Notations for PLOS One ﻿15(10): e0241539, Wang et al.

**1. *h*m and *h*:**

The mass transfer (*h*m) and heat transfer (*h*) coefficients are used in the equations as follows:

![Text, letter

Description automatically generated]()

Note that the term in Eq. (4) of the PLOS paper should be (mass concentration of water vapor in air – kg m-3) instead of (water vapor pressure – Pa). This would maintain a mass transfer coefficient *h*m a unit of m s-1. I think we made this error when we transcribed the equations.

In the simulation, *Nu* and *Sh* of 2 are assumed, because the relative movement between the droplet and the exhaled air are relatively small (Aissa, et al. “Ranz and Marshall correlations limits on heat flow between a sphere and its surrounding gas at high temperature.” Thermal science 19, no. 5 (2015): 1521-1528.). Note that for droplets that are at the edge of the exhaled air, there will be stronger relative movement, leading to larger heat and mass transfer coefficients, so the evaporation will be faster, leading to a longer residence time of particles in the air.

Therefore, we have , where is the diffusion coefficient of the water vapor molecule, and , where is the heat conductivity of air. The two equations can be rewritten as (Wang, et al. “A fast integrated mobility spectrometer for rapid measurement of sub-micrometer aerosol size distribution, Part I: Design and model evaluation.” Journal of Aerosol Science, 108, pp.44-55., **Eqs. (10-11)**):

A picture containing calendar

Description automatically generated

Text, letter

Description automatically generated

Note in the above two equations, *D*p is the droplet size (*D*d), *T*s is the droplet surface temperature (*T*d). You can find more annotations of the above equations for the paper referred above. The Matlab code solves the two equations in function EvapEquation.m (Line 51-64)

Since the respiratory droplet is mostly composed of water, I used the *c*p of water throughout the simulation. I set the initial droplet temperature to be 25 oC, but that can be changed by varying the initial conditions. I did a quick comparison of the results but did not found observable difference.

**2. Gravity term**

This is a typo in the equation and unfortunately, we did not catch this in our proof. Gravity term is considered in the solving the differential equation of the function EvapEquation.m (Line 84).

**3. Sigma values**

The sigma values are actually . The actual values will be 1.73 and 1.65

**4. Droplet evaporation**

What Joshin described is correct. The final core droplet diameter is determined by the volume of the salt and the volume of virus in the particles (EvapEquation.m (Line 75)). Here, we assumed that the virus behaved similar to the salt, which forms a solid sphere upon complete drying. This assumption is reasonable because the final droplet diameter is largely determined by the salt component in the droplet, and the number of viruses in the droplet is relatively low. As you can estimate from the general concentration of SARS-CoV-2 virus in the saliva (7 x 106 ml-1 on average for SARS-COV-2 ref: <https://www.thelancet.com/action/showPdf?pii=S1473-3099%2820%2930196-1>), if you have a respiratory droplet of 100 μm, the average number of viruses in this droplet is 3.7 copies. Here, the assumption is that viruses distribute uniformly in the respiratory droplet, which may not be the case if they form aggregates of viruses.

**5. Other notes about the Matlab code**

MonteCarlo.m: You can use this function to generate “n” number of respiratory droplets, and it is going to plot the size distributions after a time of “t.” Running this function at conditions of 3000 and 10 will reproduce Fig. 1 of the manuscript (note that two viral loads need to be used). This function calls DropletEvolution.m

ContinuousSpeaking.m: You can use this function to generate the evolution of respiratory droplets generated from certain period time of speaking. This function calls the MonteCarlo.m.

DropletEvolution: You can use this function to adjust the parameters used in the simulation, such as the respiratory droplet size distributions, numbers, and environmental conditions. This function will solve the differential equation function EvapEquations.m

EvapEquations.m: This is the set of differential equations used in the manuscript solving for droplet size, droplet surface temperature, vertical distance, and vertical velocity.

Potential improvements: When we first submit the manuscript to other journals, reviewers also raised the point that the simulation of the droplet trajectories should use turbulent flows, which would enhance the evaporation and heat transfer further and elongate the residence time of airborne particles.