PyCHAM (v1.3.4): a Python box model for simulating aerosol chambers

Simon Patrick O'Meara[1,2], Shuxuan Xu[1], David Topping[1], M. Rami Alfarra[1,2], Gerard Capes[3], Douglas Lowe[3], and Gordon McFiggans[1]

[1]Department for Earth and Environmental Sciences, University of Manchester, UK, M13 9PL
[2]National Centre for Atmospheric Science, University of Manchester, UK, M13 9PL
[3]Research Computing Services, University of Manchester, UK, M13 9PL

Correspondence: Gordon McFiggans (g.mcfiggans@manchester.ac.uk)

Abstract. In this paper the CHemistry with Aerosol Microphysics in Python (PyCHAM) box model software for aerosol chambers is described and assessed against benchmark simulations for accuracy. The model solves the coupled system of ordinary differential equations for gas-phase chemistry, gas-particle partitioning and gas-wall partitioning. Additionally, it can solve for coagulation, nucleation and particle loss to walls. PyCHAM is open source, whilst the graphical user interface, modular structure, manual and suite of tests for troubleshooting and tracking the effect of modifications to individual modules have been designed for optimal usability. In this paper, the modelled processes are individually assessed against benchmark simulations, and key parameters described. Examples of output when processes are coupled are also provided. Sensitivity of individual processes to relevant parameters is illustrated along with convergence of model output with increasing temporal and spatial resolution. The latter sensitivity analysis informs our recommendations for model setup. Where appropriate, parameterisations for specific processes have been chosen for their general applicability with their rationale detailed here. It is intended that PyCHAM aids the design and analysis of aerosol chamber experiments, with comparison of simulations against observations allowing improvement of process understanding that can be transferred to ambient atmosphere simulations.

1 Introduction

Many major advances in atmospheric modeling have arisen from chamber observations. For example, the partitioning of vapours to particles (Odum et al., 1996), the gas-phase chemistry of ozone as part of the Master Chemical Mechanism (MCM) (Jenkin et al., 1997), the gas-phase chemistries of limonene (Carslaw et al., 2012) and β-caryophyllene (Jenkin et al., 2012). Such advances can be incorporated into improved chamber models (e.g. Charan et al., 2019), aiding the design of experiments to interrogate further processes and systems (e.g. Peräkylä et al., 2020). As chamber use has multiplied, so too have chamber models, with many now published (Naumann, 2003; Pierce et al., 2008; Lowe et al., 2009; Roldin et al., 2014; Sunol et al., 2018; Topping et al., 2018; Charan et al., 2019; Roldin et al., 2019). Chamber scientists without modelling expertise or access

may be limited in the design, interpretation and advancement of both chamber experiments and their contribution to models. To address this requirement PyCHAM (CHemistry with Aerosol Microphysics in Python) has been developed in the framework of the EUROCHAMP2020 Simulation Chamber Research Infrastructure (Oliveri, 2018).

In this paper the processes represented in PyCHAM are described, along with details of software application. Where relevant, equations are presented and output from PyCHAM is compared against benchmark simulations to assess accuracy and determine whether calculations are performing as intended. It is not the intention of this paper to compare PyCHAM against observations, which is the focus of future work. In the following two sections the objectives, rationale and structure of the software are explained.

2 Purpose and scientific basis

Consistent with the criteria set by the EUROCHAMP2020 research project (Oliveri, 2018), PyCHAM is open source (available at https://github.com/simonom/PyCHAM), user-friendly and aims to be capable of representing the latest scientific understanding. It has been designed and tested on desktop computers for Windows, Linux and Mac operating systems. Python is the chosen language for two key reasons: code can be transferred between computers without the limitation of requiring a native or proprietary compiler (thereby improving ease of use and portability), and the relatively versatile parsing capability which allows the user to readily vary model inputs. The accessibility, usability and basic functionality of PyCHAM has been reviewed in O'Meara et al. (2020). The current paper presents a detailed description and introductory analysis of the PyCHAM functionality that was not the focus of O'Meara et al. (2020).

Aerosol chambers (interchangeably called smog chambers), defined as those used for interrogating gas- and particle-phase processes, provide a method for isolating specific processes of interest without the conflating effects present in the ambient atmosphere. Ultimately the goal of the chamber is to improve understanding and quantitative constraint on the evolution of the physicochemical properties of the gas- and particle-phase (Schwantes et al., 2017; Charan et al., 2019; Hidy, 2019). A description of chamber processes first requires consideration of the chamber characteristics, including: wall material (frequently fluorinated ethylene-propene film (FEP Teflon), though others are used), lighting, and dimensions. Two approaches are used to inlet components: batch mode whereby set volumes of gas or particle are injected at specific times, or in flow mode with a constant influx of gas or particle (Jaoui et al., 2014). The model variables input file for PyCHAM allows users to setup simulations for both modes along with other experiment descriptors that allow simulation of a broad range of chamber investigations: with or without seed particles (for absence of seed particles nucleation can be simulated); variable temperature, pressure and relative humidity; for illuminated experiments, either natural light intensity (for open roof chambers) or known actinic flux (for chambers with bulbs) that can be turned on and off at set times. The full introduction to model variables is given in Section 4.

Two previous models act as platforms on which PyCHAM developed: the Microphysical Aerosol Numerical model Incorporating Chemistry (MANIC) (Lowe et al., 2009) and PyBox (Topping et al., 2018), with the former guiding multi-phase processes and the latter guiding python parsing and automatic generation of chemical reaction modules. PyCHAM treats gas-phase photochemistry, gas-particle and gas-wall partitioning, coagulation, nucleation and particle deposition to walls in zero

dimensions. A key feature is its aim to be generally applicable, such that processes with outstanding uncertainties are parameterised and may be fitted to observations. Consequently the full list of PyCHAM applications is numerous and will increase as chamber experiments evolve. Key applications include designing chamber experiments, developing gas-phase chemistry mechanisms, quantifying gas-wall partitioning parameters, developing nucleation models and interrogating the effects of processes on secondary organic aerosol (SOA) evolution.

The processes included in PyCHAM are typically a subset of those represented in large-scale (regional and global) atmospheric models and it is intended that once a process has been successfully modelled by PyCHAM it can be transferred, possibly via parameterisation, to a large-scale model for evaluation and application (as illustrated by the gas-particle partitioning and gas-phase chemistry advances cited in the introduction). When interrogating the simulation of a given process it is necessary that conflating processes are modelled accurately, such that uncertainty around their effects is not compromising. Therefore, the main objective of this paper is to assess the accuracy and precision of the fundamental representation of the individual processes represented in PyCHAM through comparison with benchmark simulations.

Demonstration of PyCHAM application in which all major processes are influential is provided by simulating an experiment based on the role of nitrate radical (NO_3) oxidation of limonene in SOA evolution (Fig. 1). Such an experiment has implications for indoor air quality at night time when the photolysis of NO_3 ceases (Waring and Wells, 2015), therefore lights were turned off for this simulation. Following a similar approach to the experiment of Fry et al. (2011), the effect of NO_3 in the presence of ozone (O_3) can be replicated through introduction of O_3 and nitrogen dioxide (NO_2) to the chamber whilst removing the effect of the hydroxyl radical (OH) through addition of excess carbon monoxide (CO). At sufficient concentrations, this mixture initiates particle nucleation (Fry et al., 2011), which we simulate here through the nucleation parameterisation described below. O_3, NO_2 and limonene are injected again later but with the addition of seed aerosol for reproduction of indoor environments with substantial existing particulate matter.

In Fig. 1 the particle organic nitrate curve demonstrates that around 15 % of SOA comprises organonitrates that result from NO_3 reaction. The particle inorganic nitrate curve represents the contribution of dinitrogen pentoxide (N_2O_5) and nitric acid (HNO_3) to particles. Here we simulate the hydrolysis of N_2O_5 into the aqueous phase of particles and wall by setting its activity coefficient to zero and its accommodation coefficient according to Lowe et al. (2015). Studies have recently revealed the role of highly oxidised molecules (HOM) (Ehn et al., 2014), with the Peroxy Radical Autoxidation Mechanism (PRAM) simulating their chemistry (Roldin et al., 2019). For the results in Fig. 1, the PRAM scheme has been coupled with that of the Master Chemical Mechanism (MCM) (Jenkin et al., 1997; Saunders et al., 2003). It is intended that simulations such as Fig. 1 can help constrain uncertainties in new chemical schemes through comparison with observed particulate loading and composition analysis.

The primary difference between multiphase processes in simulation chambers and the real atmosphere is the presence of walls. Accurate representation of these processes in chamber models requires reasonable and realistic representation of the chamber walls (e.g Matsunaga and Ziemann, 2010; Zhang et al., 2015). Clearly PyCHAM must reasonably capture the partitioning of components and deposition of particles to chamber walls as incorrect reproduction makes comparison against

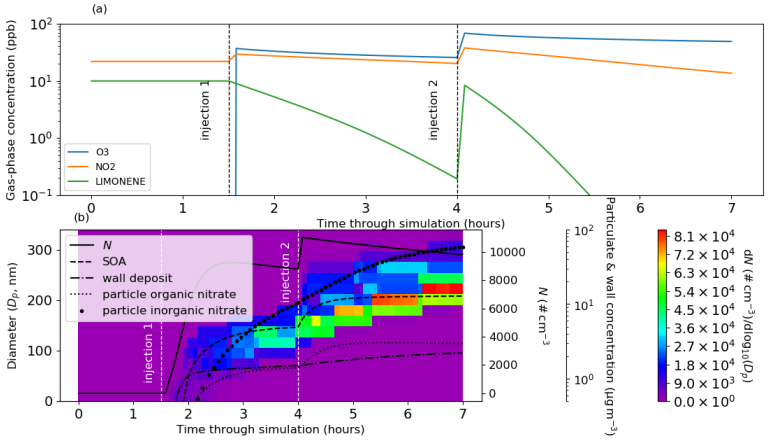

Figure 1. Limonene oxidation in the dark with and without seed particles. In (a), the gas-phase concentrations of key components and in (b), the particle properties. At the start 10 ppb limonene, 22 ppb NO_2 and 500 ppm CO are introduced. At 1.5 hours 38 ppb O_3 and 8 ppb NO_2 are injected (injection 1). At 4 hours a further injection of O_3 (45 ppb), limonene (10 ppb) and NO_2 (19 ppb) is coincident with an injection of seed aerosol (10 µg m^{-3} with a mean diameter of 0.2 µm) (injection 2). In (b), the total particle number concentration ([N]) corresponds to the the first of the right axes, mass concentrations of: secondary organic aerosol mass concentration (SOA), components deposited to walls (wall deposit), sum of particulate organic components with a nitrate functional group (particle organic nitrate) and sum of particulate inorganic components with a nitrate functional group (particle inorganic nitrate) correspond to the second of the right axes. Number size concentrations correspond to the filled contours, colour bar and left axis. Whilst [N] includes seed particles, SOA excludes the seed material and all mass concentrations exclude water.

measurements misleading or impossible. However, with correct wall loss constraint, simulations such as Fig. 1 can be compared against observations for verifying process understanding.

3 PyCHAM structure

For ease of navigation, PyCHAM has a modular structure with each key physicochemical process assigned an individual module. Unit tests are provided for key modules, which allow the user to check a particular process is functioning correctly. It is intended that these tests will be useful for troubleshooting and for analysing the effects of modifications to modules.

At the core of PyCHAM lies simultaneous numerical integration of three coupled processes: gas-phase photochemistry, vapour-particle partitioning and vapour-wall partitioning. The ordinary differential equations (ODEs) for these processes are solved by the backward differentiation formula, for which utility has been demonstrated in similar atmospheric applications (Jacobson, 2005), from the CVODE Sundials software (Hindmarsh et al., 2005). We use Assimulo (Andersson et al., 2015), a python wrapper for sundials allowing communication between the solver and Python code. The model structure is outlined in the schematic of Fig. 2, where we introduce the user-defined 'update time interval', which is the time between updates to

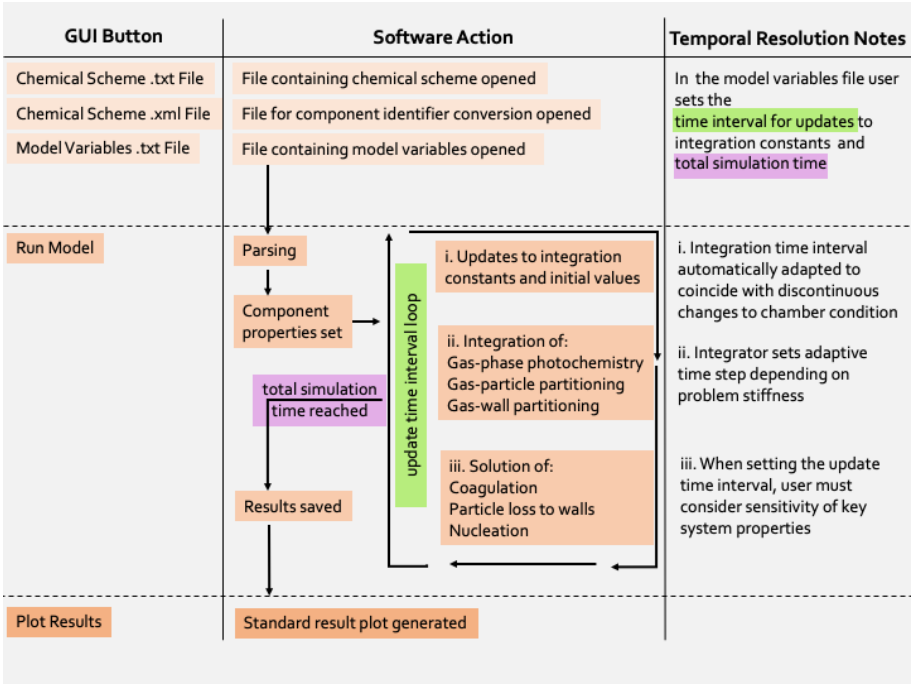

Figure 2. Schematic outlining the PyCHAM structure. In the left column are the GUI buttons that initiate the action(s) in the central column. On pressing the 'Run Model' button PyCHAM loops over the 'update time interval' until the experiment end time is reached. On each loop PyCHAM first checks whether any discontinuous changes to the chamber condition occur during the proposed time interval, and automatically reduces the interval if confirmed, such that the change can be implemented at the correct time at the start of the subsequent interval. Depending on problem stiffness the integrator sets sub-time steps and generates results at each. The final stage of a loop where the update time interval has been reached is solution of coagulation, particle deposition to walls and nucleation, which update the particle number concentration. The right column notes these temporal resolution aspects, with the mentioned sensitivity of key system properties investigated in Section 11.

natural light flux and particle number concentration due to coagulation, particle deposition to wall and nucleation (the former applicable during open roof experiments and the latter applicable for experiments with particles). Particle number concentration and photolysis rates are constants in the ODEs for gas-particle partitioning and gas-phase chemistry, respectively. The update time interval is passed to the integrator, which adaptively sets sub-time steps depending on problem stiffness. For discontinuous changes to chamber conditions, such as injection of gas-phase components or seed particles, automatic adaption ensures they occur at the start of a time interval.

