Response to the comments about the submitted paper "driftScalarDyFoam: An OpenFOAM-based multi-stage solver for drifting snow and its distribution around buildings"

We thank the reviewers for their constructive comments. We have addressed all of them and modified the paper accordingly. Our detailed answers follow.

Answers to Reviewer 1

> **Comment R1.1** Lines 45-49: definitions of u, j, and the physical definition of turbulent Schmidt number need to be added.

Answer to R1.1 u_j is the airflow velocity($j = 1, 2, 3$ for three directions). Turbulent Schmidt number is the ratio between the rates of turbulent transport of momentum and the turbulent transport of mass.

Modification to R1.1 The definitions of u_j(or u_i) and turbulent Schmidt number are added at Lines 45 and 54-55.

> **Comment R1.2** Since it is a paper about a new model, it's better to describe more details about the model, like what flow governing equation the authors have used, is there coupling between the flow phase and the granular phase, what method is used to save Eq. 11, and so on...

Answer to R1.2 The flow governing equation is a typical Navier–Stokes equation, which is now introduced in Eq. 1 and Eq. 2, and there is a one-way coupling between the granular phase and airflow, which is now stated at Lines 103-105. Eq. 11(Eq. 13 in revised manuscript) is a simple Laplacian equation of cell displacement C_i, so we discretize and solve this equation with the finite volume method for unstructured grids, just like the solution of the airflow velocity. The boundary conditions of C_i is easily obtained -- the grid size is static at locations far from the snow surface, where $C_i = 0$, and the C_i value on the snow surface is related to the mass exchange rate M_{total} caused by erosion/deposition(see Eq. 12 in revised manuscript).

Modification to R1.2 we have added two flow governing equations(Eq. 1 and 2), and added some details for our model(Line 103-105 and 129-133).

> **Comment R1.3** The authors should give more explanation on the number of stages. What's the relationship between stages and time step?

Answer to R1.3 "Stage" is a sub-concept of "Time step". As we know, in transient calculations, time integration is performed according to a specific time step, and the time step size must meet an appropriate Courant number to ensure robustness. Therefore, when the physical time is very long and the computational domain's grid size is tiny, the computational cost of directly using the transient model is very high. As a solution, we define a **large time step** as a **stage** and find a steady-state solution in it, and then update the snow distribution of the next time step with the previous calculation result. This actually simplifies the evolution of snow drifting in a specific time step as a linear process. The parameter sensitivity problem caused by this simplification has been discussed in Section 4.1.

Modification to R1.3 The concept of "Stage" is now introduced in Line 99-102.

> **Comment R1.4** The author should discuss more about the advantage of this model comparing to those previous numerical models.

Answer to R1.4 The numerical model used in this paper is a continuation of previous research, so there is no significant difference in nature. Our solution, on the other hand, can better analyze the relationship between time-varying snow distribution and snow transport by incorporating stage computations with dynamic mesh. Simultaneously, modern programming models can assist researchers in conducting parametric research on models in a easier way.

Modification to R1.4 We have added a discussion of the advantages of our model in Line 201-205.

> **Comment R1.5** Please add a short note whether this model could also apply in mountainous area, such as these 2 papers...

Answer to R1.5 Limited by the author's knowledge, this paper mainly discusses the snow transport around the building. The simulation for snow transport in mountainous areas with our model still needs further verification.

Modification to R1.5 In conclusion, we see the deployment of this methodology in mountainous area as a future work(Line 207-208).

Answers to Reviewer 2

> **Comment R2.1** It can be expected that OpenFOAM can be used in simulating the snowdrift. What is the advantage by using OpenFOAM than commercial CFD software?

