European Biophysics Journal (2023) 52:293–302
https://doi.org/10.1007/s00249-023-01637-0

METHODS PAPER

SViMULATE: a computer program facilitating interactive, multi-mode simulation of analytical ultracentrifugation data

Chad A. Brautigam[1]

Received: 27 October 2022 / Revised: 4 February 2023 / Accepted: 18 February 2023 / Published online: 9 March 2023

Abstract

The ability to simulate sedimentation velocity (SV) analytical ultracentrifugation (AUC) experiments has proved to be a valuable tool for research planning, hypothesis testing, and pedagogy. Several options for SV data simulation exist, but they often lack interactivity and require up-front calculations on the part of the user. This work introduces SViMULATE, a program designed to make AUC experimental simulation quick, straightforward, and interactive. SViMULATE takes user-provided parameters and outputs simulated AUC data in a format suitable for subsequent analyses, if desired. The user is not burdened by the necessity to calculate hydrodynamic parameters for simulated macromolecules, as the program can compute these properties on the fly. It also frees the user of decisions regarding simulation stop time. SViMULATE features a graphical view of the species that are under simulation, and there is no limit on their number. Additionally, the program emulates data from different experimental modalities and data-acquisition systems, including the realistic simulation of noise for the absorbance optical system. The executable is available for immediate download.

Keywords Analytical ultracentrifugation · Sedimentation velocity · Proteins · Viral vectors · Computer simulations

Introduction

When properly applied, simulation can be a vital component of planning a biophysical experiment. This utility is especially important in the field of analytical ultracentrifugation (AUC) in the sedimentation velocity (SV) mode. In this experimental modality, a high centrifugal field is applied to a solution of a macromolecule or a mixture of several macromolecules. As the molecules migrate along the vector of centrifugal force, concentration profiles of the solutes are acquired via on-board absorbance optics or a Rayleigh interferometer. These "scans" are taken at all relevant radii and occur at discrete times. In addition to their centrifugal transport, the molecules also undergo translational diffusion due to the concentration gradients inherent in the experiment. Examination of the velocity of the migration and the properties of the diffusion allow the discernment of the sedimentation coefficient (s) and the translational diffusion coefficient (D_T), and these two quantities can be used to determine the molar mass (M) of a species via the Svedberg equation:

$$M = \frac{sRT}{D_\mathrm{T}(1-\bar{v}\rho)}, \tag{1}$$

where R is the universal gas constant, T is the temperature in kelvins, $\bar{v}$ is the partial-specific volume of the species, and ρ is the solution density.

The acquired data can be modeled directly in data space using solutions the Lamm equation (Lamm 1929):

$$\frac{\partial c}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}\left[rD_\mathrm{T}\frac{\partial c}{\partial r} - c\omega^2 r^2 s\right], \tag{2}$$

where c is the concentration of the solute, t is time from the start of centrifugation, r is the distance from the center of rotation, and ω is the rotation speed. Because the solutions to this partial differential equation can be used to model data, they obviously can also be used to simulate data given a physically rational set of parameters. However, no exact analytical solution of the Lamm equation is known. Rather, analysts today predominantly solve the equation numerically, although there are good approximate analytical solutions

Special Issue: Analytical Ultracentrifugation 2022.

✉ Chad A. Brautigam
chad.brautigam@utsouthwestern.edu

[1] Departments of Biophysics and Microbiology, UT Southwestern Medical Center, Dallas, TX 75390, USA

(Behlke and Ristau 2002). Prominent AUC data-analysis software programs (e.g., SEDFIT (Brown and Schuck 2008), UltraScan (Cao and Demeler 2008), and SedAnal (Stafford and Sherwood 2004)) use numerical finite-element solutions of the Lamm equation, an approach pioneered by Claverie and colleagues in the 1970s (Claverie et al. 1975).

Simulation is common in AUC because the method, while very robust, is not amenable to quick pilot experiments. Thus, simulation offers the possibility of performing preliminary experiments in silico without the investment in time and materials needed for in vitro experiments. Common questions that can be addressed are (1) "How long will the experiment take at a given rotor speed?" (2) "What combination of rotor speed and experimental duration will result in an optimal analysis?" (3) "Can standard analyses resolve two (or more) putative species?" and (4) "How will the signal-to-noise ratio affect my analysis?" Thus, the ability to simulate AUC data is a rapid, cost-free means to augment the planning of AUC experiments. Simulation has obvious pedagogic value as well.

In recognition of the usefulness of simulation in AUC, the software packages mentioned above all contain simulation functionalities that work well. But they have features that can make them difficult to use for an inexperienced experimentalist. For example, a protein chemist will most often be familiar with the molar mass and perhaps the shape of a macromolecule under study, but the relationship between these parameters and those needed for simulation, namely s and D_T, are not straightforward. This fact imposes a burden on the experimenter to properly calculate the necessary quantities and enter them into the simulation software of choice. Another disadvantage of the available simulators is the lack of interactivity, i.e., adding or adjusting the parameters of a sedimenting species after examining the results of an initial simulation can be cumbersome. Also, some extant simulators require the user to input a finite time of sedimentation, but this may be unknown, forcing the user to make a difficult guess that may result in unwanted data or too few scans. Finally, alternative optical systems (i.e. Schlieren optics), different modes of data collection (difference sedimentation velocity, DSV, see Kirschner and Schachman 1971b; Brautigam et al. 2020), and realistic modeling of noise from the absorbance optical system are not supported.

To address the need for flexible, rapid, and interactive AUC simulations, a computer program called SViMULATE (Sedimentation Velocity in silico Mock experiments Using numerical Lamm and analytic Archibald-Type Equations) is introduced herein. This software has a built-in, on-the-fly hydrodynamics calculator that quickly provides the relationships between s, D_T, frictional ratio, and M under user-provided experimental conditions. The program can also receive such information from HullRad, which efficiently calculates s and D_T from structure files in the PDB format (Fleming and Fleming 2018). Results from the simulation are generally displayed within 1 s, and any necessary adjustments are easy to make, followed by additional simulation. There are simulation modes in which no finite time of simulation need be provided; instead, the software senses (based on user-adjustable criteria) whether the sedimentation is "complete", and the simulation is halted at that point. There are no limits on the number of species that can be simulated, and the simulated data may be displayed as standard signal-based or Schlieren profiles. Additionally, DSV experiments aimed at discerning small changes in s-values (Kirschner and Schachman 1971b; Brautigam et al. 2020) can be simulated, and noise features of the absorbance optical system can be emulated. Finally, the generated data may be outputted for analysis with other software. SViMULATE is freely available as a pre-compiled executable for 64-bit Windows-based computers and is distributed with all dependencies.