Whilst coagulation and particle loss to wall have timescales of minutes to hours, nucleation can cause substantial changes within seconds (Section 11). Users may increase the update time interval, which has the advantage of decreased simulation time, and the disadvantage of divergence from high resolution estimates. Simulation sensitivity to temporal resolution is investigated in Section 11, including recommendations for maximum time intervals.

In the example model output of Fig. 1 several features of PyCHAM are demonstrated. First, the coupling of gas-phase chemistry and resulting partitioning of vapours with sufficiently low volatility to particles and walls. Nucleation has been simulated prior to the introduction of seed particles with approximate values given for the tuned nucleation parameters (Section 10) that ensure nucleation begins at the introduction of ozone, has a duration of thirty minutes and produces a peak number concentration similar to that of the seed particle. Finally, coagulation and particle wall loss (the latter using the model of McMurry and Rader (1985)), contribute to the decay in particle number concentration.

The PyCHAM software is initiated via the command line to generate a graphical user interface (GUI). Via the GUI, users select three files (Fig. 2) representing: i) the chemical scheme, ii) a file associating the chemical identifiers inside the chemical scheme to their Simplified Molecular Input Line Entry System (SMILE) strings (Weininger, 1988), and iii) a model variables file. A fourth button on the GUI starts the simulation.

A parsing module interprets the chemical scheme and uses the chemical identifier conversion file to match component identifiers to their SMILE strings. Additionally, modules are automatically created that will calculate chemical tendencies and track the process tendencies (change due to chemistry and partitioning) of specified components. A gas-phase initiation module interprets the user-defined starting concentrations of components, whilst a particle-phase initiation module establishes any seed particles at experiment start. The integration module is then called where the ODEs for gas-phase photochemistry and partitioning are solved (Fig. 2). A saving module stores results by default for gas and wall concentrations, corresponding time and constants such as component molecular weight. If the user has setup the simulation to include particles, then component particulate concentration and particle number size distributions (with and without water) are also saved, furthermore, if the user has defined components to track, then the process tendencies of these are saved.

The fifth and final button on the GUI will display and save graphs of the temporal profiles of number size distribution, secondary aerosol mass concentration, total particle number concentration, and the gas-phase concentrations of the components whose initial concentrations are user-defined. Besides these default plots, users can find additional examples of plotting scripts (those used for figures here) in the PyCHAM repository. The programme can be stopped via the terminal when in integration mode, or outside this mode it can be terminated by closing the GUI.

Below we describe and verify the processes described above as coded in PyCHAM. Necessarily each process is examined in isolation, however, Fig. 1 and its associated text illustrate the coupling of mechanisms for a real world application.

4 Model variables and component properties

As described above, to initiate PyCHAM the user selects a completed model variables input file (an example is provided with the software). The available variables are extensive to allow adaptability to a range of experiments, consequently for a given experiment, many of the variables in this file may be left empty Here we introduce the available variables, whilst details such as default values and units are provided in the appendix Table A. Several model variables (PyCHAM names of variables given in brackets) are purely functional, these include the name of the output file (res_file_name), whether to update the component property estimation files (umansysprop_update), which are described below in this section, the markers used to separate sec-

tions of the chemical scheme (chem_scheme_markers), names of files containing actinic flux (act_flux_file) and absorption cross-sections and quantum yields for photochemistry (photo_par_file). Total simulation time (total_model_time), the time interval for updating integration constants (update_step) and the time interval for recording results (recording_time_step) are also available.

Chamber temperature (temperature) can change during a simulation by stating the corresponding time (tempt), whilst pressure (p_init) and relative humidity (rh) are also input. For simulations involving natural light, latitude (lat), longitude (lon), day of year (DayOfYear) and start time (daytime_start) are required inputs. Whether artificial or natural, users specify when (light_time) light is on or off (light_status). Any dilution rate (dil_fac) should be stated, or else the default is zero.

Initial concentrations (C0) of specified trace gases (Comp0) are stated separately to the concentrations (Ct) of specified trace gases (Compt) injected effectively instantaneously at set times (injectt) during the experiment. Specified components (const_comp) will have a constant concentration for the entire experiment. The final option for introducing named components (const_infl) is to state the rate of their influx (Cinfl) during a set period of the experiment (const_infl_t). The process tendencies of certain components (tracked_comp) can be recorded, which is helpful for analysis and troubleshooting.

For specific components (vol_Comp), liquid-phase saturation vapour pressures can be manually assigned (volP). As can activity coefficients (act_user) and accommodation coefficients (accom_coeff_user).

To simulate gas-wall partitioning, the mass transfer coefficient (kgwt) and effective absorbing wall mass concentrations (eff_abs_wall_massC) can be set.

Whether the experiment is seeded or involves nucleation, users state the number of size bins (number_size_bins), size at lowermost size bin boundary (lower_part_size) and at uppermost boundary (upper_part_size), and whether to have linear or logarithmic spacing of size bins (space_mode). Setting size bin number to zero turns off particle considerations.

For seeded experiments, the component comprising the seed (seed_name), its molecular weight (seed_mw), density (seed_dens) and dissociation constant (core_diss) can be input. Either the particle concentration per size bin or the total particle concentration can be input (pconc), along with the time of particle injection (pconct). If the total particle concentration is given, this can be distributed across size bins by stating the mean radius (mean_rad) and standard deviation (std).

For nucleation experiments, the nucleating component (nuc_comp) can be changed from the default, as can the radius of newly nucleated particles (new_partr). To specify the temporal profile of nucleation, three parameters (detailed in Section 10) are input (nucv1, nucv2, nucv3).

Coagulation (coag_on) and particle loss to wall (McMurry_flag) can be turned off and on. If the latter is turned on, users can specify the size-dependent loss to walls (inflectDp, Grad_pre_inflect, Grad_post_inflect and Rate_at_inflect), or can invoke the McMurry and Rader (1985) model by also inputting the chamber wall surface area (Cham_SA), the charge per particle (part_charge_num) and the chamber electric field (elec_field), which are detailed in Section 9.

The components included in the user-defined chemical scheme are automatically allocated three properties by the PyCHAM software: molecular weight, pure component liquid density and pure component liquid saturation vapour pressure. Molecular weights are estimated by passing SMILE strings to the pybel module of the Open Babel chemical toolbox (O'Boyle et al., 2011). Open Babel is installed as part of the PyCHAM package and generates unique chemical identifiers for each component based

on their SMILE string. For estimating component densities and liquid-phase saturation vapour pressures, the pybel chemical identifiers are passed to the UManSysProp module (Topping et al., 2016) which is updated on the first run of PyCHAM and at the request of the user (via the model variables file) thereafter (requires internet connection). By default the UMansSysProp module applies the liquid density estimation method of Girolami (1994) (recommended by Barley et al. (2013)) and the liquid saturation vapour pressure estimation method of Nannoolal et al. (2008) (recommended by O'Meara et al. (2014)). Component vapour pressures have a first order effect on absorptive partitioning between phases, however estimates for certain components, particularly those with relatively low vapour pressures as these are most difficult to measure experimentally and therefore inform estimation methods, are associated with considerable uncertainty (O'Meara et al., 2014). Consequently, users can also specify the vapour pressures of certain components. Similarly, although the default activity and accommodation coefficient for all components partitioning to particles and wall is unity, users may set an alternative value for specific components. At present, activity coefficient calculations are not incorporated into PyCHAM.

5 Gas-phase chemistry

For a chamber experiment including injection of reactive components, chemical reactions in the gas-phase drive the disequilibria that can affect the composition of gas, particle and wall. As mentioned above, schemes such as the MCM provide near-explicit gas-phase chemistry mechanisms for numerous organic precursors, and developments such as PRAM (Roldin et al., 2019) can be used to provide supplementary detailed updates to our understanding of atmospheric chemistry. PyCHAM is designed to accommodate any such detailed chemical schemes whilst also accepting very simplified or even empty (e.g. for a control simulation comprising only seed particles) chemical equation files. Whilst the software manual details the requirements for input chemical schemes and chemical identifier conversion files, here we describe how PyCHAM deals with chemistry. Equations of the general form:

$$s_{r_1} r_1 + s_{r_2} r_2 \ldots \rightarrow s_{p_1} p_1 + s_{p_2} p_2 \ldots \tag{R1}$$

where s represents stoichiometric number, r reactants and p products, are expressed as the ODEs:

$$\frac{d[r_i]}{dt} = -s_{r_i} k_r \Pi_{j=1}^{j=n} \left([r_j]^{s_{r_j}} \right) \tag{1}$$

$$\frac{d[p_i]}{dt} = s_{p_i} k_r \Pi_{j=1}^{j=n} \left([r_j]^{s_{r_j}} \right) \tag{2}$$

where n is the total number of reactants and r_j is a given reactant for a given reaction. k_r is the reaction rate coefficient.

For simulations involving gas-phase chemistry, users must therefore provide a reaction(s) of the form in Eq. R1 and an associated reaction rate coefficient inside a chemical scheme file. Naming of chemical components inside the chemical scheme is unrestricted, however, the software must be able to convert names to SMILES (Weininger, 1988). Therefore, users must

provide a separate file stating a unique SMILES string for every component (Fig. 2). Examples of both the chemical scheme and SMILES string conversion file are included in the software.

Inside the parsing module, reaction rate coefficients, reactant and product identities and their stoichiometric numbers are established from the chemical scheme file. To separate these properties either default formatting may be used, or a variant, so long as the appropriate changes are made inside the model variables file. By default, MCM Kinetic PreProcessor (Sander and Sandu, 2006) formatting is used (Jenkin et al., 1997; Saunders et al., 2003) and PyCHAM has been rigorously tested using schemes and SMILE conversion files from the MCM website (Rickard and Young, 2020).

Reaction rate coefficients can be functions of temperature, relative humidity, pressure and concentrations of: third body, nitrogen, oxygen and peroxy radicals. Third body, nitrogen and oxygen concentrations are calculated by the ideal gas law with the user-set temperature and pressure. As in the MCM, the chemical scheme file can include generic reaction rate coefficients (those that have an identifier which is used as the reaction rate coefficient for one or more reactions).

Photochemistry is controlled through stating light on/off times inside the model variables file. The treatment of photochemistry is determined by the user and depends on the chemical scheme employed. In the case of the MCM scheme and natural sunlight, the scattering model based on Hayman (1997) and described in Saunders et al. (2003) is invoked by stating the relevant spatial and temporal coordinates in the model variables file. For artificial lights, users must provide a file stating the wavelength-dependent actinic flux (as described in the manual). The model then calls on either the absorption cross-section and quantum yield estimates of MCM v3.3.1 or of a user-defined file.

5.1 Assessment of gas-phase chemistry accuracy

To assess the accuracy of the photochemistry section of PyCHAM, gas-particle partitioning and gas-wall partitioning were turned off, leaving only gas-phase chemistry to be solved. Here we compare against AtChem2 (Sommariva et al., 2018) as a model benchmark, with both using MCM chemical schemes. Figure 3 shows the deviation with experiment time for two standard aerosol chamber characterisation experiments: $\alpha-\mathrm{pinene}$ ozonolysis in the presence (plot a) and absence (plot b) of $\mathrm{NO_x}$. To test both dark and illuminated scenarios, the simulation is for an environmental chamber with an open roof, starting at midnight and finishing at midday. Initial concentrations of $\alpha-\mathrm{pinene}$ and $\mathrm{O_3}$ were equal at 21.1 ppb for both experiments, whilst for $\mathrm{NO_x}$ the initial concentration was 9.8 ppb in Fig. 3a and 0 ppb in Fig. 3b. Latitude was set to 51.51, longitude to 0.13 (London, UK) and the date to 1 July. The deviation between PyCHAM and AtChem2 was calculated using:

$$\sigma_{i,t} = \left(\frac{s_{i,t} - b_{i,t}}{\vee(b_i)} \right) 100, \tag{3}$$

where $\sigma_{i,t}$ is the percentage deviation (%) for component i at time t, s is the PyCHAM result, b is the AtChem2 result. $\vee(b_i)$ is the AtChem2 maximum for a given component during the simulation which is the chosen scaling factor for deviations as it means any difference between model estimates is referenced against a reasonable value for that component (in contrast scaling by $b_{i,t}$ when $b_{i,t} \ll \vee(b_i)$ may introduce a very large percentage deviation for a relatively very small difference between model estimates).

