)$ which is what Knudsens cosine law stipulates. This is shown in Fig.7.

2.8.0.6 Speed profiles We can also sample the distribution of the incident velocities of test particles on a facet. The 100 histogram bins correspond to the speed range $0..4 \cdot v_{\text{most\_probable}} = 0..4 \cdot \sqrt{\frac{2RT}{M_{\text{kg/mole}}}}$. As with the angles, a surface-volume conversion is possible: in this case each value of the distribution is divided by the speed it belongs to, then the distribution is renormalized (effectively converting from $f(v)_{coll}$ to $f(v)_{gas}$ of Table1). There are two types of speed profiles (see Fig. 8): one for the absolute value of the incident speed (giving information on the gas temperature), and one for the orthogonal component (which determines the pressure).

2.9 Error of the results

When assessing the statistical error of Molflows results, we can apply a scientific (quantitative) or a practical (qualitative) approach. I first present the quantitative. In mathematical termos, a Monte

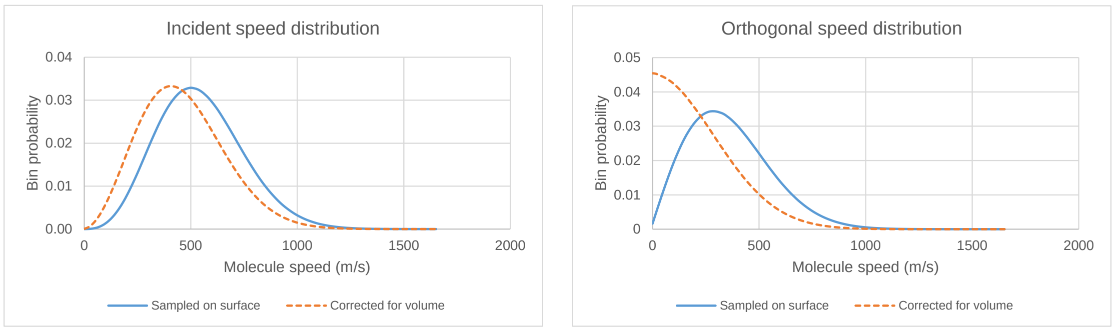


Figure 8: Incident speed and orthogonal velocity distribution on a facet in equilibrium state

Carlo simulator is approximating a physical result by carrying out a given number of experiments and counting the number of positive outcomes.

In case of calculating the transmission probability of a tube, for example, each traced test particle would be one experiment, and the positive outcome would be that the particle leaves the system on the side which is opposite to where it came from. The transmission probability p would correspond to the probability of positive outcomes: $p = n/N$ where n is the number of test particles leaving on the opposite side, and N the total number of traced test particles.

In case of the pressure at a certain location, for simplicity well assume that the gas has a Maxwellian behavior (i.e. the system is isothermal and beaming effects are not important). In this simpler case the pressure is directly proportional to the number of hits at the location. So omitting some normalizing constants - the pressure can be expressed as a function of $p = h/N$, where h is the number of hits at that point, N is the total number of hits and thus p is the probability that a hit happens at the given location.

The two examples above can be translated to probability theory: the discrete probability distribution of the number of successes in a sequence of N independent yes/no experiments, each of which yields success with probability p, is the binomial distribution.

The expected value of positive outcomes of such a series of experiments is $N \cdot p$, and the variance of the result is $N \cdot p \cdot (1 - p)$.

As suggested by Suetsugu [7], for Monte Carlo simulations we can assume a very large number of experiments, where using the notation of the two examples above, $p \sim n/N$ or $p \sim h/N$. In this case the normalized standard deviation, expressed as a percentage of hits (positive outcomes), can be derived from the variance formula above as:

$$\sigma_n\,(\%) = \frac{\sigma}{n} \cdot 100 = \sqrt{\frac{1}{n}\left(1 - \frac{n}{N}\right)} \cdot 100 \tag{5}$$

Suetsugu also correctly points out that as we trace a particle through consecutive hits, not each experiment is independent from the others statistical fluctuations in adjacent locations are correlated distorting the above estimation. I would argue that due to this, the exact error depends on the geometry of each simulation and as such there is no universally valid formula to estimate it. However, from eq.5 we can retain that the error of $p \sim n/N$ scales with $1/\sqrt{N}$: for example, to reduce the error of the calculated pressure to its half requires to run the simulation four times longer.

Since the quantitative formula only gives an approximation, it is important to mention the qualitative approach that most Molflow+ users go along with when simulating a vacuum system. As mentioned above, post-processing visualization tools are updated every second as simulation runs, which as shown in Fig.9 means in practice that profiles and textures become smoother. In most cases the user has an expectation of the pressure profile, so statistical scattering can be visually identified through inhomogeneities of drawn textures.

Staying at the example in Fig.9 for example, the engineer using the software either knows that the profile from desorption to the pump should be linear, thus he/she can fit a straight line on the obtained plot and isolate the statistical error. Otherwise the user can argument that the system is axisymmetric, therefore texture cells at the same longitudinal coordinate should have the same color, and deviation from it must be due to statistical scattering. The dependence of the error can be verified to vary with

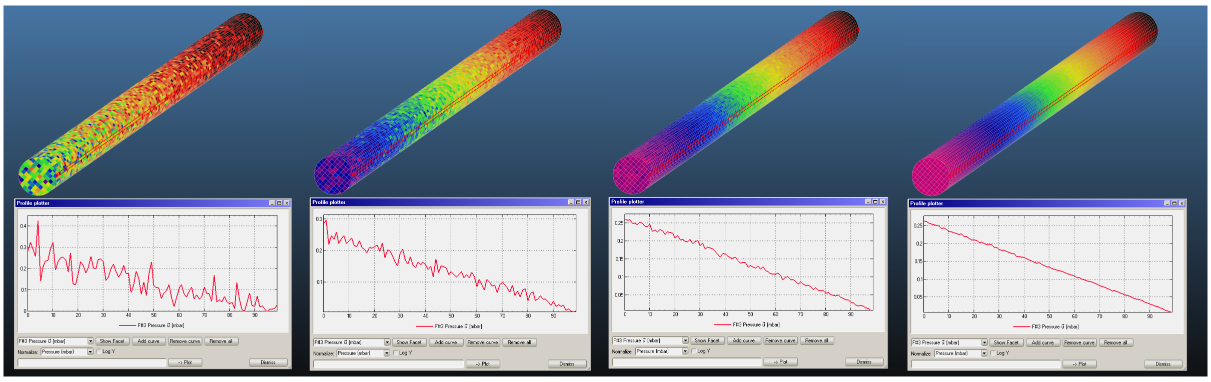


Figure 9: Texture and profile results for a vacuum tube with desorption at the left and pumping at both sides, as a function of the number of traced particles.

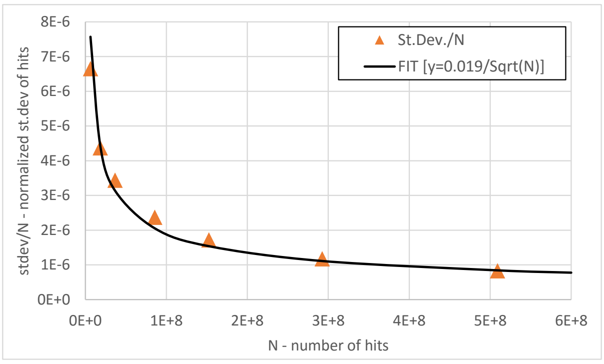


Figure 10: Normalized standard deviation of Molflow+ results as a function of the simulated test-particles

$1/\sqrt{N}$. In the above example I calculated the deviation of the number of hits [9] from the theoretical solution (which in our example is a pressure profile of a straight line) and plotted it as a number of the traced test particles in Fig.10.

The plot, containing a fit with the analytic formula confirms that the error squares approximately with the inverse of the square root of the number of hits.

3 Time-dependent Molflow+ simulations

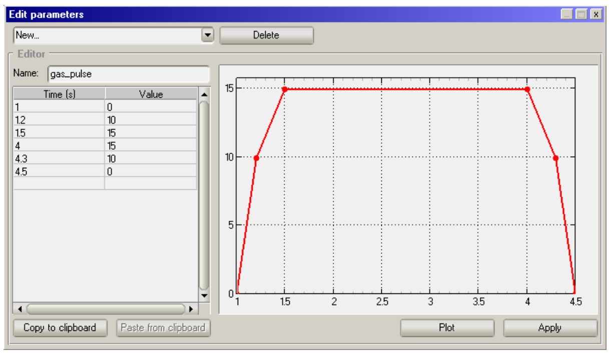
Once we keep track of the molecule speeds, apart from the possibility to calculate the pressure in a direct way (based on particle momentum change), it also becomes possible to determine the time when each hit happens, which opens the way to simulations in the time domain. On every facet, instead of calculating one global (steady-state) statistics using every registered hit, we take into account only those that happen at a certain moment - thus we can obtain the physical quantities for that instant.

The algorithm is still event-driven, and is essentially the same as that of the steady-state simulations explained above, but some corrections and fine-tuning is required.

3.1 Parameters

When our simulation treats time-dependent problems, both the outgassing and the facets physical parameters, such as sticking factor, opacity, etc. can vary over time. This is included in Molflow+ through parameters (see Fig.11): the user references a parameter (by name) at the given facets property list, then he proceeds to define that parameter by entering a list of time moments and the parameters values at those instants. For every event on that facet, Molflow+ will use the actual, time-dependent value, obtained by linear interpolation within the user-defined defined time range, and by constant extrapolation

[9] I extracted the number of Monte Carlo hits instead of the pressure because the latter is already normalized by the number of traced particles



Name: gas_pulse

Time (s)	Value
1	0
1.2	10
1.5	15
4	15
4.3	10
4.5	0

Figure 11: Time-dependent parameter definition for a gas pulse in Molflow+

outside. This is done directly for sticking factors and opacity, and with an intermediate step (integration) for time-dependent outgassing, see later.

3.2 Moments and time window

We aim to simulate a large number - possibly billions - of hits. In the original, steady state algorithm, this doesnt cause a memory problem since these hits arent stored individually: they only increment counters on facets.

In the time-dependent mode, however, the direct way to gain statistics would be to store a list of hits along with the time they occur for every facet[10]. This would, consequently, mean that the longer the simulation runs, the more memory wed need. But limiting the number of hits, and along with it the statistical accuracy is not an option, therefore we have to implement an indirect approach.