Methods

Algorithms

Numerical

For the numerical simulation of SV data, the finite-element algorithm of Claverie and others (Claverie et al. 1975) using mathematical strategies introduced by Todd and Haschemeyer (Todd and Haschemeyer 1983) and Schuck (2016) was coded into a C + + module (clavPack). Although the aforementioned authors have extensively documented the respective algorithms, a few of the concepts are recapitulated here to justify some of the strategies used in clavPack and SViMULATE. clavPack was encoded as a Python-readable module using Swig (Beazley 1996). SViMULATE imports this module, gathers parameters from the user, communicates them to the module, actuates the simulation, and finally, clears it from memory (i.e., collects the garbage). clavPack reports the results back to SViMULATE, which graphs the results.

The goal of simulating SV data is the description of the continuous function $c(r,t)$, representing the concentration of the solute as a function of radius and time after the start of centrifugation. In the formulation used herein and by others (Claverie et al. 1975; Cox and Dale 1981; Todd and Haschemeyer 1983), the radial space from the meniscus to the bottom of the solution column is divided into N equal-sized intervals; each interval thus has the size

$$\Delta r = \frac{r_b - r_m}{N}, \tag{3}$$

where r_m is the radial position of the meniscus and r_b is that of the bottom of the solution column. In SViMULATE, the user has control of the number of intervals and thus of the magnitude of Δr. This radial space is spanned by $N+1$ invariant triangular basis elements sometimes called "hat functions" (Claverie et al. 1975). Each of these elements P_i reaches its zenith (1, by definition) at r_i, slopes to 0 at r_{i-1} and r_{i+1}, and it is 0 everywhere outside of this range. Thus, a vector (**C**) with $N+1$ elements may be used to scale the $N+1$ hat functions to result (after summation) in $c(r,t)$ (Cox and Dale 1981):

$$c(r,t) = \sum_{i=1}^{N+1} C_i(t)P_i. \tag{4}$$

Consequently, the Lamm equation may be formulated thus for any element P_j at a given moment in time:

$$\sum_{i=1}^{N+1} \frac{dC_i}{dt} \int_{r_m}^{r_b} P_iP_jr\,dr - s\omega^2 \sum_{i=1}^{N+1} C_i \int_{r_m}^{r_b} P_i\frac{dP_j}{dr}r^2dr + D_{\mathrm{T}} \sum_{i=1}^{N+1} C_i \int_{r_m}^{r_b} \frac{dP_i}{dr}\frac{dP_j}{dr}r\,dr = 0. \tag{5}$$

Equation 5 represents a set of $N+1$ simultaneous equations that can be reformulated using matrices:

$$\mathbf{B}\frac{\mathbf{dC}}{\mathbf{dt}} - s\omega^2\mathbf{A_2C} + D_{\mathrm{T}}\mathbf{A_1C} = 0, \tag{6}$$

where the elements of matrices **B**, $\mathbf{A_2}$, and $\mathbf{A_1}$ can be calculated by computing the respective integrals that they substitute for (cf. Equations 6 and 5). These matrices are tridiagonal and invariant during a given simulation, and therefore they may be efficiently calculated at the outset and remain fixed. The formulas for the values in these matrices are tabulated elsewhere (Cox and Dale 1981; Todd and Haschemeyer 1983). A remaining problem is the calculation of the vectors **dC/dt** and **C**. They are estimated as

$$\begin{aligned}\frac{\mathbf{dC}}{\mathbf{dt}} &= \frac{\mathbf{C_a} - \mathbf{C_b}}{\Delta t},\\ \mathbf{C} &= (1-\theta)\mathbf{C_b} + \theta\mathbf{C_a},\end{aligned} \tag{7}$$

where $\mathbf{C_b}$ is the concentration vector before the time step at hand (which has a magnitude of Δt), $\mathbf{C_a}$ is the concentration vector after, and θ is a dimensionless value between 0 and 1 (inclusive).

Making these substitutions, rearranging, and conveniently defining $\mathbf{A} = D_{\mathrm{T}}\mathbf{A_1}$—$s\omega^2\mathbf{A_2}$, Eq. 6 becomes

$$(\mathbf{B} + \Delta t\theta\mathbf{A})\mathbf{C_a} = [\mathbf{B} - \Delta t(1-\theta)\mathbf{A}]\mathbf{C_b}. \tag{8}$$

The choice of θ underscores the main difference between the original approach of Claverie et al. (1975) and subsequent treatments by Todd and Haschemeyer (1983) and Schuck (1998). Claverie chose $\theta = 1$, an "implicit" scheme that simplifies the right-hand side of Eq. 8 to $\mathbf{BC_b}$. The θ value was set to 0.5 by Todd and Haschemeyer (1983), justifying the choice based on its inherent numerical stability. Schuck effectively made the same choice by applying a Crank-Nicolson scheme to the finite-element method (Crank and Nicolson 1947; Schuck et al. 1998). A θ value of 0.5 is used in SViMULATE, and the default value of Δt is 1.0 s.

By definition in this numerical simulation, $\mathbf{C_b}$ is known. At the start ($t=0$), it is a uniform value across all elements (i.e., radial positions). Therefore, Eq. 8 must be solved for $\mathbf{C_a}$, i.e., the concentration must be calculated following the time step Δt. This is accomplished in SViMULATE using the iterative procedure outlined by Todd and Haschemeyer (1983), with the only embellishment being the necessary recalculation of **A** at each time step during rotor acceleration (if used). After $\mathbf{C_a}$ is calculated, it is reassigned as $\mathbf{C_b}$, and the process begins again for the next time step. The $\mathbf{C_a}$ vector is not recorded for every time step; rather, the user stipulates a reporting frequency (called "scan frequency") in seconds, and only at these time points is $\mathbf{C_a}$ recorded for output before being reassigned as $\mathbf{C_b}$.