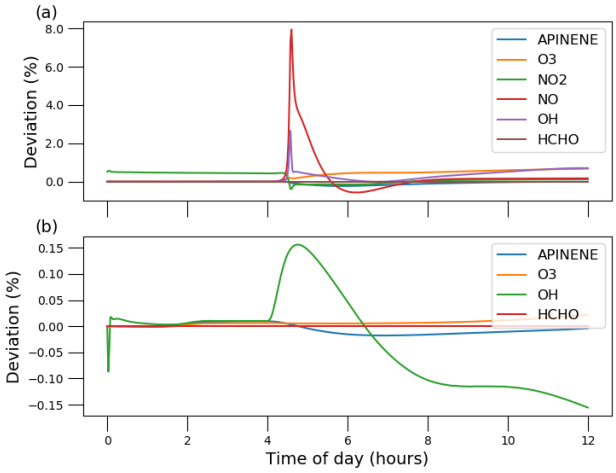

Figure 3. Gas-phase photochemistry verified; simulations of photochemistry in an aerosol chamber exposed to natural light, where deviation is defined in Eq. 3. $\alpha-$pinene ozonolysis is simulated in both plots, with $\alpha-$pinene and O_3 given the same initial concentrations of 21.1 ppb, and initial NO_x concentration in (a) 9.8 ppb and in (b) 0 ppb. For both simulations, the environmental chamber is transparent and exposed to daylight without cloud interference, with dawn at approximately 4:00 hours. The particle-phase and vapour losses to walls are turned off in PyCHAM to be consistent with the AtChem2 model.

Whilst Fig. 3 indicates that PyCHAM performs well for components with both relatively short (e.g. OH) and long (e.g. $\alpha-$pinene) lifetimes, it is necessary to ascertain that agreement is gained through the correct mechanism. The chemical tendencies of formaldehyde were tracked in both PyCHAM and AtChem2 for the $\alpha-$pinene ozonolysis simulations used for Fig. 3. Deviation of PyCHAM results from AtChem2 was calculated using Eq. 3, but with concentrations replaced by tendencies resulting from individual reaction channels. Of the loss and production channels for formaldehyde the two of each with greatest deviation are shown in Fig. 4. The low deviation values in Fig. 4 demonstrate that PyCHAM indeed solves gas-phase photochemistry correctly.

5.2 PyCHAM sensitivity to temporal resolution of continuous photolysis change

For minimising the time required for simulation, users can increase the update time interval (Fig. 2), however this decreases the frequency of update for natural light intensity (note that for artificial light simulations, PyCHAM automatically adapts the time interval to coincide with timings of lights being turned on or off). For open roof experiments, increasing the update time interval therefore reduces the accuracy of estimated photolysis rates. To illustrate and quantify the issue, the same scenario described above for Fig. 3 is used, i.e. gas-phase chemistry only simulation with increasing natural sunlight intensity. Now we compare PyCHAM low temporal resolution (update time intervals of $6x10^2$ and $6x10^3$ s) with PyCHAM high resolution (updates every $6x10^1$ s). To quantify divergence of low resolution results from high we use Eq. 3. Fig. 5 shows the loss of accuracy rising to 20 % for the lowest resolution case for both short- and long-lived components. Simulation time for the $6x10^3$ s resolution was

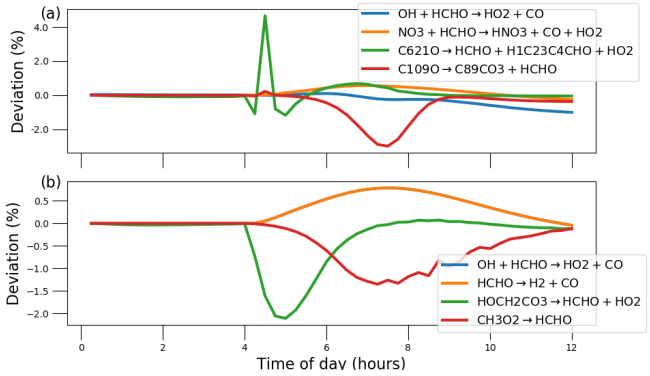

Figure 4. Deviation of PyCHAM simulated tendency of formaldehyde (HCHO) from AtChem2 simulations for the MCM reactions given in the legends (the loss and production channels of formaldehyde with greatest deviation). Where the definition for deviation is given by Eq. 3. Both plots are results for the $\alpha-$pinene ozonolysis reaction described in the main text for Fig. 3, with (a) in the presence of NO_x and (b) in the absence of NO_x.

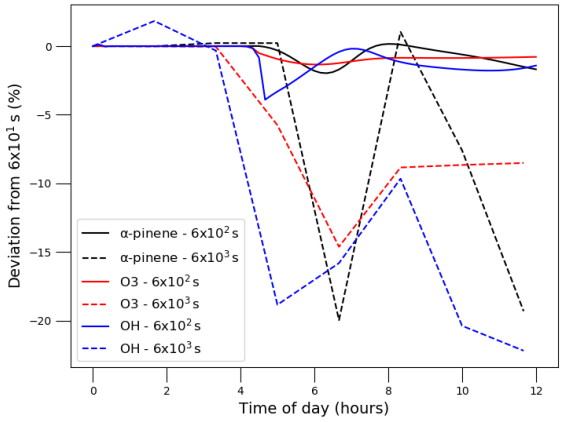

Figure 5. Illustrating the effect of update time interval resolution in PyCHAM on gas-phase concentrations of $\alpha-$pinene, O_3 and OH for the $\alpha-$pinene ozonolysis in presence of NO_x experiment described above for Fig. 3. Time intervals were set to $6x10^2$ and $6x10^3$ s as shown in the legend, and the deviation is that from results for a time interval of $6x10^1$ s.

255 52 s using a 2.5 GHz Intel Core i5 processor, with factor increases of 7 and 53 for $6x10^2$ and $6x10^1$ s resolutions, respectively. Users should conduct a similar test if their chemical scheme or environmental conditions vary significantly from those here.

6 Gas-wall partitioning

The partitioning of gases to the chamber wall is often termed wall loss as the net movement is from the gas phase to the wall (for an initially clean chamber wall). Traditionally this process has been viewed as an inconvenience since chamber results often depend on the concentration of gas- and particle-phases of certain components, whilst the fraction of these components lost to walls is poorly constrained. Several studies have focussed on partitioning to Teflon walls, which are frequently employed (Matsunaga and Ziemann, 2010; Zhang et al., 2015; Zhao et al., 2018), however, the process remains poorly modelled across the wide range of chamber materials, relative humidities, gas-phase loading, component volatilities and activity coefficients present in chamber experiments (e.g. Krechmer et al., 2017; Stefenelli et al., 2018). It is therefore preferable to allow the user to fit vapour losses to walls through the tuning of two wall loss parameters, one primarily determining equilibrium, called the effective wall mass concentration (C_w), and one determining rate of partitioning, the mass transfer coefficient (k_w). These influence gas-wall partitioning through an equation of the same framework as gas-particle partitioning (which is described in Section 7 and in Zaveri et al. (2008)):

$$\frac{dC_{i,g}}{dt} = -k_w(C_{i,g} - \frac{C_{i,w}}{C_w}p_i^0\gamma_i), \tag{4}$$

$$\frac{dC_{i,w}}{dt} = k_w(C_{i,g} - \frac{C_{i,w}}{C_w}p_i^0\gamma_i), \tag{5}$$

where p_i^0 is the liquid (sub-cooled if necessary) saturation vapour pressure of component i and γ_i is its activity coefficient on the wall. Following the conclusions of Matsunaga and Ziemann (2010) and Zhang et al. (2015), k_w represents factors such as gas- and wall-phase diffusion, turbulence, accommodation coefficient and the chamber surface area to volume ratio, whilst C_w reflects the adsorbing and/or absorbing properties of the wall, including effects of relative humidity, surface area, diffusivity and porosity. We recommend the iterative fitting of k_w and C_w to observations through minimising observation-model residuals. C_w in PyCHAM does not vary with the mass transferred to the wall, which is consistent with the findings of Matsunaga and Ziemann (2010) and Zhang et al. (2015) that indicate the effective mass concentration of the wall is much larger than the mass concentration of transferred material.

6.1 Tuning gas-wall partitioning parameters

Next we illustrate the sensitivity to k_w and C_w in Eq. 4. The same simulation setup described above for Fig. 3 was used though with $\alpha-\mathrm{pinene}$ replaced by isoprene with a concentration at experiment start of 63.4 ppb. Seed particles comprised of ammonium sulphate with mean diameter 0.5 $\mu\mathrm{m}$ and number concentration $6\mathrm{x}10^2$ cm^{-3} were introduced at experiment start. Pure component liquid saturation vapour pressures were estimated by the Nannoolal et al. (2008) method and activity coefficients for all components were assumed to be unity. To begin, both k_w and C_w were set sufficiently low to effectively eliminate gas-wall partitioning. Second, C_w was set equal to the mass concentration of seed particle (70 $\mu\mathrm{g\,m}^{-3}$) and k_w raised to $1\mathrm{x}10^{-1}\,\mathrm{s}^{-1}$ at which a notable decrease in [SOA] was observed. Third, k_w was held whilst C_w was raised three orders of magnitude greater than the seed mass concentration. Fourth, C_w was held at 70 $\mu\mathrm{g\,m}^{-3}$ and k_w was raised by three orders of

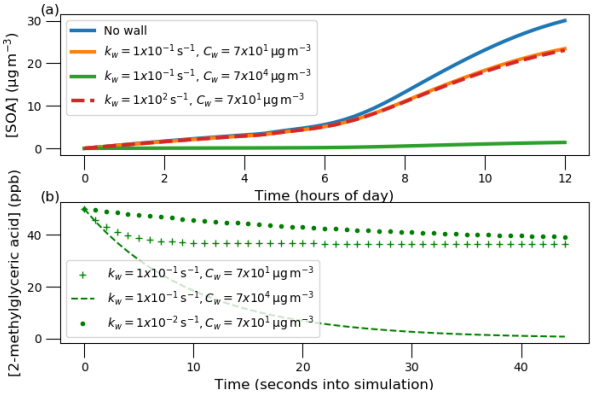

Figure 6. In (a) sensitivity of SOA mass concentration on the gas-wall partitioning parameters k_w and C_w from Eq. 4. Seed particles with a concentration of approximately 70 μg m^{-3} were present at the start of the experiment. Initial concentrations of O_3, isoprene and NO_2 were set to 21.1, 63.4 and 9.8 ppb, respectively. With regards to photolysis rates, the simulation made the same considerations as in Fig. 3, where natural sunlight drove reactions after dawn at approximately 4:00 am. In (b), the same sensitivity is assessed, but for a control experiment where only a single organic component is present.

magnitude. The effect on SOA mass concentration is given in Fig. 6 and demonstrates that at sufficiently large values of C_w, SOA production can be effectively suppressed through competitive uptake of vapours to chamber walls. However, for a given C_w, there is a limit on suppression of SOA formation due to k_w increase as it affects only the rate of partitioning with walls rather than the condensable fraction.

To guide constraint for wall loss parameters, we follow the example of Matsunaga and Ziemann (2010), with a control experiment comprising a single semi-volatile component introduced to the chamber at the start of the simulation at 50 ppb. 2-methylglyceric acid is selected as it has an estimated particle mass concentration saturation vapour pressure ($C*$) of 1.15x10^2 μg m^{-3} at 298.15 K (the simulation temperature) and is an observed oxidation product of isoprene (Surratt et al., 2006). No other components or particles are introduced. With regards to designing a control experiment for tuning C_w and k_w, the results shown in Fig. 6b demonstrate that a component with a $C*$ close to the C_w value has large sensitivity to C_w, thereby allowing greatest ease of tuning. Note, that this sensitivity can be utilised through varying chamber temperature (and therefore the $C*$ of a component), or through using a component with different volatility. Furthermore, to discern the effect of k_w a component with substantial partitioning to walls is required. When quantifying k_w it is worthwhile considering the required precision, because as Fig. 6a demonstrates, above a certain value, no further effect on SOA concentration results.

7 Gas-particle partitioning and sectional approach

PyCHAM simulations, like chamber experiments, are possible with and without seed particles. For seed particle experiments, the user defines number size distribution and composition inside the model variables input file. Furthermore, because PyCHAM

uses size bins to discretise particles, users can state the number of size bins, lower and upper bin bounds and whether to use linear or logarithmic spacing between size bins.

Particle number size concentration can change as a direct consequence of three processes modelled by PyCHAM: gas-particle partitioning, coagulation and nucleation. Whilst coagulation and nucleation are discussed below, readers are referred to Zaveri et al. (2008) for a thorough explanation of gas-particle partitioning. The transition regime correction factor required for the partitioning estimation in PyCHAM is from Fuchs and Sutugin (1971). Furthermore, the unit test test_kimt_calc is available to check that the Kelvin and Raoult effects of the PyCHAM partitioning equation are accurate (e.g. through comparison with Fig. 16.1 of Jacobson (2005)).

