https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


# MultilayerPy (v1.0): A Python-based framework for building, running and optimising kinetic multi-layer models of aerosols and films

Adam Milsom[1], Amy Lees[1], Adam M. Squires[2] and Christian Pfrang[1,3].

[1]School of Geography, Earth and Environmental Sciences, University of Birmingham, Edgbaston, Birmingham, UK
[2]Department of Chemistry, University of Bath, South Building, Soldier Down Ln, Claverton Down, Bath, UK
[3]Department of Meteorology, University of Reading, Whiteknights, Earley Gate, Reading, UK

*Correspondence to*: Christian Pfrang (c.pfrang@bham.ac.uk)

**Abstract.** Kinetic multi-layer models of aerosols and films have become the state-of-the-art method of describing complex aerosol processes at particle and film level. We present MultilayerPy: an open-source framework for building, running and optimising kinetic multi-layer models – namely the kinetic multi-layer model of aerosol surface and bulk chemistry (KM-SUB), and the kinetic multi-layer model of gas-particle interactions in aerosols and clouds (KM-GAP). The modular nature of this package allows the user to iterate through various reaction schemes, diffusion regimes and experimental conditions in a systematic way. In this way, models can be customised and the raw model code itself, produced in a readable way by MultilayerPy, is fully customisable. Optimisation to experimental data using local or global optimisation algorithms is included in the package along with the option to carry out statistical sampling and Bayesian inference of model parameters with a Markov Chain Monte Carlo (MCMC) sampler (via the *emcee* Python package). MultilayerPy abstracts the model building process into separate building blocks, increasing the reproducibility of results and minimising human error. This paper describes the general functionality of MultilayerPy and demonstrates this with use cases based on the oleic acid-ozone heterogeneous reaction system. The tutorials in the source code (written as *Jupyter* notebooks) and the documentation aim to encourage users to take advantage of this tool, which is intended to be developed in conjunction with the user base.

## 1 Introduction

Aerosols are an important atmospheric component and contribute to air quality (indoors and outdoors), public health and the climate (Abbatt and Wang, 2020; Pöschl, 2005). The composition and physical state of aerosols can affect their ability to take up water to form cloud droplets (Schill et al., 2015; Shiraiwa et al., 2011). Understanding how an aerosol particle or film interacts with common atmospheric trace gasses, some of which are reactive, affords a better description of atmospheric aerosol processes.

Kinetic multi-layer models of heterogeneous interactions of aerosols with trace gases have become popular in the last decade. In particular, those based on the Pöschl-Rudich-Ammann (PRA) framework (Pöschl et al., 2007) such as the kinetic

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


multi-layer model of surface and bulk chemistry (KM-SUB) (Shiraiwa et al., 2010) and the kinetic multi-layer model of gas-particle interactions (KM-GAP) (Shiraiwa et al., 2012) have been applied in a wide range of studies, providing particle-level insights which were often not possible to obtain experimentally.

Software packages facilitating the creation and running of box models (e.g. PyBox (Topping et al., 2018), JlBox (Huang and Topping, 2021) and AtChem, Sommariva et al., 2020), aerosol chamber experiments (e.g PyCHAM, O'Meara et al., 2021) and indoor chemistry (INCHEMPy, Shaw and Carslaw, 2021) have recently gained popularity and nurture a more accessible modelling environment, enabling more reproducible and reliable results.

This paper describes MultilayerPy, a package written in Python which is designed to facilitate the creation and optimisation of kinetic multi-layer models (namely KM-SUB and KM-GAP) in a modular and reproducible way. The key features are presented along with use cases focussing on the well-studied oleic acid-ozone heterogeneous reaction system (Berkemeier et al., 2021; Gallimore et al., 2017; King et al., 2004, 2010, 2020; Milsom et al., 2022a, 2021b, 2021a; Pfrang et al., 2011, 2017; Woden et al., 2021; Zahardis and Petrucci, 2007). An educational tool has recently been created which creates and runs simple kinetic multi-layer models with two reactants (Hua et al., 2022). *MultilayerPy* is intended for research use and can be used to create and optimise more complex kinetic multi-layer models.

It is envisaged that this paper, along with the accompanying supporting information in the form of *Jupyter* notebooks in the source code, will encourage new users to use and eventually contribute to this project.

## 2 Purpose and scientific basis

Currently, the creation of a kinetic multi-layer model requires the researcher to manually construct specific computer code which describes the set of ordinary differential equations (ODEs) which describe the model. For a basic system (*e.g.* A reacts with B to make C) this method is satisfactory. However, for more complex systems involving many model components, composition-dependent diffusivity and the inverse modelling of experimental data, this manual method quickly becomes cumbersome and prone to human error.

*MultilayerPy* provides a framework for constructing kinetic multi-layer models so that the model code is produced automatically for the user, removing potential human error in typing out the model code. This code is written in a readable way, enabling the code to be shared with an associated publication and encouraging more reproducible results. This is further supported through the *Jupyter* notebook (Perkel, 2018), which is a document that incorporates both Python code and markdown text and is gaining popularity as a way of sharing and describing scientific code.

Abstracting the model building, running and optimisation process in this way quickens this time-consuming part of a modelling study and allows the researcher to focus more on the science behind any modelling decisions made. *MultilayerPy* provides the utility for model parameter evolution via the inclusion of additional user-defined parameters not present in the original model and including time-dependent changes in model parameters (*e.g.* changes in temperature or the gas phase concentration of a component). There is also scope for model customisation. The model code can be edited and re-incorporated

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


into the *MultilayerPy* framework. This allows researchers to incorporate conditions or processes which are unique to their specific system.

A challenge at the outset of a modelling study is deciding on a reaction scheme to use. This is illustrated by the modelling of oleic acid ozonolysis where different reaction schemes have been used in the literature (Berkemeier et al., 2021; Hosny et al., 2016; Shiraiwa et al., 2010). Manually iterating through different reaction schemes can be time consuming. The object-oriented approach of *MultilayerPy*, where model components are treated as discrete objects, allows the researcher to create and test various model reaction schemes in a few lines of code. Selection of the most suitable model reaction scheme is then feasible.

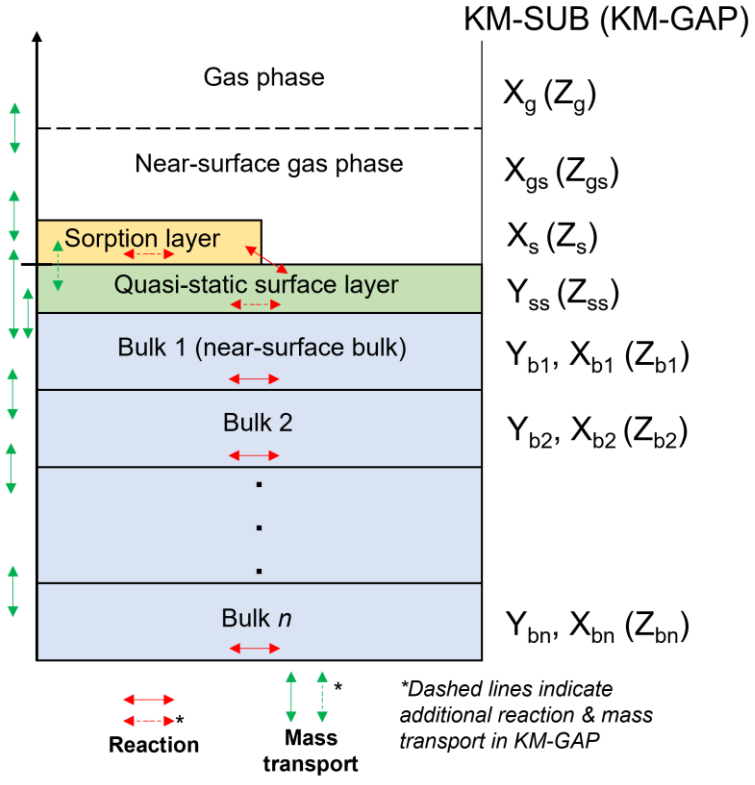


**Figure 1. KM-SUB and KM-GAP model visualisation. A particle or film is split into sorption, quasi-static surface and *n* bulk layers. KM-SUB explicitly splits components into volatile (*X*) and non-volatile (*Y*), whereas KM-GAP treats all model components as the same and they are symbolised by *Z* by convention.**