An important aspect of numerical simulation is when to stop it. SViMULATE offers four different ways to define the halt point. The first two are trivial: the user may indicate an integral number of "scans" to be outputted or may stipulate a total time of the simulation in hours and minutes. In the second two, the user tasks clavPack with the decision of when to exit. The first of these completion modes is called "Completion." At the recording points in simulation time, the algorithm compares the current values of $\mathbf{C_b}$ with the just previously recorded one at all radial values between r_{m} and r_{u}, the latter being a user-chosen "right-side limit." When the maximum difference (on an element-by-element basis) between the two "scans" falls below a user-defined level, the algorithm exits. The final mode is the "Concentration" mode, in which the signal at a user-provided radius (usually close to r_{b}) is monitored, and the algorithm exits when it falls below a user-defined threshold. It is possible for the user to set the halt criteria such that the simulation would never stop; however, as a failsafe, if the simulation reaches three days (259,200 s), clavPack automatically exits and SViMULATE displays the results. For all modes, SViMULATE displays the total time of sedimentation by default, as this value is sometimes the objective of the simulation.

The user is afforded significant control over the simulation in SViMULATE. Parameters under user control are: hydrodynamic parameters (vide infra), partial-specific volume, concentrations, r_{m}, r_{b}, T, ω (given as rotor speed in rpm), solution density, solution viscosity, N, Δt, scan frequency, rotor acceleration, completion mode/criteria, output sampling, and noise elements (vide infra).

Analytic

For analytical simulation, all calculations are performed in Python using all six terms of the Archibald-type equation promulgated by Behlke and Ristau (2002). A difficulty encountered in some simulations is that large exponents for e may need to be calculated, and these can exceed the floating-point precision used in the program. The user is warned in such cases. All of the user-adjustable parameters available to the numerical simulations are also present for the analytic ones, except for rotor acceleration and N.

Noise elements

In SViMULATE, three sources of random noise may be added to the noiseless, simulated data. First, of course, is the stochastic noise of data acquisition ($y(r,t)_\mathrm{s}$). This may be selected as normally distributed, randomly sampled noise added to each outputted data point, and the user has control of the standard deviation of the sampled distribution. However, because the absorbance scans result from the log transformation of a ratio of intensities, the noise distribution can no longer be assumed to be normal. Rather, simulations of an absorbance detector show that the noise increases and skews positively as absorbance increases; no simple analytic representation of this amplifying, skewing noise distribution could be found (see Supplemental Methods). Instead, the user may request realistic absorbance noise in two ways: (1) the user-provided parameters can be used to consult a series of tabulated parameters for an exponentially modified Gaussian function that can be sampled for noise-generation purposes, or (2) a simulation can be performed to generate the noise elements; the rationale and mathematics underpinning these protocols are presented "Results and Discussion" and more thoroughly in Supplemental Methods. Other noise sources include time-invariant (TI) noise, probably caused by imperfections in the optical path that light traverses during data acquisition, and radially invariant (RI) noise, which is usually only encountered with the Rayleigh interferometer and is due to minute changes in the vertical values of the fringes from scan to scan (i.e., "jitter") (Schuck and Demeler 1999). For TI noise, a function $y(r)_\mathrm{TI}$ is initiated with all $n\hat{}$values in this data set assigned to 0. This author has observed that the frequency of TI noise appears to be less than that of data acquisition. Thus, to mimic this "medium-frequency" noise, only every third data point from $y(r)_\mathrm{TI}$ is selected for noise generation, resulting in $\hat{n}$ data points in a subset called $\hat{y}(\hat{r})_{TI}$. Next, $\hat{n}$ values of stochastic, normally distributed noise are generated about 0 (again with a user-selected standard deviation) and, respectively, added to $\hat{y}_\mathrm{TI}$. This distribution is then subjected to a differencing procedure:

$$y'\left(\hat{r}_k\right)_\mathrm{TI} = \hat{y}\left(\hat{r}_k\right)_\mathrm{TI} - \hat{y}\left(\hat{r}_{k+1}\right)_\mathrm{TI} for\, k = 1 \ldots \hat{n} - 1. \tag{9}$$

Finally, the neglected data points from $y(r)_\mathrm{TI}$ are re-inserted to restore the full data set, and their values are interpolated (or extrapolated as necessary) between the newly calculated values of y'_TI. For RI noise ($y(t)_\mathrm{RI}$), for each time point t, a number is randomly sampled from a Gaussian distribution whose standard deviation is also specified by the user. This number is added to all radial points for a given t. Thus, the final formula for the output ($c(r,t)_\mathrm{out}$) is

$$c(r,t)_\mathrm{out} = c(r,t)_\mathrm{sim} + y(r)_\mathrm{TI} + y(t)_\mathrm{RI} + y(r,t)_\mathrm{s}, \tag{10}$$

where $c(r,t)_\mathrm{sim}$ represents the noiseless simulated data. The addition of noise in this fashion is available in SViMULATE for both the numerical and analytic simulation modes.

DSV simulations

For DSV simulations, the user is constrained to simulating two species: one for the reference sector, and one for the sample sector. The user inputs information about the reference species, and then all aspects of the species in the sample sector are kept the same except for the sedimentation coefficient (represented as Δs) and the meniscus (Δr_m). When actuated, SViMULATE calculates simulations for both species and then subtracts the concentration trace of the reference sector from that of the sample sector, plotting the result.

Schlieren optics

Simulations (except DSV) can be displayed either as signal-concentration traces (the default) or pseudo-Schlieren profiles. The latter are estimated using the central difference formula to approximate the first derivative of the profile. For a stable estimation, the concentration profile had to be interpolated with the assumption of a cubic spline connecting the successive data points. The suggestion of Cox and Dale (1981) of estimating this profile by differencing all concentration values and dividing by Δr was considered, but the resulting displacement of the radial grid by $\Delta r/2$ was not desired.

On-the-fly hydrodynamics calculations

Three of the hydrodynamic-calculation modes described in the main text essentially combine the Svedberg equation (Eq. 1), the Stokes–Einstein equation

$$D_T = \frac{kT}{f}, \tag{11}$$

where k is the Boltzmann constant and f is the frictional coefficient, Stokes' law,

$$f_0 = 6\pi\eta R_0, \tag{12}$$

and

$$R_0 = \left(\frac{3M\bar{v}}{4\pi N_A}\right)^{1/3}, \tag{13}$$

where f_0 is the frictional coefficient of a sphere with radius R_0, which is the minimum radius that a particle of molar mass M may assume, η is the solution viscosity, and N_A is Avogadro's number. For example, when the user inputs M and a frictional ratio f/f_0, Eq. 13 is used to find R_0, which is inserted into Eq. 12 to yield f_0 and, trivially, f. D_T can then be found from Eq. 11 and inserted into a rearranged Eq. 1 to yield s; s and D_T are then supplied to the simulation algorithm when the user starts the simulation. The values f/f_0, s, D_T, and M are continuously updated as appropriate in response to user inputs.

