



The Chemical Mechanism Integrator *Cminor* v1.0: a stand-alone Fortran environment for the particle-based simulation of chemical multiphase mechanisms

Levin Rug¹, Willi Schimmel², Fabian Hoffmann¹, and Oswald Knoth²

¹Institute for Meteorology, Freie Universität Berlin, Berlin, Germany

²Modeling of Atmospheric Processes, Leibniz Institute for Tropospheric Research, Leipzig, Germany

Correspondence: Levin Rug (l.rug@fu-berlin.de) and Fabian Hoffmann (f.hoffmann@fu-berlin.de)

Received: 30 January 2025 – Discussion started: 9 May 2025

Revised: 30 September 2025 – Accepted: 14 October 2025 – Published: 26 November 2025

Abstract. We present version 1.0 of the Chemical Mechanism Integrator (*Cminor*), a fully modularized modern Fortran software package for the computational simulation of skeletal and detailed chemical kinetic systems derived from atmospheric and combustion chemistry. *Cminor* aims for the efficient simulation of complex chemical mechanisms by using various mathematical techniques. These are tailored to systems of ordinary differential equations (ODEs), having the specific structure arising from chemical reaction systems. Additionally, a high-speed mechanism parser allows the user to interchange reactions or their parameters in an ASCII format text file and immediately start a new simulation without recompiling, enabling fast and numerous simulations. *Cminor*'s solver technique is based on Rosenbrock methods. Different measures of local errors and an analytical Jacobian matrix approach are implemented, where efficiency is obtained by exploiting the sparsity structure of the Jacobian.

Cminor can be run in one of three configurations:

- A box-model framework for either pure gas-phase mechanisms or a multi-modal aerosol distribution dissolved in mono-dispersed cloud droplets.
- A rising adiabatic parcel, in which the activation of multi-modal aerosols is represented by solving the droplet condensation equation.
- A constant volume environment, where thermodynamic properties are evaluated by polynomial functions of temperature according to the standards of the Chemkin thermodynamic data base.

The software package is evaluated by applying seven different chemical mechanisms. Three of them are from the field of air-quality modeling and three are from the area of combustion kinetics, ranging from 7 species and 10 reactions to 10 196 species and 23 098 reactions. The last mechanism describes sulfur accumulation in clouds, which is tested along with a rising parcel and condensating cloud droplets.

1 Introduction

The understanding of physical processes that determine atmospheric states of climatological, ecological, social, political or agricultural relevance has undergone significant improvement in theoretical and modeling aspects. While chemical processes are omnipresent in any state of the atmosphere, they have drawn relatively minor attention when physical aspects of the atmosphere are considered. Nevertheless, the problem of considering the atmosphere as a physicochemical system, rather than an entirely physical system, cannot be avoided. Therefore, models of atmospheric processes usually work with assumptions about the underlying chemical state. Major aspects, e.g., sensitivity to the chemical state, chemical processing, or inter-dependencies, are mostly ignored. In contrast to this, the community of atmospheric chemistry has shown a number of important possible influences of chemical processes. These concern some of the most relevant aspects of atmospheric physics, like radiative properties, aerosol size spectra, cloud field dynamics and precipitation (see, e.g., Murphy et al., 1998; Kulmala et al., 2000; Kreiden-

weis et al., 2003; Christner et al., 2008; Zaveri et al., 2010; Kazil et al., 2011; Goel et al., 2020; Kupc et al., 2020; Zhang et al., 2024). Atmospheric chemical processes are described by a collection of reactions, constituting a specific chemical mechanism, possibly including multiple phases (gaseous, aqueous, solid) and phase changes, i.e., nucleation or dissolution. Timescales of chemical reactions might depend on temperature, pressure, insolation, or chemical composition, but generally vary largely from reaction to reaction.

By stating an ordinary differential equation (ODE) for the time evolution of every chemical species' concentration in a mechanism, large and stiff ODE systems arise. Therefore, the simulation of comprehensive chemical reaction mechanisms requires the development of efficient algorithms, particularly suited for the characteristics of ODE systems of chemical kinetics. The software package *Cminor* (Chemical Mechanism Integrator), written in modern Fortran, was developed as an environment for the simulation of atmospheric chemistry mechanisms of any size and type.

Several software packages have been developed for the analysis of kinetic systems, e.g., ChemKin (Kee et al., 1996), SPACCIM (Wolke et al., 2005), KPP (Sandu and Sander, 2006), CAABA/MECCA (Sander et al., 2011), and Speed-Chem (Perini et al., 2012). As a matter of fact, the ongoing further development and creation of new chemical mechanisms shows that a tool providing an efficient solver and high flexibility is absolutely necessary. This goal is achieved by *Cminor*.

This paper is organized as follows. The next section introduces how a *Cminor* simulation is set up, including various input options. The third section assesses the equations solved by *Cminor*, followed by a section on the numerical methods used by *Cminor*. Afterwards, simulation results are shown. The paper is concluded by a last section summarizing results and perspectives of this work. All variables not explicitly defined in the text are listed in the Supplement, Sect. S11.

2 Model Input

The *Cminor* software package is a two-in-one simulation tool for chemical kinetics problems in atmospheric and combustion chemistry. *Cminor* reads a reaction system provided as an ASCII file, generates a set of stiff ODEs, and solves it directly. One of the advantages of *Cminor* is the flexible high-speed ASCII mechanism parser where the generation time of internally used structures grows linearly with the mechanism size. This allows the user to quickly add/remove reactions or manipulate reaction rate constants within the reaction mechanism file. The ability to quickly redefine these parameters is crucial for investigating newly developed chemical mechanisms, in contrast to KPP (Sandu and Sander, 2006), where a fixed Fortran code for every new chemical mechanism has to be generated, compiled, and executed.

2.1 General setup

For a given model scenario with initial values and prescribed emission and deposition rates (i.e., constant sources and sinks for gas phase species), the user is able to simulate the evolution of species concentrations and other variables in a box-model framework over a defined time interval. The *Cminor* box-model is depicted in Fig. 1.

The syntax rules for combustion mechanisms can be taken from the ChemKin documentation (Kee et al., 1996), which is widely used for combustion kinetics. The necessary thermodynamic data base, which includes the coefficients for fitting polynomials, follows the same ChemKin syntax rules (Kee et al., 1990). For atmospheric chemical mechanisms, a detailed description of the syntax rules is provided in the Supplement. In any case, the input data has to be provided in readable ASCII format. Assistance in the syntax rules is also given by the examples provided in the repository, as well as on request by the authors. A short summary shall be given in the following.

Running a simulation requires four files beside the actual model code:

- the *.sys-file,
- the *.dat-file,
- the *.ini-file, and
- the *.run-file.

The *.sys-file is solely to list all reactions of the mechanism to simulate. The description of a reaction consists of at least three lines. The first line specifies the reaction class (gas phase, aqueous phase, forward/backward reactions, phase transfer), the second line declares the reaction itself, and the third line contains an identifier for the reaction rate constant type, with parameter values. The third line determines the specific formula of the rate constant, i.e., the speed, of the reaction. There are various reaction rate constant types, like

- Arrhenius laws (temperature dependencies),
- photolytic processes (insolation dependencies),
- Troe reactions (pressure dependencies),
- pH-dependent formulas,
- humidity-dependent formulas,

and others. Each type comes in a multitude of slightly varying formulas, depending on parameters like activation energy of the reaction. All available rate constant formulas are listed in the Supplement, Sect. S4. The description of reactions is exemplified in Table 1. The optional fourth line, starting with "FACTOR:", sets a passive species to be included as an educt

for the reaction. Passive species are mostly primary components of air, like O_2 or N_2 , which are not included as a separate equation. Instead, their concentration is treated as constant since they are abundant. The available FACTOR species are as well listed in the Supplement, Sect. S3.5.

Cminor supports use of the SMILES notation for species throughout all files (Weininger, 1988). A species can have any name following these rules:

- the name of the species must be in capital letters,
- square brackets enclosing the whole name indicate passive species (e.g., [O₂]),
- an “a” in front of a species indicates presence in droplets, i.e., an aqueous species (e.g., aO₂),
- one or multiple “p” or “m” after the species indicate ions and their charge (e.g., CO₃mm means CO₃^{2−}), and
- allowed special characters in species names are [, (,) , = , + , and *.

The *.dat-file lists molar masses, accommodation coefficients, and diffusion coefficients for gaseous species, as well as molar masses and charges for aqueous species. Moreover, the names of organic peroxy radicals (RO₂ species), gaseous or aqueous, have to be listed here if they are needed as (non-constant) factors in the mechanism. If specified as a factor for a reaction, the respective rate constant will be multiplied with the sum of the current concentrations of all RO₂ species specified in this file.

The *.ini-file contains the species’ initial values, default is zero, and constant emission and deposition rates for gas phase species. Also, the aerosol composition and distribution are specified here.

The *.run-file is a collection of Fortran NAMELISTs to specify the remaining simulation parameters, e.g., temperature, a trigger for adiabatic parcel modeling, output-file name, time-integration method, and more. All NAMELISTs and their variables are listed in the Supplement, Sect. S9, including their default values. Multiple examples are given in the repository Rug et al. (2025).

2.2 Liquid water content

The liquid water content (LWC) describes the mass of condensed water per volume of air. The LWC for a mono-dispersed droplet population is given by $LWC = \rho_w V N$, where $\rho_w = 1000 \text{ kg m}^{-3}$ is the mass density of liquid water, V the droplet volume, and N the number concentration of droplets. Within this simplified model, the time dependent $LWC(t)$ is given by a piecewise linear function mimicking the condensation process, where t is the independent time variable, $b_1, \dots, b_6$ are the bounds of the linear function, and $LWC_{\min}$ and $LWC_{\max}$ the respective minimum and maximum LWC. If the simulation time exceeds the bound

b_6 , the function will be continued periodically, such that $b'_i = b_i + (b_6 - b_1)$ for $i = 1, \dots, 6$, as indicated in Fig. 2. From $LWC(t)$, the droplet wet radii are calculated and used for aqueous chemistry and phase transfer. In the case of an adiabatic parcel, the LWC is determined by the droplet condensation equation, which will be described in Sect. 3.3, below. For prescribed $LWC(t)$, all droplets contain the same amount of liquid water while different amounts of dissolved aerosol are possible.

2.3 Aerosol modes

The composition of aerosol is initialized at the beginning of the simulation. The syntax for the setup of the aerosol modes is described in the Supplement, Sect. S6.3. We assume that the aerosol is completely dissolved in the droplets. The aerosol is assumed to be log-normally distributed with an arbitrary number of modes. Each mode can have a specific composition, resembling, e.g., coexisting sea-salt and ammonium-salt particles, or any internally and/or externally mixed aerosol distribution. The aerosol is assigned to a user-specified number of droplet classes, each representing a number of equal hydrometeors with a specific amount of liquid water and dissolved aerosol mass. The aerosol mass is determined by discretizing the aerosol distribution using logarithmically equidistant bins, one for each droplet class. To constrain the aerosol masses to a reasonable range, the aerosol distribution of each mode is bounded by determining the 0.5 %-radius-quantile and the 99.5 %-mass-quantile. If all modes contain the same aerosol type, this procedure is applied for the sum of these modes, using the respective smallest and largest quantiles of the individual modes. If the modes differ in aerosol type, the procedure is applied to each mode individually. In addition to the initial aerosol mass, general initial values for any aqueous species can be prescribed.

If the adiabatic parcel option is used, aerosol activation is determined by Köhler theory (Sect. 3.3). Alternatively, when the change in LWC is prescribed (Sect. 2.2), only aerosols with dry radii greater than a user-specified threshold are assumed to be activated, and the LWC is distributed equally among the activated particles.

3 Model Equations

All equations solved by *Cminor* are discussed here. Any variable, meteorological parameter or value not defined here is listed in the Supplement, Sect. S11.

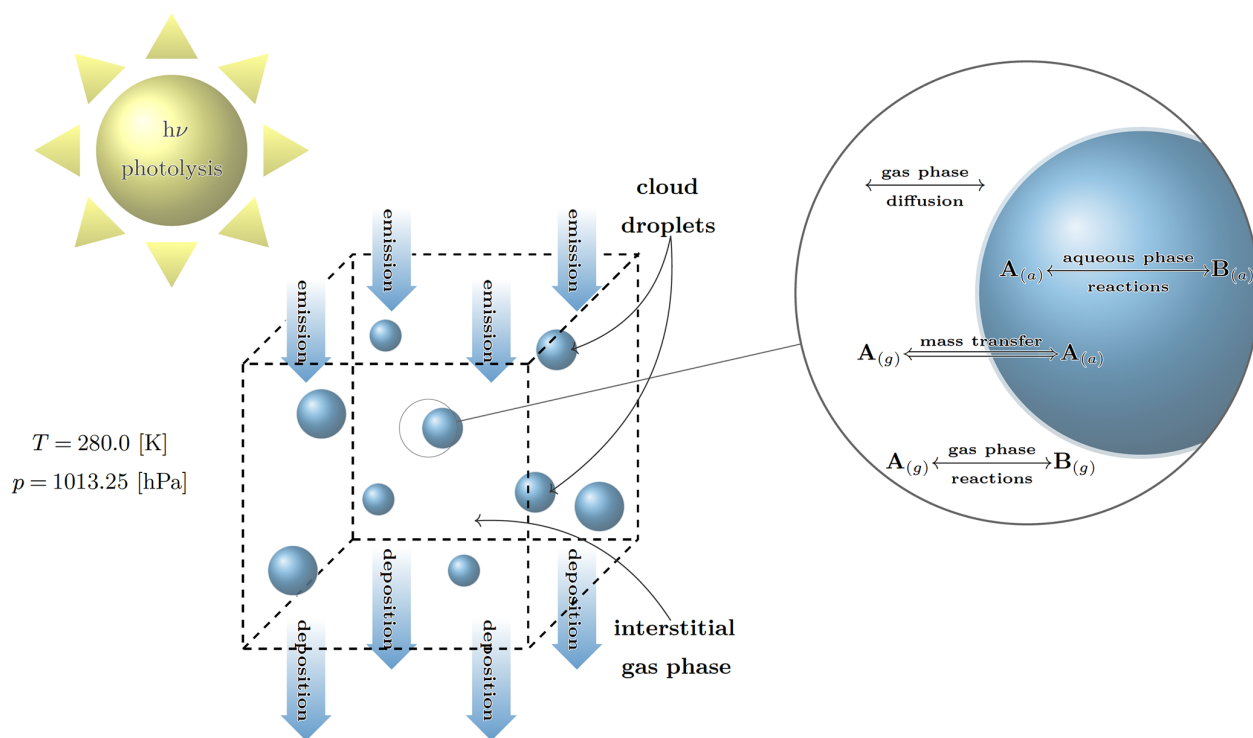


Figure 1. Graphical depiction of Cminor. Initially, the box contains a fixed number of cloud condensation nuclei that can activate to cloud droplets. Next to gas phase reactions, aqueous phase reactions take place within the cloud droplets. The letters $A_{(g)}$ and $B_{(g)}$ denote the concentrations in the gas phase and $A_{(a)}$ and $B_{(a)}$ the concentrations in the aqueous phase, respectively. The magnification shows the processes near and inside of the cloud droplets. The interchange between the gas and aqueous phase is specified according to the Schwartz approach (Schwartz, 1986).