A detailed description of the KM-SUB and KM-GAP models is presented in their respective publications (Shiraiwa et al., 2010, 2012). Figure 1 illustrates the main concept behind the two models. Essentially, the models split the particle or film into a number of shells or layers. The diffusion of reactants between each layer and the reaction of each component within each layer are resolved. Surface chemistry and the adsorption and desorption of gaseous species are resolved. Additionally, KM-GAP allows the thickness of model layers to change over time, accounting for the evaporation of volatile components in

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


the model and to follow changes in film thickness or particle size during the model run. This means that all model components can partition into and out of the film or particle.

## 3 *MultilayerPy* structure and features

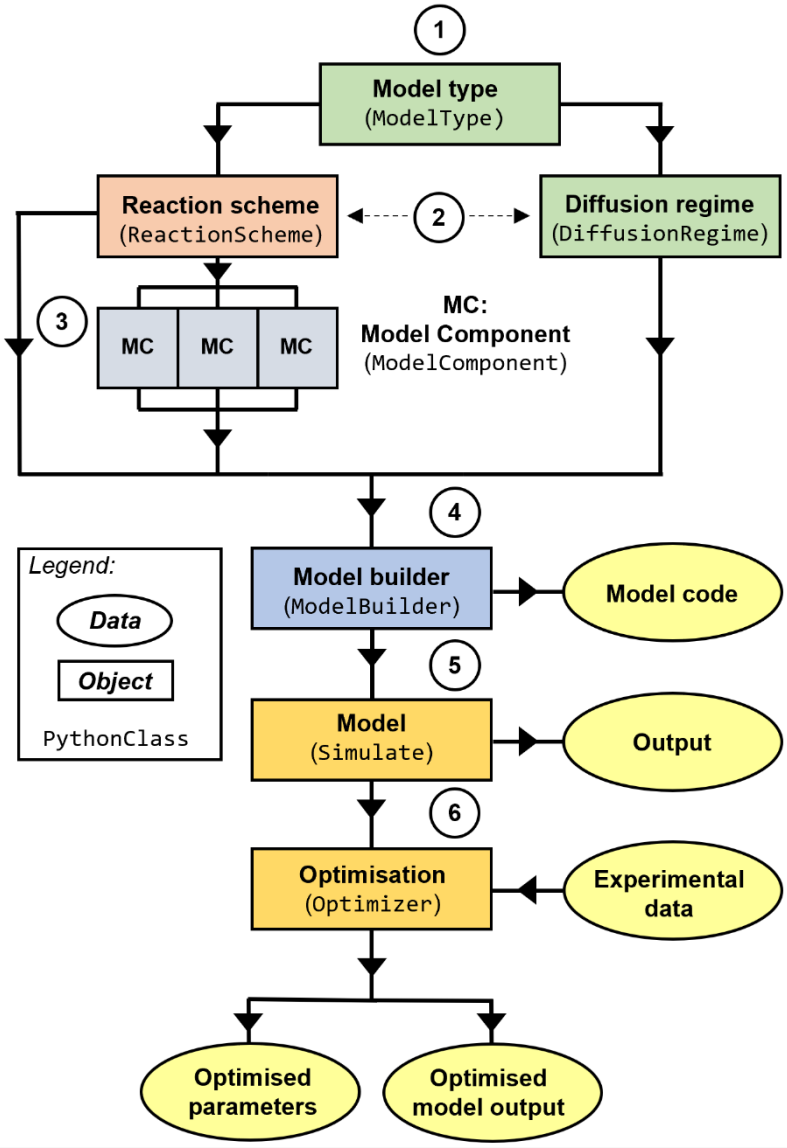


**Figure 2. High-level schematic outline of *MultilayerPy*. Circled numbers correspond to the numbered steps of the model building and optimisation process described in the main text.**

*MultilayerPy* is organised so that different combinations of reaction schemes, diffusion regimes and model types are possible and can be realised in a clear, reproducible way (Fig. 2). This is possible because of the object-oriented programming (OOP) paradigm in which this software is written. Essentially, each component of a multi-layer model is represented by an object (or

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


*class*) which has its own attributes and methods. The user sets these attributes and uses the methods associated with these objects to carry out model construction, simulation and optimisation. In relatively few lines of code, the user is able to construct this pipeline and run it.

Below is an outline of a typical model build, run and optimisation experiment. These steps are expanded on in the subsequent sections:

1. Selection of the model type which informs how the model is constructed (KM-SUB or KM-GAP).
2. Creation of a reaction scheme and diffusion regime represented in the model.
3. Creation of the model components (chemical species).
4. Model construction by combining the reaction scheme, diffusion regime and model components. Creation of the model code.
5. Model simulator object creation to run and save model outputs.
6. Optimisation of the model fit to experimental data. Determination of optimised model parameters. Optional Markov Chain Monte Carlo (MCMC) sampling of the parameter space.

This modular approach to model creation and optimisation is flexible and enables the user to experiment with and iterate through different model designs. One could imagine creating a set of reaction schemes and iterating through them in order to find the one which best describes the data and satisfies the goals of a particular study.

Only a basic knowledge of Python is required to use *MultilayerPy* and its main features. This is another advantage of abstracting the model building process into the basic building blocks described in Fig. 2.

To encourage new users, the source code for *MultilayerPy* comes with *Jupyter Notebooks* which provide example uses that can also be used as templates. A good starting point would be to work through the crash course notebook available in the repository. More detailed documentation is included in *.html* format in the source code along with instructions on how to install and test the package before use in a separate readme file. The easiest way to install the software is to either download the source code from the repository (see *code availability*) or run a *pip* install as described in the readme. The documentation will be updated with each update to the software.

### 3.1 Model construction

The first step of model construction is to define the model type and fundamental geometry (spherical particle or planar film). Currently, KM-SUB and KM-GAP models are available in *MultilayerPy*. Once defined, reaction scheme and diffusion regime are created taking into account the model type and model components, which are instantiated as separate model component objects. It is possible for the user to display the reaction scheme to check that the desired reaction scheme has been defined.

Many reaction schemes, such as the oleic acid-ozone system, have an uneven product distribution which can be described by a branching ratio. *MultilayerPy* allows the user to apply a branching ratio to a reaction scheme to account for this.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


Composition-dependent bulk diffusion is made possible by the diffusion regime defined by the user. Three different parameterisations are currently available in *MultilayerPy*: (i) Vignes-type (default) (Vignes, 1966); (ii) obstruction theory (Stroeve, 1975); (iii) linear combination (Pöschl et al., 2007; Shiraiwa et al., 2010). The evolution of particle diffusivity is of interest to the community and has been highlighted in the kinetic multi-layer modelling and experimental literature (*e.g.* (semi-) solid crust formation) (Milsom et al., 2021b, 2021a, 2022a; Nash et al., 2006; Pfrang et al., 2011; Zhou et al., 2019). If a simpler well-mixed system is modelled, diffusion evolution as a function of particle composition can be turned off by supplying a "null" argument to the diffusion regime object (see examples in the *Jupyter* notebooks in the source code repository).