Fitting simulated data

Data for accuracy testing was outputted using SViMULATE's standard output features. No noise elements were added. The data were loaded into SEDFIT version 16.1c (https://sedfitsedphat.github.io/download.htm) and analyzed using the "Non-Interacting Discrete Species" model in a mode that directly fits s and D_T. No changes to the default numerical Lamm-equation parameters were made. Because SViMULATE writes out sedimentation data with a header feature indicating correct time-stamps, SEDFIT did not attempt to automatically modify them (see (Zhao et al. 2013)). The sample meniscus, s, D_T, and concentrations were fitted in the analyses.

Results and discussion

Simulation algorithms

SViMULATE has two different means of calculating solutions to the Lamm equation (Eq. 2). The first, preferred, mode is using a finite-element numerical simulation. The simulation implemented is similar to that proposed by Claverie (Claverie et al. 1975) and essentially identical to that implemented by Todd and Haschemeyer (1983), with the exception that the rotor acceleration to the target speed can be simulated (this feature is active by default in SViMULATE). Specifics of this simulation are beyond the scope of this communication and are mostly presented elsewhere (Claverie et al. 1975; Todd and Haschemeyer 1983; Schuck et al. 1998), but some aspects are detailed in Methods. The simulation can be efficiently carried out; 50 scans (spaced 5 min apart) of a 40,000 Da species with a frictional ratio of 1.3 sedimenting at 50,000 rpm in water were completed in 0.1 s on the author's laptop computer. This efficiency was achieved by encoding the simulation in C + + and interfacing this code to the Python master program (see Methods), and it was aided by optimized calculations in Python libraries like NumPy (Harris et al. 2020), SciPy (Virtanen et al. 2020), and Matplotlib (Hunter 2007).

The second mode of calculating concentration profiles is via an approximate analytic Lamm-equation solution as detailed by Behlke and Ristau (2002). This mode does not take rotor acceleration into account, and it was included mainly as a point for comparison between it and the numerical calculation. The advantage of the method is its speed: the calculation mentioned above, in this case performed entirely in the native Python environment, only takes 0.04 s. Although numerical solutions are very frequently used for modeling SV data, accuracy testing (vide infra) demonstrates that this analytical formula can work very well. Indeed, this approach forms the computational underpinnings of the data-modeling programs SVEDBERG (Philo 1996) and LAMM (Behlke and Ristau 2002). The main disadvantage of the analytic approach is that some terms of the Behlke/Ristau formula can assume values larger than the maximum value allowed in a 64-bit floating-point number. SViMULATE tests for this problem and reports to the user when a set of parameters may produce errors.

Neither of the simulation modes currently encoded into SViMULATE account for inter-solute interactions. That is, at present, only non-interacting, ideal species may be simulated. Other authors have modified the finite-element method to account for concentration-dependent effects on sedimentation, such as hydrodynamic non-ideality (Cox and Dale 1981) and infinitely fast self-association (Schuck 1998). Further, the numerical calculations can be extended to account for finite kinetics and hetero-associations (Stafford and Sherwood 2004; Dam et al. 2005). Although none of these are currently implemented in SViMULATE, expansion of the program to include at least a few simple non-ideal and interacting models is envisioned.

Accuracy testing

In the initial publication on the finite-element numerical method, Claverie et al. (1975) noted that there was some inaccuracy in the calculation when spatial and temporal discretization is sparse. That is, when they simulated an SV data set (N=400, Δt=1 s) with a sedimentation coefficient (s) of 7.0 S and a diffusion coefficient (D_T) of 5.7 F and then analyzed it using linear-transformation methods, errors of ≤ 0.2% and 2.3%, respectively, were observed. In an initial test of SViMULATE, this simulation was exactly recapitulated: in addition to the parameters listed above, it featured a rotor speed of 60,000 rpm (with no attempt to model rotor acceleration), one observation every 200 s, a meniscus of 6.0 cm, the sector bottom at 7.0 cm, and a starting "concentration" of 1.0, using the "implicit" scheme to perform the calculations (i.e., $\theta = 1$; see Eq. 8). These noiseless data were then analyzed with SEDFIT, which uses a different finite-element method (specifically, a non-equidistant grid and different time discretization (Brown and Schuck 2008)) to model SV data. The agreement between the modeled and refined values was excellent for the s-value but evinced a + 2.4% error in D_T (Table 1).

Next, a modification to the algorithm was made to depict the actual centrifugation experiment more realistically. Specifically, the rotor acceleration was modeled at 270 rpm/s, which is approximately the acceleration value observed with the analytical ultracentrifuge in service at UT Southwestern. Only slight increases in accuracy were observed (Table 1).

Finally, a correction scheme providing better numerical stability was added to the algorithm according to the method outlined by Todd and Haschemeyer (1983) and Schuck (Schuck et al. 1998). This method abandoned the "implicit" scheme of Claverie et al. (1975) for a more numerically robust form ($\theta = 0.5$; see Eq. 8). It required roughly twice the number of calculations to model the acceleration phase of the rotor, but it resulted in substantial increases in the accuracy of D_T (− 0.06%) without sacrificing significant levels of accuracy in the s value (Table 1). Given the excellent accuracy and performance of this method (33 "scans" of this simulation were completed in 0.04 s on the author's laptop), it was adopted as the method of choice for simulation in SViMULATE.

The approximate analytical solution encoded in SViMULATE performed very well for this particular set of parameters (Table 1). Indeed, its performance exceeded that of the previously described numerical simulation, having the same error in s and a slightly smaller deviation in D_T. However, as emphasized above, the rotor acceleration was not simulated, and thus the analytic solution is not the most faithful proxy for real-world SV data.