We use binning: we ask the user to decide in advance at what moments he is interested in the pressure and density profiles. This is usually a series of moments $t_1, t_2 ... t_N$ with equal or logarithmically increasing differences, ie. the time range when the dynamics of the system is interesting. We will then create three counters, for every moment and every facet:

- $N_{hit,i}$, the number of MC hits on the facet at moment i
- $\sum I_{\perp,i}$, the sum of orthogonal momentum changes of particles hitting the facet at moment i
- $\sum \frac{1}{v_{\perp,i}}$, the sum of the reciprocals of the orthogonal speed components of particles hitting the facet at moment i

Consequently, if a hit happens at t_i, it will increment the ith counter. That way the pressure and other physical quantities at that moment can be calculated similarly to the steady state using values of the counter with the same index, and the user has several tools to visualize the dynamic behavior of the system.

One problem to tackle though is that no hit will happen exactly at t_i with infinite precision. Therefore we introduce the term time window, serving as a tolerance: we will treat all hits between $t_i - \frac{t_{\text{window}}}{2}$ and $t_i + \frac{t_{\text{window}}}{2}$ to have happened at the ith moment and accordingly, increase the ith counter.

The time window can be set by the user. In practice, it has to be small enough to reveal the fast pressure changes in the system, but large enough to contain a number of hits that is sufficient for the desired statistical accuracy. The good choice is one that matches the pressure change speed of the simulated system, and sometimes it needs to be found by trial and error across multiple simulations. This is demonstrated during the validation in section 6.1.

3.3 Particle generation

When an outgassing is time-dependent, we must:

- Generate particles so that the distribution of their time of insertion follows the user-defined desorption parameter.
- Know how many real molecules one test particle represents.

[10]As my simulations are event-driven, there is no default "time step" to store results for.

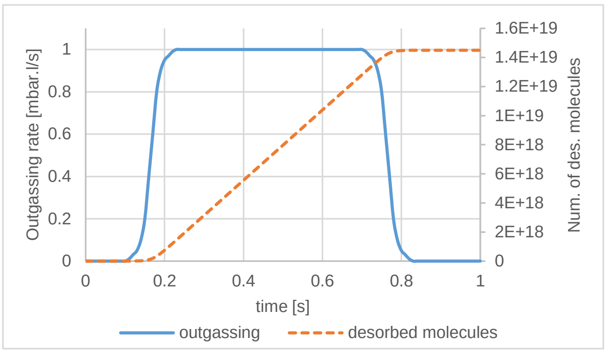


Figure 12: Desorption rate and number of desorbed molecules during injection

Both tasks can be solved by integrating the user-defined desorptions on the range $t = 0..t_{\text{last}}$, for all outgassing facets, where t_{last} is the last user-defined moment where the pressure needs to be calculated (after which we are not interested in the behavior of the system). We store the integrated desorption function - as illustrated in Fig.12. -, which will also tell us the total number $N_{f,real} = \int\limits_{0}^{t_{last}} \frac{dN_{f,real}(t)}{dt} dt$ of desorbed particles, on facet f.

For each generated virtual particle, Molflow+ then chooses a starting location. The probability of choosing a certain facet f as source is proportional to $N_{f,real}$, just like in steady-state. Once the source facet is chosen, the start position of the test particle is generated randomly with uniform distribution on its surface. To decide the desorption time, we generate a random number uniformly between 0 and $N_{f,real}$, then interpolate the desorption time in the integrated desorption function corresponding to that random number. That way, the particles will enter the system following the user-defined distribution, and when the simulation traced $N_{virtual}$ test particles, each one will represent

$$K_{\frac{real}{virtual}} = \frac{\sum_f N_{f,real}}{N_{virtual}}$$

real molecules.

3.4 Radioactive decay

As we already keep track of the flight time for every particle, it is straightforward to add radioactive decay to the algorithm: upon particle generation, the MC algorithm assigns the instant of decay, t_{decay} for each particle of user-defined half-life $T_{1/2}$ entering the system, with the method described hereafter. The probability that the particles lifetime will be t is given by the exponential probability density function:

$$P(t) = \frac{1}{\tau} \exp(-\frac{t}{\tau})$$

where $\tau = T_{1/2}/ln(2)$. Random numbers following this distribution are then generated by the inversion method: a random number rnd, generated uniformly between 0 and 1 must equal the cumulative distribution function:

$$rnd = 1 - \exp(-t/\tau)$$

The solution of this equation for t is:

$$t = -\tau \cdot ln(1 - rnd)$$

Using rnd in place of $1 - rnd$, we can express the time of decay:

$$t_{\text{decay}} = t_{des} - \tau \cdot ln(rnd)$$

where t_{des} is the time at which the particle enters the system. Each time a particle hits a facet, the time of the hit t_{hit} is compared to t_{decay}, and the particle is eliminated at the first hit when $t_{\text{hit}} > t_{\text{decay}}$. As Molflow+ is event-driven, the exact location of the decay (in the volume, between two hits) has no interest because statistics is gathered only on surfaces. One exception is if the user sets up counter facets: these transparent facets in the volume execute the lifetime check of all passing particles, therefore - if needed - they allow visualization of the decay en route between the vacuum chamber walls.

3.5 Wall sojourn time

Once the time when the particle enters the system is known, for every subsequent hit the flight time is incremented by the time elapsed since the last collision (which is calculated by dividing the distance from the previous hit location with the particles speed). That way for every hit the time it occurs is known.

One phenomenon to take into account for making these hit times more accurate is the residence time τ of a molecule on the surface.

The physical process is essentially the same as in case of the radioactive decay: each molecule on the surface has a certain, constant probability of escape, resulting in an exponential probability distribution of the residence time. Its average value has the expression[16]:

$$\tau = \tau_0 \cdot \exp(Q/RT_f)$$

where T_f is the facet temperature, τ_0 is the molecule vibration period, and Q is the molar adsorption heat. If the user defines these two parameters (two, because T_f is already defined) then each bounce time is delayed with a sojourn time that is generated following the exponential probability density function.

3.6 Calculating physical quantities

We can calculate impingement rate, pressure and density similar to the methods in the steady-state simulations, with one remarkable difference: while at steady state, each test particle represented a certain number of physical molecules entering the system *per second*, this time $K_{\frac{real}{virtual}}$ stands for a certain - absolute - number of physical molecules that enter the system during the whole simulated process. Therefore, in time-dependent calculations, the dt in the pressure and impingement rate formulas is not included anymore in $K_{\frac{real}{virtual}}$, and has to be replaced by the time window, as that represents the duration during which hits are considered to happen at the same moment and increase the same counter. Therefore, for the ith moment, physical quantities can be calculated as follows:

3.6.0.1 Impingement rate Per definition, the impingement rate is the number of collisions per second per unit area, however, our counter contains hits for the period of the time window, so to have the per second hit rate, we divide by t_{window} to get:

$$z_i = \frac{N_{real,i}}{dt \cdot A} = \frac{N_{hit,i} \cdot K_{\frac{real}{virtual}}}{t_{\text{window}} A}$$

3.6.0.2 Pressure Similar to the impingement rate, we need to divide by t_{window} which represents dt in the pressure formula at the ith moment:

$$p_i = \frac{\sum dI}{dt \cdot A} = \frac{\sum I_{\perp,i} \cdot K_{\frac{real}{virtual}}}{t_{\text{window}} A}$$

3.6.0.3 Density To calculate the density at moment i, we start from the impingement rate and the sum of reciprocal orthogonal velocities at that moment, and apply the correction by t_{window}:

$$< n >_{\text{volume},i} = \frac{z_{\text{surface},i}}{< v_\perp >_{\text{gas},i}} = z_{\text{surface},i} \cdot < \frac{1}{v_\perp} >_{\text{coll},i} = \frac{N_{hit,i} \cdot K_{\frac{real}{virtual}}}{t_{\text{window}} A} \cdot \frac{\sum \frac{1}{v_{\perp,i}}}{N_{hit,i}} = \frac{\sum \frac{1}{v_{\perp,i}} K_{\frac{real}{virtual}}}{t_{\text{window}} A}$$

3.6.0.4 Average velocity Like at steady-state, the average speed at moment i can be calculated:

$$< v_\perp >_{\text{gas},i} = \frac{1}{< \frac{1}{v_\perp} >_{\text{coll},i}} = \frac{N_{hit,i}}{\sum \frac{1}{v_{\perp,i}}}$$

4 Post-processing in Molflow+

Although results of Molflow+ simulations can be exported for external processing, the design idea was to create an all-in-one vacuum simulation tool, therefore some post-processing is provided within the graphical user interface. Since these post-processing tools are often used in the thesis' images, they are briefly presented here.

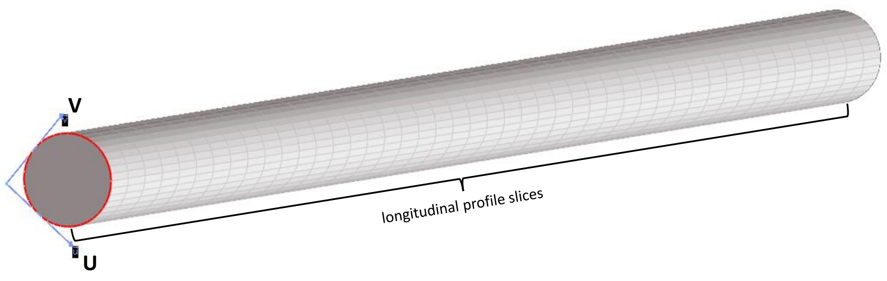


Figure 13: local U,V coordinate system of a facet and the longitudinal profile slicing