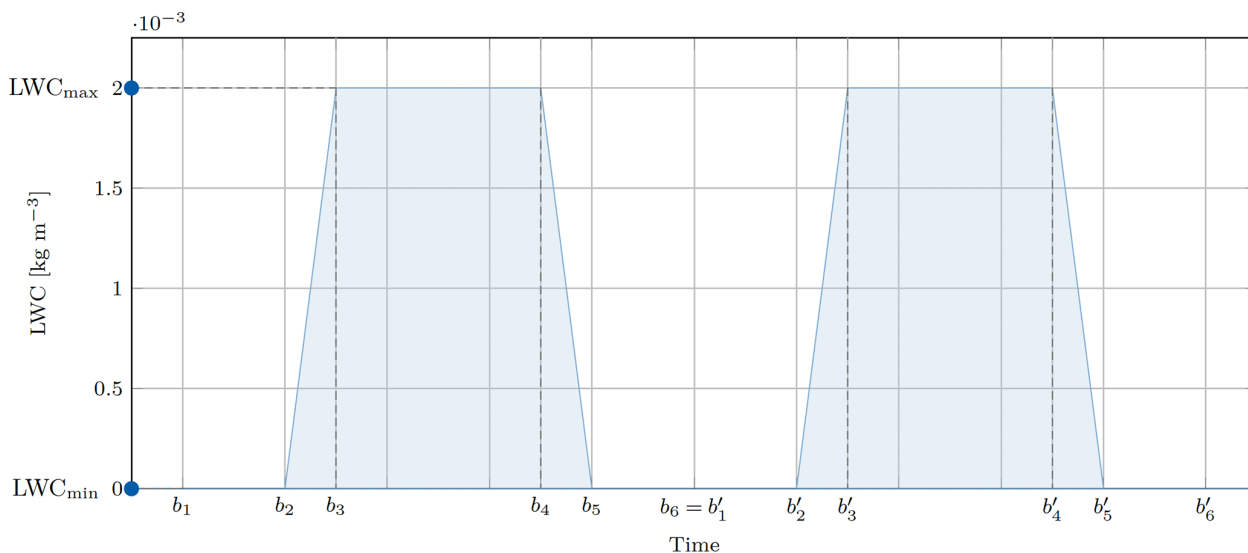


Figure 2. Pseudo liquid water content function $LWC(t)$. All boundaries can be defined by the user.

Table 1. Overview and examples of the structure describing a specific reaction in the *.sys-file. The repertoire of rate constants and the meaning of the rate parameters (A, B, E/R, ...) can be found in the Supplement, Sect. S3.5.

Pattern	Examples
CLASS: *reaction class*	CLASS: GAS
educts = *products*	NO + NO = NO2 + NO2
reaction constant type: *reaction constant parameters*	TEMP1: A: 3.3e-39 E/R: -530.0
FACTOR: \$*factor species*	FACTOR: \$O2
	CLASS: GAS
	O2 = 2.0 O
	PHOTO3: A: 2.643e-10
	CLASS: HENRY
	CO2 = aCO2
	TEMP3: A: 3.1e-2 B: 2423.0
	CLASS: DISS
	[aH2O] = OHm + Hp
	DTEMP: A: 1.8e-16 B: -6800.0 C: 1.3e11
	CLASS: DISS
	aCO2 = HCO3m + Hp
	DTEMP: A: 4.3e-7 B: -913.0 C: 5.6e4
	CLASS: AQUA
	aSO2 + aO3 = HSO4m + aO2 + Hp
	TEMP3: A: 2.4e4 B: 0.0

3.1 Description of chemical kinetic systems

Consider a system of n_R reactions involving n_S chemical species that can be represented in the general form

$$\sum_{j=1}^{n_S} v_{ij}^e S_j \xrightarrow{k_i} \sum_{j=1}^{n_S} v_{ij}^p S_j, \quad i = 1, \dots, n_R, \quad (1)$$

where the stoichiometric coefficients of educts v_{ij}^e and products v_{ij}^p are stored as sparse matrices. S_j denotes the name of the j th chemical species in the mechanism and k_i is the rate constant. Table 2 summarizes the submodules that provide the rate constant k_i for the case of atmospheric chemical mechanisms, a detailed description of all implemented types of rate constants can be seen in the Supplement, Sect. S4.

All functions are implemented in a way that minimizes numerical operations, such as (+, -, *, /, **), and calls to intrinsic functions, e.g., $\exp(\cdot)$, $\log(\cdot)$, $\sin(\cdot)$ and $\cos(\cdot)$, while performing these efficiently. Next to hard coded functions, the user is also able to declare own functions that do not match any of the functions provided, see Supplement, Sect. S4.1, to introduce custom rate constants.

The general reaction kinetics formulation Eq. (1) leads to a set of n_S ODEs expressing mass conservation in the system as

$$\frac{dc_j}{dt} = \sum_{i=1}^{n_R} v_{ij} r_i(\mathbf{c}, T, \phi) + c_j^{\text{emis}}, \quad j = 1, \dots, n_S \quad (2)$$

where dc_j/dt is the change in concentration of species j , $v_{ij} = (v_{ij}^p - v_{ij}^e)$ is the difference between stoichiometric coefficients of products and educts of species j in reaction i , and $r_i(\mathbf{c}, T, \phi)$ is the reaction rate, depending on the concentrations of all reactants in reaction i , expressed by $\mathbf{c}$, temperature T , and possibly other parameters ϕ . In addition, one can specify emission and deposition rates for gas phase species, which are summed up in the vector $\mathbf{c}^{\text{emis}}$. While the emission rates are specified as actual rate in the *.ini-file, deposition is determined by a given rate constant from the *.ini-file. The sink/source term c_j^{emis} is calculated as

$$c_j^{\text{emis}} = c_j^{\text{in}} - c_j^{\text{out}} c_j, \quad j = 1, \dots, n_S,$$

where c_j^{in} is the given emission rate and c_j^{out} the given deposition rate constant. The reaction rate is calculated by

$$r_i(\mathbf{c}, T, \phi) = k_i(T, \phi) \cdot \prod_{j=1}^{n_S} c_j^{v_{ij}^e}, \quad i = 1, \dots, n_R. \quad (3)$$

The rate constants k_i may depend on temperature and other parameters like insolation (introducing a time dependency), or third body collisions (concentration dependency). The temperature plays a major role in combustion, and a respective equation is included in the calculations, as described later. While rate constants with non-temperature dependencies are implemented and can be used, they are not treated as non-autonomous parts of the system in the numerical procedure, beside being evaluated whenever needed. This is a

Table 2. Tabular overview of possible combinations of reaction types used in simulations of atmospheric chemistry.

TYPES		CLASS			
Submodules used to calculate k	Number of functions	GAS	HENRY	DISS	AQUA
Photolysis	6	X			X
Constant	1	X	X		X
Temperature	7	X	X		X
Troe	6	X			
Add Backward Reaction	6			X	
Specials	17	X			X
Custom	*	X	X		X

common approximation and assumes not too rapid changes (e.g., a constant solar zenith angle, which is a good approximation for typical time steps below a few minutes, or third body collisions, that depend on species that are basically passive).

The right-hand side of Eq. (3) represents the law of mass action. In general, several other concentrations influence the production rate of each species, therefore the ODE system is highly coupled.

Because *Cminor* is capable of simulating gas phase combustion mechanisms, the initial value problem needs closure through energy conservation. A perfectly adiabatic constant-volume reactor is considered, as this condition is the one usually occurring when incorporating the solution of detailed chemistry into multidimensional computational fluid dynamics (CFD) codes, where the species and internal energy source terms for each cell are computed as part of a sub-cycling strategy for the whole code (Perini et al., 2012). In particular, the internal energy source term gives a change in the average reactor temperature

$$\frac{dT}{dt} = -\frac{1}{\bar{c}_v(\mathbf{c}, T)\rho} \sum_{j=1}^{n_S} \sum_{i=1}^{n_R} U_j(T) [v_{ij} r_i(\mathbf{c}, T, \phi) + c_j^{\text{emis}}], \quad (4)$$

containing the temperature dependent polynomial fitting function $U_j(T)$ describing the molar internal energy values of species j (Kee et al., 1996; Perini et al., 2012), the mass-averaged specific heat at constant volume $\bar{c}_v(\mathbf{c}, T) = \sum_{j=1}^{n_S} (c_j \partial U_j(T) / \partial T)$, and the average system density ρ .

3.2 Mass transfer between phases

The gaseous-aqueous mass transfer in multi-phase systems has to be defined by a pseudo first-order equilibrium reaction $A_{(g)} \rightleftharpoons A_{(a)}$, where subscript g denotes the concentration of species A in the gas phase and subscript a the concentration of A in the aqueous phase, respectively. The forward and backward rate coefficients are calculated using the accommodation coefficient α , the gas phase diffusion constant D_g , and Henry's law constant. The correct syntax for species names is denoted in the Supplement, Sect. S3.3.

With Henry's law coefficient $A := K_H^\ominus$ in $[\text{mol L}^{-1} \text{atm}^{-1}]$ and $B := -\Delta_{\text{soln}} H / R$ in $[\text{K}]$, where $\Delta_{\text{soln}} H$ = molar enthalpy of dissolution in $[\text{J mol}^{-1}]$ and R is the gas constant. Then, Henry's law constant k_H is computed using the temperature-dependent equation

$$k_H(T) = A \cdot \exp \left[B \cdot (T^{-1} - T_{\text{ref}}^{-1}) \right], \quad (5)$$

where A and B are the parameter given in the third line of the reaction structure (Table 1), T is the actual temperature, and $T_{\text{ref}} = 298.15 \text{ K}$ is a reference temperature. The droplet wet radius r is then used to calculate the mass transfer coefficient according to Schwartz (1986),

$$k_{\text{mt}} = (k_{\text{diff}} \cdot r^2 + k_{\text{acc}} \cdot r)^{-1}, \quad (6)$$

with $k_{\text{diff}} = 1/(3D_g)$, where D_g is the gas phase diffusion coefficient, and $k_{\text{acc}} = 4/(3\hat{\alpha}v)$, where $\hat{\alpha}$ is the mass accommodation coefficient, $v = \sqrt{8RT/(\pi m_{\text{mol}})}$ is the mean molecular speed, and m_{mol} the molar mass of the gas. The reaction rates for gas uptake by droplets is

$$\left. \frac{dA_{(a)}}{dt} \right|_{\text{gas uptake}} = - \left. \frac{dA_{(g)}}{dt} \right|_{\text{gas uptake}} = k_{\text{mt}} \frac{\text{LWC}_i}{\rho_w} [A_{(g)}], \quad (7)$$

and

$$\begin{aligned} \left. \frac{dA_{(g)}}{dt} \right|_{\text{gas escape}} &= - \left. \frac{dA_{(a)}}{dt} \right|_{\text{gas escape}} \\ &= k_{\text{mt}} [A_{(a)}] / (k_H(T) R_{\text{atm}} T) \end{aligned} \quad (8)$$

for the escape of gas from droplets, with $R_{\text{atm}} = 0.0820574 \text{ L atm mol}^{-1} \text{ K}^{-1}$. The division by ρ_w converts the LWC from $[\text{kg m}^{-3}]$ to $[\text{m}^3 \text{m}^{-3}]$, ensuring unit independent rates. The underscore i indicates that if there are multiple droplet classes, only the water volume of a single droplet class is determining the uptake of gases by this class.

3.3 Adiabatic parcel setup and condensational growth

In terms of the liquid water mixing ratio of droplet class i , $q_{l,i}$, the equation for condensational growth of water mass

reads

$$\frac{dq_{l,i}}{dt} = n_i 4\pi r_{d,i}^2 \frac{S - S_{eq,i}}{F_k S_{eq,i} (r_{d,i} + r_\beta) + F_d (r_{d,i} + r_\alpha)}, \quad (9)$$

where $r_{d,i}$ is the radius of the droplets of this class and n_i the corresponding number mixing ratio of droplets.

$$S = \frac{q_v}{q_s(T)}$$

is the ambient relative humidity with q_v and $q_s(p, T)$ being current and saturation vapor mixing ratio, respectively.

$$S_{eq,i} = \frac{n_{w,i}}{n_{w,i} + n_{s,i}} \exp\left(\frac{2\sigma}{r_{d,i} \rho_w R_v T}\right) \quad (10)$$

is the value of the Köhler curve for the particles' current radius, where $n_{w,i}$ and $n_{s,i}$ are the concentrations of liquid water and dissolved species in moles, respectively, σ is the surface tension between liquid water and air, and R_v is the gas constant of water vapor.

$$F_k = \frac{L_v}{k_T T} \left(\frac{L_v}{R_v T} - 1 \right) \quad \text{and} \quad F_d = \frac{R_v T}{D_v e_s(T)} \quad (11)$$

are variables accounting for latent heat release and diffusion of vapor (Lamb and Verlinde, 2011). Lastly,

$$r_\beta = \frac{k_T}{\beta_{H_2O} P} \frac{(2\pi R_a T)^{\frac{1}{2}}}{c_{va} + R_a/2} \quad \text{and} \quad r_\alpha = \left(\frac{2\pi}{R_v T} \right)^{\frac{1}{2}} \frac{D_v}{\alpha_{H_2O}} \quad (12)$$

are physical relaxation parameters (Fukuta and Walter, 1970). To follow current common nomenclature, we changed the names from the original reference, because they refer to what is now called “accommodation coefficient” as “condensation coefficient”. We kept the letter α referring to what is now usually called the accommodation coefficient, as for the gases in Sect. 3.2. By recommendations in Pruppacher and Klett (1978) and Fukuta and Walter (1970), we set $\beta_{H_2O} = 1.0$ and $\alpha_{H_2O} = 0.0415$. These values can be changed to the user's preferences without recompilation, as described in the Supplement, Sect. S9.4.