In early tests of the implicit Claverie algorithm implemented in SViMULATE, it was noted that large species sedimenting in a high centrifugal field suffered an even higher degree of inaccuracy than that noted above (Table 1). To illustrate this, a scenario in which two species having considerably different s values (3.244 S v. 11.516 S), molar masses (40 kDa v. 400 kDa), and frictional ratios (1.3 v. 1.7) was considered (Figs. 1 and 2A). In the implicit Claverie scheme, the SEDFIT-analyzed results featured a D_T for the larger species that was incorrect by 6.7%, leading to a faulty determination of the molar mass (see Eq. 1 and Table 2). The numerically robust scheme with modeled rotor acceleration provided far superior estimates of D_T and molar mass (errors of − 0.4% and + 0.4%, respectively). This scenario could not be simulated with the analytic algorithm, as it resulted in values in some terms exceeding the maximum for 64-bit floating-point numbers.

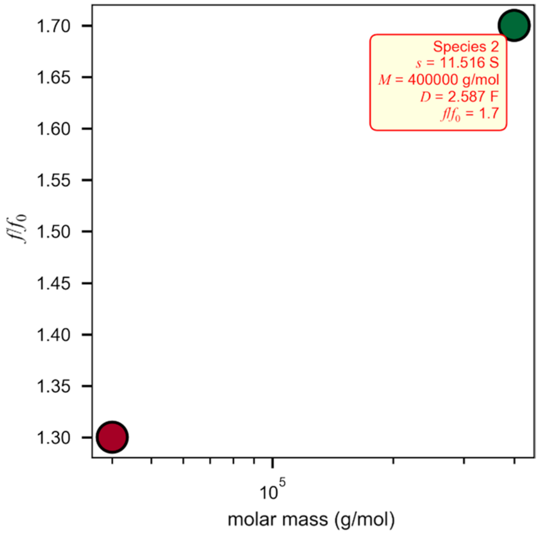

Fig. 1 The Species View in SViMULATE. Circles represent the two species currently inputted into SViMULATE. In this view, the user is hovering the mouse cursor over the circle representing Species 2; this action causes the program to pop up a yellow box that displays information about the respective species

Table 1 Simulation performance on a test solute[a]

Type	θ	Acceleration	Refined s (S)	Refined D_T (F)
Finite element	1	None	6.99991 (-1.2×10^{-3})[b]	5.8511 (+2.6)
Finite element	1	270 rpm/s	6.99993 (-1.0×10^{-3})	5.83991 (+2.4)
Finite element	0.5	270 rpm/s	7.00032 ($+4.2\times10^{-3}$)	5.69651 (−0.06)
Analytic	N/A	None	7.00032 ($+4.2\times10^{-3}$)	5.70151 (+0.03)

[a] The simulation parameters for finite-element simulations were: s: 7 S; D_T: 5.7 F; N: 400; Δt: 1 s; r_m: 6.0 cm; r_b: 7.0 cm; scan frequency: 200 s; rotor speed: 60,000 rpm

[b] Numbers in parentheses represent departures from the simulated value in percent

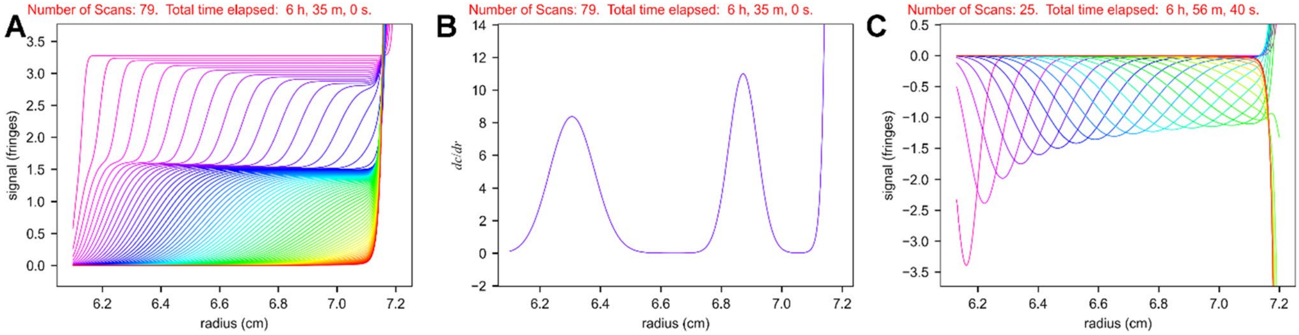

Fig. 2 Output from SViMULATE. For all parts, early scans are colored violet, and subsequent scans, respectively, advance through rainbow colors, ending with red. For parts **A** and **B**, a finite-element simulation was performed with the following parameters: *N*: 1,000; Δ*t*: 1 s; r_m: 6.1 cm; r_b: 7.2 cm; rotor speed: 50,000 rpm; rotor acceleration: 270 rpm/s; scan frequency: 300 s. Two species were simulated. Species 1 had an *s*-value of 3.244 S and a D_T of 7.287 F; Species 2 had values of 11.516 S and 2.587 F, respectively. These equate to species with *M* values of 40,000 g/mol and 400,000 g/mol, respectively, with respective frictional ratios of 1.3 and 1.7, sedimenting in water at 20 °C. Both were given concentrations of 0.5 mg/mL with a signal increment of 2.75 fringes·L/g·cm, and the path length was 1.2 cm. **A** Normal output. The "Completion" mode was chosen, and the total time of the simulation is noted at the top of the figure (this is exactly how SViMULATE displays the result). **B** Pseudo-schlieren output. For clarity, only the data from Scan 13 (3900 s after the start of centrifugation) are shown, but ordinarily, SViMULATE will show all scans overlapped. **C** DSV output. The parameters were *N*: 1,000; Δ*t*: 1 s; r_m: 6.1 cm; Δr_m: 0.03 cm; r_b: 7.2 cm; rotor speed: 50,000 rpm; rotor acceleration: 270 rpm/s; scan frequency: 1000 s. The reference sector was simulated to contain 3 mg/mL of the W145A mutant of *Treponema pallidum* protein TpMglB2, unliganded, and the sample sector was simulated with an identical concentration of the D-glucose-bound form of the same protein. Species parameters were provided by HullRad, as X-ray crystal structures of the two versions of the protein are known (accession numbers 6BGD and 6BGC, respectively (Brautigam et al. 2018)). The parameters were *s*: 3.39 S; D_T: 7.74 F; Δ*s*: 0.09 S. The "Completion" mode was also used for this simulation, automatically halting the simulation at the indicated elapsed time