Once the model type, model components, reaction scheme and diffusion regime are defined, the model can be constructed. The building blocks (objects) are supplied to a model builder object. Invoking the *build* method of this object has two main functions: (i) it writes the model code (ODE function) to a separate *.py* file; and (ii) it defines a list of required parameters for the model to run (see next section).

The model code defining the system of ODEs for each component in each layer is automatically generated. This is a key utility of *MultilayerPy*, especially when considering a complex multi-component system. Writing many lines of code manually is error prone. The removal of this risk, along with the readable nature of the model code, enhances the reproducibility and reliability of the results. This file is, however, customisable should the user want to add or remove specific processes in this template framework. Custom modifications should be checked thoroughly.

### 3.2 Running the model

Once constructed, the model can be run by incorporation into a *simulate* object, which also requires input model parameters supplied as a Python dictionary. The simulate object can also contain experimental data for optimisation.

Before the model can be run, the number of model bulk layers, initial concentration of each component in each layer, initial layer volume, initial layer surface area and initial layer thickness need to be supplied to the simulate object. *MultilayerPy* has utility functions which make this process straightforward. The number of model bulk layers is particularly important when modelling viscous systems because bulk diffusion gradients need to be resolved to describe the system sufficiently – the assumption in KM-SUB and KM-GAP models is that each bulk layer is well-mixed.

After running the model for the desired time span the model output, consisting of spatially and temporally resolved number concentration arrays for each model component, is saved and associated with the simulate object. The user can access these outputs easily for further analysis and visualisation.

Simple plotting methods are available which will quickly plot the model output including surface concentrations or the total number of each component in the model as a function of simulation time. Summary heat map plots of component bulk concentrations are also accessible via a plotting function. This allows the user to quickly visualise a model run and decide on the next course of action.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


If a KM-GAP model is implemented, the volume, surface area and thickness of each bulk layer as a function of time can be accessed via the simulate object. From this, the user can plot the change in particle diameter and volume as a function of time. An example of this is presented in the corresponding KM-GAP *Jupyter* Notebook in the source code.

Sometimes modification of model input parameters is required. For example, the concentration of a reactive gas can be changed partway through an experiment. This can be implemented in the *MultilayerPy* framework by supplying a function which changes the concentration of the reactive gas after a certain time point (see the "parameter evolution" *Jupyter* Notebook in the source code for an example). This is possible for any of the model input parameters. Additionally, extra input parameters can be supplied to the model in this manner. These parameters can themselves be optimised.

### 3.3 Model optimisation

After constructing and running the kinetic multi-layer model, the user may want to optimise the model input parameters. Data can be associated with the simulate object which was created to run the model. Optimisation requires a minimum of a simulate object with data associated with it.

Currently, the built-in cost function is the mean squared error (MSE), which is defined in Eq. 1:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} w_i(Y_{data,i} - Y_{model,i})^2 \quad (1)$$

where $n$ is the number of datapoints, $Y_{\text{data}}$ and $Y_{\text{model}}$ are the experimental and model datapoints, respectively. A weighting factor ($w_i$) is applied if there are uncertainties associated with the data and is equivalent to the inverse square of the uncertainty. This is equal to one if no uncertainties are provided. The user can supply their own cost function if desired.

There are two main methods of model optimisation available in *MultilayerPy*: (i) local optimisation (least-squares simplex method); (ii) global optimisation (differential evolution method). Both methods use the implementations found in the *SciPy* Python package (Virtanen et al., 2020).

Local optimisation is the least computationally expensive way of optimising the model. However, as the name implies, only a local minimum in the cost function will be found (though this could also be the global minimum). This algorithm does not search the entire parameter space looking for a global minimum and will only settle in the nearest local minimum. The user must therefore be confident that the initial model parameters are close to what represents their system.

Global optimisation is more computationally expensive. A description of the Monte Carlo Genetic Algorithm (MCGA) has been presented by Berkemeier *et al.* in the context of kinetic multi-layer modelling (Berkemeier et al., 2017). The implementation in *MultilayerPy* is similar, with an initial Latin Hypercube sampling step (instead of a Monte Carlo sampling step), followed by a differential evolution algorithm implementation which searches the parameter space and applies concepts of natural selection to mutate and select each successive generation of parameter sets (Storn and Price, 1997).

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


### 3.4 Estimating parameter uncertainty with a Markov Chain Monte Carlo (MCMC) method

A range of model outputs could be considered consistent with experimental data due to the uncertainty associated with each datapoint. Global optimisation algorithms, such as differential evolution, focus on achieving a single parameter set with the lowest cost function. However, running the same global optimisation algorithm multiple times can return optimised parameter sets with different values. This is indeed the case with the MCGA algorithm where uncertainties in the optimised parameters are presented as the distribution of optimised parameter values after a set number of MCGA runs (Berkemeier et al., 2017).

MCMC sampling seeks to define the probability distribution for each model parameter by finding the region of highest probability in a given parameter space. First, a parameter set is initiated within pre-defined bounds. Then the parameter set is allowed to "walk" around the parameter space. The likelihood of the next proposed step being accepted is dependent on the likelihood of the current position in the parameter space (i.e., the goodness of the model-data fit). This means that the chain will tend towards regions of higher probability. This is the Markov Chain aspect of the MCMC algorithm. The Monte Carlo aspect arises from the randomness associated the proposed next step in the chain, the next *sample*. When a run is successful, the chain of samples will equilibrate around the region of highest probability. A probability distribution for each varying model parameter can be obtained from the values returned by this equilibrated chain of samples (see Fig. 4(c) later for an example).

The MCMC algorithm infers the posterior probability distribution function, $p(\theta|D)$. This is the probability of the model parameters ($\theta$) given the data ($D$). This is calculated via Bayes' rule (Eq. 2).

$$p(\theta|D) = \frac{1}{Z}p(D|\theta)p(\theta) \quad (2)$$

$p(\theta|D)$ is proportional to the likelihood ($p(D|\theta)$ – essentially the goodness of fit) and the prior probability for the parameters ($p(\theta)$) which is normally set to 1 (uniform) if the prior probability distribution function for each parameter is unknown. The evidence ($Z$) is a constant which is generally hard to calculate and is ignored in MCMC as it is constant for a given model-experiment system and does not affect the outcome of an MCMC sampling run. Because of this, MCMC cannot be used to compare two different models as $p(D|\theta)$ would not be normalised to the same scale. For a more detailed description of MCMC sampling and best practices, a paper by Hogg and Foreman-Mackey walks the reader through the algorithm and troubleshoots common problems (Hogg and Foreman-Mackey, 2018).

*MultilayerPy* employs the well-established *emcee* python package for MCMC sampling (Foreman-Mackey et al., 2013). This is an *ensemble* sampler which initiates a number of walkers in the parameter space. The ensemble of walkers can then proceed with the MCMC routine. Parallelisation of the algorithm can be implemented as each walker can be iterated independently. An example of this is given in the "MCMC sampling" *Jupyter* notebook.

Though MCMC can technically be used as a global optimiser, this is not recommended as there are dedicated global optimisation algorithms which are much more efficient at finding the global minimum, such as the differential evolution algorithm employed in *MultilayerPy*.

In practice, the model-data system can be optimised (locally or globally) before initiating the walkers in a tight gaussian "ball" around the optimum point in the parameter space. Initialising the walkers in this way can reduce the time for

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


them to equilibrate. This is the recommended procedure in *MultilayerPy* and is the way presented in the "MCMC sampling" *Jupyter* notebook associated with the source code.