We have to include a few other equations to account for an adiabatically rising parcel. The water vapor mixing ratio q_v is only affected by condensation and evaporation, i.e., the change in liquid water mixing ratio q_l , such that

$$\frac{dq_v}{dt} = -\frac{dq_l}{dt}$$

with

$$q_l = \sum_{i=1}^{n_D} q_{l,i},$$

where n_D is the number of droplet classes. A user-specified updraft velocity

$$\frac{dz}{dt} = w$$

determines the rate of pressure change, where we assume instantaneous adjustment of the parcel's pressure to the ambient conditions. From the ideal gas law,

$$\rho_a = \frac{p}{R_a T}, \quad (13)$$

where ρ_a is the mass density of the air in the parcel and R_a the specific gas constant of dry air. Differentiation in time gives

$$\frac{d\rho_a}{dt} = -\frac{\rho_a g}{R_a T} \frac{dz}{dt} - \frac{p}{R_a T^2} \frac{dT}{dt}. \quad (14)$$

Here, dT/dt considers the expanding parcel (dry adiabatic lapse rate) and latent heat release following Eq. (9), such that

$$\frac{dT}{dt} = -\frac{g}{c_{pa}} \frac{dz}{dt} + \frac{L_v}{c_{pa}} \frac{dq_l}{dt}. \quad (15)$$

4 Numerical Integration

The variety of time constants inherent to most sets of kinetic equations has long been recognized as a barrier to their efficient integration by traditional methods, as stability requirements dictate an integration step size limited by the smallest time constant. From the numerical point of view, atmospheric chemistry is challenging due to the coexistence of very stable (e.g., CH_4) and very reactive (e.g., $\text{O}(^1\text{D})$) species. Mathematically speaking, this phenomenon is better known as *stiffness*. Therefore, a major task is the integration of stiff systems of ODEs in reasonable time (Sandu and Sander, 2006; Sandu et al., 1997). Since we are dealing with extremely stiff ODE systems, the solution usually relies on robust solvers.

For this purpose, Rosenbrock methods are employed (Hairer et al., 1991). Cminor's implementation allows the user to change the method coefficients via Fortran NAMELIST in order to find the most effective integration scheme for each specific reaction mechanism. A variety of Rosenbrock methods (i.e., sets of coefficients) to choose of comes with the Cminor code already, but the implementation allows the user to easily incorporate different methods or additional solvers.

The novelty in this section lies in the compilation of all of the numerical efforts in one consistent formulation of the Rosenbrock method and describing the system in terms of linear algebra operations. Cminor provides a framework for efficient simulation of pure gas-phase mechanisms, multi-phase mechanisms, polydisperse droplets and aerosol, the droplet condensation equation including chemical effects, the temperature equation for combustion (Wolke and Knoth, 2002; Wolke et al., 2005; Perini et al., 2012), along with a fast mechanism parser, common as well as more complicated rate constants, and the possibility of quick simulations of different scenarios.

4.1 Rosenbrock methods

This section will give a brief introduction to Rosenbrock methods in order to clarify the notation. The starting point is the autonomous initial value problem

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}), \quad t > t_0, \quad \mathbf{y}(t_0) = (y_{1,0}, \dots, y_{n,0})^T. \quad (16)$$

This assumption does not restrict the problem for non-autonomous systems $\dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y})$, because it can easily be transformed by treating the time variable t as dependent, such that $i = 1$. The ODE system contains only first order derivatives $\mathrm{d}\mathbf{y}/\mathrm{d}t$, which are usually non-linear functions $\mathbf{f}$ of concentration. Most stiff solvers benefit from using implicit formulations such as Rosenbrock methods which arise from diagonal-implicit Runge-Kutta (DIRK) methods. The simplest scheme of this kind is the backward Euler method

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}[\mathbf{y}_{n+1}], \quad (17)$$

where h is the step size and $\mathbf{y}_n$ the current state vector. To avoid implicit systems and iterative solvers, Eq. (17) can be linearized, leading to

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}[\mathbf{y}_n] + h[\mathbf{y}_{n+1} - \mathbf{y}_n]\mathbf{f}'[\mathbf{y}_n]. \quad (18)$$

This is commonly referred to as a linear-implicit method. It can be seen as restricting the iterative solver to one step of Newton's method, see Hairer et al. (1991) for elaborate derivations. To achieve a higher order of consistency, Rosenbrock (1963) proposed to generalize this linearly implicit approach to methods using more stages. The general form of a non-autonomous s -stage Rosenbrock method produces a next step solution as

$$\begin{aligned} \mathbf{y}_{n+1} &= \mathbf{y}_n + \sum_{i=1}^s b_i \mathbf{k}^{(i)}, \\ \text{with } \mathbf{k}^{(i)} &= h\mathbf{f}\left[t_n + \alpha_i h, \mathbf{y}_n + \sum_{j=1}^{i-1} \alpha_{ij} \mathbf{k}^{(j)}\right] \\ &\quad + \gamma_i h^2 \mathbf{f}_t[t_n, \mathbf{y}_n] + h\mathbf{J}[t_n, \mathbf{y}_n] \cdot \sum_{j=1}^i \gamma_{ij} \mathbf{k}^{(j)}, \end{aligned} \quad (19)$$

where the coefficients b_i , α_{ij} , and γ_{ij} define the particular method, and are chosen such that a certain accuracy and stability is granted (Zhang et al., 2011). $\mathbf{J}$ refers to the Jacobian of the system. For linear-implicit methods the method coefficients have the additional feature that $\alpha_{ij} = 0$ for all $j \geq i$, $\gamma_{ij} = 0$ for all $j \geq i + 1$, and

$$\alpha_i = \sum_{j=1}^{i-1} \alpha_{ij}, \quad \gamma_i = \sum_{j=1}^i \gamma_{ij}. \quad (20)$$

The partial derivative $\mathbf{f}_t$ in Eq. (19) is equal to zero for autonomous systems. Reducing the computation cost is crucial for large systems because each time step requires not only $s - 1$ matrix-vector products, but also s matrix decompositions. It is therefore advantageous to choose the same diagonal elements for $\gamma_{ii} = \gamma$, $i = 1, \dots, s$. Thus, only one decomposition is needed per time step. In order to avoid the

matrix-vector products on the right-hand side of Eq. (19), the stage vectors $\mathbf{k}^{(i)}$ are substituted by $\mathbf{u}^{(i)} = \sum_{j=1}^i \gamma_{ij} \mathbf{k}^{(j)}$ for $i = 1, \dots, s$ (see Hairer et al., 1991). The modified Rosenbrock method with s internal stages is then given by

$$\begin{aligned} \mathbf{y}_{n+1} &= \mathbf{y}_n + \sum_{j=1}^s m_j \mathbf{u}^{(j)}, \\ \text{with } (\mathbf{I} - h\gamma \mathbf{J})\mathbf{u}^{(i)} &= h\mathbf{f}\left(t_n + \alpha_i h, \mathbf{y}_n + \sum_{j=1}^{i-1} a_{ij} \mathbf{u}^{(j)}\right) \\ &\quad + \gamma_i h^2 \mathbf{f}_t(t_n, \mathbf{y}_n) + \sum_{j=1}^{i-1} d_{ij} \mathbf{u}^{(j)}, \quad i = 1, \dots, s, \end{aligned} \quad (21)$$

where $\mathbf{I}$ is the identity matrix, and $\Gamma = (\gamma_{ii})$, $(a_{ij}) = (\alpha_{ij})\Gamma^{-1}$, $(d_{ij}) = \text{diag}(\gamma_{11}^{-1}, \dots, \gamma_{ss}^{-1}) - \Gamma^{-1}$, $(m_j) = (b_j)\Gamma^{-1}$, and γ the method parameters.

4.2 Adaptive step size control

Usually, the purpose of adaptive step size control is to achieve some predetermined accuracy in the solution with minimum computational effort. The use of Rosenbrock methods allows us to calculate the local error of the current state vector $\mathbf{y}$. This task can be carried out by computing a second state vector $\hat{\mathbf{y}}$, which is given by an embedded formula where the order of the embedded method $\hat{p}$ is usually lower than order p of the actual formula ($\hat{p} = p - 1$). The step size control is implemented by estimating the local error $\hat{e}_j = \mathbf{y}_j - \hat{\mathbf{y}}_j$, and is then scaled by the denominator $\text{scal}_j = \text{tol}_j^A + \max\{|\mathbf{y}_j|, |\hat{\mathbf{y}}_j|\} \cdot \text{tol}^R$, where tol_j^A is the absolute and tol^R the relative tolerance given by the user. Note that tol_j^A can be set for each species separately, where tol^R is equal for all species. *Cminor*'s implementation allows to specify two different absolute tolerances, for gaseous and aqueous species. The actual measurement of the error induced by the Rosenbrock method is computed either via the maximum norm or Euclidean norm, denoted by $\|\cdot\|$. The new step size is then calculated by the following expression:

$$h^{\text{new}} = \begin{cases} \max\{h_{\min}, h^{\text{old}} \max\{0.1, 0.8 \|\hat{e}\|^{-\delta}\}\}, & \text{if } \|\hat{e}\| > 1, \\ \min\{h_{\max}, 0.8 h^{\text{old}} \|\hat{e}\|^{-\delta}, 2 h^{\text{old}}\}, & \text{otherwise.} \end{cases} \quad (22)$$

The exponent δ is defined as $\delta = 1/(p + 1)$. A new step size h^{new} is estimated that will yield an error of tol^R on the next step or the next try at taking this step, as the case may be. The step size is multiplied by 0.8 to avoid failures. In addition, the step size h^{new} is bounded by user-given values for a minimal and a maximal step size, $h_{\min}$ and $h_{\max}$, respectively.

4.3 Analytical Jacobian matrix formulation

The initial value problem is described by Eqs. (2) and (4), where the dimension of the ODE system is $n = n_S + 1$, with $\dot{\mathbf{c}} \in \mathbb{R}^{n_S}$ and $\dot{T} \in \mathbb{R}$. The ODE system can then be represented in a very short manner as matrix-vector products,

$$\mathbf{y}' = f(\mathbf{c}, T) = \begin{pmatrix} \dot{\mathbf{c}} \\ \dot{T} \end{pmatrix} = \begin{pmatrix} \mathbf{v}^T \mathbf{r} + \mathbf{c}^{\text{emis}} \\ -\frac{1}{\bar{c}_v \rho} U^T [\mathbf{v}^T \mathbf{r} + \mathbf{c}^{\text{emis}}] \end{pmatrix},$$

$$\mathbf{y}(t_0) = (c_1^0, \dots, c_{n_S}^0, T^0)^T. \quad (23)$$

The use of Rosenbrock methods Eq. (21) requires the calculation of the Jacobian matrix of the ODE system Eq. (23),

$$J = \frac{\partial f(\mathbf{c}, T)}{\partial(\mathbf{c}, T)} = \begin{bmatrix} \ddots & & \ddots & \vdots \\ & \frac{\partial \dot{c}_l}{\partial c_j} & & \frac{\partial \dot{c}_l}{\partial T} \\ \ddots & & \ddots & \vdots \\ \dots & \frac{\partial \dot{T}}{\partial c_j} & \dots & \frac{\partial \dot{T}}{\partial T} \end{bmatrix},$$

$$j = 1, \dots, n_S \quad l = 1, \dots, n_S, \quad (24)$$

where the first block $\partial \dot{c}_l / \partial c_j$ is a square sparse matrix of dimension n_S containing the derivatives of the change of c_l with respect to c_j . The simplification of this part of the Jacobian matrix is related to the assumption that $k = k(T)$ in Eq. (3) depends only on temperature and not on other variables, e.g., time, pressure, or concentrations of catalytic species. This simplification is justified by the assumed slow change of these parameters, such that

$$J_{cc} = \frac{\partial}{\partial c_j} \left(\frac{dc_l}{dt} \right) = \sum_{i=1}^{n_R} \left[v_{il} \frac{\partial r_i}{\partial c_j} \right] = \mathbf{v}^T D_r \mathbf{v}^e D_c^{-1},$$

$$j = 1, \dots, n_S, \quad l = 1, \dots, n_S, \quad (25)$$

where $D_r = \text{diag}(r_1, \dots, r_{n_R})$ and $D_c^{-1} = \text{diag}(1/c_1, \dots, 1/c_{n_S})$ are diagonal matrices containing the reactions rates and the inverse concentrations, respectively. The second block $\partial \dot{c}_l / \partial T$ consists of a full column vector and represents the derivatives of change in c_l with respect to T . These are expressed as

$$J_{cT} = \frac{\partial}{\partial T} \left(\frac{dc_l}{dt} \right) = \sum_{i=1}^{n_R} \left[v_{il} \frac{\partial r_i}{\partial T} \right] = \mathbf{v}^T D_r \mathbf{K}_i,$$

$$l = 1, \dots, n_S, \quad (26)$$

where $\mathbf{K}_i = \partial k_i / \partial T \cdot k_i^{-1}$ contains the derivatives of the i th rate constants with respect to T , multiplied by the inverse rate constant of reaction i . As mentioned before, k_i is assumed to be only temperature-dependent. The third part of the Jacobian matrix $\partial \dot{T} / \partial c_j$ is a full row vector and contains the derivatives of change in T with respect to c_j . It is expressed

as

$$J_{Tc} = \frac{\partial}{\partial c_j} \left(\frac{dT}{dt} \right) = -\frac{1}{\bar{c}_v \rho} \left[C_{v,j} \dot{T} + \sum_{l=1}^{n_S} U_l \frac{\partial \dot{c}_l}{\partial c_j} \right],$$

$$j = 1, \dots, n_S$$

$$= -\frac{1}{\bar{c}_v \rho} \left[\mathbf{C}_v \dot{T} + U^T \mathbf{v}^T D_r \mathbf{v}^e D_c^{-1} \right], \quad (27)$$

where $C_{v,j} = \partial U_j / \partial T$ is the constant volume specific heat of the j th species. The first item in the Jacobian matrix, i.e., the entry in position $(n_S + 1, n_S + 1)$, is the derivative of the average mixture temperature change rate with respect to the system temperature itself,

$$J_{TT} = \frac{\partial}{\partial T} \left(\frac{dT}{dt} \right) =$$

$$- \frac{1}{\bar{c}_v \rho} \left\{ \rho \dot{T} \frac{\partial \bar{c}_v}{\partial T} + \sum_{l=1}^{n_S} \left[C_{v,l} \dot{c}_l + U_l \frac{\partial \dot{c}_l}{\partial T} \right] \right\} =$$

$$- \frac{1}{\bar{c}_v \rho} \left[\rho \dot{T} \frac{\partial \bar{c}_v}{\partial T} + \mathbf{C}_v \dot{\mathbf{c}} + U^T \mathbf{v}^T D_r \mathbf{K} \right]. \quad (28)$$