Table 2 Finite-element simulation performance for two disparate species sedimenting simultaneously[a]

θ	Refined s_1 (S)	Refined $D_{T,1}$	Refined s_2 (S)	Refined $D_{T,2}$ (F)
1	3.24435 (+0.011)[b]	7.3150 (+0.4)	11.51528 (−0.006)	2.76211 (+7)
0.5	3.24442 (+0.013)	7.29871 (+0.16)	11.51628 (+0.002)	2.57713 (−0.4)

[a] The simulation parameters were: s_1: 3.244 S; $D_{T,1}$: 7.287 F; s_2: 11.516 S; $D_{T,2}$: 2.587 F; *N*: 1,000; Δ*t*: 1 s; r_m: 6.1 cm; r_b: 7.2 cm; scan frequency: 300 s; rotor speed: 50,000 rpm; rotor acceleration: 270 rpm/s

[b] Numbers in parentheses represent departures from the simulated value in percent

Given the recent interest in performing SV on gene-therapy vectors, particularly adeno-associated viruses (AAVs) (Burnham et al. 2015; Nass et al. 2018; Maruno et al. 2021), a simulation was conducted with large species meant to mimic empty and full AAV capsids (Table 3). The *s*-values of these species were approximately 64 S and 100 S, respectively, which are similar to results garnered in this lab and others. Although performing this simulation only resulted in 34 scans (with a scan frequency of 300 s), the analysis results (Table 3) show that the SViMULATE simulations accord with those from SEDFIT to a very high degree.

Table 3 Performance of the finite-element algorithm with large species[a]

Parameter	Simulated value	Refined value	% deviation
s_1 (S)	63.889	63.8916	+0.004
$D_{T,1}$ (F)	1.852	1.85063	−0.07
s_2 (S)	99.224	99.22568	+0.002
$D_{T,2}$ (F)	1.486	1.48957	+0.2

[a] Other simulation parameters: *N*: 1,000; Δ*t*: 1 s; r_m: 6.1 cm; r_b: 7.2 cm; scan frequency: 300 s; rotor speed: 20,000 rpm; rotor acceleration: 270 rpm/s

On-the-fly hydrodynamics calculations

Five modes of inputting macromolecular parameters were enabled in SViMULATE, named for the information that the user provides: (1) frictional ratio/*M*, (2) *s*/*M*, (3) *s/D*, (4) *s*/frictional ratio, and (5) HullRad. For example, in the first mode, the user can input the known molar mass and a guess regarding the frictional ratio (along with solution parameters), and all other parameters necessary for the simulation will be calculated on-the-fly and displayed to the user. Once these are adjusted to the user's satisfaction, the simulation can be actuated, with the result immediately displayed. Any number of species can be simulated, and each one can have its own mode of macromolecular-property input (Fig. 1). The fifth method, "HullRad," utilizes the convex-hull method introduced by Fleming and Fleming (2018) to generate *s* and D_T from a structure-coordinate file, and all other parameters are calculated from knowledge of these two.

If desired, SViMULATE allows the user to inspect a graph that summarizes the hydrodynamic properties of all currently inputted species. For example, the simulation used to produce Table 2 could be visualized as in Fig. 1. SViMULATE can generate three such views: *M* vs. f/f_0 (Fig. 1), *s* vs. f/f_0, and *s* vs. D_T. It is straightforward to switch between them.

Data display and output

The SV data resulting from the simulation can be displayed and outputted in several different ways. First and most commonly, the user may specify that signal profiles, as collected by the AUC data-acquisition software, be displayed (Fig. 2A). The user may enter "concentrations" in signal, molar, or mass-concentration units. Signal increments of course must be provided for the latter two values. A Schlieren-type data-output mode, i.e., dc/dr vs. r (Fig. 2B), is available, but the program enforces a requirement for the mass-concentration mode of concentration input, as this data-acquisition method is based on refractive-index changes.

A specialized mode offered only in SViMULATE is the ability to simulate DSV data. In this experimental strategy, samples of identical concentration are placed in both sectors of an AUC centerpiece, and the Rayleigh interferometer is used to measure the refractive-index differences between them. The usual objective is to find differences in sedimentation coefficient between the samples in the two sectors. This method is a sensitive means to detect ligand-induced conformational changes in proteins (Kirschner and Schachman 1971a; Brautigam et al. 2020). SViMULATE, when used for such simulations, expects the user to define two species, one for each sector. Upon actuation, it simulates both curves, computes the difference between them, and displays the result in signal units (Fig. 2C).

Systematic noise designed to mimic the noise generated by the AUC optics can also be simulated. Three major types of noise in AUC data are (1) the stochastic noise of data acquisition, (2) time-invariant (TI) noise, and (3) radially invariant (RI) noise (Schuck and Demeler 1999). Sources of these noise elements are briefly discussed in Methods and are elaborated elsewhere (Stafford 1992; Schuck and Demeler 1999; Kar et al. 2000; Schuck et al. 2016). SViMULATE can add all three types of noise in every possible combination. The user has control of the magnitude of noise added in all cases. An example of only TI noise added to the simulation in Fig. 2A is shown in Fig. 3.

Realistically modeling the stochastic noise from the absorbance optical system represents a particular challenge. This is because the absorbance reading is the base-ten logarithm of the ratio of two intensity readings (one from the reference sector, and one from the sample sector). Simulation of the intensity readings, considering the likely noise features, suggested that realistic noise for the absorbance optics has two trends as the reading increases: (1) it becomes higher, and (2) it becomes more asymmetrically distributed (Fig. 4A). Extensive modeling of theoretical noise led to the conclusion that it could be simulated with an exponentially modified Gaussian (EMG) distribution. Although a simple analytic relationship between the absorbance, user-selected noise, and the EMG's parameters could not be found, the modeling of 30,100 achievable combinations of parameters allowed the construction of parametric tables that can be consulted by SViMULATE (see Supplemental Methods). Thus, when the user selects realistic absorbance noise, the tables are referred to, and noise from an appropriate EMG is sampled. Notably, a single scan can feature readings from 0.0 to near the maximum of absorbance (Fig. 4B), and thus the noise should increase correspondingly. This feature is also a part of the SViMULATE absorbance modeling. The user may turn this realistically skewed noise feature on or off on demand. An alternative mode for calculating realistic absorbance noise is to simulate the noise elements directly as if they resulted from the logarithm of the ratio of two noisy intensity readings. Although this second method is effective and is provided as an option in SViMULATE, it is time consuming and imposes significant limitations on the magnitudes of the noise and the absorbance readings. For these reasons, the EMG-based method is preferred. An important aspect of these noise-generation protocols is that they do not guarantee accurate modeling for all absorbance