An example of how MCMC sampling is implemented in *MultilayerPy* is presented in the oleic acid monolayer case study (see *case study 2*).

## 4 Case studies: oleic acid ozonolysis

The oleic acid-ozone reaction system is a well-established model compound for heterogeneous reactions of organic aerosols due to its prevalence as a cooking emission tracer (Lyu et al., 2021; Vicente et al., 2021; Wang et al., 2020) and has been the subject of numerous experimental studies (Dennis-Smither et al., 2012; González-Labrada et al., 2007; Hearn and Smith, 2004; Hosny et al., 2013; Hung et al., 2005; King et al., 2004, 2020; Knopf et al., 2005; Milsom et al., 2021b, 2021a; Pfrang et al., 2017; Sebastiani et al., 2018; Smith et al., 2002; Woden et al., 2021; Zahardis and Petrucci, 2007). For this reason, it has been a popular system to model (Berkemeier et al., 2021; Gallimore et al., 2017; Milsom et al., 2022a Pfrang et al., 2010, 2011; Shiraiwa et al., 2010, 2012). The precise reaction scheme is still not constrained, especially when considering the impact of reactive intermediates, although recent work has advanced our understanding of this system and highlighted experimental gaps to be filled (Berkemeier et al., 2021). Here, we use the oleic acid-ozone system as a case study. Case study 1 demonstrates fitting a KM-SUB and KM-GAP model to the same oleic acid particle ozonolysis dataset. Case study 2 demonstrates an application to an insoluble oleic acid monolayer and the MCMC sampling procedure employed in *MultilayerPy* in order to estimate the uncertainty associated with a model parameter.

### 4.1 Case study 1: fitting KM-SUB and KM-GAP to oleic acid particle ozonolysis data

In this case study, KM-SUB and KM-GAP models are fitted experimental data for the ozonolysis of oleic acid particles from Ziemann (2005). This is the same example case study used in the kinetic double-layer model (K2-SUB) (Pfrang et al., 2010), KM-SUB (Shiraiwa et al., 2010) and KM-GAP (Shiraiwa et al., 2012) papers.

The radius of the particles was 0.2 μm and the gas phase ozone concentration was $7.0 \times 10^{13}$ $cm^{-3}$ (2.8 ppm at 101325 Pa). 10 bulk layers were initiated in our case study. A table with all other optimised parameters along with bounds for varied parameters is presented in the Supplement (Table S1).

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


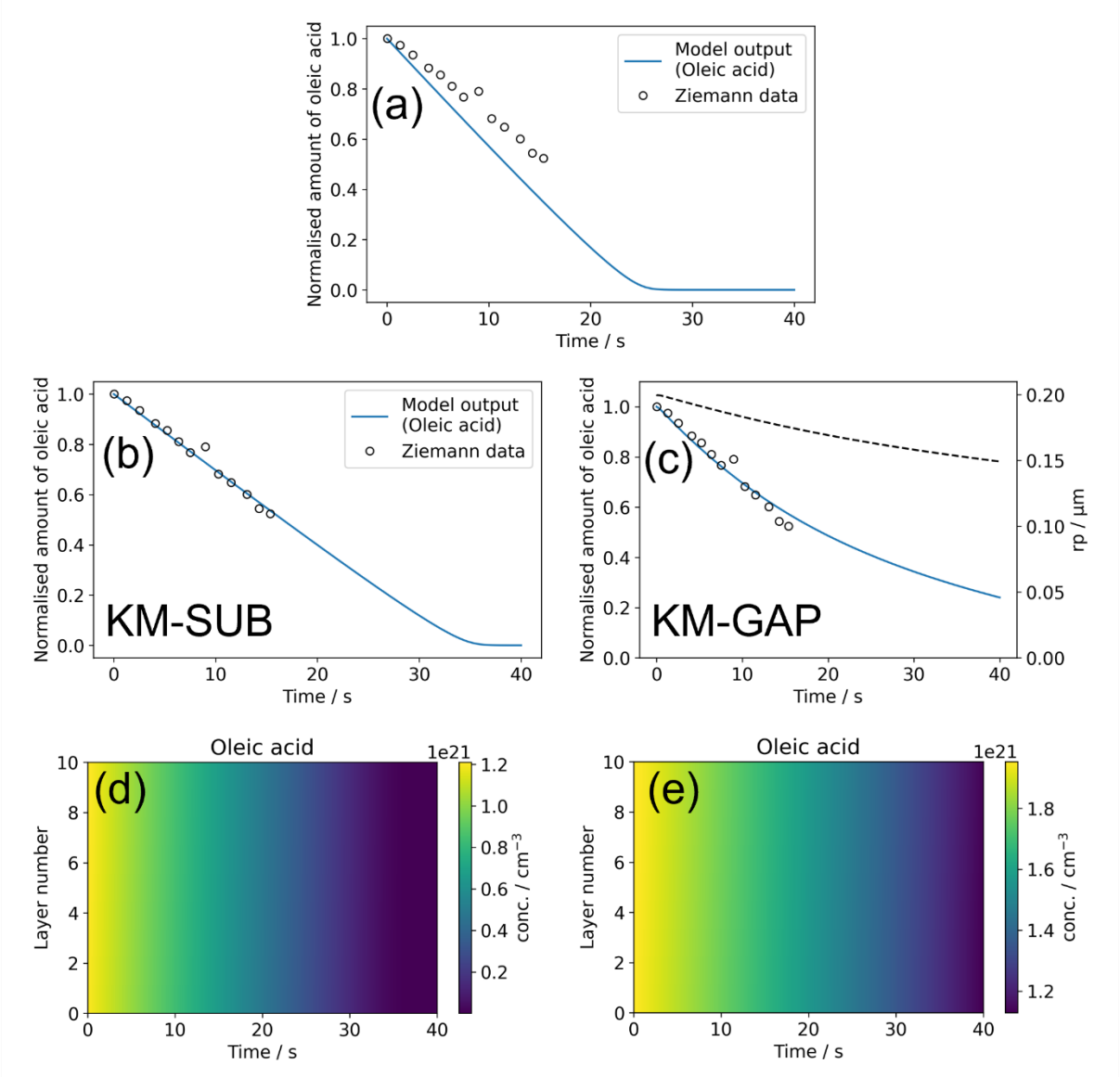


**Figure 3. Model optimisation to experimental data from Ziemann (Ziemann, 2005) using the *MultilayerPy* model building and optimisation tool. (a) The initial model output before optimisation. (b) KM-SUB model output optimised by varying $\alpha_{s,0,ozone}$. (c) KM-GAP model output optimised by varying $\alpha_{s,0,ozone}$ and the desorption lifetime of the ozonolysis product nonanal ($\tau_{d,nonanal}$). The radius of the particle ($r_p$) is also plotted as a dotted line ($r_p$ is not displayed in figure (b) since particle size changes are not described in KM-SUB). (d) Depth-resolved oleic acid concentration profile for the optimised KM-SUB model. (e) Depth-resolved oleic acid concentration profile for the optimised KM-GAP model.**

A modelling experiment was carried out using *MultilayerPy* with the data from Ziemann (Ziemann, 2005) (Fig. 3). The code used to generate these outputs is available in "crash course" and "KM-GAP model creation" *Jupyter* notebooks within the source code.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


The differential evolution algorithm was used as the global optimiser for the KM-SUB and KM-GAP model runs in this case study (Storn and Price, 1997). The surface accommodation coefficient of ozone on a free surface ($\alpha_{s,0,ozone}$) was varied in the KM-SUB model. Fig. 3(b)&(c) demonstrate good fits to the experimental data. The parameters for the fitted KM-SUB model presented here are identical to those used to fit to the same dataset in the KM-SUB model description paper (Shiraiwa et al., 2010).