4.4 Integration scheme for box-model chemistry

In this section, the resulting numerical method using the classical linear algebra formulation is shown. The ODE system Eq. (23) is assumed to be autonomous. Substituting the values of J , Eq. (24), into the Rosenbrock method Eq. (21) yields the classic linear algebra formulation for computing a new state vector. Note that the partial derivative f_t , see Sect. 4.1, will be dropped. Also, still, for box-model atmospheric chemistry systems, the equation for T will be dropped. Consequently, for atmospheric systems, no dependencies of the rate constants k_i are considered in the numerical procedure, i.e., the system is assumed to be fully autonomous from the numerical point of view. Still, rate constants of photolytic reactions are updated during integration, according to current time and the specified location, i.e., zenith angle of the sun. Accordingly, this gives

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \sum_{i=1}^s m_i \begin{bmatrix} \mathbf{u}^{(i)} \\ \tilde{\mathbf{u}}^{(i)} \end{bmatrix}, \text{ with}$$

$$\begin{bmatrix} I - h\gamma \mathbf{v}^T D_r \mathbf{v}^e D_c^{-1} & -h\gamma \mathbf{v}^T D_r \mathbf{K} \\ -\frac{h\gamma}{\bar{c}_v \rho} (\mathbf{C}_v \dot{T} + U^T J_{cc}) & 1 + \frac{h\gamma}{\bar{c}_v \rho} \left(\rho \frac{\partial \bar{c}_v}{\partial T} \dot{T} + \mathbf{C}_v \dot{\mathbf{c}} + U^T J_{cT} \right) \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{u}^{(i)} \\ \tilde{\mathbf{u}}^{(i)} \end{bmatrix} = h \begin{bmatrix} \mathbf{v}^T \mathbf{r}^{(i)} + \mathbf{c}^{\text{emis}} \\ -\frac{1}{\bar{c}_v^{(i)} \rho} (U^{(i)})^T (\mathbf{v}^T \mathbf{r}^{(i)} + \mathbf{c}^{\text{emis}}) \end{bmatrix}$$

$$+ \sum_{j=1}^{i-1} d_{ij} \begin{bmatrix} \mathbf{u}^{(j)} \\ \tilde{\mathbf{u}}^{(j)} \end{bmatrix}, \quad i = 1, \dots, s. \quad (29)$$

The values of both diagonal matrices D_r and D_c are calculated at the beginning of each time step (i.e., at $\mathbf{y}^{(n)}$), meaning that they do not form intermediates. In contrast, the reaction rates r_i , the mass-averaged specific heat at constant volume $\bar{c}_{v,i}$, and the internal energy values U_i^T on the right-hand-side form the intermediate solution for each stage

$$r^{(i)} = r^{(i)} \left(t_n + \alpha_i h, \mathbf{y}_n + \sum_{j=1}^{i-1} a_{ij} \begin{bmatrix} \mathbf{u}^{(i)} \\ \tilde{\mathbf{u}}^{(i)} \end{bmatrix} \right),$$

$$\tilde{\mathbf{c}}_v^{(i)} = \tilde{\mathbf{c}}_v^{(i)} \left(\mathbf{y}_n + \sum_{j=1}^{i-1} a_{ij} \begin{bmatrix} \mathbf{u}^{(i)} \\ \tilde{\mathbf{u}}^{(i)} \end{bmatrix} \right),$$

$$\mathbf{U}^{(i)} = \mathbf{U}^{(i)} \left(T + \sum_{j=1}^{i-1} a_{ij} \tilde{\mathbf{u}}^{(i)} \right)$$

Note that the value for T is located in the last entry in the state vector $\mathbf{y}^{(n)}$, see Eq. (23).

4.5 Direct sparse linear solver

In this work, direct linear solvers are considered to calculate the solution of the linear systems in the integration procedure. For reasons of simplicity, the linear systems are denoted as $\mathbf{A}\mathbf{x} = \mathbf{b}$ in this section. A direct solver strategy requires the factorization of the coefficient matrix $\mathbf{A}$ into a lower and an upper triangular matrix $\mathbf{L}$ and $\mathbf{U}$, where $\mathbf{A} = \mathbf{L} \cdot \mathbf{U}$. The implemented sparse solver consists of three parts, (i) the symbolic phase, where the actual factor matrix $\mathbf{LU}(\mathbf{A})$ is built and stored as separate sparse matrix as it has the same structure (i.e., non-zero pattern) in every time step, (ii) the numerical factorization within the Rosenbrock method, and (iii) a solution phase for the triangular systems. While factorizing sparse matrices, generally fill-in occurs and the factors of $\mathbf{LU}(\mathbf{A})$ become more dense than the original matrix $\mathbf{A}$. However, to take advantage of the sparseness of the coefficient matrix $\mathbf{A}$, the equations must be arranged in special order. To find a good ordering for the sparse matrix $\mathbf{A}$, we have to determine a permutation P that minimizes the fill-in in the factors $\mathbf{L}$ and $\mathbf{U}$. Since the problem of finding a permutation P that minimizes the amount of fill-in can be reduced to an NP-complete task (Yannakakis, 1981), finding the optimal solution becomes computationally infeasible for larger systems due to the exponential growth of required computational resources. Yet, *Cminor* uses the minimum-degree ordering heuristic proposed by Markowitz (1957) in a symmetric manner, meaning the pivot always lies on the diagonal to preserve existing stability. Thus, if rows i and j are swapped, then columns i and j are swapped as well. The (symbolic) elimination process is denoted in Algorithm 1. $\mathbf{A}$ is assumed to be unsymmetric and regular, which does generally not ensure that a solution exists. Nevertheless, numerical pivoting is not implemented in this approach. The numerical phase (ii) takes place once at each time step within the Rosenbrock method. By using the Markowitz (1957) strategy, the growth of the number of non-zero elements is roughly bounded by the factor 2.

Algorithm 1 Gaussian Elimination with a Minimum-Degree heuristic: Minimizing fill-in by selecting pivots with the smallest row-column non-zero product, guided by a restriction vector *Restr* to prioritize certain species for permutation to the final rows and columns.

```

1: procedure Eliminate_MD( $A_0$ )
2: for ( $i = 1 : \dim(A_0)$ ) do
3:   # count non-zero elements of every row and column
   in sub-matrix  $\mathbf{A}_i$ 
4:   # choose pivot element:  $j_p =$ 
   min{[row( $k$ ) - 1] · [col( $k$ ) - 1]}
   for
   {  $k \geq i, k \notin \text{Restr}$  if  $\exists k \geq i : k \notin \text{Restr}$ 
      $k \geq i$  else
5:   # swap row and column  $i \leftrightarrow j_p$ 
6:   # elimination of column  $j_p$  in sub-matrix  $\mathbf{A}_i$ 
7: end for
8: end procedure

```

4.6 Vectorized LU decomposition

For multi-phase systems in a box-model, droplets of the same size will behave equally as long as the initial concentrations are equal in each droplet. If this is not the case, as is for specifying log-normally distributed aerosol, each aqueous species has to be considered once for each droplet class. A droplet class refers to a number of droplets of equal size and chemical composition. This corresponds to the concept of super-droplets commonly used in cloud microphysics (Shima et al., 2009; Hoffmann et al., 2015).

When considering multiple droplet classes, the number of equations in atmospheric systems grows from

$$n_{\text{Gas}} + n_{\text{Henry}} + n_{\text{Aq}} \quad \text{to} \quad n_{\text{Gas}} + n_{\text{Henry}} + n_D \cdot n_{\text{Aq}},$$

where n_{Gas} is the number of purely gaseous species, i.e., not dissolving, n_{Henry} the number of gaseous species that may dissolve in droplets, n_{Aq} the number of aqueous species in the mechanism and n_D the number of droplet classes. This is a significant increase for mechanisms with a prominent aqueous phase. For instance, a number of 50 droplet classes is common in super-droplet models. Since the equations for each droplet class are equal in structure, as the same reactions occur in every droplet class, the matrix $\mathbf{A}$ has a specific, repeating pattern. Examples are shown in Fig. 3, displaying the sparsity pattern of the iteration matrix of a sulfur oxidation mechanism in clouds for one (a) and five droplet classes (b, c). The repeating pattern, in turn, can be represented by considering the matrix for one droplet class, with vectors representing the entries of aqueous species and mass transfer terms.

By doing so, the (symbolic) matrix and its (symbolic) LU decomposition stay the same for any number of droplet classes. Therefore, only the solution phase (iii) changes from scalar operations to vector operations. These are fast and minimize indirect accesses to the respective arrays, which

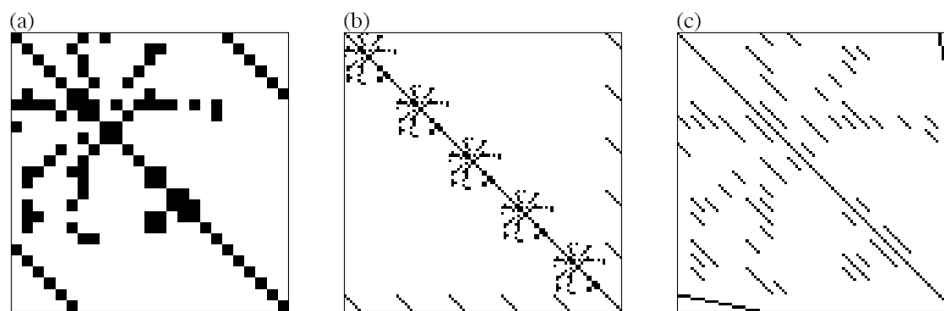


Figure 3. Sparsity patterns: iteration matrix of sulphate oxidation mechanism in clouds; one droplet class (a), five droplet classes ordered by droplet classes (b), five droplet classes ordered by species (c).

make up a major portion of the total solution process, and therefore also of the total simulation. Of course, since there are five droplet classes in this example, the matrix to be decomposed is (b) or (c), in Fig. 3, which is just a matter of ordering.

In the following, we explore whether the decomposition can be determined using only (a). As shown in (c), the vector structure can be represented by arranging all entries corresponding to aqueous species into diagonal matrix blocks. If every zero entry modification (fill-in), i.e., change of a zero entry to a non-zero entry while decomposing the matrix, is part of a new diagonal matrix block, then it is possible to work with (a) to find the decomposition of (c). Consequently, when a diagonal block matrix transforms into a fully populated matrix during the decomposition process, the vector representation becomes impractical. In such iteration matrices, the vectorized decomposition approach is no longer viable. Also, any single entry appearing off-diagonal in a diagonal matrix block prevents the vector representation of the matrix as its position would have to be stored.

Every row in the iteration matrix corresponds to an equation of a specific species, while a column resembles the influence of a specific species on the other species. This means, every row and every column relate to one specific species. By permuting the rows and columns representing Henry species, i.e., species that are gaseous but may dissolve in droplets, to the last rows and columns, the vector structure will be preserved during decomposition. Ordered like this, the iteration matrix for multi-phase mechanisms exhibits the four regions

$$\mathbf{A} = \begin{pmatrix} \text{in-phase reactions} & \begin{matrix} \text{Henry species} \\ \rightarrow \text{aqueous or purely gaseous species} \end{matrix} \\ \begin{matrix} \text{aqueous or purely gaseous species} \\ \rightarrow \text{Henry species} \end{matrix} & \begin{matrix} \text{Henry species} \\ \rightarrow \text{Henry species} \end{matrix} \end{pmatrix},$$

where purely gaseous species means not dissolving in droplets.

In the following, the preservation of the vector structure by this ordering is reasoned by an induction approach. It is shown that, having a matrix which can be represented with vector entries and which is ordered in the proposed way, a new fill-in will always occur vector-wise, preserving the

possibility to represent the matrix with vector entries. The following Algorithm 2 generates the LU decomposition of a matrix $\mathbf{A}$.

Algorithm 2 LU Decomposition: Generating a lower triangular and an upper triangular matrix, whose product yields $\mathbf{A}$. The matrices $\mathbf{L}$ and $\mathbf{U}$ are stored in the upper and lower part of the over-written matrix $\mathbf{A}$.

```

1: for i = 1 : nspc do
2:   for j = 1 : i-1 do
3:     A[i, j] = A[i, j] / A[j, j]
4:     for k = j+1 : nspc do
5:       A[i, k] = A[i, k] - A[i, j] · A[j, k]
6:     end for
7:   end for
8: end for

```

It can be seen that a necessary condition for fill-in at entry $\mathbf{A}[i, k]$ is

$$\exists j > 0, j < i, j < k : \mathbf{A}[i, j] \cdot \mathbf{A}[j, k] \neq 0. \quad (30)$$

With this necessary condition, the logical path of the following reasoning is

fill-in $\rightarrow$ Eq. (30) $\rightarrow$ appropriate fill-in (vector-wise where needed).