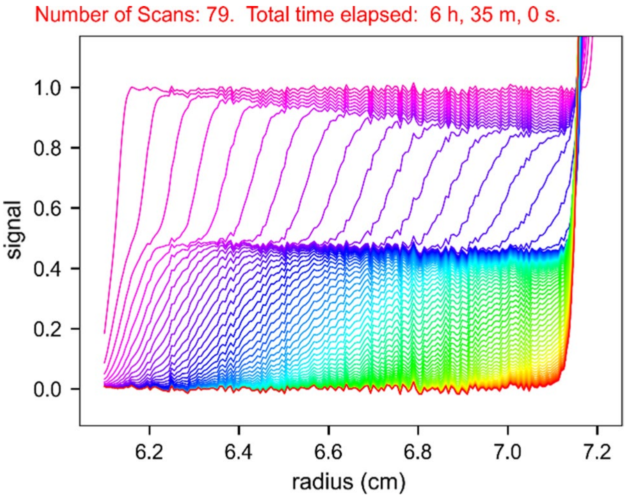

Fig. 3 TI Noise added to a simulation. The same simulation as in Fig. 2A is shown, but TI noise elements have been added by SViMULATE. The TI-Noise amplitude level was set to 0.1 for this simulation

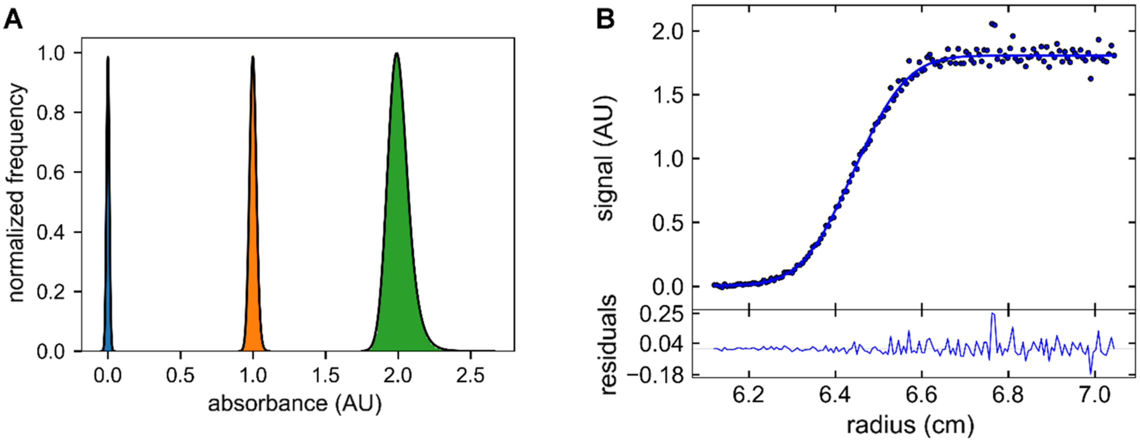

Fig. 4 Simulated realistically skewed noise from the absorbance optics. **A** Histograms of expected noise and EMG functions. The three histograms (blue, orange, and green) are normalized to a maximum value of 1.0. They, respectively, show the noise distribution expected for an absorbance optical system experiencing readings of 0.0, 1.0, and 2.0 AU (the latter is generally not readily achievable in most AUC instruments; nonetheless, the program allows readings up to 3.0 AU). These distributions were simulated by assuming that (1) the intensity reading from the reference sector stayed constant, (2) the noise from the detector was normally distributed, (3) the detector noise scaled as the square root of the intensity, and (4) the root-mean-square (RMS) noise level at 0.0 AU is 0.01 AU. The black lines are not fits to the respective histograms; rather, they are EMG distributions plotted with the parameters that were tabulated in a sparse but comprehensive sampling of RMS/Absorbance space (see Supplemental Methods). In other words, they are the distributions that would have been sampled by SViMULATE to provide realistically skewed absorbance noise given a user-provided noise level and the absorbance magnitudes. To compare the histograms and the distributions, the *H* statistic was adopted (Ma et al. 2015), with the sum of the squared frequencies from the EMG serving as the normalizing quantity; the respective *H* values were 0.04%, 0.04%, and 0.07%. **B** An example of EMG-sampled noise outputted from SViMULATE. A single species with a molar mass of 40,000 g/mol and a f/f_0 of 1.3 was simulated. The starting signal was 2.0 AU. An RMS noise of 0.01 AU was selected (per convention, SViMULATE makes this the root-mean-square noise of an absorbance reading of 0.0 AU). The upper panel shows the 20th scan (markers; only every 3rd data point is shown), along with the fit (from SEDFIT; line), and the lower panel shows the residuals between the shown data and the fit line

optical systems; rather, they generate noise features that are plausible for absorbance optics that behave as described in Supplemental Methods. Future modifications will seek to augment the verisimilitude of the noise (e.g., adding a sloping baseline to TI noise for data simulated to be from the Rayleigh interferometer).

The simulation can be saved in two ways. First, SViMULATE can write out a binary file that contains all species' respective parameters and the global experimental parameters. The user may thus load these data later and exactly recapitulate the simulation. The second means of saving the data is to write to disk the simulated scans using the Beckman-Coulter file format. Because the output grid may not exactly match the radial points specified by the user in the numerical simulation, linear interpolation is used to provide values for all the outputted radial points. The outputted files may be opened by any analytic software package for examination and analysis. An informational text file is also written in the same directory as the simulated data files; it contains all relevant details of the simulation.

In summary, the software SViMULATE is an accurate, quick, easy, and interactive tool for simulating AUC data in the sedimentation velocity mode. It may be downloaded immediately from https://www.utsouthwestern.edu/research/core-facilities/mbr/software, and it is designed for use on 64-bit Windows-based computers. It is hoped that it can serve as a tool to be utilized by the scientific community for experimental planning and hypothesis testing, facilitating the informed use of limited centrifuge time and maximizing throughput. Also, its ease of use should incentivize AUC neophytes to explore the principles of the method.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00249-023-01637-0.