In this section we focus on redistribution of particles between size bins following a change in size due predominately to gas-particle partitioning. PyCHAM does this by adopting the moving-centre structure (Jacobson, 2005). The moving-centre approach has the advantage of minimal numerical diffusion and the ability to accommodate populations of particles of varying modes (e.g. a nucleation event in the presence of pre-existing particles). However, it suffers from loss of accuracy due to averaging of particles originally from different size bins that have grown, shrunk or coagulated to a given size bin (Zhang et al., 1999). In contrast the full-moving structure does not average particles of different size bin together, and can therefore exactly model certain chamber scenarios, one of which we use below to verify the moving-centre approach against. Full-moving is not used in PyCHAM however, as it lacks the general applicability of moving-centre, since it cannot account for additions to the particle population after experiment start (such as through injection of seed or nucleation).

To maintain stability in numerical solutions and improve model accuracy, a condition inside the moving-centre module of PyCHAM iteratively reduces the boundary condition time interval if particles in a size bin change volume sufficiently to be allocated to a size bin beyond the adjacent one, or if particles have an unrealistic negative volume (possibly due to evaporation).

Here we assess the moving-centre method through analysis of output during two periods of relatively substantial (and therefore testing) condensational growth and compare to benchmark simulations. The simulations also illustrate two further means of component influx to chambers using PyCHAM in addition to the simulations above where components were introduced with an initial pulse. In the first case a constant flux of sulphuric acid is added to a chamber with seed aerosol typical of hazy conditions following the benchmark simulation of Zhang et al. (1999). For consistency with the benchmark, gas and particle partitioning to walls was turned off and sulphuric acid was assumed to be non-volatile. The analysis section of Zhang et al. (1999) notes that to resolve the growth of smallest particles in this scenario, spatial resolution must be at least 100 size bins, therefore we use this value and set the update time interval to 90 s for a total 12 hour simulation.

The exact solution to this condensational growth problem is given by the full-moving structure in Fig. 7a (taken from Fig. 3 of Zhang et al. (1999)). When PyCHAM is compared against the exact solution, the tri-modal distribution is present with mean values at the correct particle size though with some disagreement in the peak height and spread. The degree of agreement is significantly better than for the 13 size bin moving-centre simulation presented in Zhang et al. (1999) and indicates that PyCHAM is operating as intended. Our results in Fig. 7a are a two-point moving average which is often necessary for the moving-centre structure because its requirement that all particles in a size bin be transferred to the adjacent bin means that some bins will intermittently have zero particles.

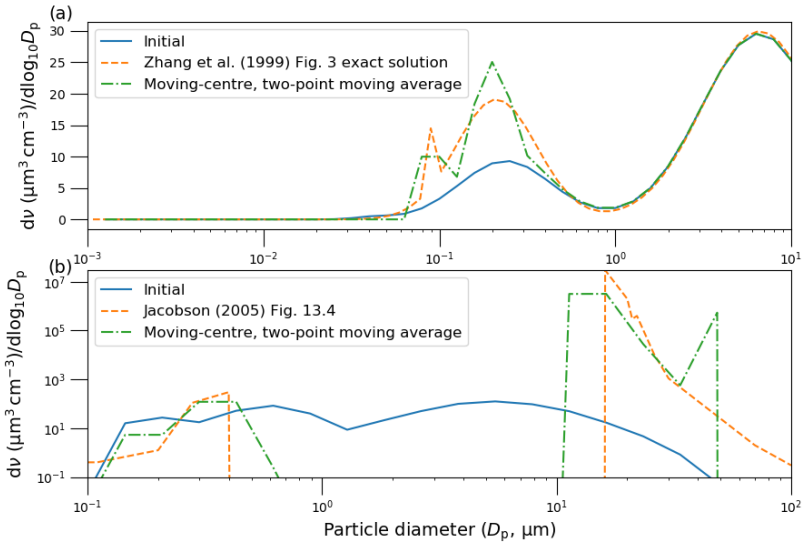

Figure 7. In (a), replication of Fig.3 of Zhang et al. (1999) where a constant influx of sulphuric acid condenses to seed particles with the shown initial volume-size distribution, with final results shown after 12 hours. In (b), replication of Fig. 13.8 of Jacobson (2005), where an initial distribution of particles are subject to a relative humidity of 100.002 % at minute intervals for 9 minutes, with results shown after 10 minutes.

Another case of relatively intense vapour-particle partitioning is provided by the example of cloud condensation nuclei experiencing varying degrees of water vapour supersaturation. Chamber experiments may involve injections of a component at specific times and the model variables input file can accommodate such a scenario. Making use of this function we reproduce the benchmark simulation of Jacobson (2005) (Fig. 13.8) where relative humidity is increased to 100.002 % every minute (including at simulation start) for nine minutes, with results analysed after ten minutes. Seed particles are assumed non-volatile and wall interactions are turned off. The parameters: temperature, seed component dissociation constant, molecular weight and density are not disclosed by the reference simulation, therefore we set these as: 318.15 K, 1.0, 200 $\mathrm{g\,mol^{-1}}$ and 1 $\mathrm{g\,cm^{-3}}$, respectively. The comparison between the Jacobson (2005) result in Fig. 7b and PyCHAM certainly shows agreement in the main feature of this simulation, which is the initially larger particles out competing smaller particles for water condensation to grow to water droplet size ($D_p > 10$ μm). It should be noted that this is a very much more stringent test of the representation of partitioning than is ever intended for PyCHAM, which will not generally be used for the huge mass flux of condensing material experienced under water supersaturated conditions. Nevertheless, the PyCHAM result gives reasonable agreement considering that key parameters (such as seed component dissociation) may vary between simulations and taken together with Fig. 7a verifies the operation of gas-particle partitioning and the moving-centre structure.

8 Coagulation

Equations of coagulation kernels for Brownian diffusion, convective Brownian diffusion enhancement, gravitational collection, turbulent inertial motion, turbulent shear and Van der Waals collision were taken from Jacobson (2005). The unit test test_coag produces a plot of coagulation kernels that can be compared to Fig. 15.7 of Jacobson (2005) to verify accuracy. Once the coagulation kernel for each pair of particle size bins (β) has been found, the combinations of size bins (denoted j and z) whose coagulation produces a particle of size bin k at time t are identified:

$$Vb_{l,k} \leq (V_{j,t-h} + V_{z,t-h}) < Vb_{u,k}, \tag{6}$$

where V is particle volume, $Vb_{l,k}$ and $Vb_{u,k}$ are the lower and upper volume bounds of the size bin and h is the time interval for coagulation to occur over. The semiimplicit coagulation equation from Jacobson (2005) is then used to estimate the new number concentration per size bin (N ($\#\,\mathrm{cm}^{-3}$)):

$$N_{k,t} = \frac{N_{k,t-h} + \frac{1}{2}h\sum_{j=1}^{j_{max}} \beta_{z,j} N_{z,t} N_{j,t-h}}{1 + h\sum_{j=1}^{\infty} \beta_{k,j} N_{j,t-h}}, \tag{7}$$

where j_{max} is the largest size bin that can undergo coagulation to produce a particle in size bin k. This equation is not mass-conserving. However, mass transfer between size bins is estimated using the number fraction of particles coagulating. For example, for a size bin k, the gain in molecular concentration of component i due to coagulation between size bins j and z (according to Eq. 6 and Eq. 7) is:

$$\Delta C_{i,k} = \frac{h(\frac{1}{2}\beta_{z,j} N_{z,t} N_{j,t-h} + \frac{1}{2}\beta_{j,z} N_{j,t} N_{z,t-h})}{N_{j,t-h}} C_{i,j,t-h}, \tag{8}$$

and the loss of molecular concentration from size bin k due to coagulation is:

$$\Delta C_{i,k} = -\left(1 - \frac{1}{1 + h\sum_{j=1}^{\infty} \beta_{k,j} N_{j,t-h}}\right) C_{i,k,t-h}, \tag{9}$$

Equations 7- 9 imply that coagulation directly influences the particle number- and mass-size distributions. We now asses the sensitivity of the number size distribution and mass conservation to temporal (represented by the time interval for updating coagulation) and spatial (represented by number of size bins) resolution. A relatively complex initial distribution with four number modes is taken from ambient observations at Claremont, California on August 27, 1987 (Jacobson, 2005) and assumed to comprise non-volatile material. Results are presented for a six hour simulation in Fig. 8 where particle wall loss was turned off to allow clearer assessment of the coagulation sensitivity. In the top row of Fig. 8 no gas-phase chemistry was allowed, whilst in the bottom row, a single chemical reaction with reaction rate $5.6\mathrm{x}10^{-17}\,\mathrm{molec}^{-1}\mathrm{s}^{-1}$ between $\alpha-\mathrm{pinene}$ and O_3 (both with initial concentrations 100 ppb) was modelled to produce a single low volatility product with saturation vapour

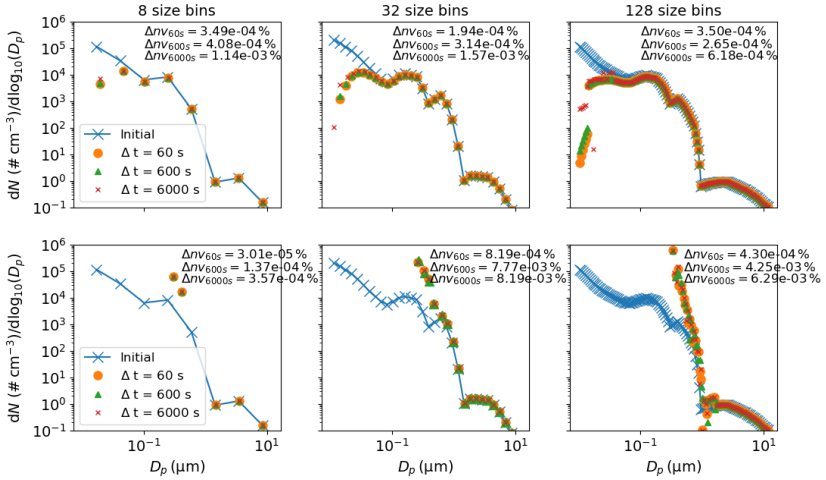

Figure 8. Sensitivity of the coagulation process to changes in temporal resolution (given in the legend) and spatial resolution (given in column titles). In the top row no chemistry occurred whilst in the bottom row a semi-volatile species was produced, as detailed in the main text. Results are for the end of a simulated six hour experiment. The $\Delta \mathrm{nv}_{\mathrm{temporal\ resolution}}$ value given in the inset text is the percentage change in total non-volatile particle-phase material from the start to finish of the experiment, demonstrating mass conservation in the model.

pressure of $1\mathrm{x}10^{-10}$ Pa, whilst gas-wall partitioning was turned off. For the chemistry case, approximately 500 $\mu\mathrm{g\,m^{-3}}$ of secondary material was formed, compared to 90 $\mu\mathrm{g\,m^{-3}}$ of seed material. Columns in Fig. 8 are distinguished by the size bin resolution as presented in the column titles, and within each plot temporal resolution is varied.

The inset text of Fig. 8 (Δnv) gives the fractional change in non-volatile material from the start to end (six hours) of the simulation for the three temporal resolutions. It is clear that the coagulation equations introduce negligible error to mass conservation. Two features are present in the top row (no chemistry) of Fig. 8: first, that in terms of number concentration, coagulation overwhelmingly affects the number concentration of smaller particles - note that such particles are sufficiently small in volume that they may coagulate with a larger particle without causing it to grow a size bin; second, that only for the smallest particles (below a diameter of $3\mathrm{x}10^{-2}$ μm in this case) is a sensitivity to temporal resolution clear across all size bin resolutions. For the no chemistry case there is demonstrable coupling of spatial and temporal resolution, with an increase in the former indicating greater sensitivity to the latter. However, the resolution considerations change substantially when we consider the case with gas-particle partitioning (bottom row of Fig. 8). In this instance, the effect of partitioning dominates the change in number-size distribution and no sensitivity of coagulation to spatial or temporal resolution is discernible. We recommend users consider these examples in addition to the nature of their simulation and objective when deciding whether temporal or spatial resolution will significantly impact results.

9 Particle deposition to walls

As with gas-wall partitioning, the loss of particles to chamber walls can significantly invalidate chamber results if unaccounted for and has been detailed in previous publications (Crump and Seinfeld, 1981; McMurry and Rader, 1985; Nah et al., 2017; Wang et al., 2018). During control experiments the deposition rate of particles to walls can be inferred through observations of the rate of decay of particles of varying size (with coagulation accounted for) (Charan et al., 2019). Several studies have published results from such experiments (McMurry and Rader, 1985; Wang et al., 2018), including a relatively large dataset from the EUROCHAMP2020 project (Oliveri, 2018). Comparison of inferred wall loss rates indicate that diffusion and settling enhance the loss rates of relatively small and relatively large particles, respectively (Crump and Seinfeld, 1981), however the absolute values and size-dependent gradient of the loss rates vary significantly between control experiments. Even for a given chamber, significant variations appear with changes to relative humidity, disturbance to walls due to air conditioning, and, for teflon chambers, with time since the chamber walls experienced frictional force to create electrostatic charge (Wang et al., 2018). Currently no method is available to measure the required inputs that a particle deposition model would need to satisfactorily reproduce observations across all chambers and conditions, therefore in PyCHAM users have three options to estimate particle wall deposition. Here we describe the options and provide examples of their use.