Nonanal was assumed to be volatile in the KM-GAP reaction scheme as it is the only product known to be volatile (Vesna et al., 2009). The desorption lifetime of nonanal ($\tau_{d,nonanal}$) and $\alpha_{s,0,ozone}$ were varied in this case study. As KM-GAP resolves changes in particle size via the loss or gain of volatile species to the particle, $\tau_{d,nonanal}$ will have an impact on the size of the particle during the model run. Time-resolved particle size information is not known for this experiment, accounting for the significant differences seen between the KM-SUB and KM-GAP outputs at times greater than the last experimental datapoint (~17 min).

In both KM-SUB and KM-GAP models presented here the bulk phase is well-mixed, demonstrated by the lack of oleic acid concentration gradient occurring throughout the particle during ozonolysis (Fig. 3(d)&(e)). The concentration of oleic acid in each model layer is generally higher in the KM-GAP output compared with KM-SUB due to the shrinking of the model layers accounted for in KM-GAP and caused by the removal of nonanal from the particle.

### 4.2 Case study 2: fitting a KM-SUB model to oleic acid monolayer ozonolysis data, including MCMC sampling

Woden *et al.* ozonised insoluble floating monolayers of oleic acid deposited on water (Woden et al., 2021). The reaction kinetics were followed using neutron reflectometry (NR) and fitted parameters from an interfacial model applied to the NR data – a common method of extracting kinetic information from NR experiments (King et al., 2009, 2020; Pfrang et al., 2014; Sebastiani et al., 2018; Woden et al., 2018, 2021). The ozone concentration for the example model-data system was 323 ± 29 ppb.

The dissolution of oleic acid and products into the aqueous phase was turned off in the model building process by setting bulk diffusion parameters to 0. The model in this case was particularly sensitive to $\alpha_{s,0,ozone}$. This was selected as the fitting parameter.

$\alpha_{s,0,ozone}$ can range between 0 – 1. This was set as the fitting bound for both the differential evolution (global) optimisation algorithm and the MCMC sampling procedure.

An initial differential evolution procedure was carried out on this model-data system. This optimised value of $\alpha_{s,0,ozone}$ was then used to initialise the ensemble of walkers for the MCMC sampling procedure – this is handled by *MultilayerPy*. The MCMC algorithm can then be run in series or in parallel. The result of this MCMC sampling procedure is presented in Fig. 4.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


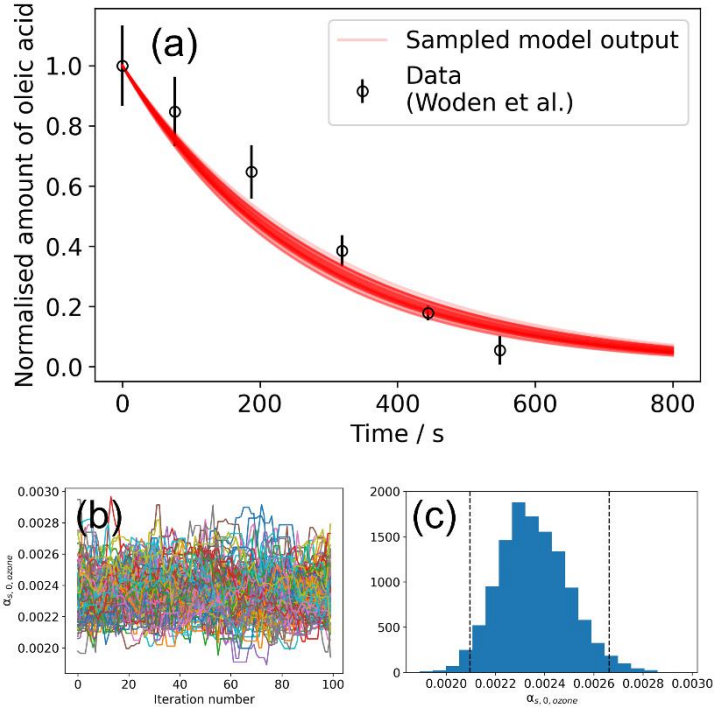


**Figure 4. Result of the global optimisation-MCMC sampling procedure. (a) 200 model outputs sampled from the MCMC sampling procedure, showing the range of model outputs consistent with the data. (b) A plot of $\alpha_{s,0,ozone}$ vs. iteration number of the MCMC algorithm for a converged set of walkers (Markov chains). 100 samples were discarded (burn-in step) before chains converged. (c) The histogram of $\alpha_{s,0,ozone}$ values derived from the walkers presented in panel (b). Vertical dashed lines represent the interval in which 95 % of the data lie.**

The mean value of $\alpha_{s,0,ozone}$ obtained from MCMC sampling is $(2.35 \pm 0.14) \times 10^{-3}$. In this case, the distribution of $\alpha_{s,0,ozone}$ is Gaussian (Fig. 4(c)). For other distributions, quoting an interquartile range may be more representative. The lower and upper bound for each optimised parameter is included when the user exports their modelling results. In this case, the lower and upper bounds of the inter-quartile range for $\alpha_{s,0,ozone}$ are $2.36 \times 10^{-3}$ and $2.44 \times 10^{-3}$, respectively.

Constraining model input parameters experimentally remains the best way to improve the estimation of unknown model parameters. For example, simultaneously varying the desorption lifetime of ozone ($\tau_{d,ozone}$), the surface reaction rate coefficient ($k_{surf}$) and $\alpha_{s,0,ozone}$ returns a range of possible combinations of these three parameters – all of which are associated with surface processes. This is demonstrated by the strong correlation observed between these parameters during MCMC sampling (Fig. S1, the Supplement). *MultilayerPy* could be used to identify which experimental parameters should be constrained and inform experimental work.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


## 5 Conclusions: summary and future developments

This open-source model construction framework represents a first step towards more reproducible kinetic multi-layer modelling of atmospheric aerosols and films. The way that models are constructed and the ease with which model code can be generated in *MultilayerPy* encourages the user to share their models with the community.

*MultilayerPy* provides a simple model construction and optimisation pipeline for the user to follow with a few lines of code needed to produce results. This is a major practical advance compared to manually typing out ODEs – which can be an unnecessary source of error and is time consuming. There is sufficient flexibility to suit more complex systems with parameterisation of model input parameters, along with customisation of the source model code "under the hood".

The trade-off when integrating systems of ODEs written in a high-level programming language such as Python or MATLAB is that, compared with C/C++ and FORTRAN, these integrations are relatively slow. Future work will consider ways of speeding up ODE integration. This is partially addressed via the ability to parallelise the global optimisation and MCMC algorithms. The Julia programming language is also an attractive proposition, allowing human-readable code to be written and run at speeds faster than that of Python and MATLAB. Indeed, Julia has recently been applied in an atmospheric context to construct box models (Huang and Topping, 2021). These are feasible options for the direction of *MultilayerPy* development in the future.

The modular way in which *MultilayerPy* is constructed makes future modular developments relatively straightforward. Other modelling systems based on the kinetic multi-layer model presented here (KM-SUB and KM-GAP) have been developed and include other "compartments" such as the skin (KM-SUB-skin) (Lakey et al., 2017), indoor air boundary layer (KM-BL) (Morrison et al., 2019), film formation and growth (KM-FILM) (Lakey et al., 2021), and epithelial lining fluid (KM-SUB-ELF) (Lelieveld et al., 2021). Future iterations of *MultilayerPy* could incorporate such multi-compartment models, coupling different processes occurring in various environments and contexts.