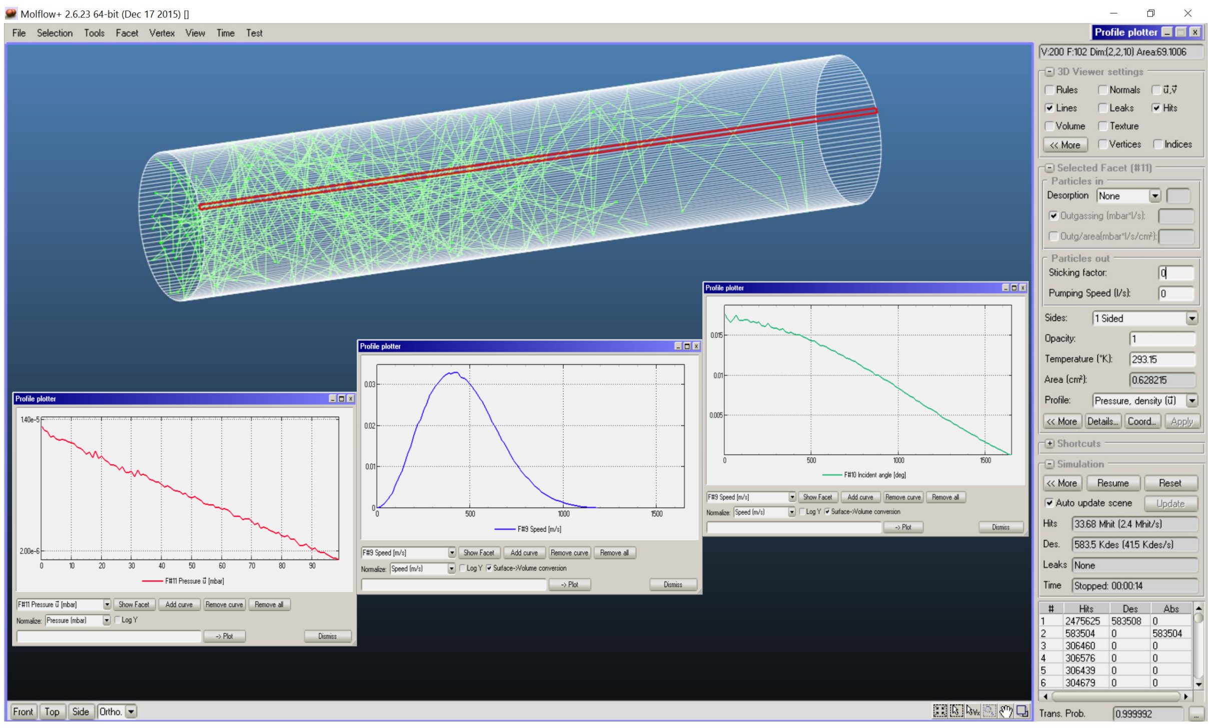

Molflow+ 2.6.23 64-bit (Dec 17 2015)
File Selection Tools Facet Vertex View Time Test
Profile plotter
F#11 Pressure [mbar]
F#9 Speed [m/s]
F#10 Incident angle [deg]
Front Top Side Ortho.

Figure 14: Pressure, speed and angular profiles for a simple tube within Molflow's interface

4.1 Profiles

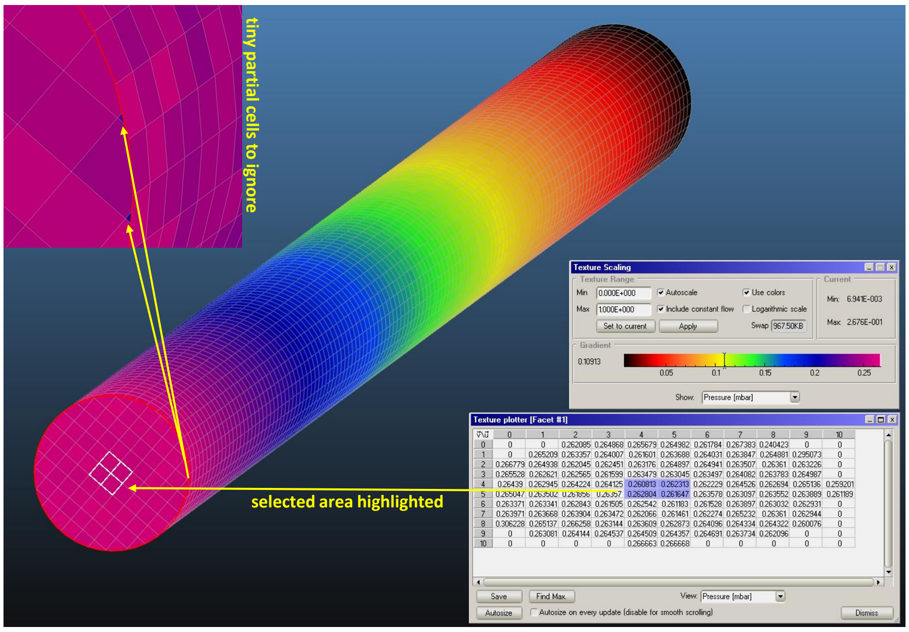
Each facet has a local, 2D coordinate system, with the origin being one of the facets vertices, and its local U, V vectors delimiting the bounding box, orthogonal to each other. By default, the U vector is aligned with the facets longest side, though this is adjustable by the user.

As shown in Fig.13, a profile splits the facet to a given number of slices (currently 100) along the U axis, and samples the pressure, density, number of Monte Carlo hits or the impingement rate independently for each slice. The goal is to plot within the user interface the evolution of a physical quantity along a direction.

There are three special profiles, presented earlier which count a distribution of a quantity for the entire facet:

- Angular profiles sample the distribution of the incident angles
- Speed profiles sample the distribution of the incident molecular speed
- Orthogonal velocity profiles are similar to speed profiles, but sample only the component of the molecule speed orthogonal to the facet

As an example, for a simple tube with a length/radius ratio of 10, I desorb gas at one side with $Q = 1 \times 10^{-4}$ mbar l/s and plot the pressure, speed and angular distributions on a wall facet in Fig.14, after 500.000 desorbed test particles.



Min 0.000E+000 [✓ Autoscale] [✓ Use colors]
Max 1.000E+000 [✓ Include constant flow] Logarithmic scale
Swap 967.50KB
Min: 6.941E-003
Max: 2.676E-001
Show: Pressure [mbar]

Texture plotter [Facet #1]

$\vec{v}\backslash\vec{u}$	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0.262085	0.264868	0.265679	0.264982	0.261784	0.267383	0.240423	0	0
1	0	0.265209	0.263357	0.264007	0.261601	0.263688	0.264031	0.263847	0.264881	0.295073	0
2	0.266779	0.264938	0.262045	0.262451	0.263176	0.264897	0.264941	0.263507	0.26361	0.263226	0
3	0.265528	0.262621	0.262565	0.261599	0.263479	0.263045	0.263497	0.264082	0.263783	0.264987	0
4	0.26439	0.262945	0.264224	0.264125	0.260813	0.262313	0.262229	0.264526	0.262694	0.265136	0.259201
5	0.265047	0.263502	0.261856	0.26357	0.262804	0.261647	0.263578	0.263097	0.263552	0.263889	0.261189
6	0.263371	0.263341	0.262843	0.261505	0.262542	0.261183	0.261528	0.263897	0.263032	0.262931	0
7	0.263971	0.263668	0.263904	0.263472	0.262066	0.261461	0.262274	0.265232	0.26361	0.262944	0
8	0.306228	0.265137	0.266258	0.263144	0.263609	0.262873	0.264096	0.264334	0.264322	0.260076	0
9	0	0.263081	0.264144	0.264537	0.264509	0.264357	0.264691	0.263734	0.262096	0	0
10	0	0	0	0	0.266663	0.266668	0	0	0	0	0

View: Pressure [mbar]
Autosize on every update (disable for smooth scrolling)

Figure 15: Textures and mesh cells within Molflow+

4.2 Textures

Textures similar to profiles can sample a physical quantity within a facet, but split it in not one but two directions (along the local U and V vectors on the plane of the facet). The resolution (cell size) is chosen by the user. Results can be read with a Texture Plotter tool numerically, or visualized by a linear or logarithmic color scale, with parameters adjustable by the user.

As most physical quantities depend on the cell area, it must be calculated carefully for cells on the edges. The area of these partial cells can be very tiny compared to full cells, and therefore they might receive only a very small number of MC hits during the whole simulation. For plotting and autoscaling purposes, it is the best to ignore their values this can be done automatically in Molflow+ if their area is smaller than a given percent of whole cells (as in Fig.15).

Visualization is supported for the pressure, the impingement rate and the density.

4.3 Direction vectors

For each texture cell, the user has the option to evaluate the prevailing molecular direction. This is the average [11] of all velocity vectors incident to the surface (or crossing it in case of transparent facets). This is shown in Fig.16 for a simple effusion example, where a gas is injected to an infinite, empty volume (modelled by a box with sticking walls) with an outgassing rate of $Q = 1 \times 10^{-4}$ mbar l/s.

The importance of these direction vectors is emphasized when simulating non-isothermal systems, such as the hot filament / cold walls example in Fig.17, where we can clearly observe a deviation from the isotropic directional distribution of equilibrium gases.

4.4 Formulas

We also let the user get more insight into the system by allowing him to define formulas, which are symbolic expressions referring to facet parameters. One can refer to the number of hits, the pressure, impingement rate, etc. of a facet or a group of facets, and even extract global simulation parameters, such as the mean free path (in this sense, the average distance of flight between two wall hits), mean pumping path (average flight distance from desorption to pumping or to radioactive decay point), total particle influx of the system or the gas mass.

As common mathematical operators, trigonometric functions and physical constants are also included, among others, these formulas are extensively used for conductance calculations, as in Fig.18.

[11] averaging is done vectorially: two velocity vectors in the opposing directions cancel each other. Consequently, in equilibrium, this average velocity vector is zero.

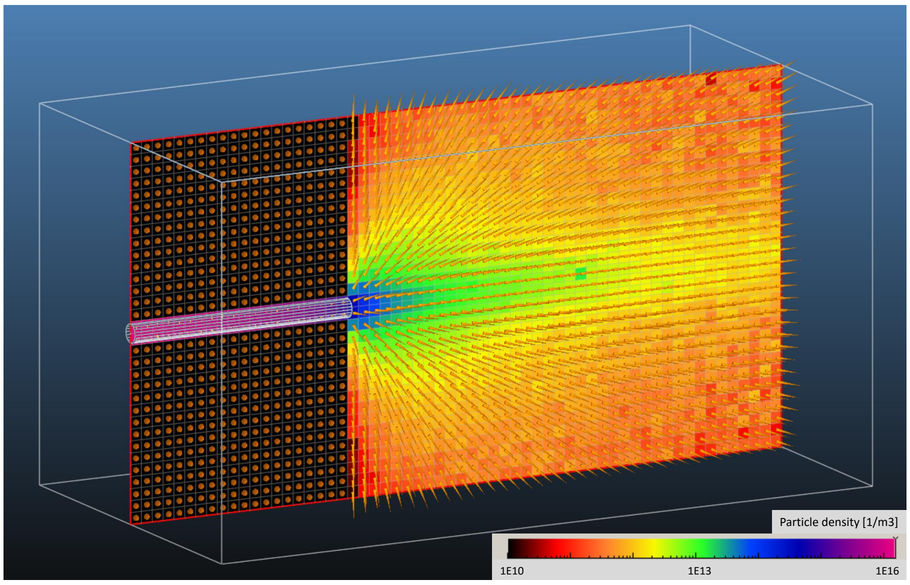


Figure 16: Direction vectors and density texture in an effusion example

4.5 Time-dependent visualization tools

When performing time-dependent simulations, all the post-processing tools above are available. The user can step through the predefined moments and the on-screen visualizations are updated for that given moment. Succession of these visualized system states can be used as frames to compile short videos showing the system behavior.[12]

In the time-dependent mode, two additional tools are available:

- A *Timewise Plotter* (Fig.19) that graphs the pressure profile evolution in a certain direction across different moments on a chosen surface.
- The *Pressure Evolution* (Fig.20) that shows how the pressure on a profile changes over time: the X scale contains all the user-defined moments, and the Y value is the pressure, density or velocity at that moment.