The bottom left and upper right regions of $\mathbf{A}$ can in any case be represented by vector entries as the vector representation only prevents storing zeros in the upper left region. The lower right region has scalar entries anyhow, representing gaseous reactions of dissolving species (not shown in Fig. 3). Therefore, it is sufficient to show that if fill-in occurs for the case of i and k both representing aqueous species, fill-in in the form of a diagonal matrix block follows.

Let $i, j, k > 0$ be a triple of indices, for which $\mathbf{A}[i, j] \cdot \mathbf{A}[j, k] \neq 0$ holds for a given matrix $\mathbf{A}$, where the gaseous, dissolving species are permuted to the last rows and columns. If $i, k \leq n_{\text{Aq}} \cdot n_{\text{D}} + n_{\text{Gas}}$, the entry $\mathbf{A}[i, k]$ is in the region of either diagonal matrix entries, i.e., aqueous species, or single-value entries, i.e., purely gaseous species. As mentioned above, this is the only important case for which preser-

vation of the vector structure of $\mathbf{A}$ needs to be shown. Any entry $\mathbf{A}[i, j] \neq 0$ comes from a reaction of species j to species i (or an equivalent previous fill-in). Since there are no reactions between purely gaseous species and aqueous species, it follows that i, j , and k all belong to the same type of species, either purely gaseous or aqueous. It also follows that fill-in can never occur between aqueous and purely gaseous species. If they are gaseous, no vector structure has to be preserved, in this case the fill-in is irrelevant. If they are aqueous, i, j , and k can be written as

$$\begin{aligned} i &= n_{\text{Gas}<i} + (S_i - 1) \cdot n_D + d, \\ j &= n_{\text{Gas}<j} + (S_j - 1) \cdot n_D + d, \\ k &= n_{\text{Gas}<k} + (S_k - 1) \cdot n_D + d, \end{aligned} \quad (31)$$

where $n_{\text{Gas}<i}$ is the number of purely gaseous species that were permuted before species i , $S_i \in \{1, \dots, n_{\text{Aq}}\}$ is the aqueous species number disregarding droplet classes, and $d \in \{1, \dots, n_D\}$ is the droplet class in which species i is dissolved. The analogous nomenclature is used for the variables containing j and k . Note that d is the same for i, j , and k . Before fill-in, the matrix is assumed to have proper vector entries, and we re-iterate that there is no reaction from one droplet class to another. Thus, d is the same for i, j , and k . Now, w.l.o.g., it is assumed that $d = 1$. For other values of d , analogous calculations yield the same result. The vector structure is preserved if

$$\begin{aligned} \forall l_1, l_2 \in \{0, \dots, n_D - 1\}, l_1 \neq l_2, \forall j^* > 0, j^* < i^* \\ := i + l_1, j^* < k^* := k + l_2 : \mathbf{A}[i^*, j^*] \cdot \mathbf{A}[j^*, k^*] = 0. \end{aligned} \quad (32)$$

Under this condition, the entry $\mathbf{A}[i^*, k^*]$ will not undergo fill-in, i.e., every off-diagonal entry in the submatrix

$$\mathbf{A}[i : i + n_D - 1, k : k + n_D - 1]$$

stays equal to zero. Then, the possible fill-in is a diagonal matrix, which is what needs to be shown. The updated representation of Eq. (31), which is

$$\begin{aligned} i^* &= i + l_1 = n_{\text{Gas}<i} + (S_i - 1) \cdot n_D + d + l_1, \\ k^* &= k + l_2 = n_{\text{Gas}<k} + (S_k - 1) \cdot n_D + d + l_2, \end{aligned}$$

helps recognizing that, since $l_1 \neq l_2$ by Eq. (32), species i^* and k^* now belong to separate droplet classes, namely $d + l_1$ and $d + l_2$, respectively. Since the matrix is assumed to have proper vector entries before the examined fill-in, it can only have non-zero values at entries which belong to the same droplet class. Any $j^* > 0$ with $j^* < i^*$ and $j^* < k^*$ may correspond to an aqueous species of a specific droplet class, where either $\mathbf{A}[i^*, j^*]$ or $\mathbf{A}[j^*, k^*]$ or both are equal to zero because of the aforementioned reason, or it may correspond to a purely gaseous species, where the respective matrix entries are zero, too, because purely gaseous species do not interact with the aqueous species i^* and k^* . Thus, Eq. (32) holds.

Hence, the vector structure of $\mathbf{A}$, e.g., the rightmost matrix in Fig. 3, is preserved during decomposition for the proposed ordering, which is to permute the gaseous dissolving species to the last rows and columns. In opposition to this, if i represents a Henry species, which is permuted between aqueous and purely gaseous ones, Eq. (32) is violated for $l_1 = 1$, $l_2 = 0$ and $j^* = j$. In this case, the vector structure is not preserved.

4.7 Numerical treatment of parcel equations

For the Jacobian J , we need derivatives of Eqs. (9), (14), and (15) with respect to the rest of the variables. To minimize floating point operations, we neglect these derivatives, resulting in a sort of explicit scheme for these equations, which we assume is reasonable as they behave not as stiff as the chemical species' equations. The only exception is for the condensational equation, where we calculate the derivative with respect to the water masses, where $i, j \in \{1, \dots, n_D\}$ and n_D is again the number of droplet classes. Here,

$$\begin{aligned} \frac{\partial}{\partial q_{1,j}} \left(\frac{dq_{1,i}}{dt} \right) &= \frac{\partial}{\partial q_{1,j}} [D_i \cdot (S - S_{\text{eq},i})] \\ &= \frac{\partial D_i}{\partial q_{1,j}} \cdot (S - S_{\text{eq},i}) + D_i \cdot \frac{\partial (S - S_{\text{eq}})}{\partial q_{1,j}} \\ &\approx D_i \cdot \frac{\partial (S - S_{\text{eq},i})}{\partial q_{1,j}}. \end{aligned} \quad (33)$$

Since supersaturations in liquid clouds are small, we neglect $(\partial D_i / \partial q_{1,j}) \cdot (S - S_{\text{eq},i})$, and D_i summarizes all remaining factors in Eq. (9). The remaining term splits up in the derivatives

$$\frac{\partial S}{\partial q_{1,j}} \text{ and } -\frac{\partial S_{\text{eq},i}}{\partial q_{1,j}}. \quad (34)$$

The first term is zero since S has no direct relation to the liquid water mixing ratio. For the second term, we omit the index indicating the droplet class because for $i \neq j$ the term is zero. Also, we again summarize factors to R and K , the Raoult and the Kelvin term, respectively. Thus,

$$\begin{aligned} \frac{\partial}{\partial q_1} S_{\text{eq}} &= \frac{\partial}{\partial q_1} (R \cdot K) \\ &= \frac{\partial}{\partial q_1} \left[\frac{q_1 / M_w}{q_1 / M_w + n_s} \cdot \exp \left(\frac{a}{r_d} \right) \right] \\ &= R' \cdot K + R \cdot K' \end{aligned} \quad (35)$$

The derivative of the Raoult term with respect to water mass is easily calculated as

$$R' = \frac{n_s}{M_w (q_1 / M_w + n_s)^2}. \quad (36)$$

For the derivative of the Kelvin term, we note that the radius r_d might indicate the radius of a droplet not only consisting

of water, but also soluble and insoluble material. However, assuming ideal solution behavior for droplets of a size larger than a few nanometers, the volume of a single droplet might be calculated as $V = V_w + V_{\text{rest}}$ with $V_w = q_l/(\rho_w n)$, n is the number of droplets (in the droplet class) per kg of air, and V_{rest} the volume of dissolved species. Doing so, we get

$$\begin{aligned} K' &= \frac{\partial \exp\left(\frac{a}{r_d}\right)}{\partial r_d} \cdot \frac{\partial r_d}{\partial V_w} \cdot \frac{\partial V_w}{\partial q_l} \\ &= \left[-\frac{a}{r_d^2} \exp\left(\frac{a}{r_d}\right) \right] \cdot \frac{1}{4\pi} \left[\frac{3}{4\pi} (V_w + V_{\text{rest}}) \right]^{-\frac{2}{3}} \cdot \frac{1}{\rho_w n} \quad (37) \\ &= -\frac{a}{4\pi \rho_w n r_d^4} \exp\left(\frac{a}{r_d}\right). \end{aligned}$$

The resulting classical linear algebra scheme is

$$\begin{aligned} \mathbf{y}_{n+1} &= \mathbf{y}_n + \sum_{i=1}^s m_i \begin{bmatrix} \mathbf{u}^{(i)} \\ \tilde{\mathbf{u}}^{(i)} \end{bmatrix}, \text{ with} \\ \begin{bmatrix} I - h\gamma v^T D_r v^e D_c^{-1} & \mathbf{0} \\ \mathbf{0} & I - h\gamma J_{\text{parcel}} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{(i)} \\ \tilde{\mathbf{u}}^{(i)} \end{bmatrix} &= \\ h \begin{bmatrix} v^T \mathbf{r}^{(i)} + \mathbf{c}^{\text{emis}} + \mathbf{c}^{\frac{\rho}{\rho}} \\ \dot{\mathbf{q}}_1^{(i)} \\ -\frac{g}{c_{\text{pa}}} w + \frac{L_v}{c_{\text{pa}}} \dot{\mathbf{q}}_1^{(i)} \\ -\dot{\mathbf{q}}_1^{(i)} \\ -\frac{\rho_a g}{R_a T} w - \frac{p}{R_a T^2} \left(-\frac{g}{c_{\text{pa}}} w + \frac{L_v}{c_{\text{pa}}} \dot{\mathbf{q}}_1^{(i)} \right) \\ w \end{bmatrix} &+ \sum_{j=1}^{i-1} d_{ij} \begin{bmatrix} \mathbf{u}^{(j)} \\ \tilde{\mathbf{u}}^{(j)} \end{bmatrix}, \quad i = 1, \dots, s, \quad (38) \end{aligned}$$

where $\tilde{\mathbf{u}}^{(j)}$ denotes the j th intermediate of the transformed parcel variables,

$$\dot{\mathbf{q}}_1^{(i)} = \text{diag}(\mathbf{D}^{(i)})(S^{(i)} \cdot \mathbf{1} - \mathbf{S}_{\text{eq}}^{(i)}) \quad (39)$$

and

$$J_{\text{parcel}} = \begin{pmatrix} \mathbf{J}_{mm} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\mathbf{J}_{mm} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad (40)$$

with

$$\mathbf{J}_{mm} = -\text{diag}(\mathbf{D}) [\text{diag}(\mathbf{R}')\mathbf{K} + \text{diag}(\mathbf{R})\mathbf{K}']. \quad (41)$$

The bold letters indicate vectors containing values of all droplet classes. This represents an approximate Jacobian matrix. Note that the concept of vector entries of Sect. 4.6 is applied here (if $n_D > 1$). Wherever no index to indicate the stage value is written, the value of the first stage is taken, i.e., $i = 1$, as the Jacobian is only calculated at the beginning of a

time step. Again, the reaction rates $r^{(i)}$ form the intermediate solution for each stage as

$$\mathbf{r}^{(i)} = \mathbf{r} \left(t_n + \alpha_i h, \mathbf{y}_n + \sum_{j=1}^{i-1} a_{ij} \begin{bmatrix} \mathbf{u}^{(j)} \\ \tilde{\mathbf{u}}^{(j)} \end{bmatrix} \right).$$

5 Evaluation and First Results

Cminor is evaluated by the numerical simulation of six different chemical mechanisms in a box-model setup, followed by a comparison to two other chemical solvers with a box-model gas-phase simulation, and a simulation of a rising adiabatic parcel. Five reaction mechanisms are derived from atmospheric chemistry modeling, and three mechanisms from the field of combustion chemistry, which are used in internal combustion engine CFD, autoignition and flame simulations. These simulations are meant to present the functionality of the solver. All results can be reproduced by a python script, provided in the repository Rug et al. (2025). Tables 3 and 4 provide an overview of the conditions simulated and the references to the chemical mechanisms. All simulations are carried out on a MacBook Pro 2015 with a 2.5 GHz Quad-Core Intel Core i7 processor.

The first atmospheric mechanism is the Chapman mechanism (Sandu and Sander, 2006), using 7 species and 10 reactions to explain the presence of the ozone layer in earth's stratosphere. The results are shown in Fig. 4a.

The second atmospheric mechanism, RACM+CAPRAMv2.4, consists of the gas phase Regional Atmospheric Chemistry Model (RACM) by Stockwell et al. (1997), coupled with the Chemical Aqueous Phase Radical Mechanism version 2.4 (CAPRAM2.4) by Ervens et al. (2003). RACM is an upgrade of previously developed mechanisms for acid deposition in the troposphere, and intends to be valid for remote to polluted conditions in the whole troposphere. It is still a condensed mechanism, i.e., not a near-explicit description of all resolved processes, but it resembles thoroughly revised inorganic as well as organic chemistry. The aqueous phase mechanism CAPRAM2.4 contains inorganic, extended organic, and transition metal chemistry. Simulation results have been analyzed earlier by Tilgner et al. (2008), including time resolved source and sink studies focusing particularly on multiphase phase processing of radical oxidants and of C2–C4 organic compounds. Results are presented in Fig. 4b.