As an open-source project, contributions from the particle and film modelling community are strongly encouraged and will help push the project forward and achieve collective goals. This will also facilitate scientific collaboration and encourage more reproducible modelling studies as a result.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


## Appendix

### A. List of acronyms

| *Parameter name* | *Description (units)* |
|---|---|
| delta_1 | molecular diameter of component 1 (cm) |
| w_1 | mean thermal velocity of component 1 in the gas phase (cm $s^{-1}$) |
| H_1 | Henry's law coefficient (mol $cm^{-3}$ $atm^{-1}$) |
| alpha_s_0_1 | Surface accommodation coefficient of component 1 on a clear surface |
| Db_1 | Bulk diffusion coefficient of component 1 ($cm^2$ $s^{-1}$) |
| Db_1_2 | Bulk diffusion coefficient of component 1 in component 2 ($cm^2$ $s^{-1}$) |
| T | Temperature (K) |
| k_1_2 | Second-order rate coefficient for the bulk reaction of component 1 and component 2 ($cm^3$ $s^{-1}$) |
| k_1_2_surf | Second-order rate coefficient for the surface reaction of component 1 and component 2 ($cm^2$ $s^{-1}$) |
| k1_1 | first order decay constant for component 1 ($s^{-1}$) |
| Xgs_1 | Near-surface gas phase concentration of component 1 ($cm^{-3}$) |
| Zgs_1 | Near-surface gas phase concentration of component 1 with KM-GAP notation ($cm^{-3}$) |
| p_1 | equilibrium vapour pressure of component 1 (Pa) |
| Td_1 | Desorption lifetime of component 1 (s) |
| scale_bulk_to_surf | scaling factor applied to bulk reaction rate coefficient to return the surface reaction rate coefficient (cm) |

**Table A1. Table of variable names, descriptions and units supplied to models created in *MultilayerPy*.**

## Code availability

The *MultilayerPy* software, including tutorials and documentation, is available at https://github.com/tintin554/kinetic-multilayer-model-builder (last access: 27 April 2022) and https://doi.org/10.5281/zenodo.6411188 (Milsom et al., 2022b). The code is released under the GPL v3.0 license.

## Author contribution

AM designed, wrote and tested *MultilayerPy* and the manuscript. AL tested and helped optimise the software. CP supervised the project, contributed to the discussion and manuscript. AMS co-supervised the project and contributed to the discussion and manuscript.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


**Acknowledgements**

CP would like to thank Uli Pöschl and Manabu Shiraiwa for the opportunity to be part of the PRA modelling developments and especially for support during his visits to the Max Planck Institute for Chemistry in Mainz since 2009. AM acknowledges support from NERC SCENARIO and CENTA DTPs (grant number NE/L002566/1) and NERC (grant number NE/T00732X/1).

**References**

Abbatt, J. P. D. and Wang, C.: The atmospheric chemistry of indoor environments, Environ. Sci. Process. Impacts, 22(1), 25–48, doi:10.1039/c9em00386j, 2020.

Berkemeier, T., Ammann, M., Krieger, U. K., Peter, T., Spichtinger, P., Pöschl, U., Shiraiwa, M. and Huisman, A. J.: Technical note: Monte Carlo genetic algorithm (MCGA) for model analysis of multiphase chemical kinetics to determine transport and reaction rate coefficients using multiple experimental data sets, Atmos. Chem. Phys., 17(12), 8021–8029, doi:10.5194/acp-17-8021-2017, 2017.

Berkemeier, T., Mishra, A., Mattei, C., Huisman, A. J., Krieger, U. K. and Pöschl, U.: Ozonolysis of Oleic Acid Aerosol Revisited: Multiphase Chemical Kinetics and Reaction Mechanisms, ACS Earth Sp. Chem., 5(12), 3313–3323, doi:10.1021/acsearthspacechem.1c00232, 2021.

Dennis-Smither, B. J., Miles, R. E. H. and Reid, J. P.: Oxidative aging of mixed oleic acid/sodium chloride aerosol particles, J. Geophys. Res. Atmos., 117(20), 1–13, doi:10.1029/2012JD018163, 2012.

Foreman-Mackey, D., Hogg, D. W., Lang, D. and Goodman, J.: emcee : The MCMC Hammer , Publ. Astron. Soc. Pacific, 125(925), 306–312, doi:10.1086/670067, 2013.

Gallimore, P. J., Griffiths, P. T., Pope, F. D., Reid, J. P. and Kalberer, M.: Comprehensive modeling study of ozonolysis of oleic acid aerosol based on real-time, online measurements of aerosol composition, J. Geophys. Res., 122(8), 4364–4377, doi:10.1002/2016JD026221, 2017.

González-Labrada, E., Schmidt, R. and DeWolf, C. E.: Kinetic analysis of the ozone processing of an unsaturated organic monolayer as a model of an aerosol surface, Phys. Chem. Chem. Phys., 9(43), 5814–5821, doi:10.1039/b707890k, 2007.

Hearn, J. D. and Smith, G. D.: Kinetics and product studies for ozonolysis reactions of organic particles using aerosol CIMS, J. Phys. Chem. A, 108(45), 10019–10029, doi:10.1021/jp0404145, 2004.

Hogg, D. W. and Foreman-Mackey, D.: Data Analysis Recipes: Using Markov Chain Monte Carlo, Astrophys. J. Suppl. Ser., 236(1), 11, doi:10.3847/1538-4365/aab76e, 2018.

Hosny, N. A., Fitzgerald, C., Tong, C., Kalberer, M., Kuimova, M. K. and Pope, F. D.: Fluorescent lifetime imaging of

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


atmospheric aerosols: A direct probe of aerosol viscosity, Faraday Discuss., 165, 343–356, doi:10.1039/c3fd00041a, 2013.

Hosny, N. A., Fitzgerald, C., Vyšniauskas, A., Athanasiadis, A., Berkemeier, T., Uygur, N., Pöschl, U., Shiraiwa, M., Kalberer, M., Pope, F. D. and Kuimova, M. K.: Direct imaging of changes in aerosol particle viscosity upon hydration and chemical aging, Chem. Sci., 7(2), 1357–1367, doi:10.1039/c5sc02959g, 2016.

Hua, A. K., Lakey, P. S. J. and Shiraiwa, M.: Multiphase Kinetic Multilayer Model Interfaces for Simulating Surface and Bulk Chemistry for Environmental and Atmospheric Chemistry Teaching, J. Chem. Educ., doi:10.1021/acs.jchemed.1c00931, 2022.

Huang, L. and Topping, D.: Jlbox v1.1: A Julia-based multi-phase atmospheric chemistry box model, Geosci. Model Dev., 14(4), 2187–2203, doi:10.5194/gmd-14-2187-2021, 2021.

Hung, H. M., Katrib, Y. and Martin, S. T.: Products and mechanisms of the reaction of oleic acid with ozone and nitrate radical, J. Phys. Chem. A, 109(20), 4517–4530, doi:10.1021/jp0500900, 2005.

King, M. D., Thompson, K. C. and Ward, A. D.: Laser tweezers raman study of optically trapped aerosol droplets of seawater and oleic acid reacting with ozone: Implications for cloud-droplet properties, J. Am. Chem. Soc., 126(51), 16710–16711, doi:10.1021/ja044717o, 2004.