5 Sectioning the geometry

The complexity limits of the systems that we can simulate depend on the computation speed. In general, the more facets we have, the longer the ray-tracing algorithm takes to determine the closest collision point and thus the slower the simulations are. There are some ways, though, to include large or complex vacuum systems and at the same time mitigate the speed impact of the large number of facets.

5.1 Teleports

Molflow+ allows the definition of periodic boundary conditions. When the ray-racing algorithm finds a collision with such a boundary, the virtual particle is moved to an other, referenced facet (the opposite side in case of a periodic B.C.) hence the name "teleport". If we can replace a large geometry with only one pattern that is repeated, we reduce the number of facets accordingly.

In the example below, a simple RF structure is simulated with gas injection on one side and pumping on the other. Two models are built for comparison: one containing the full revolution, and one with only a 45° slice (Fig.21). In the latter case, the walls of the slice are set as teleports referencing each other. The total desorption of the slice is one-eighth of the full revolution (1×10^{-5} mbar l/s vs. 8×10^{-5} mbar l/s) resulting in the same influx density, and the pumping speed is also one-eighth (in practice, this results in the same sticking factor of 1 for both surfaces). As the surface normals are tilted by 45 degrees compared

[12]Two examples of such videos are shared on these permalinks:
https://goo.gl/fgtTQR (Gas expansion and pumpdown following an RF cavity electric breakdown)
https://goo.gl/Pd31Q6 (Gas expansion and pumpdown in a straight tube)

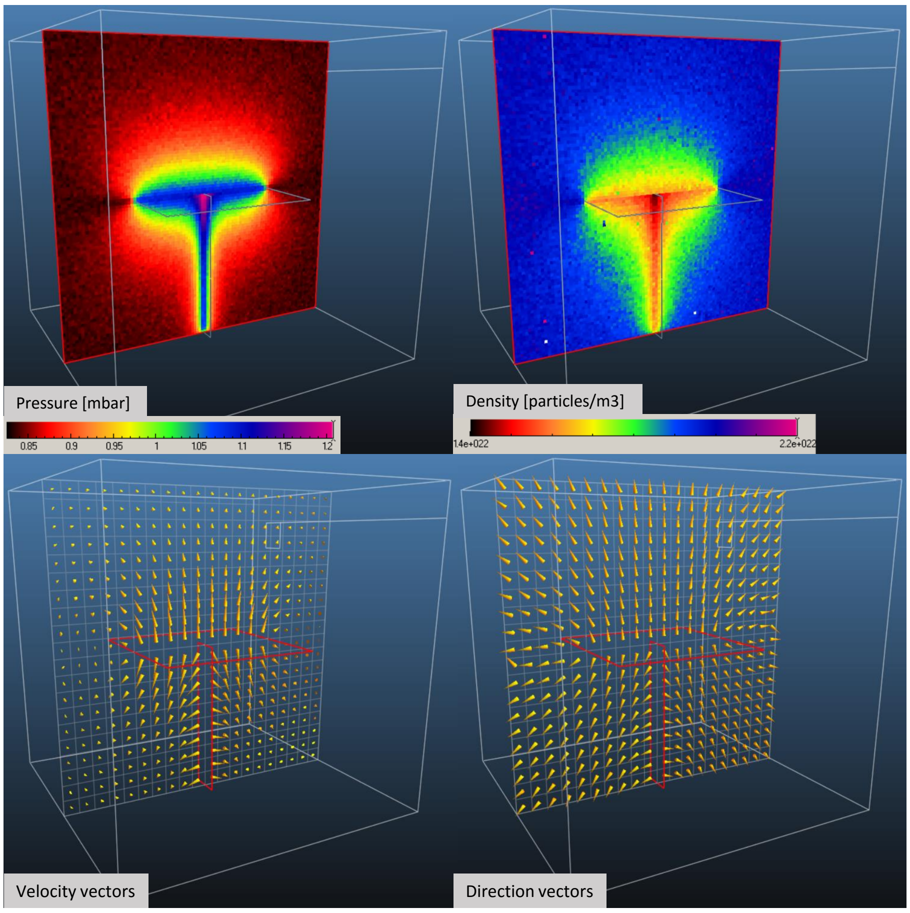


Figure 17: Pressure, density, velocity and direction vectors in a non-isothermal system at equilibrium; with a hot filament at $T = 800\,\mathrm{K}$ at the center and walls at room temperature

to each other, when a teleport happens, the particle direction is also rotated (Fig.22) to correspond to the real case.

The resulting simulation speed improvement is summed in Table2, and the resulting pressure profile (Fig.23) is the same.

One other use of teleports is to include a reusable geometry feature only once. This can be for example a pumping group. Molflow+ can memorize where test particles were teleported from, and if molecules find their way back to the teleporting surface, they are transported back to the origin. This as shown on the example in Fig24 is essentially a single-to-multiple logical assignment, once again reducing the number of facets.

5.2 Structures

As stated before, the ray-tracing checks for collisions with all facets and then among the collisions found, it selects the shortest (although due to optimizations the exact implementation is somewhat different).

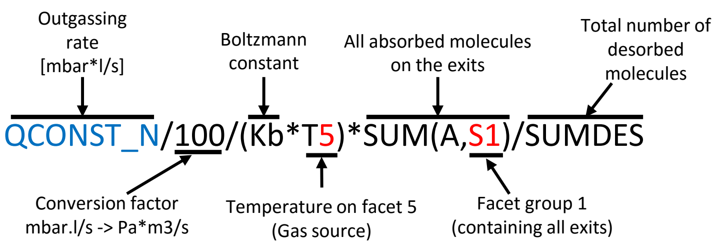

Outgassing rate [mbar*l/s]
Boltzmann constant
All absorbed molecules on the exits
Total number of desorbed molecules
QCONST_N/100/(Kb*T5)*SUM(A,S1)/SUMDES
Conversion factor mbar.l/s -> Pa*m3/s
Temperature on facet 5 (Gas source)
Facet group 1 (containing all exits)

Figure 18: Molflow+ formula for calculating the outgoing flux of particles on a vacuum part

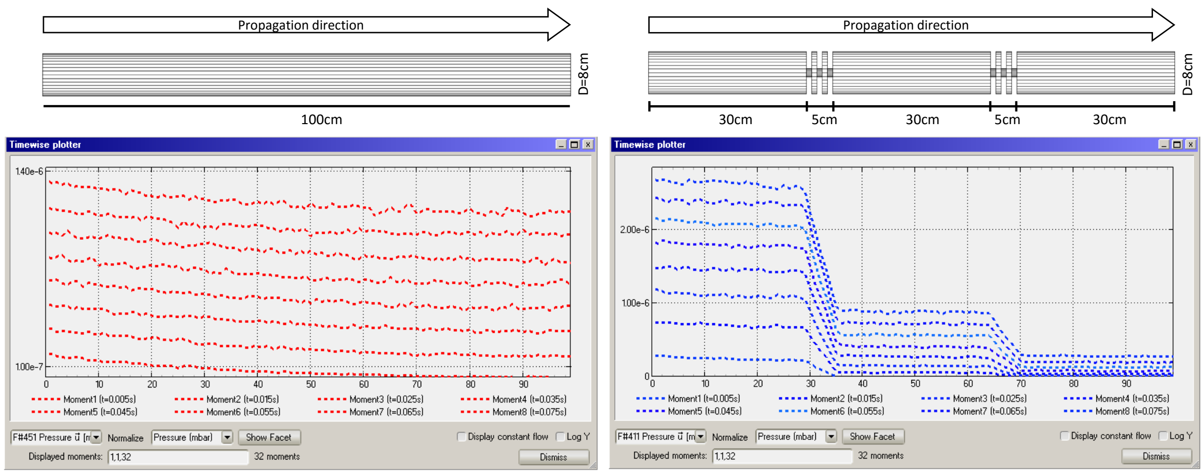


Figure 19: Timewise Plotter in Molflow+ 2.6 showing the pressure evolution at eight different moments in a tube without (left) and with (right) an acoustic delay device

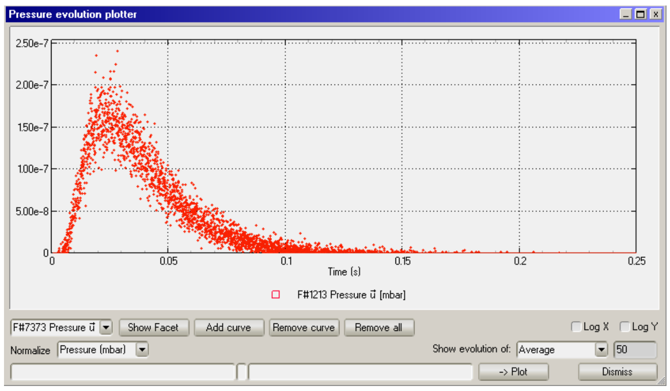


Figure 20: Pressure evolution plotter window showing the pressure response signal on a target after a desorption pulse at $t = 0$

In Molflow, it is possible to speed up the process by dividing a usually linear geometry to distinct sections, and defining them as separate structures. In this case collisions are only sought with facets in the actual structure. Connection between the structures is done by link facets. When a test particle crosses such a link facet, its actual structure is updated.

A very simple example is shown in Fig.25, where a short pipe is split in half, with a link facet in the middle. The speed advantage is almost linear: in theory splitting to N structures would yield an N times faster simulation, but the speedup is mitigated by the extra collisions with link facets, the fact that not all structures have the same number of facets, and the non-negligible user time spent with setting up the sectioning and the links.

To give a real-life example, the method has been applied for a static analysis of the radioactive ion-beams facility ISOLDE. Working together on the problem with my colleague M. Maietta, as explained in detail in her MSc thesis [17], the model of the ISOLDE vacuum line was split to 9 different structures, as shown in Fig.26, making the simulation approximately four times faster, as shown in Table3.