The third atmospheric mechanism, MCMv3.2+CAPRAMv4.0 α , consists of the detailed Master Chemical Mechanism version 3.2 (MCMv3.2) developed by Jenkin et al. (2012), which is a near-explicit, pure gas phase mechanism of the troposphere. Coupled with version 4.0 α of the Chemical Aqueous Phase Radical Mechanism (CAPRAM4.0 α) (Bräuer, 2015), it represents a detailed documentation of chemical processes in the gaseous and aqueous phase of the troposphere. The Master

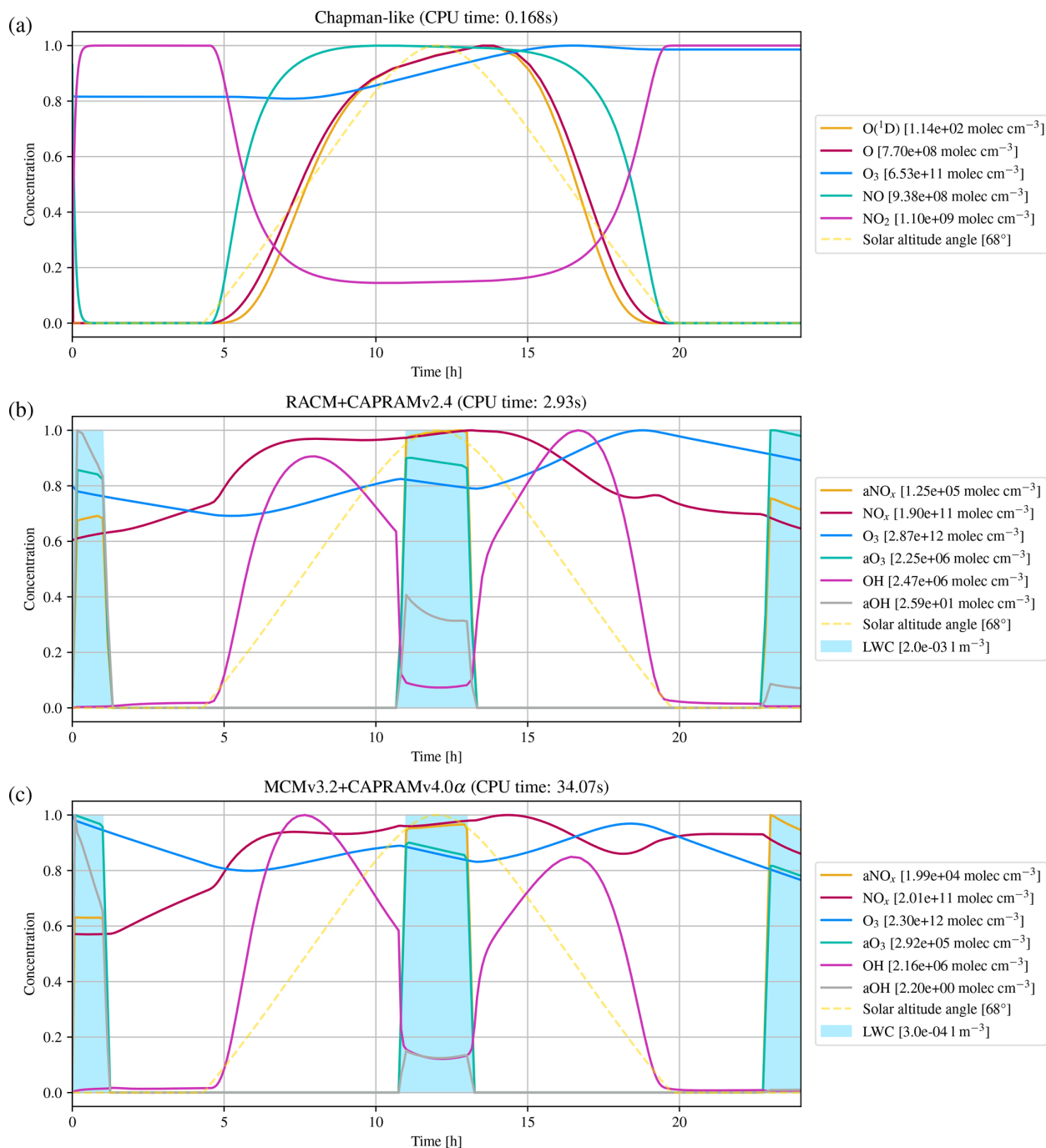


Figure 4. Numerical simulation results of three atmospheric (multi-phase) chemistry mechanisms for a one day time interval, with panel (a) presenting the Chapman, (b) the RACM+CAPRAMv2.4, and (c) the MCMv3.2+CAPRAMv4.0 α mechanism. To visualize all species' concentrations in a single plot, the concentrations have been normalized by individual scaling factors, as indicated in the legend. The blue columns indicate the formation of cloud droplets, which is required for aqueous-phase reactions. CPU time is the total time spent for integration with a relatively strict relative error tolerance of 1.0×10^{-4} .

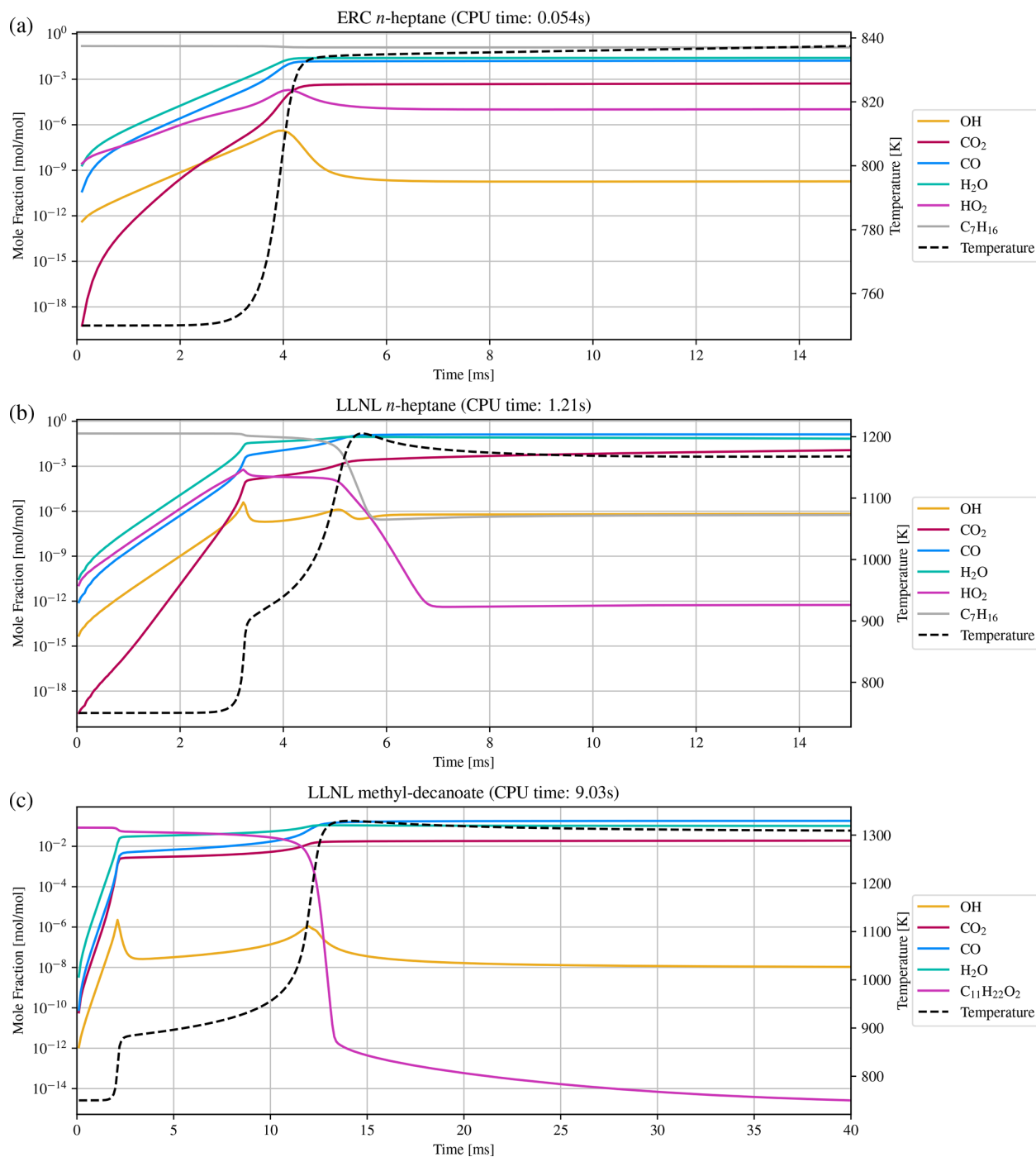


Figure 5. Numerical simulation results of three combustion mechanisms: (a) ERC *n*-heptane, (b) LLNL *n*-heptane, and (c) LLNL methyl-decanoate. CPU time is the total time spent for integration with a relatively strict relative error tolerance of 1.0×10^{-4} .

Table 3. Mechanism features for the four atmospheric (multi-phase) systems. For multiple droplet classes, aqueous species and reactions need to be considered once for every droplet class in the ODE system, due to different concentrations. The resulting number of species and reactions in the ODE system is shown in parenthesis, if multiple droplet classes are present.

mechanism	Chapman	RACM+CAPRAMv2.4	MCMv3.2+CAPRAMv4.0 α	Sulfur Oxidation
class	pure gas phase	multiphase (gas, aqua)		
scenario	no emissions no cloud	urban conditions mono-disperse droplet distribution		relatively unpolluted 50 superdroplets
n_S	7	250	10 196	26 (906)
n_R	10	787	23 098	36 (1800)
reference	Sandu and Sander (2006)	Stockwell et al. (1997); Ervens et al. (2003)	Jenkin et al. (2012); Bräuer (2015)	Kreidenweis et al. (2003)

Table 4. Mechanism features for the three constant-volume reactor combustion systems.

mechanism	ERC <i>n</i> -heptane	LLNL <i>n</i> -heptane	LLNL methyl-decanoate
class	gas phase combustion – constant volume reactor		
n_S	29	160	2878
n_R	104	1540	16 831
initial mixture	[C ₇ H ₁₆] = 15.3846 % [O ₂] = 17.7689 % [N ₂] = 66.8465 %	[C ₇ H ₁₆] = 15.3846 % [O ₂] = 17.7689 % [N ₂] = 66.8465 %	[C ₁₁ H ₂₂ O ₂] = 8.43882 % [O ₂] = 19.2275 % [N ₂] = 72.3337 %
reference	Patel et al. (2004)	Seiser et al. (2000)	Herbinet et al. (2008)

Chemical Mechanism was developed to investigate the degradation of emitted volatile organic compounds (VOCs), which have a major influence on the chemistry of the troposphere, contributing to the formation of ozone, secondary organic aerosol (SOA), and other secondary pollutants. CAPRAM4.0 α is the most comprehensive version of the CAPRAM mechanism and is also used to study oxidant budgets and SOA formation in the aqueous phase. Figure 4c shows the corresponding results.

In homogeneous charge compression ignition engines, the mixture of fuel and oxidizer is compressed to the point of auto-ignition, creating heat to be transformed into work by the engine. Investigating and optimizing such engines requires deep understanding of the ongoing chemistry and intermediate chemical states of the fuel as well as of physical properties, especially temperature. The following three combustion mechanisms were chosen because they were used as representatives for combustion chemistry in Perini et al. (2012).

Being used as a reference fuel, *n*-heptane studies are crucial and a number of mechanisms exist to simulate ignition delay times, heat release rates, and more, some being more reduced versions and some containing more detailed descriptions of the chemical processes. Two are shown here, a strongly reduced version from the Engine Research Center

(ERC) at the University of Wisconsin-Madison (Patel et al., 2004), and a slightly larger mechanism developed at the Lawrence Livermore National Laboratory (LLNL) in California (Seiser et al., 2000). The results are shown in Fig. 5a and b. The third combustion mechanism, also developed at LLNL (Herbinet et al., 2008), describes the consumption of an alternative fuel, driven by methyl decanoate. Such large methyl esters are characteristic for rapeseed or soybean derived biodiesels and such fuels show different behavior, e.g., early formation of carbon dioxide, which this larger mechanism is able to reproduce. It serves as an example for state-of-the-art comprehensive combustion mechanisms and results are shown in Fig. 5c.

The last two mechanisms are used to validate the correctness of the solver. The MCMv3.2 (Jenkin et al., 2012) is used to analyze *Cminor*'s ability to solve pure chemistry. The mechanism is simulated using *Cminor* and KPP with the same initial and environmental conditions, numerical methods, and tolerances. The results match, while the times to perform a simulation differ, demonstrating different usages of the two solvers. Also, KPP has been maintained and optimized over many years and creates a source code perfectly fitting one specific mechanism. *Cminor* has unused optimization potential in matrix-vector operations and possibly elsewhere. The simulation times are 4.55 and 3.08 s for *Cminor*

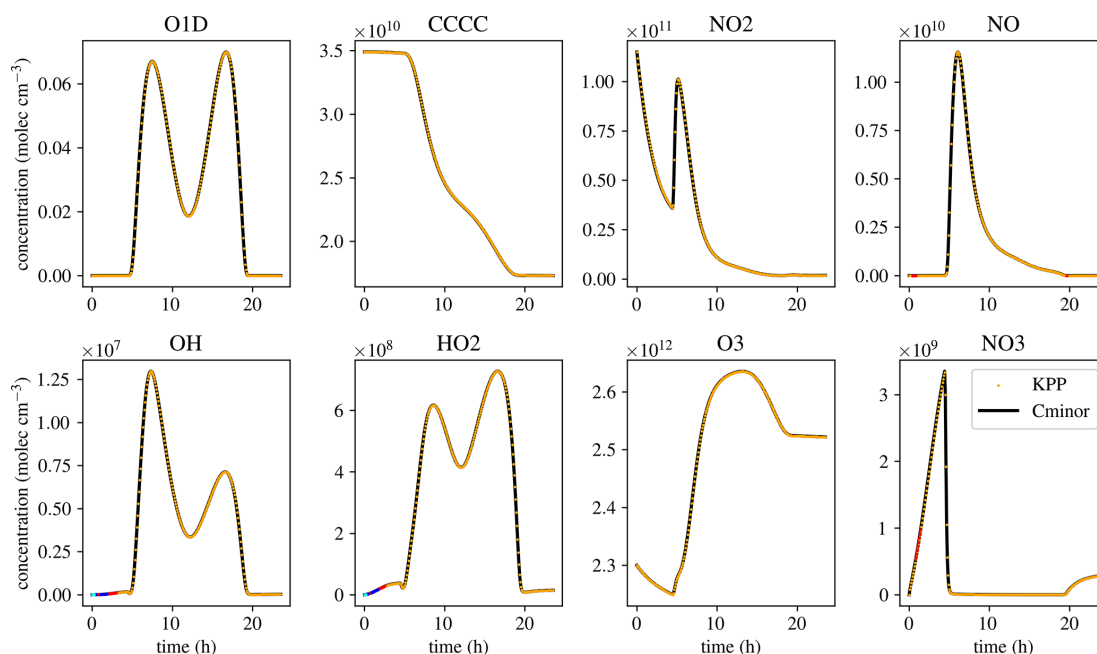


Figure 6. Comparison of eight species of a simulation of the MCMv3.2 mechanism with KPP and Cminor. Black lines show simulation results of Cminor, yellow dots indicate the results of the simulation using KPP. Red dots indicate a 2 % relative difference between the results of KPP and Cminor, blue dots indicate 5 % relative difference, and cyan dots 10 % relative difference between the two models.

and KPP, respectively (a factor of 1.48). This is primarily because KPP needs less computation time per numerical time step. While KPP does 1358 time steps in total, Cminor does 1394. Reading and preparing the simulation, on the other hand, takes 5 h and 52 min for KPP, and 2.2 s for Cminor. While this showcases that Cminor enables fast and flexible modification and analysis of mechanisms, sensitivities, and different scenarios, an optimization effort should be made to reach closer to KPP's simulation times.