King, M. D., Rennie, A. R., Thompson, K. C., Fisher, F. N., Dong, C. C., Thomas, R. K., Pfrang, C. and Hughes, A. V.: Oxidation of oleic acid at the air-water interface and its potential effects on cloud critical supersaturations, Phys. Chem. Chem. Phys., 11(35), 7699–7707, doi:10.1039/b906517b, 2009.

King, M. D., Rennie, A. R., Pfrang, C., Hughes, A. V. and Thompson, K. C.: Interaction of nitrogen dioxide (NO2) with a monolayer of oleic acid at the air-water interface - A simple proxy for atmospheric aerosol, Atmos. Environ., 44(14), 1822–1825, doi:10.1016/j.atmosenv.2010.01.031, 2010.

King, M. D., Jones, S. H., Lucas, C. O. M., Thompson, K. C., Rennie, A. R., Ward, A. D., Marks, A. A., Fisher, F. N., Pfrang, C., Hughes, A. V. and Campbell, R. A.: The reaction of oleic acid monolayers with gas-phase ozone at the air water interface: The effect of sub-phase viscosity, and inert secondary components, Phys. Chem. Chem. Phys., 22(48), 28032–28044, doi:10.1039/d0cp03934a, 2020.

Knopf, D. A., Anthony, L. M. and Bertram, A. K.: Reactive uptake of O3 by multicomponent and multiphase mixtures containing oleic acid, J. Phys. Chem. A, 109(25), 5579–5589, doi:10.1021/jp0512513, 2005.

Lakey, P. S. J., Wisthaler, A., Berkemeier, T., Mikoviny, T., Pöschl, U. and Shiraiwa, M.: Chemical kinetics of multiphase reactions between ozone and human skin lipids: Implications for indoor air quality and health effects, Indoor Air, 27(4), 816–828, doi:10.1111/ina.12360, 2017.

Lakey, P. S. J., Eichler, C. M. A., Wang, C., Little, J. C. and Shiraiwa, M.: Kinetic multi-layer model of film formation, growth, and chemistry (KM-FILM): Boundary layer processes, multi-layer adsorption, bulk diffusion, and heterogeneous reactions,

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


Indoor Air, (April), ina.12854, doi:10.1111/ina.12854, 2021.

Lelieveld, S., Wilson, J., Dovrou, E., Mishra, A., Lakey, P. S. J., Shiraiwa, M., Pöschl, U. and Berkemeier, T.: Hydroxyl Radical Production by Air Pollutants in Epithelial Lining Fluid Governed by Interconversion and Scavenging of Reactive Oxygen Species, Environ. Sci. Technol., 55(20), 14069–14079, doi:10.1021/acs.est.1c03875, 2021.

Lyu, X., Huo, Y., Yang, J., Yao, D., Li, K., Lu, H., Zeren, Y. and Guo, H.: Real-time molecular characterization of air pollutants in a Hong Kong residence: Implication of indoor source emissions and heterogeneous chemistry, Indoor Air, 31(5), 1340–1352, doi:10.1111/ina.12826, 2021.

Milsom, A., Squires, A. M., Boswell, J. A., Terrill, N. J., Ward, A. D. and Pfrang, C.: An organic crystalline state in ageing atmospheric aerosol proxies: Spatially resolved structural changes in levitated fatty acid particles, Atmos. Chem. Phys., 21(19), 15003–15021, doi:10.5194/acp-21-15003-2021, 2021a.

Milsom, A., Squires, A. M., Woden, B., Terrill, N. J., Ward, A. D. and Pfrang, C.: The persistence of a proxy for cooking emissions in megacities: a kinetic study of the ozonolysis of self-assembled films by simultaneous small and wide angle X-ray scattering (SAXS/WAXS) and Raman microscopy, Faraday Discuss., 226, 364–381, doi:10.1039/D0FD00088D, 2021b.

Milsom, A., Squires, A. M., Ward, A. D. and Pfrang, C.: The impact of molecular self-organisation on the atmospheric fate of a cooking aerosol proxy, Atmos. Chem. Phys., 22(7), 4895–4907, doi:10.5194/acp-22-4895-2022, 2022a.

Milsom, A., Lees, A., Squires, A. M. and Pfrang, C.: MultilayerPy, Zenodo [code], doi:10.5281/zenodo.6411188, 2022b.

Morrison, G., Lakey, P. S. J., Abbatt, J. and Shiraiwa, M.: Indoor boundary layer chemistry modeling, Indoor Air, 29(6), 956–967, doi:10.1111/ina.12601, 2019.

Nash, D. G., Tolocka, M. P. and Baer, T.: The uptake of O3 by myristic acid-oleic acid mixed particles: Evidence for solid surface layers, Phys. Chem. Chem. Phys., 8(38), 4468–4475, doi:10.1039/b609855j, 2006.

O'Meara, S. P., Xu, S., Topping, D., Rami Alfarra, M., Capes, G., Lowe, D., Shao, Y. and McFiggans, G.: PyCHAM (v2.1.1): A Python box model for simulating aerosol chambers, Geosci. Model Dev., 14(2), 675–702, doi:10.5194/gmd-14-675-2021, 2021.

Perkel, J. M.: Why Jupyter is data scientists' computational notebook of choice, Nature, 563(7729), 145–146, doi:10.1038/d41586-018-07196-1, 2018.

Pfrang, C., Shiraiwa, M. and Pöschl, U.: Coupling aerosol surface and bulk chemistry with a kinetic double layer model (K2-SUB): Oxidation of oleic acid by ozone, Atmos. Chem. Phys., 10(10), 4537–4557, doi:10.5194/acp-10-4537-2010, 2010.

Pfrang, C., Shiraiwa, M. and Pöschl, U.: Chemical ageing and transformation of diffusivity in semi-solid multi-component organic aerosol particles, Atmos. Chem. Phys., 11(14), 7343–7354, doi:10.5194/acp-11-7343-2011, 2011.

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


Pfrang, C., Sebastiani, F., Lucas, C. O. M., King, M. D., Hoare, I. D., Chang, D. and Campbell, R. A.: Ozonolysis of methyl oleate monolayers at the air-water interface: Oxidation kinetics, reaction products and atmospheric implications, Phys. Chem. Chem. Phys., 16(26), 13220–13228, doi:10.1039/c4cp00775a, 2014.

Pfrang, C., Rastogi, K., Cabrera-Martinez, E. R., Seddon, A. M., Dicko, C., Labrador, A., Plivelic, T. S., Cowieson, N. and Squires, A. M.: Complex three-dimensional self-assembly in proxies for atmospheric aerosols, Nat. Commun., 8(1), 1724, doi:10.1038/s41467-017-01918-1, 2017.

Pöschl, U.: Atmospheric Aerosols: Composition, Transformation, Climate and Health Effects, Angew. Chemie Int. Ed., 44(46), 7520–7540, doi:10.1002/anie.200501122, 2005.

Pöschl, U., Rudich, Y. and Ammann, M.: Kinetic model framework for aerosol and cloud surface chemistry and gas-particle interactions - Part 1: General equations, parameters, and terminology, Atmos. Chem. Phys., 7(23), 5989–6023, doi:10.5194/acp-7-5989-2007, 2007.

Schill, S. R., Collins, D. B., Lee, C., Morris, H. S., Novak, G. A., Prather, K. A., Quinn, P. K., Sultana, C. M., Tivanski, A. V., Zimmermann, K., Cappa, C. D. and Bertram, T. H.: The impact of aerosol particle mixing state on the hygroscopicity of sea spray aerosol, ACS Cent. Sci., 1(3), 132–141, doi:10.1021/acscentsci.5b00174, 2015.