Table 2: Periodic B.C. performance results

	Full revolution	**45° slice with teleports**
Number of facets	25491	3492
Traced molecules/sec	10.5	36
Hits/sec	8200	28000

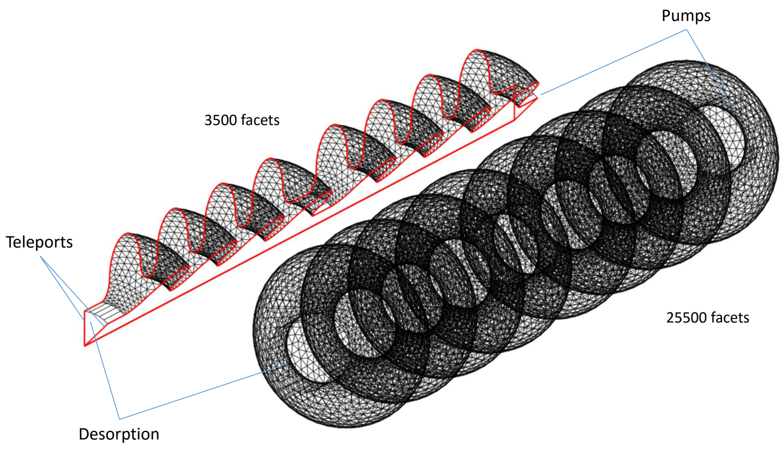


Figure 21: Periodic structure (left) and its full equivalent (right)

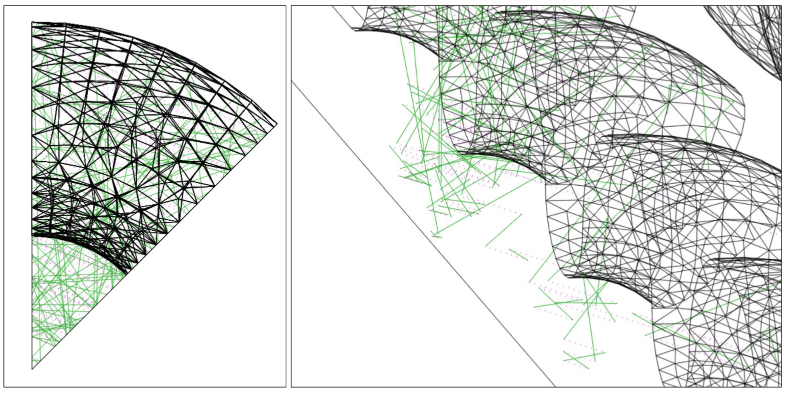


Figure 22: Particle trajectories rotating as following teleports

5.3 Simulating by parts (Convolution method)

When simulating linear vacuum systems with powerful pumps, a relatively frequent problem is the large pressure difference between different parts. When Molflow+ traces molecules, a test particle always represents the same number of physical particles (or in case of steady-state simulations, a flux of a certain number of particles per second). This means that if there is a part of the geometry where the pressure is orders of magnitude higher than at other parts, most of the ray-tracing will ”focus” on that high-pressure part.

This is shown on an example in Fig.27 showing a linear vacuum system with a desorption of 1×10^{-3} mbar l/s on one side and a series of pumps with sticking factor 1 placed downstream. Most of the test particles are caught by the pumps before they can reach the low-pressure region on the right side. Since the pressure difference is of 5 orders of magnitude (1×10^{-3} mbar vs. 1×10^{-8} mbar), to simulate a single hit on the low-pressure side, 100.000 hits must be calculated on the high-pressure side. This results in an inefficient distribution of test particles, and the statistical scattering will be very high on the low-pressure side.

An idea to tackle this issue is to do two (or even more) different simulations instead of one: If we determine that as shown in Fig.27 only 0.1% of the particles desorbed from point A will reach the middle (point B) of the system (we will refer to this ratio as transmission probability), then we could

Table 3: Teleport performance results

	Four pumping ports	**One p.p. with teleport**
Number of facets	701	601
Traced molecules/sec	1.600	2.200
Hits/sec	560.000	776.000

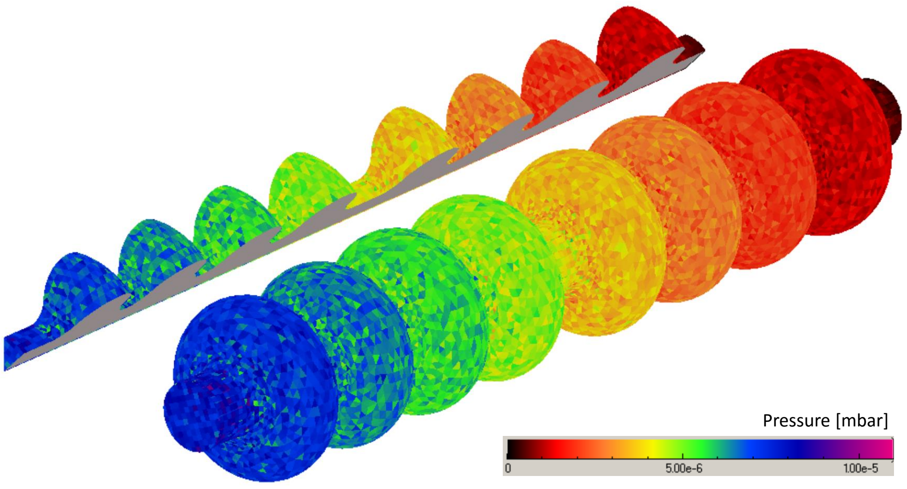


Figure 23: Pressure results for the periodic and the full RF model

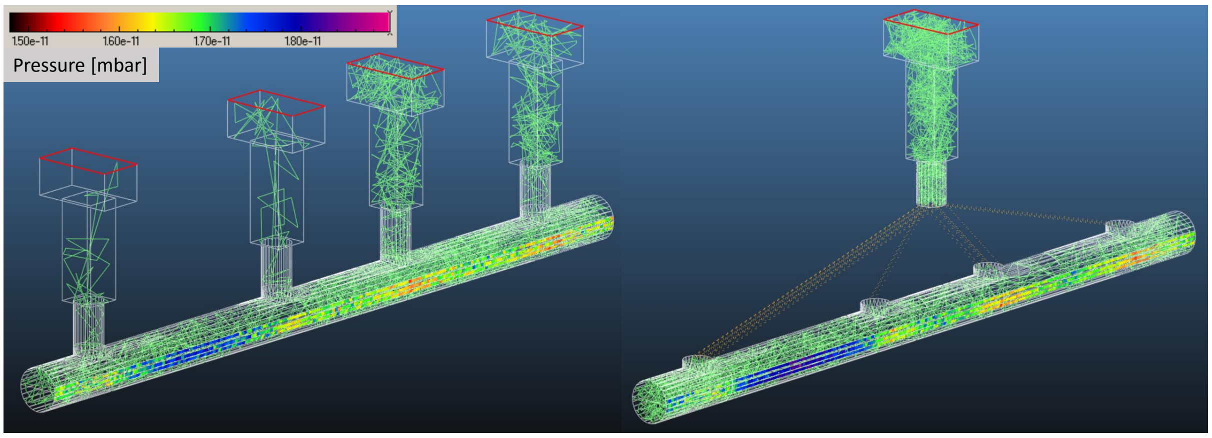


Figure 24: The same system modeled with 4 distinct pumping groups (left) and a reused structure with teleports (right). Teleports are marked with yellow dashed lines. The resulting pressure profile (color code, on the nearest side of the pipes) is the same.

launch a second simulation, desorbing particles only from point B to the right side, with a physical desorption rate which is 0.1% of the original. That way the speedup on the low-pressure side is of a factor of 1000. We could apply this method repeatedly, launching a new simulation from every point where the pressure drops significantly (launching a third simulation at point C would improve statistics downstream by an additional factor of 100).

When applying this method, we have to take into account the beaming effect, though: in a tubular geometry where there is a dominant molecular direction (from the desorption point to the right, in our case), the molecular motion is not isotropic. In Molflow, we can calculate the distribution of the incident angles at the desired relaunch point through angular profiles, and if significantly different from isotropic we can adjust the directional distribution of the new simulations desorption to one that is similar to what we sampled.

Staying at the ISOLDE vacuum system, this method was applied to calculate the time of flight of radioactive isotopes from point A to C, as shown in Fig.28. The method is described in detail in Chapter 6 of [17]. To summarize it, the idea is the following:

If the time of flight from point A to B is described by the (discrete) distribution f, and the T.O.F. from B to C is described by g, then the T.O.F. from A to C will be a sum of two random variables, and its distribution will be described by the convolution of f and g [18]:

$$(f * g)[n] = \sum_{\infty} f[m]\, g[n-m]$$

Since due to the pumps upstream of point B, there is a significant pressure drop between points A

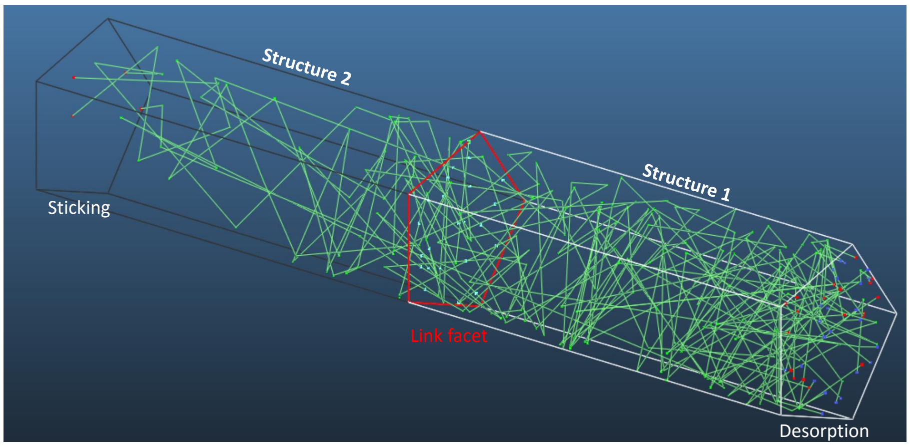


Figure 25: A simple system split to two structures, with the link facets (red) in the middle

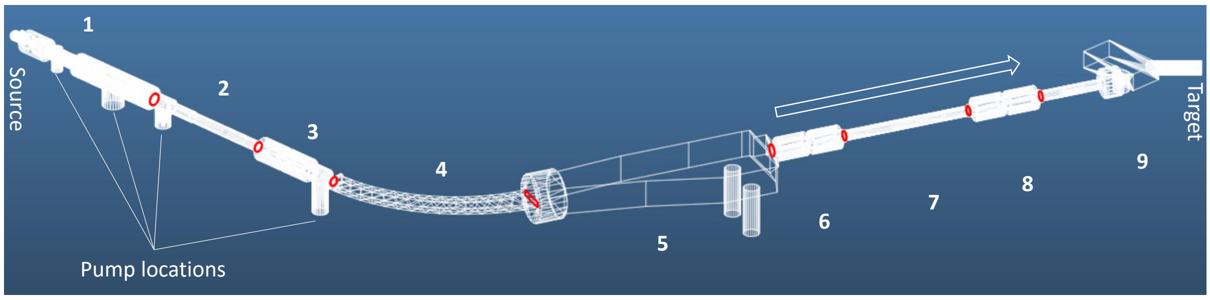


Figure 26: Sectioned system of the ISOLDE transfer line, drawn based on geometry from Maddalena Maietta[17]

and B, it is a good idea to as described before perform two separate simulations.