For the parcel model, we chose to reproduce the case used in the intercomparison study by Kreidenweis et al. (2003) and in Jaruga and Pawlowska (2018). The sulfur oxidation mechanism deployed there describes the irreversible accumulation of sulfur in cloud droplets via oxidation pathways with hydrogen peroxide and ozone. This models the aging of aerosol, in this case the growth, by existence of the aqueous phase, i.e., a cloud. Results are shown in Fig. 7, where yellow dots indicate the results presented in Jaruga and Pawlowska (2018). Most distinctly, Cminor predicts a cloud droplet concentration of 397 cm^{-3} , while Jaruga and Pawlowska (2018) estimated a concentration of 269 cm^{-3} . This difference is due to the lower water accommodation coefficient of 0.0415 used by Cminor, while Jaruga and Pawlowska (2018) used a value of 1.0. Using a water accommodation coefficient of 1.0, Cminor predicts a cloud droplet number concentration of 338 cm^{-3} . All values are well within the range presented in the intercomparison study Kreidenweis et al. (2003), the results of Jaruga and Pawlowska (2018) being at the lower end, while Cminor is located more in the upper region.

Finally, Fig. 8 shows a collection of statistics of the mechanisms presented here. It gives a glimpse on how the computational effort and simulation times scale with the number of species and reactions, how the sparsity grows in the respective matrices, and of the robustness of the symmetric Markowitz ordering strategy (Markowitz, 1957), used to prevent sparse matrices from becoming dense matrices upon decomposition. This can only be seen as an empirical demonstration since the computational effort depends heavily on aspects of the chemical mechanism and cannot be predicted in terms of species number or other simple measures. The computation time for one integration time step shows linear behavior on the log-log scale, but this cannot be used as a general estimate as there are more factors determining computation times. For total simulation time, especially the number of steps needed is important. The total simulation times can be seen in Figs. 4 and 5.

6 Summary and Outlook

In this work, we presented Cminor – a highly efficient stand-alone solver environment designed to simulate chemical kinetic systems, from skeletal mechanisms to detailed descriptions of the atmosphere's chemical processes in the gaseous and aqueous phase. The optimized implementation allows for fast and accurate simulations of these chemical systems. Additionally, the lifting of an adiabatic parcel and cloud droplet formation can be simulated, which are influenced by

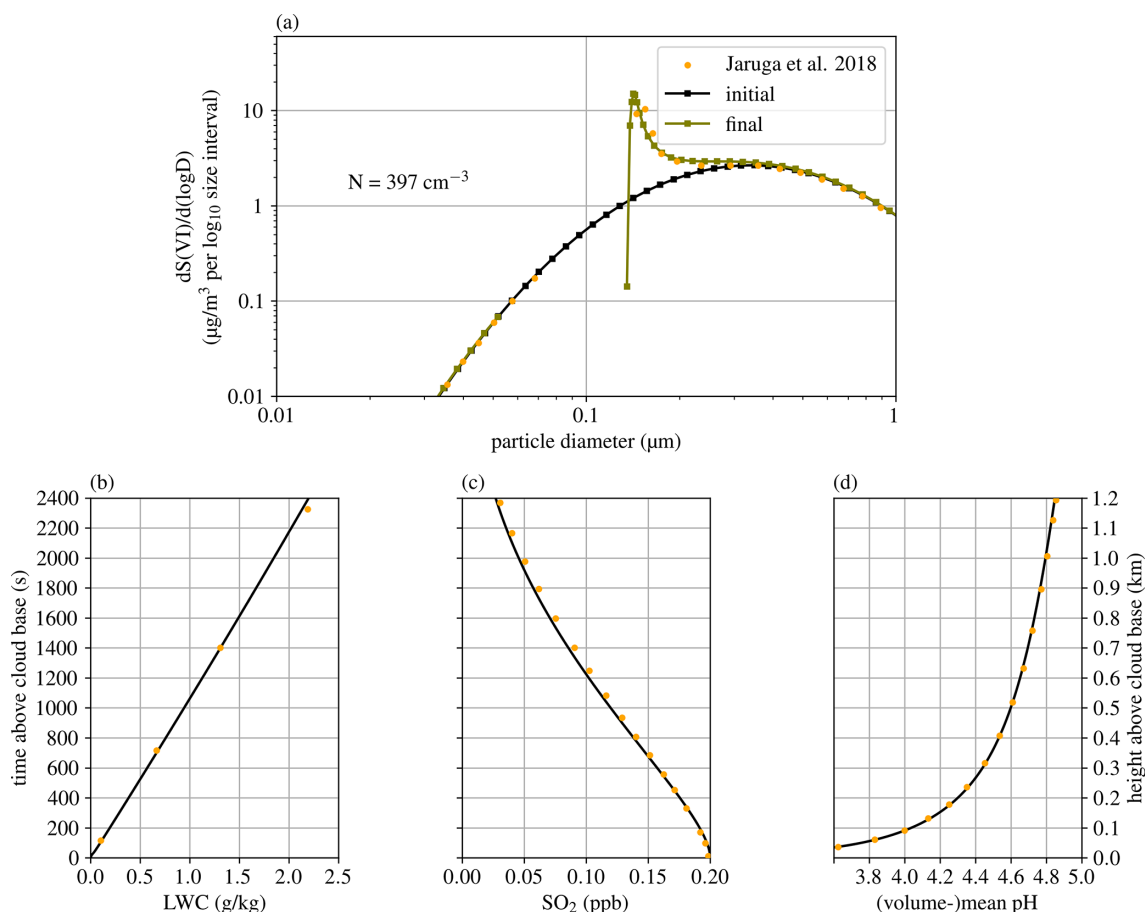


Figure 7. Simulated conditions in the adiabatic parcel setup (Kreidenweis et al., 2003). Subplot (a) shows the evolution of the aerosol size distribution with the number concentration of activated aerosol, denoted by N , (b) the liquid water mixing ratio, (c) the SO_2 concentration (both gaseous and dissolved), and (d) the negative logarithm of the volumetric mean H^+ concentration in the droplets. Yellow dots indicate the results presented in Jaruga and Pawlowska (2018).

chemical processes and vice versa. Furthermore, Cminor can solve the energy conservation equation needed for a perfectly adiabatic constant-volume reactor to investigate combustion chemistry instead of atmospheric systems.

The solver of Cminor uses linear-implicit Rosenbrock numerical schemes, which are well-suited for the integration of stiff systems of chemical kinetics. Moreover, Cminor's solver is implemented in a way that is tailored to specifically consider and exploit the characteristics of ODE systems of chemical kinetics. The following list of key features illustrates the improvements of the solver. Using sparse linear algebra is essential, but also exploiting matrix block structures, the sparsity of the analytical Jacobian matrix, only considering relevant derivatives which corresponds to an approximated Jacobian matrix, efficient evaluation of the rate constants, a symbolic and droplet-wise vectorized algebra and LU decomposition, and automated adaptive step size with error control.

The input syntax of atmospheric mechanisms for Cminor is outlined in the text, and elaborated on in the Supple-

ment. A collection of rate constants can be chosen of to simulate Arrhenius-type laws, photolytic reactions, third-body interactions, pressure-dependencies, humidity-dependencies, gas-aqua phase transfer, dissociation, pH-independence, and special reaction types, including custom formulas. The setup of a simulation is designed to be intuitive and only requires use of readable ASCII files to list the chemical reactions, initial values, and simulation parameters such as temperature, output frequency, and error tolerances. For combustion systems, the system description follows the widely used ChemKin syntax rules and JANAF standards.

Flexibility is achieved by the stand-alone nature of Cminor, which reads the mechanism and other inputs to construct the required ODE system on the fly with a high-speed mechanism parser. Chemical mechanisms, ambient conditions, and all simulation parameters can be changed and studied without recompiling, by typing a new parameter in the respective text file and re-executing the Cminor run command. Output is written in NetCDF format, or binary files for more abundant data.

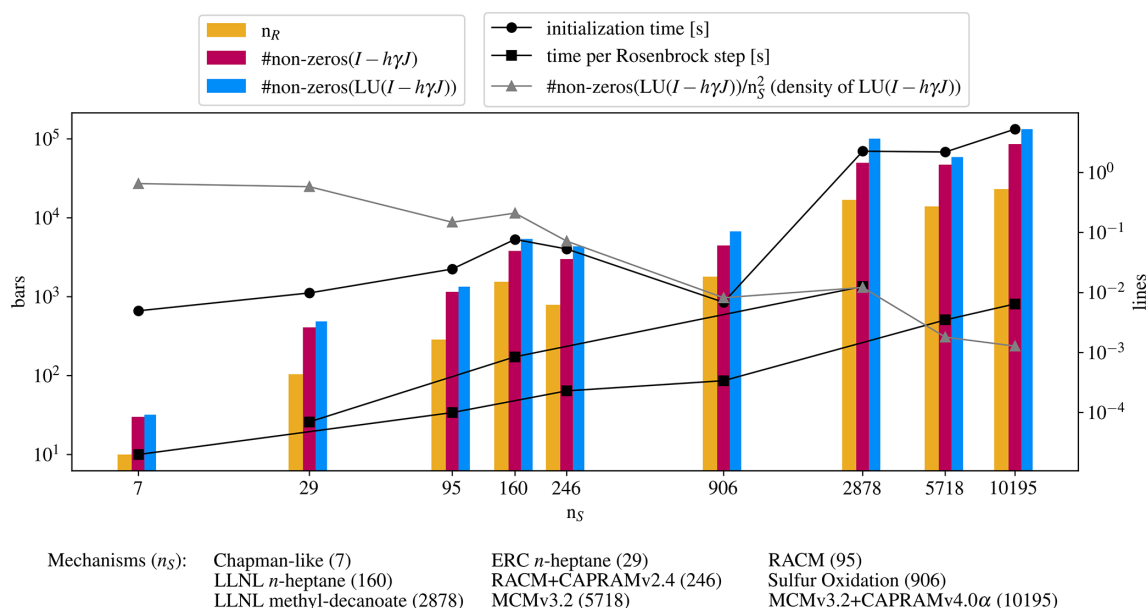


Figure 8. Overview of some statistics of the presented mechanisms. The abscissa is the number of species, determining where the mechanism statistics are placed, to preserve any scaling behavior with the number of species (which is the number of equations in the ODE system). The ordinates do not have units, unless indicated in the legends. The bars indicate n_R , the number of reactions, the number of non-zeros in the iteration matrix $I - h\gamma J$, and the number of non-zero entries in the LU decomposition of the iteration matrix. The lines with circles indicate the time to prepare the mechanism, i.e., everything what has to be done before integration, namely, reading reactions and symbolic decomposition. Lines with squares show the time spent for one Rosenbrock step. Times for atmospheric and combustion mechanisms are shown separately, because the temperature equation for combustion increases the computational effort, as clearly visible in the figure. The grey line with triangles shows the decreasing density of the decomposed iteration matrix, i.e. the number of non-zero entries divided by the matrix size n_s^2 .

The code is fully modularized and does not require the use of additional 3rd party software, except the output data format (NetCDF) and the basic Linear Algebra Pack (LAPack). *Cminor* v1.0 is open source and available under the terms of the GNU General Public License version 3.0.

We showed that *Cminor* is applicable to smaller as well as modern detailed descriptions of chemical processes in atmospheric and combustion chemistry. State-of-the-art multiphase mechanisms, e.g., the Master Chemical Mechanism version 3.2 (Jenkin et al., 2012) coupled with the Chemical Aqueous Phase Radical Mechanism version 4.0 α (Hermann et al., 2005), with more than 10 000 species and 23 000 reactions, or a methyl decanoate mechanism (Herbinet et al., 2008), derived from bio-fuel chemistry, with nearly 3000 species and 17 000 reactions can be simulated. Also, *Cminor* is able to accurately reproduce previously studied effects of chemical processes on aerosol size distributions (Kreidenweis et al., 2003). We found that it takes 3.08 s to simulate the Master Chemical Mechanism version 3.2 with a state-of-the-art KPP code, while *Cminor* required 4.55 s, which is slower by a factor of 1.48. Reading the mechanism and preparing the simulation, however, took 5 h and 52 min for KPP, and 2.2 s for *Cminor*. This overhead is required any time a reaction, a parameter, or an environmental variable is

changed, demonstrating the flexibility and handiness of *Cminor*. Nonetheless, further optimization is desirable to reach closer to KPP's simulation time, for example in *Cminor*'s linear algebra routines.