Acknowledgements The author wishes to thank Dr. Peter Schuck for helpful discussions, Dr. Walter Stafford for providing exemplary code, and Drs. Lake Paul and Alexander Yarawsky for beta-testing SViMULATE and offering helpful suggestions.

Funding No funding was received for this work.

Data availability A compiled version of the software is freely available at https://www.utsouthwestern.edu/research/core-facilities/mbr/software.

Declarations

Conflict of interest The author has no relevant financial or non-financial interests to disclose.

References

Beazley DM (1996) Using SWIG to control, prototype, and debug C programs with Python. https://www.legacy.python.org/workshops/1996-06/papers/. Accessed 6 Feb 2023

Behlke J, Ristau O (2002) A new approximate whole boundary solution of the Lamm differential equation for the analysis of sedimentation velocity experiments. Biophys Chem 95:59–68

Brautigam CA, Deka RK, Liu WZ, Norgard MV (2018) Crystal structures of MglB-2 (TP0684), a topologically variant D-glucose-binding protein from *Treponema pallidum*, reveal a ligand-induced conformational change. Protein Sci 27:880–885

Brautigam CA, Tso S-C, Deka RK et al (2020) Using modern approaches to sedimentation velocity to detect conformational changes in proteins. Eur Biophys J 49:729–743

Brown PH, Schuck P (2008) A new adaptive grid-size algorithm for the simulation of sedimentation velocity profiles in analytical ultracentrifugation. Comput Phys Commun 178:105–120

Burnham B, Nass S, Kong E et al (2015) Analytical ultracentrifugation as an approach to characterize recombinant adeno-associated viral vectors. Hum Gene Ther Methods 26:228–242

Cao W, Demeler B (2008) Modeling analytical ultracentrifugation experiments with an adaptive space-time finite element solution for multicomponent reacting systems. Biophys J 95:54–65

Claverie J-M, Dreux H, Cohen R (1975) Sedimentation of generalized systems of interacting particles. I. Solution of systems of complete Lamm equations. Biopolymers 14:1685–1700

Cox DJ, Dale RS (1981) Simulation of transport experiments for interacting systems. In: Frieden C, Nichol LW (eds) Protein-protein interactions. John Wiley & Sons, New York, pp 173–211

Crank J, Nicolson P (1947) A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. Math Proc Cambridge Philos Soc 43:50–67

Dam J, Velikovsky CA, Mariuzza RA et al (2005) Sedimentation velocity analysis of heterogeneous protein-protein interactions: Lamm equation modeling and sedimentation coefficient distributions c(s). Biophys J 89:619–634

Fleming PJ, Fleming KG (2018) HullRad: Fast calculations of folded and disordered protein and nucleic acid hydrodynamic properties. Biophys J 114:856–869

Harris CR, Millman KJ, van der Walt SJ et al (2020) Array programming with NumPy. Nature 585:357–362

Hunter JD (2007) Matplotlib: a 2D graphics environment. Comput Sci Eng 9:90–95

Kar SR, Kingsbury JS, Lewis MS et al (2000) Analysis of transport experiments using pseudo-absorbance data. Anal Biochem 285:135–142

Kirschner MW, Schachman HK (1971a) Conformational changes in proteins as measured by difference sedimentation studies. II. Effect of stereospecific ligands on the catalytic subunit of aspartate transcarbamylase. Biochemistry 10:1919–1926

Kirschner MW, Schachman HK (1971b) Conformational changes in proteins as measured by difference sedimentation studies. I. A technique for measuring small changes in sedimentation coefficient. Biochemistry 10:1900–1919

Lamm O (1929) Die differentialgleichung der ultrazentrifugierung. Ark För Mat Astron Och Fys 21B:1–4

Ma J, Zhao H, Schuck P (2015) A histogram approach to the quality of fit in sedimentation velocity analyses. Anal Biochem 483:1–3

Maruno T, Usami K, Ishii K et al (2021) Comprehensive size distribution and composition analysis of adeno-associated virus vector by multiwavelength sedimentation velocity analytical ultracentrifugation. J Pharm Sci 110:3375–3384. https://doi.org/10.1016/j.xphs.2021.06.031

Nass SA, Mattingly MA, Woodcock DA et al (2018) Universal method for the purification of recombinant AAV vectors of differing serotypes. Mol Ther Methods Clin Dev 9:33–46

Philo JS (1996) An improved function for fitting sedimentation velocity data for low- molecular-weight solutes. Biophys J 72:435–444

Schuck P (1998) Sedimentation analysis of noninteracting and self-associating solutes using numerical solutions to the Lamm equation. Biophys J 75:1503–1512

Schuck P (2016) Sedimentation velocity analytical ultracentrifugation: discrete species and size-distributions of macromolecules and particles. CRC Press, Boca Raton

Schuck P, Demeler B (1999) Direct sedimentation analysis of interference optical data in analytical ultracentrifugation. Biophys J 76:2288–2296

Schuck P, MacPhee CE, Howlett GJ (1998) Determination of sedimentation coefficients for small peptides. Biophys J 74:466–474

Schuck P, Zhao H, Brautigam CA, Ghirlando R (2016) Basic principles of analytical ultracentrifugation. CRC Press, Boca Raton

Stafford WF (1992) Boundary analysis in sedimentation transport experiments: a procedure for obtaining sedimentation coefficient distributions using the time derivative of the concentration profile. Anal Biochem 203:295–301

Stafford WF, Sherwood PJ (2004) Analysis of heterologous interacting systems by sedimentation velocity: curve fitting algorithms for estimation of sedimentation coefficients, equilibrium and kinetic constants. Biophys Chem 108:231–243

Todd GP, Haschemeyer RH (1983) Generalized finite element solution to one-dimensional flux problems. Biophys Chem 17:321–336

Virtanen P, Gommers R, Oliphant TE et al (2020) SciPy 1.0: fundamental algorithms for scientific computing in python. Nat Methods 17:261–272

Zhao H, Ghirlando R, Piszczek G et al (2013) Recorded scan times can limit the accuracy of sedimentation coefficients in analytical ultracentrifugation. Anal Biochem 437:104–108

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.