- The first simulation will determine the T.O.F. distribution f and the transmission probability from point A to B (distance: 661 cm)
- The second will launch test particles from point B, with a physical outgassing rate adjusted by the previously determined transmission probability, and give the T.O.F. distribution g between points B and C (of 690 cm distance)
- The resulting T.O.F. distribution for the whole system will be the convolution of f and g, as shown in Fig.29.

Table 4: Sectioning details

Structure	**Number of facets**
1	9180
2	699
3	959
4	321
5	192
6	682
7	143
8	741
9	1049
SUM	13966

Table 5: Sectioning performance

	No sectioning	9 structures
Traced molecules / sec	2.2	8.2
Hits / sec	4000	17000

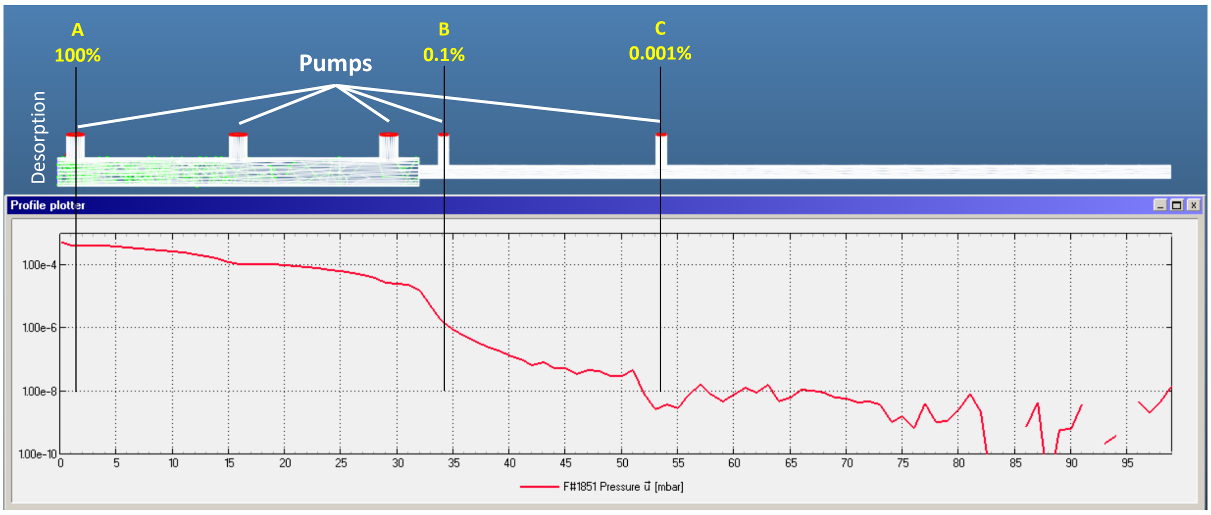


Figure 27: Linear system with desorption from the left and pumps at locations marked by red. The pressure profile, as calculated after 1 million desorbed test particles, is shown on a logarithmic scale under the geometry, with transmission probabilities marked in yellow at three locations. Statistical scattering is high at low-pressure parts.

6 Theoretical validation

The original, steady state algorithm of Molflow has been validated and used on many occasions in accelerator design since its first publication over the last 20 years. We concentrate therefore on the new algorithm capabilities:

6.1 Time-dependent simulations

First we compare a simulation in the time domain against textbook results. We simulate injection and pump-down processes in a cylinder of 10 cm length and 10 cm diameter (thus a volume of $V = 0.785\,\mathrm{l}$). We inject nitrogen from one side with a rate of $Q = 1 \times 10^{-6}\,\mathrm{mbar\,l/s}$ between $t_{\mathrm{start}} = 0.005\,\mathrm{s}$ and $t_{\mathrm{stop}} = 0.020\,\mathrm{s}$ while constantly pumping at the other side with $S = 500\,\mathrm{l/s}$. The gas and the walls are kept at 20 °C, and the cylinders length/radius ratio is small enough to treat it as a single volume where conductance doesnt limit the flow of the gas.

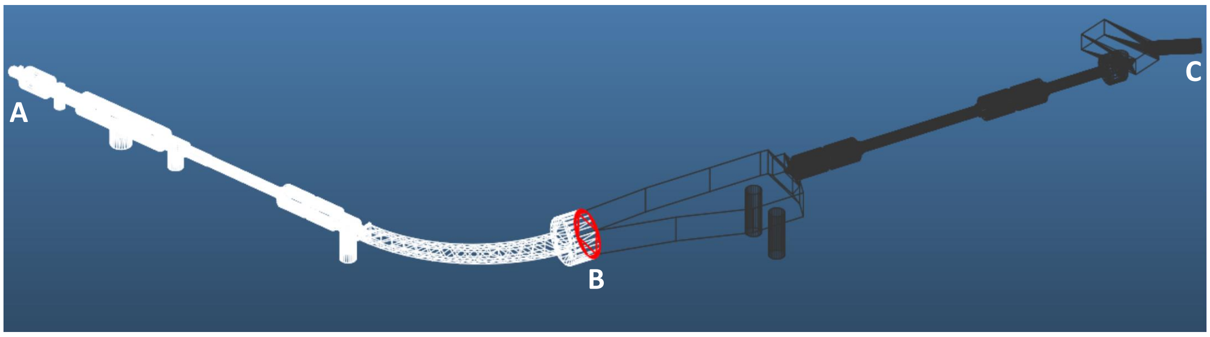


Figure 28: ISOLDE transfer line used for Time of Flight calculations. Redrawn based on Figure 6.1 of [17]

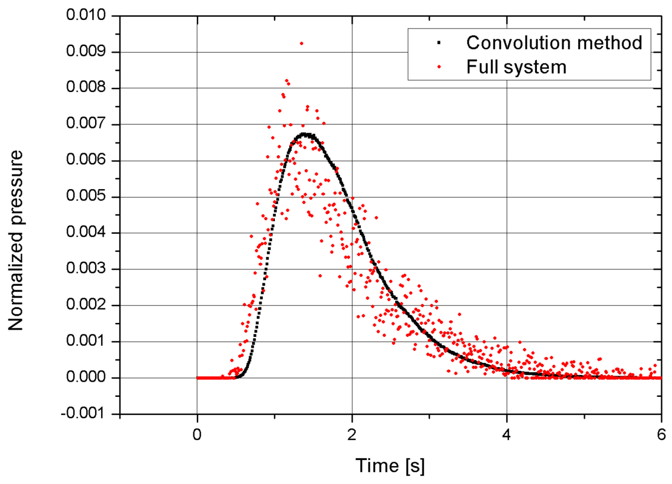


Figure 29: Results after the same computing time, with and without the convolution method. Redrawn based on Figure 6.2 of [17]

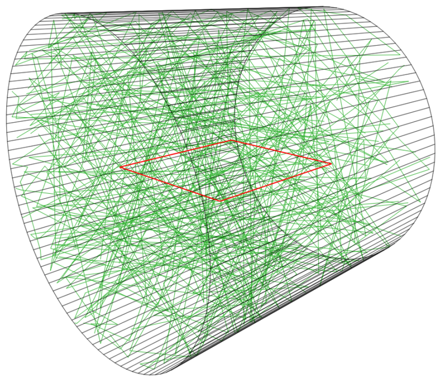


Figure 30: The 3D model representing the cylinder. The green lines show the test particle trajectories; the transparent facet in the middle (shown in red) samples the pressure.

While the pump is running, every second it extracts S/V ratio of the N molecules in the system. Including the gas injection, the governing differential equation will be:

$$\frac{dN}{dt} = \frac{dN_{\text{inject}}}{dt} - \frac{S}{V}N$$

Knowing that the pressure - in this case - is proportional to the number of molecules in the system, and using vacuum theory to express the steady-state pressure as $p_{\text{const}} = Q/S$, we can solve the equation for the injection and the pump-down durations:

$$p(t)_{\text{injection}} = \frac{Q}{S}(1 - \exp(-\frac{t - t_{\text{start}}}{\tau}))$$

$$p(t)_{\text{pumpdown}} = \frac{Q}{S}\exp(-\frac{t - t_{\text{stop}}}{\tau})$$

To simulate the above process, we create a model in Molflow+, presented in Fig.30. It consists of 100 sides and 2 cap facets of the cylinder, and an additional transparent facet in the middle to measure the average pressure in the system. The wall facets use diffuse reflection, one cap facet serves as gas source (governed by a time-dependent desorption function, which is $Q = 1 \times 10^{-6}$ mbar l/s between t_{start} and t_{stop} and 0 elsewhere), and the other as the 500 l/s pump, which converts to a sticking factor of 0.54.

Three simulations are performed with a time window of 1×10^{-3} s, 1×10^{-4} s and 1×10^{-5} s respectively. We record the system state at 40, 400 and 4000 equidistant moments between $t = 0$ and $t = 0.04$ s, therefore the distance between the recorded moments corresponds to the time window length in all three cases. We run the simulation with 1 million test particles. This results in approximately 11 million hits,

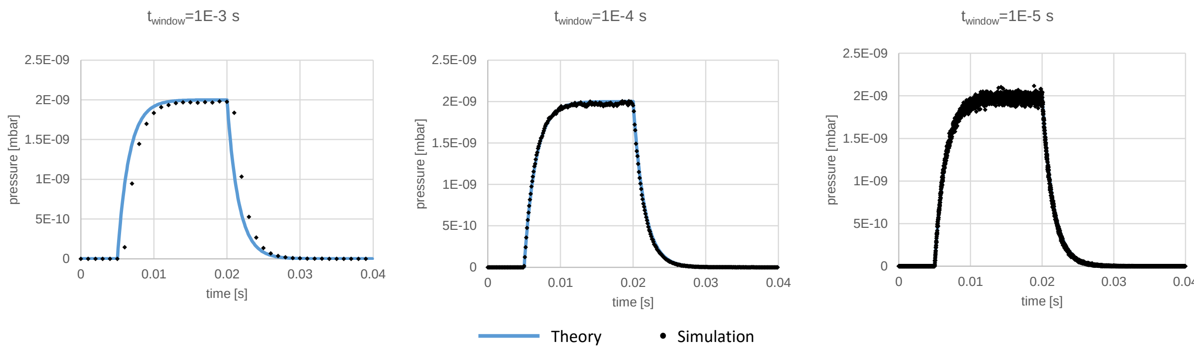


Figure 31: Pressure profiles for an injection-pumpdown process with too large, optimal and too small time windows.