Users select wall loss treatment with the McMurry_flag option in the model variables input file. The default (if left empty) is no loss of particles to wall, which can be used for estimating wall loss corrected values such as aerosol yield. If set to one, the model of McMurry and Rader (1985) is used, which is based on the particle deposition model of Crump and Seinfeld (1981) but with electrostatic effects. Studies have found the Crump and Seinfeld (1981) and McMurry and Rader (1985) approach to reproduce measured particle wall losses well (Chen et al., 1992; Kim et al., 2001). Selecting McMurry and Rader (1985) requires the user to also input the chamber surface area, the average charge per particle and the average electric field inside the chamber, where the latter two may be set to zero for nullifying electrostatic effects. With the test_wallloss module users can confirm that PyCHAM accurately reproduces Fig. 2 of McMurry and Rader (1985), as shown here in Fig. 9, which demonstrates the effect of changing the charge number per particle.

If user sets the McMurry_flag option to zero then a customised particle deposition rate dependence on particle size is available. This option allows application of known or best estimate deposition rates (β) to the model, as recommended by Wang et al. (2018). Four further inputs are required for this option: the particle diameter at which the inflection in deposition rates occurs ($D_{p,flec}$) (where the inflection point marks a change in dependance of deposition rate with particle size), the rate of particle deposition to wall at the inflection point (β_{flec}), and the gradients of the deposition rate with respect to particle diameter before (∇_{pre}) and after the inflection (∇_{pro}), where a linear dependence in log-log space is assumed, consistent with observations (Charan et al., 2019). The equations for deposition rate in this instance are given in Eq. 10, and example dependencies of rate with particle size provided by Fig. 9.

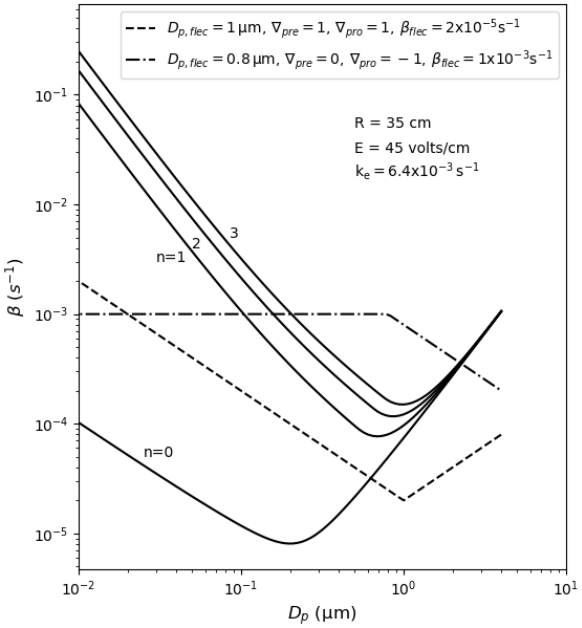

Figure 9. Example dependencies of the particle deposition to wall rate using the model of McMurry and Rader (1985) in the solid lines, where the charge per particle is given by n and other inputs given by inset text (R is spherical-equivalent chamber radius, E is the average electric field in the chamber and k_e is the coefficient of eddy diffusion). The dashed lines demonstrate the observation-based deposition rate utility of PyCHAM given in Eq. 10, with inputs at the top of the plot.

$D_p < D_{p,flec}$

$$\log_{10}(\beta(D_p)) = \log_{10}(D_{p,flec}) - \log_{10}(D_p)\nabla_{pre} + \beta_{flec}$$

$D_p \geq D_{p,flec}$

$$\log_{10}(\beta(D_p)) = \log_{10}(D_p) - \log_{10}(D_{p,flec})\nabla_{pro} + \beta_{flec} \quad (10)$$

10 Nucleation

The simulation of nucleation to produce newly-formed suspended particles is one of the most active areas of ongoing atmospheric research and many important advances in observing the nucleation process have been, and will continue to be, made through appropriate measurements in chamber experiments and their interpretation (Dada et al., 2020). PyCHAM is not intended to interpret and examine chamber experiments designed to resolve the mechanisms involved in molecular clustering,

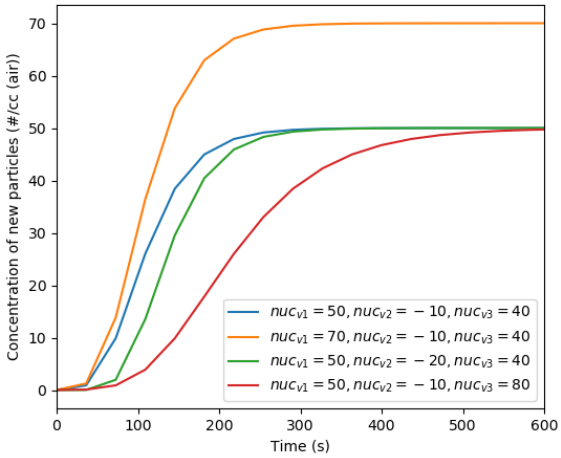

Figure 10. Effect of varying the nucleation parameters on simulated particle number concentration when considering only nucleation.

nucleation and early growth in particle formation and there are tools much better suited to these processes. However, the use of PyCHAM in simulating chamber processes in the presence of new particle formation necessitates a phenomenological accommodation of the process. Users are therefore able to provide parameters to a Gompertz function for cumulative new particle number, allowing them to fit to observed number size distributions without inferring mechanistic insight:

$$P_1(t) = \mathrm{nuc_{v1}}\left(\exp\left(\mathrm{nuc_{v2}}\left(\exp\left(-t/\mathrm{nuc_{v3}}\right)\right)\right)\right) \tag{11}$$

where P_1 is the number concentration of new particles after time t that enter the smallest size bin, and $\mathrm{nuc_{vn}}$ are the user-defined parameters. The resulting function forms an asymmetrical sigmoidal curve with time, whilst the parameters allow the amplitude ($\mathrm{nuc_{v1}}$), onset ($\mathrm{nuc_{v2}}$), and duration ($\mathrm{nuc_{v3}}$) of the curve to be adjusted, as shown in Fig. 10. The Gompertz function provides a sigmoidal form with faster increase in new particle number prior to peak rate of production than after (Fig. 10). This characteristic is consistent with observations of new particle formation (Riccobono et al., 2014; Dada et al., 2020; Wang et al., 2020).

As with gas-wall partitioning parameters, nucleation parameters should be fitted to measurements by minimising model-observation residuals. For this process the total particle number concentration may be used, however, the greater amount of data in number size distributions introduces stronger constraint, making it the preferred observation for fitting.

With the shape, size, composition and growth mechanism of the clusters that act as the nucleus of particles subject to ongoing research, in PyCHAM default properties are currently assigned, with a view to advance representation as understanding develops and an appreciation of their physical limitation. An arbitrary involatile component is assumed to form spherical nucleating clusters with a radius of 2 nm. Growth of clusters is assumed to follow absorptive partitioning (as for particulates of all sizes in PyCHAM). At the current stage of development, this representation of new particle formation in PyCHAM aims

to enable simulations of coupled photochemistry and aerosol microphysics in seeded and unseeded experiments. However, a more rigorous mechanistic representation of nucleation and early growth should be readily accommodated and will be required before PyCHAM is suitable for investigating new particle formation.

11 Sensitivity to temporal and spatial resolution

In PyCHAM, temporal resolution is represented by the time interval for updating ODE constants, as described in Section. 3 and spatial resolution is determined by the number of size bins. Whilst both resolutions can be decreased to decrease the time required for simulation, it can also introduce inaccuracies because PyCHAM processes are sensitive to changes to number size distributions (updated after each time interval) and particle size. Although it is beyond the scope of this paper to assess resolution sensitivity across all possible PyCHAM parameter space, in this section we compare the divergence of outputs from simulations with decreasing temporal and spatial resolution against a high resolution reference for extremes of the relevant parameter space: seeded experiments with no gas-particle partitioning and both seeded and nucleation experiments with relatively large condensational growth of particles. As in Section 6, two-methylglyceric acid is used in the simulations with partitioning as its vapour pressure at simulation temperature (298 K) makes it semi-volatile. Results here determine the recommended temporal and spatial resolution, provide a useful illustration of sensitivity and may help users perform sensitivity tests for their individual model inputs.

For the simulations without partitioning, the effect of resolution on particle number size distribution and total number concentration is considered, whilst for the partitioning simulations, concentration of secondary material is also relevant. To allow comparison of low resolution simulations with the high simulation reference, output from the former is interpolated to the resolution of the latter. Divergence between a low resolution simulation (g) and the high resolution reference is represented by a single absolute percentage deviation (σ_g). For number size distribution, divergence is averaged over size bins containing particles and time steps, with Y of the former and Z of the latter:

$$\sigma_g = \frac{\sum_{t_i=1}^{t_i=Z} \sum_{k=1}^{k=Y} \frac{|(n_{g,t_i,k} - \bar{n}_{t_i,k})|}{\vee(n_{g,t_i,k}, \bar{n}_{t_i,k})}}{ZY} 100, \tag{12}$$

where $n_{g,t_i,k}$ is the particle number concentration at time step t_i in size bin k from a lower resolution simulation and $\bar{n}_{t_i,k}$ is the output from the reference maximum resolution simulation. Where the two agree exactly the contribution to σ is zero, and where one output is zero and the other is greater, σ is at a maximum of 100. The term $\vee(n_{g,t_i,k}, \bar{n}_{t_i,k})$ means the greater of $n_{g,t_i,k}$ and $\bar{n}_{t_i,k}$ is used as denominator. Where the outputs from the two resolutions are similar this choice of denominator makes negligible difference, however, where one is much greater than the other it limits the divergence to a helpful (for interpretation) maximum of 100.

For total number concentration and total secondary material concentration, divergence is calculated as the percentage deviation averaged over time steps:

$$\sigma_g = \frac{\sum_{t_i=1}^{t_i=Z} \frac{|(N_{g,t_i} - \bar{N}_{t_i})|}{\vee(N_{g,t_i}, \bar{N}_{t_i})}}{Z} 100, \tag{13}$$

where N represents either total number concentration or total secondary material concentration.

For simulations assessing sensitivity to temporal resolution, 128 logarithmically spaced size bins are used, for which Fig. 8 indicates no limitation to accuracy due to spatial resolution. For seeded simulations, we use the same initial number size distribution as in Fig. 8, as this gives a relatively broad range of particle sizes, which is necessary to fully appreciate the size-dependent effects of coagulation, particle loss to wall and nucleation. All simulations were run for 24 hours and the reference simulation had an update time interval of 6 s.

Results for temporal resolution sensitivity are shown across three plots. The first, given in Fig. 11a represents the no partitioning case, with sensitivity assessed for two setups: only coagulation, and both coagulation and wall loss turned on. Coagulation proceeds as described in Section 8, whilst wall loss is described in Section 9, with the following inputs to recreate a size-dependent wall loss profile similar to n=3 in Fig. 9: $D_{p,flec}$ = 1.0 $\mu\mathrm{m}$, β_{flec} = $1.0\mathrm{x}10^{-4}$ s^{-1}, $\nabla_{pre} = \nabla_{pro}$ = 1.5. Consequently, the particle loss to wall is relatively large and the sensitivity results are conservative. Fig. 11a indicates that under this scenario, particle loss to walls considerably increases sensitivity to temporal resolution compared to the only coagulation case, with average deviations of 10 % for both total particle number concentration and number size distribution occurring at resolutions around two orders of magnitude finer than for coagulation alone.

Contours in Fig. 11 show orders of magnitude variation in simulation times with changing resolutions. All simulations in this section were using a 2.5 GHz Intel Core i5 processor.

For Fig. 11b, two-methylglyceric acid is introduced at a rate of $1.0\mathrm{x}10^{-2}\,\mathrm{ppb\,s}^{-1}$ and increases sensitivity to temporal resolution compared to Fig. 11a. A resolution of around 60 s is required to attain divergences less than 10 % for total number concentrations and secondary material, whilst for number size distribution, not even the lowest temporal resolution of 12 s can produce average divergence below 10 % when compared against the reference case of 6 s. This reflects the steep gradients in particle size-number space that are generated during intense condensational growth periods (e.g. Fig. 7), since small changes to the update time interval can vary the size bins that particles concentrate in. This effect is even further pronounced when an extremely low volatility organic component is injected at the simulation start at 1 ppb to act as a nucleating agent in an unseeded simulation, with results given in Fig. 11c. We use the nucleation parameters for Eq. 11 of: $\mathrm{nuc_{v1}}$ = $1\mathrm{x}10^{4}$, $\mathrm{nuc_{v2}}$ = $-1\mathrm{x}10^{1}$ and $\mathrm{nuc_{v3}}$ = $1\mathrm{x}10^{2}$, for a relatively rapid nucleation period (lasting only ten minutes) and therefore conservative assessment of sensitivity. Fig. 11c indicates that although particles do not grow to the same size bin(s) as the reference case, the total concentration of number and secondary material diverges from the reference case by around 10 % for an update time interval of 60 s. Given the conservative nature of these simulations we therefore recommend a maximum update time interval of 60 s. However, with the coagulation case effectively representing zero wall loss and showing considerably less divergence, if users can demonstrate relatively low particle wall loss, a coarser resolution could be applied.