Cminor's current box-model setup can be used to study sensitivities, new mechanisms, specific interactions or other detailed research interests. The workflow of *Cminor* also enables ensemble simulations and analysis. Moreover, *Cminor* can easily be coupled with three-dimensional models, enabling the integration with comprehensive modeling frameworks, such as chemical transport or physical processes of the atmosphere. The droplet condensation equation resembles an important interaction of chemical and physical processes, which is accurately modeled by *Cminor*.

In future work, we aim to couple *Cminor* to a large-eddy simulation (LES) model with Lagrangian cloud microphysics. The droplet classes of *Cminor* match the concept of superdroplets in Lagrangian cloud models, while the LES simulates the transport of chemical constituents of the atmosphere. This will enable the investigation of the interplay of atmospheric chemistry with dynamics and cloud physics on larger scales. Due to its modular structure, *Cminor* is easily extendable. Resolving processes including a particulate phase would be a valuable addition, as well as ionic

strength effects and other peculiarities for small water contents. Adding the possibility to include particle nucleation will be subject of future work. In this way, *Cminor* will contribute to a more systematic understanding of the influences of atmospheric chemistry on physical, meteorological, and climatological processes.

Code availability. The code is published on Zenodo (Rug et al., 2025, DOI: <https://doi.org/10.5281/zenodo.17186119>) and maintained on the Github repository linked there.

Supplement. The supplement related to this article is available online at <https://doi.org/10.5194/gmd-18-9039-2025-supplement>.

Author contributions. LR and WS developed the draft and the source code. OK and FH supervised the process and contributed to editing and improving it.

Competing interests. The contact author has declared that none of the authors has any competing interests.

Disclaimer. Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors. Views expressed in the text are those of the authors and do not necessarily reflect the views of the publisher.

Acknowledgements. The authors thank Andreas Tilgner and Erik Hoffmann for their assistance in all questions related to atmospheric chemistry.

Financial support. This research has been supported by the Deutsche Forschungsgemeinschaft (grant nos. HO 6588/1-1 and HO 6588/3-1) and the EU Horizon 2020 (EUROCHAMP-2020 (grant no. 730997)).

The article processing charges for this open-access publication were covered by the Freie Universität Berlin.

Review statement. This paper was edited by Simon Unterstrasser and reviewed by two anonymous referees.

References

Bräuer, P.: Extension and application of a tropospheric aqueous phase chemical mechanism (CAPRAM) for aerosol and

cloud models, Ph.D. thesis, Leipzig, Univ., Diss., 2015., <http://nbn-resolving.de/urn:nbn:de:bsz:15-qucosa-183743> (last access: 11 November 2025), 2015.

Christner, B. C., Cai, R., Morris, C. E., McCarter, K. S., Foreman, C. M., Skidmore, M. L., Montross, S. N., and Sands, D. C.: Geographical, seasonal, and precipitation chemistry influence on the abundance and activity of biological ice nucleators in rain and snow, *Proceedings of the National Academy of Sciences*, 105, 18854–18859, <https://doi.org/10.1073/pnas.0809816105>, 2008.

Ervens, B., George, C., Williams, J. E., Buxton, G. V., Salmon, G. A., Bydder, M., Wilkinson, F., Dentener, F., Mirabel, P., Wolke, R., and Herrmann, H.: CAPRAM 2.4 (MODAC mechanism): An extended and condensed tropospheric aqueous phase mechanism and its application, *J. Geophys. Res.*, 108, 4426, <https://doi.org/10.1029/2002jd002202>, 2003.

Fukuta, N. and Walter, L.: Kinetics of hydrometeor growth from a vapor-spherical model, *Journal of Atmospheric Sciences*, 27, 1160–1172, [https://doi.org/10.1175/1520-0469\(1970\)027<1160:KOHGFA>2.0.CO;2](https://doi.org/10.1175/1520-0469(1970)027<1160:KOHGFA>2.0.CO;2), 1970.

Goel, V., Mishra, S. K., Pal, P., Ahlawat, A., Vijayan, N., Jain, S., and Sharma, C.: Influence of chemical aging on physico-chemical properties of mineral dust particles: A case study of 2016 dust storms over Delhi, *Environmental Pollution*, 267, 115338, <https://doi.org/10.1016/j.envpol.2020.115338>, 2020.

Hairer, E., Nørsett, S. P., and Wanner, G.: *Stiff and differential algebraic problems*, Springer, Berlin, ISBN 0387537759, <https://doi.org/10.1007/978-3-642-05221-7>, 1991.

Herbinet, O., Pitz, W. J., and Westbrook, C. K.: Detailed chemical kinetic oxidation mechanism for a biodiesel surrogate, *Combustion and Flame*, 154, 507–528, <https://doi.org/10.1016/j.combustflame.2008.03.003>, 2008.

Herrmann, H., Tilgner, A., Barzaghi, P., Majdik, Z., Gligorovski, S., Poulain, L., and Monod, A.: Towards a more detailed description of tropospheric aqueous phase organic chemistry: CAPRAM 3.0, *Atmospheric Environment*, 39, 4351–4363, <https://doi.org/10.1016/j.atmosenv.2005.02.016>, 2005.

Hoffmann, F., Raasch, S., and Noh, Y.: Entrainment of aerosols and their activation in a shallow cumulus cloud studied with a coupled LCM–LES approach, *Atmospheric Research*, 156, 43–57, <https://doi.org/10.1016/j.atmosres.2014.12.008>, 2015.

Jaruga, A. and Pawlowska, H.: libcloudph++ 2.0: aqueous-phase chemistry extension of the particle-based cloud microphysics scheme, *Geosci. Model Dev.*, 11, 3623–3645, <https://doi.org/10.5194/gmd-11-3623-2018>, 2018.

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.

Kazil, J., Wang, H., Feingold, G., Clarke, A. D., Snider, J. R., and Bandy, A. R.: Modeling chemical and aerosol processes in the transition from closed to open cells during VOCALS-REx, *Atmos. Chem. Phys.*, 11, 7491–7514, <https://doi.org/10.5194/acp-11-7491-2011>, 2011.

Kee, R., Rupley, F., and Miller, J.: The Chemkin Thermodynamic Data Base, <https://doi.org/10.2172/7073290>, 1990.

Kee, R., Rupley, F., Meeks, E., and Miller, J.: CHEMKIN-III: A Fortran Chemical Kinetics Package for the Analysis of Gas Phase

- Chemical and Plasma Kinetics, Sandia National Laboratories Report, <https://doi.org/10.2172/481621>, 1996.
- Kreidenweis, S. M., Walcek, C. J., Feingold, G., Gong, W., Jacobson, M. Z., Kim, C.-H., Liu, X., Penner, J. E., Nenes, A., and Seinfeld, J. H.: Modification of aerosol mass and size distribution due to aqueous-phase SO₂ oxidation in clouds: Comparisons of several models, *Journal of Geophysical Research: Atmospheres*, 108, <https://doi.org/10.1029/2002jd002697>, 2003.
- Kulmala, M., Pirjola, L., and Mäkelä, J. M.: Stable sulphate clusters as a source of new atmospheric particles, *Nature*, 404, 66–69, <https://doi.org/10.1038/35003550>, 2000.
- Kupc, A., Williamson, C. J., Hodshire, A. L., Kazil, J., Ray, E., Bui, T. P., Dollner, M., Froyd, K. D., McKain, K., Rollins, A., Schill, G. P., Thames, A., Weinzierl, B. B., Pierce, J. R., and Brock, C. A.: The potential role of organics in new particle formation and initial growth in the remote tropical upper troposphere, *Atmos. Chem. Phys.*, 20, 15037–15060, <https://doi.org/10.5194/acp-20-15037-2020>, 2020.
- Lamb, D. and Verlinde, J.: *Physics and chemistry of clouds*, Cambridge University Press, <https://doi.org/10.1017/cbo9780511976377>, 2011.
- Markowitz, H. M.: The Elimination Form of the Inverse and Its Application to Linear Programming, *Management Science*, 3, 255–269, <https://doi.org/10.1287/mnsc.3.3.255>, 1957.
- Murphy, D., Anderson, J., Quinn, P., McInnes, L., Brechtel, F., Kreidenweis, S., Middlebrook, A., Pósfai, M., Thomson, D., and Buseck, P.: Influence of sea-salt on aerosol radiative properties in the Southern Ocean marine boundary layer, *Nature*, 392, 62–65, <https://doi.org/10.1038/32138>, 1998.
- Patel, A., Kong, S.-C., and Reitz, R.: Development and validation of a reduced reaction mechanism for HCCI engine simulations, *SAE Paper 2004-01-0558*, 63–76, <https://doi.org/10.4271/2004-01-0558>, 2004.
- Perini, F., Galligani, E., and Reitz, R. D.: An analytical Jacobian approach to sparse reaction kinetics for computationally efficient combustion modeling with large reaction mechanisms, *Energy and Fuels*, 26, 4804–4822, <https://doi.org/10.1021/ef300747n>, 2012.
- Pruppacher, H. R. and Klett, J. D.: *Microphysics of clouds and precipitation*: Reprinted 1980, Springer Science & Business Media, <https://doi.org/10.1007/978-94-009-9905-3>, 1978.
- Rosenbrock, H. H.: Some general implicit processes for the numerical solution of differential equations, *The Computer Journal*, 5, 329–330, <https://doi.org/10.1093/comjnl/5.4.329>, 1963.
- Rug, L., Schimmel, W., Hoffmann, F., and Knöth, O.: *Cminor* v1.0, Zenodo [code], <https://doi.org/10.5281/zenodo.17186119>, 2025.
- Sander, R., Baumgaertner, A., Gromov, S., Harder, H., Jöckel, P., Kerkweg, A., Kubistin, D., Regelin, E., Riede, H., Sandu, A., Taraborrelli, D., Tost, H., and Xie, Z.-Q.: The atmospheric chemistry box model CAABA/MECCA-3.0, *Geosci. Model Dev.*, 4, 373–380, <https://doi.org/10.5194/gmd-4-373-2011>, 2011.
- Sandu, A. and Sander, R.: 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.
- Sandu, A., Verwer, J. G., Blom, J. G., Spee, E. J., Carmichael, G. R., and Potra, F. A.: Benchmarking stiff ODE solvers for atmospheric chemistry problems II: Rosenbrock solvers, *Atmospheric Environment*, 31, 3459–3472, [https://doi.org/10.1016/S1352-2310\(97\)83212-8](https://doi.org/10.1016/S1352-2310(97)83212-8), 1997.
- Schwartz, S. E.: Mass-transport considerations pertinent to aqueous-phase reactions of gases in liquid-water clouds, *Chemistry of Multiphase Atmospheric Systems*, 415–471, https://doi.org/10.1007/978-3-642-70627-1_16, 1986.
- Seiser, R., Pitsch, H., Seshadri, K., Pitz, W., and Gurran, H.: Extinction and autoignition of n-heptane in counterflow configuration, *Proceedings of the Combustion Institute*, 28, 2029–2037, [https://doi.org/10.1016/S0082-0784\(00\)80610-4](https://doi.org/10.1016/S0082-0784(00)80610-4), 2000.
- Shima, S.-I., Kusano, K., Kawano, A., Sugiyama, T., and Kawahara, S.: The super-droplet method for the numerical simulation of clouds and precipitation: A particle-based and probabilistic microphysics model coupled with a non-hydrostatic model, *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 135, 1307–1320, <https://doi.org/10.1002/qj.441>, 2009.
- Stockwell, W. R., Kirchner, F., Kuhn, M., and Seinfeld, S.: A new mechanism for regional atmospheric chemistry modeling, *Journal of Geophysical Research*, 102, 25847, <https://doi.org/10.1029/97JD00849>, 1997.
- Tilgner, A., Wolke, R., and Herrmann, H.: Capram Modeling Of Aqueous Aerosol And Cloud Chemistry, in: *Simulation and Assessment of Chemical Processes in a Multiphase Environment*, edited by Barnes, I. and Kharytonov, M. M., Springer Netherlands, Dordrecht, 107–122, ISBN 978-1-4020-8846-9, https://doi.org/10.1007/978-1-4020-8846-9_9, 2008.
- Weininger, D.: SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules, *Journal of chemical information and computer sciences*, 28, 31–36, <https://doi.org/10.1021/ci00057a005>, 1988.
- Wolke, R. and Knöth, O.: Time-integration of multiphase chemistry in size-resolved cloud models, *Applied Numerical Mathematics*, 42, 473–487, [https://doi.org/10.1016/S0168-9274\(01\)00169-6](https://doi.org/10.1016/S0168-9274(01)00169-6), 2002.
- Wolke, R., Sehili, A. M., Simmel, M., Knöth, O., Tilgner, A., and Herrmann, H.: SPACCIM: A parcel model with detailed microphysics and complex multiphase chemistry, *Atmospheric Environment*, 39, 4375–4388, <https://doi.org/10.1016/j.atmosenv.2005.02.038>, 2005.
- Yannakakis, M.: Computing the Minimum Fill-In is NP-Complete, *SIAM Journal on Algebraic Discrete Methods*, 2, 77–79, <https://doi.org/10.1137/0602010>, 1981.
- Zaveri, R. A., Barnard, J. C., Easter, R. C., Riemer, N., and West, M.: Particle-resolved simulation of aerosol size, composition, mixing state, and the associated optical and cloud condensation nuclei activation properties in an evolving urban plume, *Journal of Geophysical Research: Atmospheres*, 115, <https://doi.org/10.1029/2009jd013616>, 2010.
- Zhang, G., Peng, X., Sun, W., Fu, Y., Yang, Y., Liu, D., Shi, Z., Tang, M., Wang, X., and Bi, X.: Fog/cloud processing of atmospheric aerosols from a single particle perspective: a review of field observations, *Atmospheric Environment*, 120536, <https://doi.org/10.1016/j.atmosenv.2024.120536>, 2024.
- Zhang, H., Linford, J. C., Sandu, A., and Sander, R.: Chemical mechanism solvers in air quality models, *Atmosphere*, 2, 510–532, <https://doi.org/10.3390/atmos2030510>, 2011.