Table 6: Analytic and Molflow+ results for the thermal creep problem

Temperature	**Pressure**	**Density**
$T_1 = 100\,\mathrm{K}$	$p_1 = 2.89 \times 10^{-5}\,\mathrm{mbar}$	$n_1 = 2.10 \times 10^{18}\,\mathrm{m}^{-3}$
$T_2 = 450\,\mathrm{K}$	$p_2 = 6.13 \times 10^{-5}\,\mathrm{mbar}$	$n_2 = 9.91 \times 10^{17}\,\mathrm{m}^{-3}$
Theoretical ratio:	Molflow+ ratio:	Molflow+ ratio:
$\sqrt{T_2/T_1} = 2.121$	$p_1/p_2 = 2.122$	$n_2/n_1 = 2.115$

and for reference, the computing takes around 10 seconds on a standard desktop PC. The pressure values for all moments are plotted in Fig.31.

Looking at the results we can make two observations. The first is that choosing the time window correctly is important: on the left plot of Fig.31 we can see that a large time window will smooth out thus hide the beginning of the injection and the pumpdown. The optimal time window (middle), $t_{\mathrm{window}} = 1 \times 10^{-4}\,\mathrm{s}$ shows a very good match between theory and simulation. On the other end, choosing a too small time window will introduce excessive statistical scattering: to gain the same number of hits (i.e. accuracy) for every moment with a ten times smaller time window, we would need to run the simulation ten times longer.

There is no universally correct size for the time window, but the systems dynamics can give us the order of magnitude: in this case, the characteristic pump-down time of the system is $\tau = \frac{V}{S} = 1.57 \times 10^{-3}\,\mathrm{s}$, and the optimal time window size was approximately one-tenth of it.

6.2 Non-isothermal systems

Results of the new algorithm for non-isothermal systems can be demonstrated and checked against theory through a simple vacuum part with two vessels connected by a narrow pipe. One vessel is hot ($T_1 = 450\,\mathrm{K}$), the other is cold ($T_2 = 100\,\mathrm{K}$), and the pipes temperature changes linearly in between. The cubes have a volume of 2 l, the connecting pipe has a 3x3 cm rectangular cross-section and a length of 30 cm.

Gas in this system will exhibit a phenomenon called thermal transpiration: even at equilibrium, due to higher molecular velocities, pressure will be locally higher on the hot side, while the molecule density will be higher on the colder side. The theoretical[19] pressure and density ratios are given by the formulas

$$p_1/p_2 = \sqrt{T_1/T_2}$$

$$n_1/n_2 = \sqrt{T_2/T_1}$$

The Molflow+ model shown in Fig.32 consists of 36 facets. The pipe connection is split to six 5 cm long sections to model the temperature gradient in 50 K steps. All facets have textures with 1x1 cm cell size to sample and visualize the pressure, impingement rate and density. One side facet has a $Q = 1 \times 10^{-7}\,\mathrm{mbar\,l/s}$ outgassing rate and an $s = 10^{-6}$ sticking factor, so low that the system is in quasi-equilibrium (each inserted test-particle has an average of 12 million hits all around the system before being pumped). 2 billion hits were simulated, taking about 10 minutes of computing time. The pressure and density ratios in table 6 show a good match between simulation and theory.

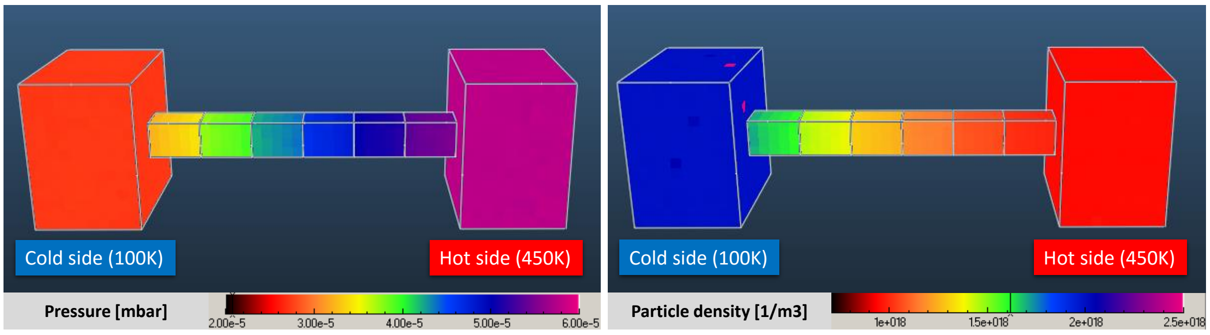


Figure 32: Pressure and density at equilibrium in a non-isothermal system

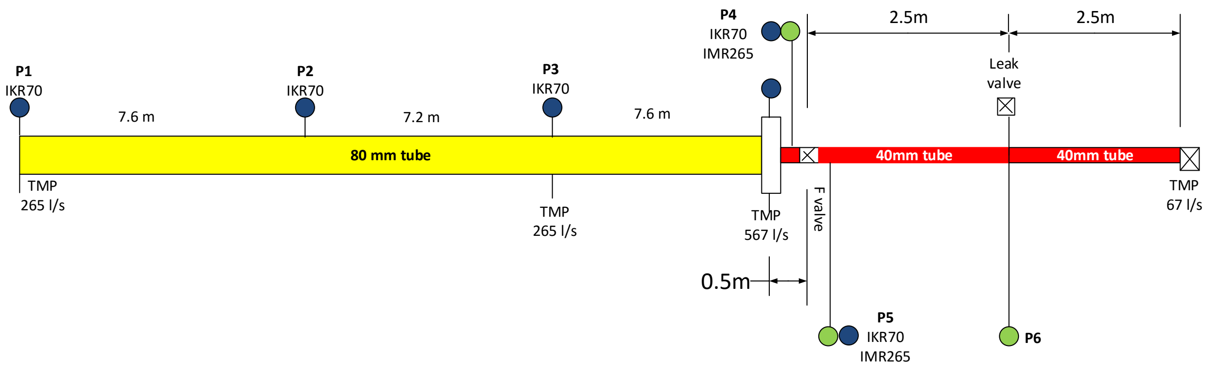


Figure 33: Overview of the experimental bench at CERN (not to scale)

7 Experimental validation (CERN)

In 2014, when the new time-dependent code was first introduced internally at CERN, one of its immediate practical applications was the simulation of the AWAKE plasma cell opening process. In the AWAKE experiment[20], a plasma cell must be periodically opened to allow a laser pulse to pass through, which in turn allows the pressurized Rubidium in the cell to propagate down the vacuum line. The pressure gradient created by this process has an undesired defocussing effect, therefore precise modeling of the system is necessary. The new code was used to simulate the process, and subsequently an experimental station was assembled at CERN to test the prototype of the system.

7.1 Experimental setup

The test station, as shown in Fig.33, consists of a 4 m long, 40 mm diameter cylinder that represents the plasma cell. It is separated from the vacuum line by a VAT gate valve. The vacuum line consists of three approx. 7 m long, 80 mm diameter stainless steel tubes. Pumping is performed at the vacuum side of the gate valve by a 500 l/s turbo-molecular pump. There were additional pumps installed on the system which helped the initial pump-down process but were separated during our tests.

The experiment is performed in a way to simulate the actual cycle the AWAKE plasma cell goes through:

- We pressurize the plasma cell prototype to 1×10^{-2} mbar, the reference pressure of the AWAKE plasma cell, through the variable leak valve at $P6$
- By pressing a button on its controller, we open the gate valve (actuated by compressed air), allowing gas to flow out of the pressurized cell
- Approximately one second later, we close the gate valve
- During a 30 second pause, the gas that entered the vacuum line is pumped down, and the cell is pressurized to 1×10^{-2} mbar again

During the cycles, pressure is measured at each interconnection of the 7 m pipes (referred to as $P1$, $P2$ and $P3$), on each side of the gate valve ($P4$ and $P5$), and also at the middle of the plasma cell prototype

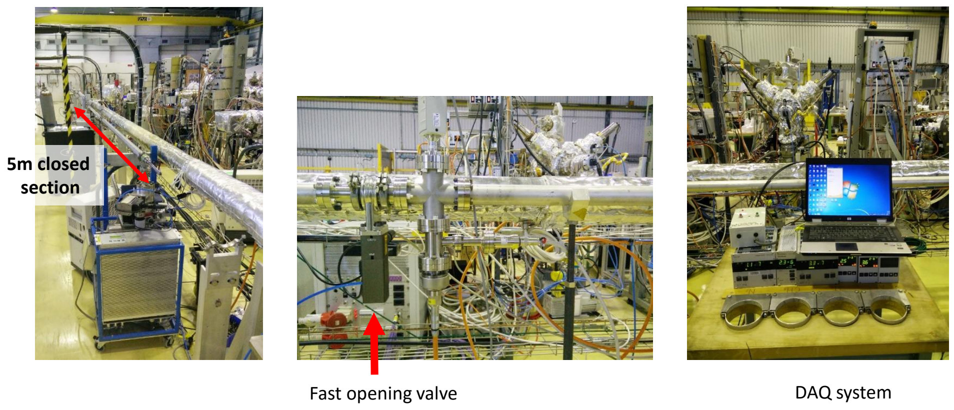


Figure 34: Parts of our experimental bench at CERN

($P6$). We have placed IKR070 (Penning) gauges of range $10^{-8}..10^{-3}$ mbar to $P1$, $P2$ and $P3$ where pressure is expected to be low at all times, IMR260 (all-range) gauges of range $10^{-6}..1$ mbar at $P5$ and $P6$ since those points are pressurized to 1×10^{-2} mbar, and both types of gauges at $P4$ which are at low pressure for most of the cycle but reach a high-pressure peak after the opening of the valve. Sampling of the gauge voltages is performed at 19.2 kHz and converted to pressures during post-processing, taking into account the different gauge sensitivities for the different types of gases used: we performed our tests with either air or Krypton (choosing the latter because its mass is close to that of Rubidium).