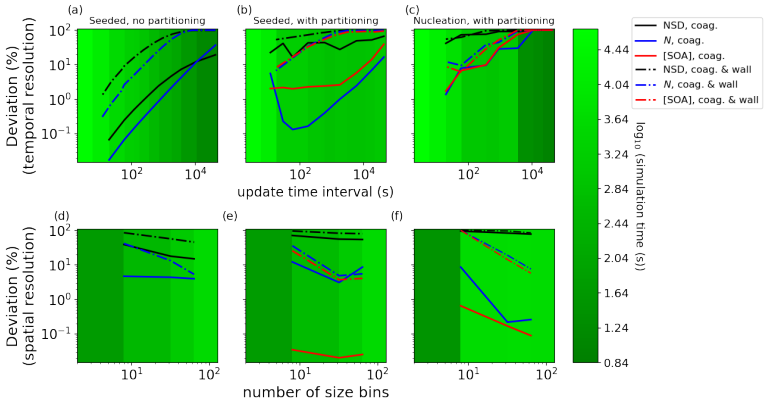

Figure 11. Sensitivity of number size distribution (NSD), total particle number concentration (N) and total concentration of secondary material ([SOA]) to temporal resolution (a-c) and to spatial resolution (d-f). The effects of coagulation alone (coag.) and combined with particle loss to wall (coag. & wall) are probed. In (a) and (d) no gas-particle partitioning is allowed, whilst in (b),(c), (e) and (f) it is, as two-methylglyceric acid is continuously injected at a rate of $1.0\text{x}10^{-2}\ \text{ppb}\,\text{s}^{-1}$. Also in (c) and (f) an extremely low volatility organic component is present at simulation start at 1 ppb, and set as the nucleating component for an unseeded simulation. The legend for lines in all plots is given in the upper right. Contours represent the time taken for simulation.

In Fig. 11d-f, the same simulation scenarios are applied as for temporal resolution sensitivity, but now we investigate spatial resolution sensitivity. Using a fixed temporal resolution of 60 s, results show the divergence of 8, 32 and 64 size bins compared against results for 128 size bins (all logarithmically spaced). In Fig. 11d, when coagulation alone is effective, reasonable agreement is seen in total number concentrations across size bin resolutions. However, when wall loss is also considered, the relatively high loss rate of small particles to the wall leads to a strong dependence of divergence on resolution. The number size distribution divergence is poor across all scenarios, indicating that if this is an important output for users (e.g. when fitting nucleation parameters, comparison against number size distribution is very useful), users should employ 128 size bins. Results for the partitioning cases in Fig. 11e and f show that without wall loss, total number concentration and secondary material concentration gives reasonable agreement of 10 % or less when 8 size bins are used. However, when partitioning is active, 32 size bins is the minimum resolution for divergence of approximately 10 % or less. Consequently, whilst recognising substantial differences between scenarios and user requirements, we recommend a size bin number of 32.

Whilst the contours in Fig. 11 provide the simulation time taken using a 1 reaction chemical scheme (ensuring that two-methylglyceric acid is recognised), Table 1 demonstrates simulation times using the $\alpha-$pinene ozonolysis scheme of the MCM, which comprises approximately $1\text{x}10^3$ reactions. Relevant combinations of spatial and temporal resolution are provided for a 6 hour unseeded experiment with $\alpha-$pinene and ozone introduced at the start to generate a nucleation episode.

Table 1. log10 of simulation times (s) for a 6 hour experiment of $\alpha-$pinene ozonolysis including nucleation. Spatial resolutions are in columns and temporal resolutions are in rows.

	2 size bin	8 size bin	32 size bin
60 s	3.4	4.0	5.2
600 s	2.6	3.3	4.7
6000 s	1.9	2.6	3.8

12 Conclusions

The PyCHAM (CHemistry with Aerosol Microphysics in Python) software for aerosol chambers has been described. Its open source repository is given in Section 2. PyCHAM has been designed for optimal ease of use (from online access to output) whilst being broadly able to address scientific problems of current relevance across a range of aerosol chamber and experimental configurations (Section 2). We have provided a model output for the dark oxidation of limonene to illustrate the coupling of modelled processes: gas-phase chemistry, gas-particle partitioning, gas-wall partitioning, redistribution of particles between size bins, particle loss to wall, coagulation and nucleation (Sections 2 and 3).

The steps to run a simulation using the software's GUI were described in Section 3 and the methods for estimating or setting component properties explained in Section 4. The setting up and solution of gas-phase photochemical reactions is detailed in Section 5, including comparison against the AtChem2 model (Sommariva et al., 2018) for verification and illustration of the effect of varying temporal resolution on model output for a system subject to varying natural light intensity.

For gas-wall partitioning this paper details (Section 6) a parameterisation that aims to satisfy the breadth of chamber characteristics and recommends a method for tuning to observations. In Section 7, gas-particle partitioning and the moving-centre structure for redistributing particles between size bins was introduced and assessed against benchmark simulations.

Coagulation was detailed in Section 8 and shown to introduce negligible loss of mass conservation. With Section 9, the three options for treating particle losses to walls were detailed and the resulting deposition rates as a function of particle diameter were exemplified, including assessment against the benchmark of McMurry and Rader (1985). Similar to gas-wall partitioning, nucleation in PyCHAM is treated with a parameterisation that aims to optimise model versatility, with examples of parameter effects provided (Section 10).

In Section 11 the sensitivity of key outputs to temporal and spatial resolution were illustrated and informed our recommendations of a minimum update time interval of 60 s and a minimum of 32 size bins. We also show a high sensitivity of model accuracy to the rate of particle wall loss and note that users could use lower resolutions if wall loss is lower than used in our tests.

Papers in preparation demonstrate further the utility of PyCHAM and its evaluation when assessed against observations. These papers include both phenomenological and mechanistic approaches to coupled photochemistry and aerosol microphysics, both of which the model readily accommodates.

Code availability. The PyCHAM software, figures in this manuscript and code to plot figures is available at: https://github.com/simonom/PyCHAM.

Appendix A: Model variable inputs

Below is the table of model variables required for input to PyCHAM accompanied by a description.

Input Name	Description
res_file_name	Name of folder to save results to
total_model_time	Total experiment time to be simulated (s)
op_spl_step	Time interval (s) for updating ordinary differential equation constants. Default is 60 s. Can be set to more than the total_model_time variable above to allow uninterrupted integration.
recording_time_step	Time interval (s) for recording results. Default is 60 s.
number_size_bins	Number of size bins (excluding wall); to turn off particle considerations set to 0 (which is also the default), likewise set pconc and seed_name variables below off. Must be integer (e.g. 1) not float (e.g. 1.0).
lower_part_size	Radius of smallest size bin boundary (um)
upper_part_size	Radius of largest size bin boundary (um)
space_mode	Set to lin for linear spacing of size bins in radius space, or to log for logarithmic spacing of size bins in radius space, if empty defaults to linear spacing
kgwt	Mass transfer coefficient of vapour-wall partitioning (/s), if left empty defaults to zero
eff_abs_wall_massC	Effective absorbing wall mass concentration (g/m3 (air)), if left empty defaults to zero
temperature	Air temperature inside the chamber (K). At least one value must be given for the experiment start (times corresponding to temperatures given in tempt variable below). If multiple values, representing temperatures at different times, then separate with a comma. For example, if the temperature at experiment start is 290.0 K and this increases to 300.0 K after 3600.0 s of the experiment, input is 290.0, 300.0.
tempt	Times since start of experiment (s) at which the temperature(s) set by the temperature variable above, are reached. Defaults to 0.0 if left empty as at least the temperature at experiment start needs to be known. If multiple values, representing temperatures at different times, then separate with a comma. For example, if the temperature at experiment start is 290.0 K and this increases to 300.0 K after 3600.0 s of the experiment, input is 0.0, 3600.0.
p_init	Pressure of air inside the chamber (Pa)
rh	Relative Humidity (fraction, 0-1)
lat	Latitude (degrees) for natural light intensity (if applicable, leave empty if not (if experiment is dark set light_status below to 0 for all times))
lon	Longitude (degrees) for natural light intensity (if applicable, leave empty if not (if experiment is dark set light_status below to 0 for all times))
DayOfYear	Day of the year for natural light intensity (if applicable, leave empty if not (if experiment is dark set light_status below to 0 for all times)), must be integer between 1 and 365
daytime_start	Time of the day (s since midnight) for natural light intensity (if applicable, leave empty if not (if experiment is dark set light_status below to 0 for all times))
act_flux_file	Name of csv file stored in PyCHAM/photofiles containing actinic flux values; use only if artificial lights inside chamber are used during experiment. The file should have a line for each wavelength, with the first number in each line representing the wavelength in nm, and the second number separated from the first by a comma stating the flux (Photons/cm2/nm/s) at that wavelength. No headers should be present in this file. Example of file given by /PyCHAM/photofiles/Example_act_flux and example of the act_flux_path variable is: act_flux_path = Example_act_flux.csv. Note, please include the .csv in the variable name if this is part of the file name. Defaults to empty.
photo_par_file	Name of txt file stored in PyCHAM/photofiles containing the wavelength-dependent absorption cross-sections and quantum yields for photochemistry. If left empty defaults to MCMv3.2, and is only used if

Input Name	Description
ChamSA	Chamber surface area (m2), used if the Rader and McMurry wall loss of particles option (Rader_flag) is set to 1 below
coag_on	Set to 1 (the default if left empty) for coagulation to be modelled, or set to zero to omit coagulation
nucv1	Nucleation parameterisation value 1
nucv2	Nucleation parameterisation value 2
nucv3	Nucleation parameterisation value 3
nuc_comp	Name of component contributing to nucleation (only one allowed), must correspond to a name in the chemical scheme file. Defaults to empty. If empty, the nucleation module (nuc.py) will not be called.
new_partr	Radius of newly nucleated particles (cm), if empty defaults to 2.0e-7 cm.
inflectDp	The particle diameter (m) at the inflection point of the size-dependent wall deposition rate.
Grad_pre_inflect	Negative log10 of the gradient of particle wall deposition rate against the log10 of particle diameter before inflection (/s). For example, for the rate to decrease by an order of magnitude every order of magnitude increase in particle diameter, set to 1.
Grad_post_inflect	Log10 of the gradient of particle wall deposition rate against the log10 of particle diameter after inflection (/s). For example, for the rate to increase by an order of magnitude for every order of magnitude increase in particle diameter, set to 1.
Rate_at_inflect	Particle deposition rate to wall at the inflection point for size-dependent particle loss to walls (/s)
part_charge_num	Average number of charges per particle, only required if the McMurry and Rader (1985) model for particle deposition to walls is selected
elec_field	Average electric field inside the chamber (g.m/A.s3), only required if the McMurry and Rader (1985) model for particle deposition to walls is selected
McMurry_flag	Set to 0 to use the particle wall loss parameter values given above or 1 to use the McMurry and Rader (1985, doi: 10.1080/02786828508959054) method for particle wall loss, which uses the chamber surface area given by ChamSA above, average number of charges per particle (part_charge_num above) and average electric field inside chamber (elec_field above), defaults to no particle wall loss if empty, similarly -1 turns off particle wall loss
C0	Initial concentrations of any trace gases input at the experiment start (ppb), must correspond to component names in Comp0 variable below. Separate concentrations of multiple components with a comma.
Comp0	Names of trace gases present at experiment start (in the order corresponding to their concentrations in C0). Note, this is case sensitive, with the case matching that in the chemical scheme file. Separate multiple component names with a comma.
Ct	Concentrations of component achieved when injected at some time after experiment start (ppb), if multiple values (representing injection at multiple times), please separate with commas. If multiple components are injected after the start time, then this input should comprise the injected concentrations of components with times separated by commas and components separated by semicolons. E.g., if k ppb of component A injected after m seconds and j ppb of component B injected after n (n>m) seconds, then Ct should be k,0;0,j. The value here is the increase in concentration from the moment before the injection to the moment after (ppb)

continued on next page

Input Name	Description
Compt	Name of component injected at some time after experiment start. Note, this is case sensitive, with the case matching that in the chemical scheme file. If more than one component, separate with a comma.
injectt	Time(s) at which injections occur (seconds), which corresponds to the concentrations in Ct, if multiple values (representing injection at multiple times), please separate with commas. If multiple components are injected after the start time, then this input should still consist of just one series of times as these will apply to all components. E.g., if k ppb of component A injected after m seconds and j ppb of component B injected after n (n>m) seconds, then this input should be m, n.
const_comp	Name of component with continuous gas-phase concentration inside chamber. Note, this is case sensitive, with the case matching that in the chemical scheme file. Defaults to nothing if left empty. To specifically account for constant influx, see const_infl variable below.
const_infl	Name of component(s) with continuous gas-phase influx to chamber. Note, this is case sensitive, with the case matching that in the chemical scheme file. Defaults to nothing if left empty. For constant gas-phase concentration see const_comp variable above. Should be one dimensional array covering all components. For example, if component A has constant influx of K ppb/s from 0 s to 10 s and component B has constant influx of J ppb/s from 5 s to 20 s, the input is: const_infl = A, B.
const_infl_t	Times during which constant influx of each component given in the const_infl variable occurs, with the rate of their influx given in the Cinfl variable. Should be one dimensional array covering all components. For example, if component A has constant influx of K ppb/s from 0 s to 10 s and component B has constant influx of J ppb/s from 5 s to 20 s, the input is: const_infl_t = 0, 5, 10, 20.
Cinfl	Rate of gas-phase influx of components with constant influx (stated in the const_infl variable above). In units of ppb/s. Defaults to zero if left empty. If multiple components affected, their influx rate should be separated by a semicolon, with a rate given for all times presented in const_infl_t (even if this is constant from the previous time step for a given component). For example, if component A has constant influx of K ppb/s from 0 s to 10 s and component B has constant influx of J ppb/s from 5 s to 20 s, the input is: Cinfl = K, K, 0, 0; 0, J, J, 0.
vol_Comp	Names of components with vapour pressures to be manually assigned in the volP variable below, names must correspond to those in the chemical scheme file and if more than one, separated by commas. Can be left empty, which is the default.
volP	Vapour pressures (Pa) of components with names given in vol_Comp variable above, where one vapour pressure must be stated for each component named in vol_Comp and multiple values should be separated by a comma. Acceptable for inputs to use e for standard notation, such as 1.0e-2 for 0.01 Pa
act_comp	Names of components (corresponding to those the chemical scheme file) with activity coefficients stated in act_user variable below (if multiple names, separate with a comma). Must have same length as act_user.
act_user	Activity coefficients of components with names given in act_comp variable above, if multiple values then separate with a comma. Must have same length as act_comp.