Sebastiani, F., Campbell, R. A., Rastogi, K. and Pfrang, C.: Nighttime oxidation of surfactants at the air-water interface: Effects of chain length, head group and saturation, Atmos. Chem. Phys., 18(5), 3249–3268, doi:10.5194/acp-18-3249-2018, 2018.

Shaw, D. and Carslaw, N.: INCHEM-Py: An open source Python box model for indoor air chemistry, J. Open Source Softw., 6(63), 3224, doi:10.21105/joss.03224, 2021.

Shiraiwa, M., Pfrang, C. and Pöschl, U.: Kinetic multi-layer model of aerosol surface and bulk chemistry (KM-SUB): The influence of interfacial transport and bulk diffusion on the oxidation of oleic acid by ozone, Atmos. Chem. Phys., 10, 3673–3691, doi:10.5194/acp-10-3673-2010, 2010.

Shiraiwa, M., Ammann, M., Koop, T. and Poschl, U.: Gas uptake and chemical aging of semisolid organic aerosol particles, Proc. Natl. Acad. Sci. U. S. A., 108(27), 11003–11008, doi:10.1073/pnas.1103045108, 2011.

Shiraiwa, M., Pfrang, C., Koop, T. and Pöschl, U.: Kinetic multi-layer model of gas-particle interactions in aerosols and clouds (KM-GAP): Linking condensation, evaporation and chemical reactions of organics, oxidants and water, Atmos. Chem. Phys., 12(5), 2777–2794, doi:10.5194/acp-12-2777-2012, 2012.

Smith, G. D., Woods, E., DeForest, C. L., Baer, T. and Miller, R. E.: Reactive uptake of ozone by oleic acid aerosol particles: Application of single-particle mass spectrometry to heterogeneous reaction kinetics, J. Phys. Chem. A, 106(35), 8085–8095, doi:10.1021/jp020527t, 2002.

Sommariva, R., Cox, S., Martin, C., Borońska, K., Young, J., Jimack, P. K., Pilling, M. J., Matthaios, V. N., Nelson, B. S.,

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


Newland, M. J., Panagi, M., Bloss, W. J., Monks, P. S. and Rickard, A. R.: AtChem (version 1), an open-source box model for the Master Chemical Mechanism, Geosci. Model Dev., 13(1), 169–183, doi:10.5194/gmd-13-169-2020, 2020.

Storn, R. and Price, K.: Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, J. Glob. Optim., 11, 341–359, doi: 10.1023/A:1008202821328, 1997.

Stroeve, P.: On the Diffusion of Gases in Protein Solutions, Ind. Eng. Chem. Fundam., 14(2), 140–141, doi:10.1021/i160054a017, 1975.

Topping, D., Connolly, P. and Reid, J.: PyBox: An automated box-model generator for atmospheric chemistry and aerosol simulations., J. Open Source Softw., 3(28), 755, doi:10.21105/joss.00755, 2018.

Vesna, O., Sax, M., Kalberer, M., Gaschen, A. and Ammann, M.: Product study of oleic acid ozonolysis as function of humidity, Atmos. Environ., 43(24), 3662–3669, doi:10.1016/j.atmosenv.2009.04.047, 2009.

Vicente, A. M. P., Rocha, S., Duarte, M., Moreira, R., Nunes, T. and Alves, C. A.: Fingerprinting and emission rates of particulate organic compounds from typical restaurants in Portugal, Sci. Total Environ., 778, 146090, doi:10.1016/j.scitotenv.2021.146090, 2021.

Vignes, A.: Diffusion in Binary Solutions. Variation in Diffusion Coefficient with Composition, Ind. Eng. Chem. Fundam., 5(2), 189–199, doi: 10.1021/i160018a007, 1966.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., Vijaykumar, A., Bardelli, A. Pietro, Rothberg, A., Hilboll, A., Kloeckner, A., Scopatz, A., Lee, A., Rokem, A., Woods, C. N., Fulton, C., Masson, C., Häggström, C., Fitzgerald, C., Nicholson, D. A., Hagen, D. R., Pasechnik, D. V., Olivetti, E., Martin, E., Wieser, E., Silva, F., Lenders, F., Wilhelm, F., Young, G., Price, G. A., Ingold, G. L., Allen, G. E., Lee, G. R., Audren, H., Probst, I., Dietrich, J. P., Silterra, J., Webber, J. T., Slavič, J., Nothman, J., Buchner, J., Kulick, J., Schönberger, J. L., de Miranda Cardoso, J. V., Reimer, J., Harrington, J., Rodríguez, J. L. C., Nunez-Iglesias, J., Kuczynski, J., Tritz, K., Thoma, M., Newville, M., Kümmerer, M., Bolingbroke, M., Tartre, M., Pak, M., Smith, N. J., Nowaczyk, N., Shebanov, N., Pavlyk, O., Brodtkorb, P. A., Lee, P., McGibbon, R. T., Feldbauer, R., Lewis, S., Tygier, S., Sievert, S., Vigna, S., Peterson, S., More, S., Pudlik, T., et al.: SciPy 1.0: fundamental algorithms for scientific computing in Python, Nat. Methods, 17(3), 261–272, doi:10.1038/s41592-019-0686-2, 2020.

Wang, Q., He, X., Zhou, M., Huang, D. D., Qiao, L., Zhu, S., Ma, Y. G., Wang, H. L., Li, L., Huang, C., Huang, X. H. H., Xu, W., Worsnop, D., Goldstein, A. H., Guo, H., Yu, J. Z., Huang, C. and Yu, J. Z.: Hourly Measurements of Organic Molecular Markers in Urban Shanghai, China: Primary Organic Aerosol Source Identification and Observation of Cooking Aerosol

https://doi.org/10.5194/egusphere-2022-143
Preprint. Discussion started: 2 May 2022


Aging, ACS Earth Sp. Chem., 4(9), 1670–1685, doi:10.1021/acsearthspacechem.0c00205, 2020.

Woden, B., Skoda, M., Hagreen, M. and Pfrang, C.: Night-Time Oxidation of a Monolayer Model for the Air–Water Interface of Marine Aerosols—A Study by Simultaneous Neutron Reflectometry and in Situ Infra-Red Reflection Absorption Spectroscopy (IRRAS), Atmosphere, 9(12), 471, doi:10.3390/atmos9120471, 2018.

Woden, B., Skoda, M. W. A., Milsom, A., Gubb, C., Maestro, A., Tellam, J. and Pfrang, C.: Ozonolysis of fatty acid monolayers at the air–water interface: organic films may persist at the surface of atmospheric aerosols, Atmos. Chem. Phys., 21(2), 1325–1340, doi:10.5194/acp-21-1325-2021, 2021.

Zahardis, J. and Petrucci, G. A.: The oleic acid-ozone heterogeneous reaction system: Products, kinetics, secondary chemistry, and atmospheric implications of a model system - A review, Atmos. Chem. Phys., 7(5), 1237–1274, doi:10.5194/acp-7-1237-2007, 2007.

Zhou, S., Hwang, B. C. H., Lakey, P. S. J., Zuend, A., Abbatt, J. P. D. and Shiraiwa, M.: Multiphase reactivity of polycyclic aromatic hydrocarbons is driven by phase separation and diffusion limitations, Proc. Natl. Acad. Sci. U. S. A., 116(24), 11658–11663, doi:10.1073/pnas.1902517116, 2019.

Ziemann, P. J.: Aerosol products, mechanisms, and kinetics of heterogeneous reactions of ozone with oleic acid in pure and mixed particles, Faraday Discuss., 130, 469–490, doi:10.1039/b417502f, 2005.