For each test run we repeated the cycle four times. System behavior differed significantly between the first and second cycles (at the first cycle we started from a clean state), and differed only slightly between subsequent cycles (manual valve operation couldnt guarantee the same valve opening duration). For the comparison with our simulations below, we have selected a representative cycle, using air as injected gas.

7.2 Simulation

A CAD model representing the vacuum part of the real system was created and imported to Molflow+. After merging coplanar surfaces, it was reduced to 1700 facets. The fast-opening valve, as seen in red in Fig.35 is represented by a facet with time-dependent opacity, whereas the 567 l/s turbo-molecular pump is modeled with a single facet (of sticking factor 1) at the aperture of the real pump. The experiment cycle was modeled with the following time-dependent parameters:

- Gas desorption was set at point $P6$ to be 0.321 mbar l/s between $t = 0.1..0.3$ s and 0 outside of that time range. This pressurizes the valve-delimited cell volume to 1×10^{-2} mbar.
- Valve opacity was set to 1 (closed state) until $t = 2s$. Then it was set to decrease to 0 (fully open state) within 100 ms. It is left open until $t = 3.2$s, after which its opacity changes back to 1 during 50 ms. With these values the simulated valve opening/closing times coincide with those measured electronically.
- Pressure is recorded in 600 equidistant steps between $t = 0$ and $t = 6$s, and the time window was chosen to match the time step: t_{window} =0.01 s.

30 billion test particles were desorbed to reach a good statistical accuracy. For reference, the calculation took 11 hours on a standard desktop PC. This relatively long simulation time comes from the fact that all hits up to the flight time of 6 s need to be calculated, taking 48.000 collisions per desorbed test particle on average. Some speedup was achieved by removing particles once their flight time reached 6 s, as opposed to tracing them until they are pumped. As the simulation doesnt account for the residual gases in the real system, the measured background pressure was added to all calculated pressure values.

7.3 Results

For comparison, the measured values were offset in time so that the valve opening happens at $t = 2$s for both the simulated and the measured datasets. Results (measured vs. simulated pressures) are shown in

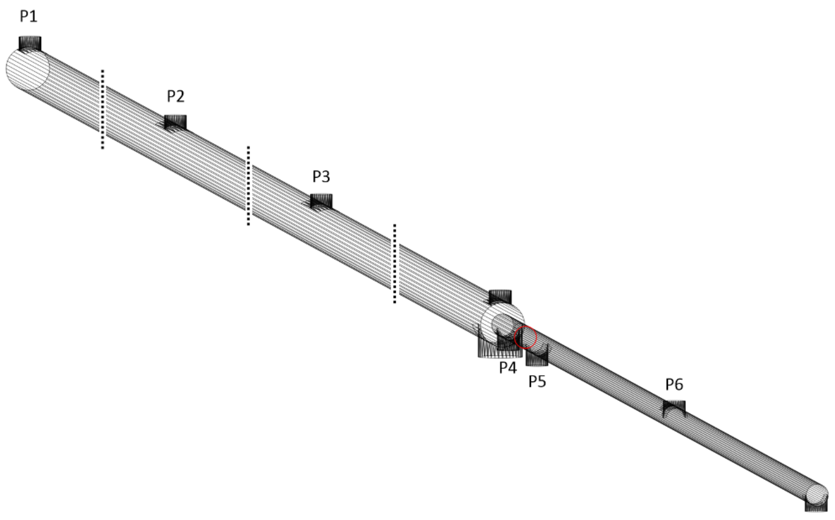


Figure 35: Molflow+ model of the system, with the facet representing the fast valve in red

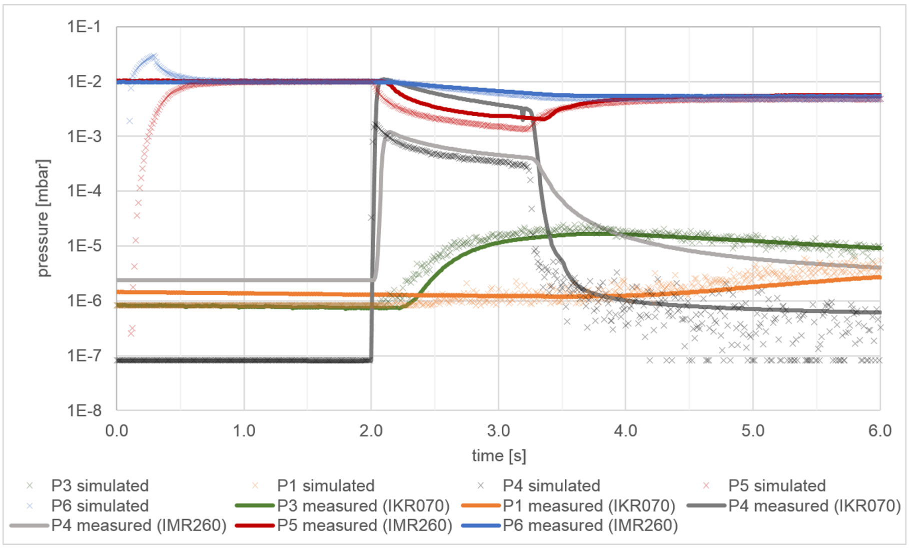


Figure 36: Pressure evolution - simulation vs. measurement

Fig.36. Error of the measured values can be estimated to be 15% (precision of an average vacuum gauge whose temperature is not controlled), and the error of the simulated values, the statistical scattering depends on the pressure (the lower the pressure, the smaller the number of hits thus the larger the scattering). The following can be observed:

- At $P5$ and $P6$ before $t = 1$s (red and blue markers), simulation over/undershoots the initial pressure of 1×10^{-2} mbar. This is because the initial state was achieved in the simulation with a short, pulsed gas injection at $P6$ (overshoot), which then spreads in the plasma cell reaching equilibrium with $P5$.

- At $P4$, the measured pressure from both the IKR070 and the IMR060 gauges is displayed (dark and light gray): in the high pressure region, the Penning gauge is over range, therefore the IMR060 signal should be trusted; as opposed to the low pressure region, where the latter is underrange and the real pressure is indicated by the IKR070 gauge. The transition can be well observed at the valve closing moment, $t \approx 3.3$s.

- Most importantly, the measured pressures have delays, originating from the gauges. We can approximate it by using the valve opening moment, $t = 2$ s as reference: whereas the simulation predicts and immediate pressure change at $P4$ and $P5$ (which is the physical solution as gas will flow from the plasma cell as soon as the valve is open), looking at the measurement curves, we can see a significant delay at both points.

- Correcting for this approx. 0.2 s of delay would result in a significantly better match between the measured and simulated values, however it was deliberately not done to benchmark the accuracy of Molflow+ simulations "out of the box".

Apart from the gauge delays and the trick necessary to measure in both low and high pressures at $P4$, the simulated values match reasonably well with the measured results. Differences can be attributed to physical phenomena not included in the simulation, like different components of air propagating with different velocities, pumping speed inexactitude, etc. It is also important to consider that Molflow+ assumes molecular flow condition, which is not strictly the case in the 1×10^{-2} mbar region.

References

[1] Marton Ady, Roberto Kersevan, and Rivkin Leonid. *Monte Carlo simulations of ultra high vacuum and synchrotron radiation for particle accelerators.* PhD thesis, Ecole Polytechnique, Lausanne, May 2016. Presented 03 May 2016.

[2] Graeme Austin Bird. Molecular gas dynamics. *NASA STI/Recon Technical Report A*, 76:40225, 1976.

[3] M. Knudsen. Die Gesetze der Molekularströmung und der inneren Reibungsstromung der Gase durch Rohren. *Annalen der Physik*, 333(1):75–130, 1909.

[4] Y. Kusumoto. Reflection rules preserving molecular flow symmetry in an arbitrarily shaped pipe. *Journal of Vacuum Science & Technology A*, 25(2):401–408, 2007.

[5] J. Greenwood. The correct and incorrect generation of a cosine distribution of scattered particles for monte-carlo modelling of vacuum systems. *Vacuum*, 67(2):217–222, 2002.

[6] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

[7] Y. Suetsugu. Application of the monte carlo method to pressure calculation. *Journal of Vacuum Science & Technology A*, 14(1):245–250, 1996.

[8] C Garion. Monte Carlo method implemented in a finite element code with application to dynamic vacuum in particle accelerators. *Vacuum*, 84(1):274–276, 2009.

[9] J.C. Maxwell. Illustrations of the dynamical theory of gases. *The Kinetic Theory Of Gases. Series: History of Modern Physical Sciences, ISBN: 978-1-86094-347-8., vol. 1, pp. 148-171*, 1:148–171, 2003.

[10] W. Hörmann and J. Leydold. Continuous random variate generation by fast numerical inversion. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 13(4):347–362, 2003.

[11] R. Kersevan. Analytical and numerical tools for vacuum systems. *CAS CERN Accelerator School, Vacuum in Accelerators: Platja D'Aro, Spain, 16-24 May 2006*, (3):285, 2007.

[12] D.J. Rader. *Measurements of thermal accommodation coefficients.*

[13] VL Kovalev, AN Yakunchikov, Deborah A Levin, Ingrid J Wysong, and Alejandro L Garcia. Accommodation coefficients for hydrogen molecules on graphite surface. In *AIP Conference Proceedings-American Institute of Physics*, volume 1333, page 481, 2011.

[14] H. Matsumoto and K. Kanamori. Monte carlo simulation of rarefied gas flow induced by wall temperature gradient. In *28th International Symposium on Rarefied Gas Dynamics 2012*, volume 1501, pages 661–666. AIP Publishing, 2012.

[15] Low-Pressure effusion of gases, Hope College, Krueger Laboratory for Biophysics.

[16] C Benvenuti. Molecular surface pumping: the getter pumps. *CERN-Reports*, pages 43–50, 1999.

[17] M. Maietta: A study of the propagation of neutral isotopes at Isotopes Separator On Line Device (ISOLDE): an analysis by Monte Carlo simulation and spectroscopic methods; Universita degli Studi di Napoli "Federico II".

[18] S.B. Damelin and W. Miller Jr. *The mathematics of signal processing.* Number 48. Cambridge University Press, 2012.

[19] F. Sharipov. Non-isothermal gas flow through rectangular microchannels. *Journal of Micromechanics and Microengineering*, 9(4):394, 1999.

[20] R Assmann, R Bingham, T Bohl, C Bracco, A Butterworth, A Caldwell, S Chattopadhyay, S Cipiccia, E Feldbaumer, RA Fonseca, et al. Proton-driven plasma wakefield acceleration: a path to the future of high-energy particle physics. *Plasma Physics and Controlled Fusion*, 56(8):084013, 2014.