continued on next page

Input Name	Description
accom_coeff_comp	Names of components (corresponding to names in chemical scheme file) with accommodation coefficients set by the user in the accom_coeff_user variable below, therefore length must equal that of accom_coeff_user. Multiple names must be separated by a comma. For any components not mentioned in accom_coeff_comp, accommodation coefficient defaults to 1.0
accom_coeff_user	Accommodation coefficients (dimensionless) of the components with names given in the accom_coeff_comp variable above, therefore number of accommodation coefficients must equal number of names, with multiple coefficients separated by a comma. Can be a function of radius (m), in which case use the variable name radius, e.g: for NO2 and N2O5 with accommodation coefficients set to 1.0 and 6.09e-08/Rp, respectively, where Rp is radius of particle at a given time (m), the inputs are: accom_coeff_comp = NO2, N2O5 accom_coeff_user = 1.0, 6.09e-08/radius. For any components not mentioned in accom_coeff_comp, accommodation coefficient defaults to 1.0.
pconct	Times (seconds) at which seed particles of number concentration given in pconc variable below are introduced to the chamber. If introduced at multiple times, separate times by a semicolon. For example, for a two size bin simulation with 10 and 5 particles/cc in the first and second size bin respectively introduced at time 0 s, and later at time 120 s seed particles of concentration 6 and 0 particles/cc in the first and second size bin respectively are introduced, input is: pconct = 0; 120 (and the number_size_bins variable above = 2).
pconc	Either total particle concentration, in which case should be a scalar, or particle concentration per size bin, in which case length should equal number of particle size bins (# particles/cc (air)). If an array of numbers, then separate numbers by a comma. If a scalar, the particles will be spread across size bins based on the values in the std and mean_rad variables below. To turn off particle considerations leave empty. If seed aerosol introduced at multiple times during the simulation, separate times using a semicolon. For example, for a two size bin simulation with 10 and 5 particles/cc in the first and second size bin respectively introduced at time 0 s, and later at time 120 s seed particles of concentration 6 and 0 particles/cc in the first and second size bin respectively are introduced, the input is: pconc = 10, 5; 6, 0 (and the number_size_bins variable above = 2).

continued on next page

Input Name	Description
seed_name	Name of component comprising the seed particles, can either be core for a component not present in the chemical scheme file, a name from this file, or H2O for water, note no quotation marks needed
seed_mw	Molecular weight of seed component (g/mol), if empty defaults to that of ammonium sulphate - 132.14 g/mol
seed_dens	Density of seed material (g/cc), defaults to 1.0 g/cc if left empty
mean_rad	Mean radius of particles (um), defaults to a flag that tells software to estimate mean radius from the particle size bin radius bounds given by lower_part_size and upper_part_size variables above. If more than one size bin the default is the mid-point of each. If the lognormal size distribution is being found (using the std input below), mean_rad should be a scalar representing the mean radius of the lognormal size distribution. If seed particles are introduced at more than one time, then mean_rad for the different times should be separated by a semicolon. For example, if seed particle with a mean_rad of 1.0e-2 um introduced at start and with mean_rad of 1.0e-1 um introduced after 120 s, the input is: mean_rad = 1.0e-2; 1.0e-1 and the pconct input is pconct = 0; 120.
std	Geometric mean standard deviation of seed particle number concentration (dimensionless) when scalar provided in pconc variable above, role explained online in scipy.stats.lognorm page, under pdf method: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.lognorm.html. If left empty defaults to 1.1. If seed particles introduced after the experiment start, then separate std for different times using a semicolon. For example, if seed particle with a standard deviation of 1.2 introduced at start and with standard deviation of 1.3 introduced after 120 s, the std input is: std = 1.2; 1.3 and the pconct input is: pconct = 0; 120
core_diss	Core dissociation constant (for seed component) (dimensionless), if empty defaults to 1.0.
light_time	Times (s) for lighting condition, corresponding to the elements of the light_status variable below, if empty defaults to lights off for whole experiment. Use this variable regardless of whether light is natural or artificial (chamber lamps). For example, for a 4 hour experiment, with lights on for first half and lights off for second, use: light_time = 0.0, 7200.0. If light_time doesn't include the experiment start (0.0 s), default is lights off at experiment start.
light_status	Set to 1 for lights on and 0 for lights off, with times given in the light_time variable above, if empty defaults to lights off for whole experiment. Setting to off (0) means that even if variables above that define light intensity are submitted the simulation will be dark. Use this variable for both natural and artificial (chamber lamps) light. The lighting condition for a particular time is recognised when the simulated time meets the time given in light_time. For example, for a 4 hour experiment, with lights on for first half and lights off for second, use: light_status = 1, 0. If status not given for the experiment start (0.0 s), default is lights off at experiment start.

continued on next page

Input Name	Description
tracked_comp	Name of component(s) to track rate of concentration change (molecules/cc.s); must match name given in chemical scheme, and if multiple components given they must be separated by a comma. Can be left empty and then defaults to tracking no components.
umansysprop_update	Flag to update the UManSysProp module via internet connection: set to 1 to update and 0 to not update. If empty defaults to no update. In the case of no update, the module PyCHAM checks whether an existing UManSysProp module is available and if not tries to update via the internet. If update requested and either no internet or UManSysProp repository page is down, code stops with an error.
chem_scheme_markers	Markers denoting various sections of the user's chemical scheme. If left empty defaults to Kinetic Pre-Processor (KPP) formatting. If filled, must have following elements separated with commas: marker for punctuation at start of reaction lines (just the first element), marker for peroxy radical list starting, punctuation between peroxy radical names, prefix to peroxy radical name, string after peroxy radical name, number of lines taken by peroxy radical list (including the line containing the marker for peroxy radical list starting), punctuation at the end of lines for generic rate coefficients. For example, for the MCM FACSIMILE format: chem_scheme_markers = %, RO2, +, , , 20, ; would be used.
int_tol	Integration tolerances, with absolute tolerance first followed by relative tolerance, if left empty defaults to the maximum required during testing for stable solution: 1.0e-3 for absolute and 1.0e-4 for relative
dil_fac	Volume fraction per second chamber is diluted by, should be just a single number. Defaults to zero if left empty.

Table A1 containing the PyCHAM variable inputs and their associated descriptions.

Author contributions. Gordon McFiggans was principal investigator for the PyCHAM project. Simon O'Meara and Shuxuan Xu equally contributed to writing of the PyCHAM software. David Topping wrote the PyBOX software, Douglas Lowe and Gerard Capes wrote the MANIC software, both MANIC and PyBOX were used as starting points for PyCHAM. Rami Alfarra provided guidance on chamber experiments. Simon O'Meara wrote this manuscript, with edits provided by Gordon McFiggans, Rami Alfarra, Shuxuan Xu and David Topping

Competing interests. The authors declare that they have no conflict of interest.

Disclaimer. The PyCHAM software is provided under the GNU General Public License v3.0.

Acknowledgements. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 730997. Simon O'Meara has received funding from the National Centre for Atmospheric Science.

References

Andersson, C., Führer, C., and Åkesson, J.: Assimulo: A unified framework for {ODE} solvers, Math. Comput. Simulat., 116, 26 – 43, https://doi.org/http://dx.doi.org/10.1016/j.matcom.2015.04.007, 2015.

Barley, M., Topping, D., and McFiggans, G.: Critical Assessment of Liquid Density Estimation Methods for Multifunctional Organic Compounds and Their Use in Atmospheric Science, J. Phys. Chem. A, 117, 3428–3441, https://doi.org/10.1021/jp304547r, 2013.

Carslaw, N., Mota, T., Jenkin, M. E., Barley, M. H., and McFiggans, G.: A Significant Role for Nitrate and Peroxide Groups on Indoor Secondary Organic Aerosol, Environ. Sci. Technol., 46, 9290–9298, https://doi.org/10.1021/es301350x, 2012.

Charan, S. M., Huang, Y., and Seinfeld, J. H.: Computational Simulation of Secondary Organic Aerosol Formation in Laboratory Chambers, Chem. Rev., 119, 11 912,11 944, https://doi.org/10.1021/acs.chemrev.9b00358, 2019.

Chen, B. T., Yeh, H. C., and Cheng, Y. S.: Evaluation of an Environmental Reaction Chamber, Aerosol Sci. Tech., 17, 9–24, https://doi.org/10.1080/02786829208959556, 1992.

Crump, J. G. and Seinfeld, J. H.: Turbulent deposition and gravitational sedimentation of an aerosol in a vessel of arbitrary shape, J. Aerosol Sci., 12, 405–415, https://doi.org/10.1016/0021-8502(81)90036-7, 1981.

Dada, L., Lehtipalo, K., Kontkanen, J., Nieminen, T., Baalbaki, R., Ahonen, L., Duplissy, J., Yan, C., Chu, B., Petäjä, T., Lehtinen, K., Kerminen, V.-M., Kulmala, M., and Kangasluoma, J.: Formation and growth of sub-3-nm aerosol particles in experimental chambers, Nat. Protoc., 15, 1013–1040, https://doi.org/10.1038/s41596-019-0274-z, 2020.

Ehn, M., Thornton, J., Kleist, E., Sipilä, M., Junninen, H., Pullinen, I., Springer, M., Rubach, F., Tillmann, R., Lee, B., Lopez-Hilfiker, F., Andres, S., Acir, I., Rissanen, M., Jokinen, T., Schobesberger, S., Kangasluoma, J., Kontkanen, J., Nieminen, T., Kurtén, T., Nielsen, L. B., Jørgensen, S., Kjaergaard, H., Canagaratna, M., Maso, M., Berndt, T., Petäjä, T., Wahner, A., Kerminen, V., Kulmala, M., Worsnop, D., Wildt, J., and Mentel, T.: A large source of low-volatility secondary organic aerosol, Nature, 506, 476–479, https://doi.org/10.1038/nature13032, 2014.

Fry, J. L., Kiendler-Scharr, A., Rollins, A. W., Brauers, T., Brown, S. S., Dorn, H.-P., Dubé, W. P., Fuchs, H., Mensah, A., Rohrer, F., Tillmann, R., Wahner, A., Wooldridge, P. J., and Cohen, R. C.: SOA from limonene: role of NO3 in its generation and degradation, Atmos. Chem. Phys., 11, 3879–3894, https://doi.org/10.5194/acp-11-3879-2011, 2011.

Fuchs, N. and Sutugin, A.: Highly dispersed aerosols, Butterworth Heinemann, Woburn, Mass., 1971.

Girolami, G. S.: A Simple "Back of the Envelope" Method for Estimating the Densities and Molecular Volumes of Liquids and Solids, J. Chem. Educ., 71, 962–964, https://doi.org/10.1021/ed071p962, 1994.

Hayman, G.: Effects of pollution control on UV Exposure, in: AEA Technology Final Report (Reference AEA/RCEC/22522001/R/002 ISSUE1) prepared for the Department of Health on Contract 121/6377, AEA Technology, Oxfordshire, UK, 1997.

Hidy, G.: Atmospheric Chemistry in a Box or a Bag, Atmosphere-Basel, 10(7), https://doi.org/10.3390/atmos10070401, 2019.

Hindmarsh, A. C., Brown, P. N., Grant, K. E., Lee, S. L., Serban, R., Shumaker, D. E., and Woodward, C. S.: SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers, ACM T. Math. Software, 31, 363–396, https://doi.org/10.1145/1089014.1089020, 2005.

Jacobson, M.: Fundamentals of Atmospheric Modeling, Cambridge University Press, Cambridge, U.K., 2 edn., 2005.

Jaoui, M., Lewandowski, M., Docherty, K., Offenberg, J. H., and Kleindienst, T. E.: Atmospheric oxidation of 1,3-butadiene: characterization of gas and aerosol reaction products and implications for PM 2.5, Atmos. Chem. Phys., 14, 13 681–13 704, https://doi.org/10.5194/acp-14-13681-2014, 2014.

Jenkin, M., Saunders, S., and Pilling, M.: The tropospheric degradation of volatile organic compounds: A protocol for mechanism development, Atmos. Environ., 31, 81–104, https://doi.org/10.1016/S1352-2310(96)00105-7, 1997.

Jenkin, M. E., Wyche, K. P., Evans, C. J., Carr, T., Monks, P. S., Alfarra, M. R., Barley, M. H., McFiggans, G. B., Young, J. C., and Rickard, A. R.: Development and chamber evaluation of the MCM v3.2 degradation scheme for β-caryophyllene, Atmos. Chem. Phys., 12, 5275–5308, https://doi.org/10.5194/acp-12-5275-2012, 2012.

Kim, H. O., Han, Y. T., Kwon, S. B., and Lee, K. W.: Wall Loss Rate of Polydispersed Aerosols, Aerosol Sci. Tech., 35, 710–717, https://doi.org/10.1080/02786820152546752, 2001.

Krechmer, J. E., Day, D. A., Ziemann, P. J., and Jimenez, J. L.: Direct Measurements of Gas/Particle Partitioning and Mass Accommodation Coefficients in Environmental Chambers, Environ. Sci. Technol., 51, 11 867,11 875, https://doi.org/10.1021/acs.est.7b02144, 2017.

Lowe, D., Archer-Nicholls, S., Morgan, W., Allan, J., Utembe, S., Ouyang, B., Aruffo, E., Le Breton, M., Zaveri, R. A., Di Carlo, P., Percival, C., Coe, H., Jones, R., and McFiggans, G.: WRF-Chem model predictions of the regional impacts of N2O5 heterogeneous processes on night-time chemistry over north-western Europe, Atmos. Chem. Phys., 15, 1385–1409, https://doi.org/10.5194/acp-15-1385-2015, 2015.

Lowe, D., Topping, D., and McFiggans, G.: Modelling multi-phase halogen chemistry in the remote marine boundary layer: investigation of the influence of aerosol size resolution on predicted gas- and condensed-phase chemistry, Atmos. Chem. Phys., 9, 4559–4573, https://doi.org/10.5194/acp-9-4559-2009, 2009.

Matsunaga, A. and Ziemann, P.: Gas-Wall Partitioning of Organic Compounds in a Teflon Film Chamber and Potential Effects on Reaction Product and Aerosol Yield Measurements, Aerosol Sci. Tech., https://doi.org/10.1080/02786826.2010.501044, 2010.

McMurry, P. and Rader, D.: Aerosol Wall Losses in Electrically Charged Chambers, Aerosol Sci. Tech., 4, 249 – 268, https://doi.org/10.1080/02786828508959054, 1985.

Nah, T., McVay, R. C., Pierce, J. R., Seinfeld, J. H., and Ng, N. L.: Constraining uncertainties in particle-wall deposition correction during SOA formation in chamber experiments, Atmos. Chem. Phys., 17, 2297–2310, https://doi.org/10.5194/acp-17-2297-2017, 2017.

Nannoolal, Y., Rarey, J., and Ramjugernath, D.: Estimation of pure component properties Part 3. Estimation of the vapour pressure of non-electrolyte organic compounds via group contributions and group interactions, Fluid Phase Equilibr., 269, 117–133, https://doi.org/10.1016/j.fluid.2008.04.020, 2008.

Naumann, K.-H.: COSIMA - a computer program simulating the dynamics of fractal aerosols, J. Aerosol. Sci., 34, 1371–1397, https://doi.org/10.1016/S0021-8502(03)00367-7, 2003.

O'Boyle, N., Banck, M., James, C., Morley, C., Vandermeersch, T., and Hutchinson, G.: Open Babel: An open chemical toolbox, J Cheminformatics, 3, https://doi.org/10.1186/1758-2946-3-33, 2011.

Odum, J., Hoffmann, T., Bowman, F., Collins, D., Flagan, R., and Seinfeld, J.: Gas-particle partitioning and secondary organic aerosol yields, Environ. Sci. Technol., 30, 2580–2585, https://doi.org/10.1021/es950943+, 1996.

Oliveri, M.: EUROCHAMP 2020, https://www.eurochamp.org/Eurochamp2020.aspx, 2018.

O'Meara, S., Booth, A., Barley, M., Topping, D., and McFiggans, G.: An assessment of vapour pressure estimation methods, Phys. Chem. Chem. Phys., 16, 19 453–19 469, https://doi.org/10.1039/c4cp00857j, 2014.

O'Meara, S., Xu, S., Topping, D., Capes, G., Lowe, D., Alfarra, M., and McFiggans, G.: PyCHAM: CHemistry with Aerosol Microphysics in Python, Journal of Open Source Software, 5, 1918, https://doi.org/10.21105/joss.01918, 2020.

Peräkylä, O., Riva, M., Heikkinen, L., Quéléver, L., Roldin, P., and Ehn, M.: Experimental investigation into the volatilities of highly oxygenated organic molecules (HOMs), Atmos. Chem. Phys., 20, 649,669, https://doi.org/10.5194/acp-20-649-2020, 2020.

Pierce, J., Englehart, G., Hildebrandt, L., Weitkamp, E., Parthak, E., and Donahue, N.: Constraining Particle Evolution from Wall Losses, Coagulation and Condensation-Evaporation in Smog Chamber Experiments: Optimal Based Size Distribution Measurements, Aerosol Sci. Tech., 42, https://doi.org/10.1080/02786820802389251, 2008.

Riccobono, F., Schobesberger, S., Scott, C. E., Dommen, J., Ortega, I. K., Rondo, L., Almeida, J., Amorim, A., Bianchi, F., Breitenlechner, M., David, A., Downard, A., Dunne, E. M., Duplissy, J., Ehrhart, S., Flagan, R. C., Franchin, A., Hansel, A., Junninen, H., Kajos, M., Keskinen, H., Kupc, A., Kürten, A., Kvashin, A. N., Laaksonen, A., Lehtipalo, K., Makhmutov, V., Mathot, S., Nieminen, T., Onnela, A., Petäjä, T., Praplan, A. P., Santos, F. D., Schallhart, S., Seinfeld, J. H., Sipilä, M., Spracklen, D. V., Stozhkov, Y., Stratmann, F., Tomé, A., Tsagkogeorgas, G., Vaattovaara, P., Viisanen, Y., Vrtala, A., Wagner, P. E., Weingartner, E., Wex, H., Wimmer, D., Carslaw, K. S., Curtius, J., Donahue, N. M., Kirkby, J., Kulmala, M., Worsnop, D. R., and Baltensperger, U.: Oxidation products of biogenic emissions contribute to nucleation of atmospheric particles., Science, 344, 717–21, https://doi.org/10.1126/science.1243527, 2014.

Rickard, A. and Young, J.: The Master Chemical Mechanism, http://mcm.leeds.ac.uk/MCM/, 2020.

Roldin, P., Ericsson, A., Nordin, E., Hermansson, E., Mogensen, D., Rusanen, A., Boy, M., Swietlicki, E., Svenningsson, B., Zelenyuk, A., and Pagels, J.: Modelling non-equilibrium secondary organic aerosol formation and evaporation with the aerosol dynamics, gas- and particle-phase chemistry kinetic multilayer model ADCHAM, Atmos. Chem. Phys., 14, https://doi.org/10.5194/acp-14-7953-2014, 2014.

Roldin, P., Ehn, M., Kurtén, T., Olenius, T., Rissanen, M., Sarnela, N., Elm, J., Rantala, P., Hao, L., Hyttinen, N., , Heikkinen, L., Worsnop, D., Pichelstorfer, L., Xavier, C., Clusius, P., Öström, E., Petäjä, T., Kulmala, M., Vehkamäki, H., Virtanen, A., Riipinen, I., and Boy, M.: The role of highly oxygenated organic molecules in the Boreal aerosol-cloud-climate system, Nat. Commun., 10, https://doi.org/10.1038/s41467-019-12338-8, 2019.

Sander, R. and Sandu, A.: Technical note: Simulating chemical systems in Fortran90 and Matlab with the Kinetic PreProcessor KPP-2.1, Atmos. Chem. Phys., 6, 187–195, https://doi.org/10.5194/acp-6-187-2006, 2006.

Saunders, S. M., Jenkin, M. E., Derwent, R., and Pilling, M. J.: Protocol for the development of the Master Chemical Mechanism, MCM v3 (Part A): tropospheric degradation of non-aromatic volatile organic compounds, Atmos. Chem. Phys., 3, 161–180, https://doi.org/10.5194/acp-3-161-2003, 2003.

Schwantes, R., McVay, R., Zhang, X., Coggon, M., Lignell, H., Flagan, R., Wennberg, P., and Seinfeld, J.: Advances in Atmospheric Chemistry, Chapter 1, in: Science of the Environmental Chamber, World Scientific, https://doi.org/10.1142/10216, 2017.

Sommariva, R., Cox, S., Martin, C., Boronska, K., Young, J., Jimack, P., Pilling, M. J., Bloss, W. J., Monks, P. S., and Rickard, A. R.: AtChem, an open source box-model for the Master Chemical Mechanism, in: Atmospheric Chemical Mechanisms Conference, 2018.

Stefenelli, G., Pieber, S. M., Bruns, E. A., Temime-Roussel, B., Bertrand, A., Pieber, S. M., Bruns, E. A., Slowik, J. G., Wortham, H., Prévôt, A. S. H., El Haddad, I., and Marchand, N.: Influence of the vapor wall loss on the degradation rate constants in chamber experiments of levoglucosan and other biomass burning markers, Atmos. Chem. Phys., 18, 10 915–10 930, https://doi.org/10.5194/acp-18-10915-2018, 2018.

Sunol, A., Charan, S., and Seinfeld, J.: Computational simulation of the dynamics of secondary organic aerosol formation in an environmental chamber, Aerosol Sci. Tech., 52, 470 – 482, https://doi.org/10.1080/02786826.2018.1427209, 2018.

Surratt, J. D., Murphy, S. M., Kroll, J. H., Ng, N. L., Hildebrandt, L., Sorooshian, A., Szmigielski, R., Vermeylen, R., Maenhaut, W., Claeys, M., Flagan, R. C., and Seinfeld, J. H.: Chemical Composition of Secondary Organic Aerosol Formed from the Photooxidation of Isoprene, J. Phys. Chem-US, 110, https://doi.org/10.1021/jp061734m, 2006.

Topping, D., Barley, M., Bane, M. K., Higham, N., Aumont, B., Dingle, N., and McFiggans, G.: UManSysProp v1.0: an online and open-source facility for molecular property prediction and atmospheric aerosol calculations, Geosci. Model Dev., 9, 899–914, https://doi.org/10.5194/gmd-9-899-2016, https://www.geosci-model-dev.net/9/899/2016/, 2016.

Topping, D., Connolly, P., and Reid, J.: PyBOX: An automated box-model generator for atmospheric chemistry and aerosol simulations, The Journal of Open Source Software, 3(28), https://doi.org/10.21105/joss.00755, 2018.

Wang, M., Kong, W., Marten, R., He, X.-C., Chen, D., Pfeifer, J., Heitto, A., Kontkanen, J., Dada, L., Kürten, A., Yli-Juuti, T., Manninen, H. E., Amanatidis, S., Amorim, A., Baalbaki, R., Baccarini, A., Bell, D. M., Bertozzi, B., Bräkling, S., Brilke, S., Murillo, L. C., Chiu, R., Chu, B., De Menezes, L.-P., Duplissy, J., Finkenzeller, H., Carracedo, L. G., Granzin, M., Guida, R., Hansel, A., Hofbauer, V., Krechmer, J., Lehtipalo, K., Lamkaddam, H., Lampiäki, M., Lee, C. P., Makhmutov, V., Marie, G., Mathot, S., Mauldin, R. L., Mentler, B., Müller, T., Onnela, A., Partoll, E., Petäjä, T., Philippov, M., Pospisilova, V., Ranjithkumar, A., Rissanen, M., Rörup, B., Scholz, W., Shen, J., Simon, M., Sipilä, M., Steiner, G., Stolzenburg, D., Tham, Y. J., Tomé, A., Wagner, A. C., Wang, D. S., Wang, Y., Weber, S. K., Winkler, P. M., Wlasits, P. J., Wu, Y., Xiao, M., Ye, Q., Zauner-Wieczorek, M., Zhou, X., Volkamer, R., Riipinen, I., Dommen, J., Curtius, J., Baltensperger, U., Kulmala, M., Worsnop, D. R., Kirkby, J., Seinfeld, J. H., El-Haddad, I., Flagan, R. C., and Donahue, N. M.: Formation and growth of sub-3-nm aerosol particles in experimental chambers, Nat. Protoc., 15, 1013–1040, https://doi.org/10.1038/s41586-020-2270-4, 2020.

Wang, N., Jorga, S., Pierce, J., Donahue, N., and Pandis, S.: Particle wall-loss correction methods in smog chamber experiments, Atmos. Meas. Tech., 11, 6577 – 6588, https://doi.org/10.5194/amt-11-6577-2018, 2018.

Waring, M. S. and Wells, J. R.: Volatile organic compound conversion by ozone, hydroxyl radicals, and nitrate radicals in residential indoor air: Magnitudes and impacts of oxidant sources, Atmos. Environ., 106, 382–391, https://doi.org/10.1016/j.atmosenv.2014.06.062, 2015.

Weininger, D.: Smiles, a chemical language and information system. 1. Introduction to methodology and encoding rules, J. Chem. Inf. Comp. Sci., 28, 31–36, https://doi.org/10.1021/ci00057a005, 1988.

Zaveri, R., Easter, R., Fast, J., and Peters, L.: Model for Simulating Aerosol Interactions and Chemistry (MOSAIC), J. Geophys. Res., 113, D13 204, https://doi.org/10.1029/2007JD008782, 2008.

Zhang, X., Schwantes, R., McVay, R., Lignell, H., Coggon, M., Flagan, R., and Seinfeld, J.: Vapor wall deposition in Teflon chambers, Atmos. Chem. Phys., 15, 4197–4214, https://doi.org/10.5194/acp-15-4197-2015, 2015.

Zhang, Y., Seigneur, C., Seinfeld, J. H., Jacobson, M. Z., and Binkowski, F. S.: Simulation of Aerosol Dynamics: A Comparative Review of Algorithms Used in Air Quality Models, Aerosol Sci. Tech., 31, 487–514, https://doi.org/10.1080/027868299304039, 1999.

Zhao, R., Charan, S. M., Kenseth, C. M., Zhang, X., Huang, Y., Charan, S. M., Kenseth, C. M., and Seinfeld, J. H.: Unified Theory of Vapor-Wall Mass Transport in Teflon-Walled Environmental Chambers, Environ. Sci. Techol., 52, 2134,2142, https://doi.org/10.1021/acs.est.7b05575, 2018.