Answer to R2.1 OpenFOAM is an open-source suite of libraries and applications designed to solve computational fluid dynamics(CFD) problems. Aiming at the simulation of snowdrift, using OpenFOAM has the following advantages:

1. The source code of OpenFOAM is available, which means that our research has good reproducibility and scalability.
2. OpenFOAM's code design features a decent level of hierarchy, with each low-level part's design isolated (mesh, numerical method, physical model...), and combined at the high-level application (solver, pre&post utilities...). In our simulation, the concept of "Stage" is proposed, which requires the simulation to perform multiple steady-state calculations and adjust the mesh after each calculation. In ANSYS Fluent, this process requires manual operation or the use of Text User Interface(TUI). In addition, the Fluent User-Defined Function(UDF) programming is not specific for different number of stages. In driftScalarDyFoam, using the Time class in OpenFOAM, the stage is easily set as a time step without time integration. In each stage, we set up some sub-cycles to simulate the iterations in the steady simulation. This means we only need to enter the number of stages and the duration of each stage in the control parameters to perform one simulation. Overall, OpenFOAM may assist us in better customizing the simulation process.
3. OpenFOAM supports direct abstraction for mathematical models. For the convection-diffusion equation like Eq. 3(in revised manuscript), we generally use User-Defined Scalars(UDS) module to implement in Fluent, and each term of the UDS transport equation is strictly limited by the DEFINE macros (DEFINE_ANISOTROPIC_DIFFUSIVITY, DEFINE_UDS_FLUX, DEFINE_UDS_UNSTEADY...).

OpenFOAM can directly implement similar functions through the following code segment more flexibly, in which a new mathematical equation for snow drifting is quickly created and solved.

```
fvScalarMatrix TEqn
(
  fvm::div(phi, T)
+ fvm::div(phiWf, T)
- fvm::laplacian(turbulence->nut()/S_ct, T)
==
  fvOptions(T)
);
TEqn.solve();
```

4. OpenFOAM's data structure is easily accessible. In the wind-induced snow simulation, an important physical process is that the snow surface is eroded due to aerodynamic entrainment, and the eroded snow particles join the airflow to produce snow transport. Simultaneously, some of the snow particles in the airflow is deposited on the snow surface. This necessitates that the simulation appropriately construct the snow concentration field ϕ and combine the snow concentration boundary conditions with the erosion/deposition on the boundary. In ANSYS Fluent, the definition of the erosion boundary condition and the adjustment of the dynamic mesh are carried out in different DEFINE macros, which poses a challenge to data transfer. Defining User-Defined Memory(UDM) or specifying global static variables may be some solutions. However, these methods have some limitations for complex requirements such as MPI parallelism and unstructured grids. In driftScalarDyFoam, the snow concentration ϕ and boundary mass exchange M_{total} are defined as scalars in each cell or face, and they can be accessed or even modified in any code segment. Thus, through the programmable boundary condition (codedFixedValue), we unify the snow distribution evolution and the snow concentration boundary condition. The author believes that this is beneficial for understanding the physical process of snowdrift and for further development.

> **Comment R2.2** It can be seen that your simulation agrees well with the existing publication. How does the conditions influence the results? Such as the snow supply and mass flux.

Answer to R2.2 For flat roofs where snow initially exists, the transport of snow comes from the aerodynamic entrainment due to high friction velocity. Therefore, the snow transport in Case I depends on the airflow, which has been discussed in Line 171-172. For Case II, where the snow concentration is explicitly specified at the inlet, the absolute snow height is related to the snow concentration, so we use the dimensionless normalized snow depth to present our calculation results, which is discussed in detail in **Answer to R2.3**.

> **Comment R2.3** In Fig.6, what is the dimension in the color bar?

Answer to R2.3 The dimension in the color bar is the dimensionless normalized snow depth. The snow depths are normalized by snow depth in the absence of building, here we use the snow depth at the outlet as the reference snow depth, which is far away from the building in the calculation domain. In other words, the dimensionless normalized snow depth at a location far from the building is 1.

Modification to R2.3 The description of the normalized snow depth has been added to the caption of Fig. 6.

> **Comment R2.4** It will be great if you could show the flow at the cross-section at y/H=0, which can be compared with the results in Fig.7.

Modification to R2.4 The airflow at the cross-section at $y/H = 0$ is now shown in Fig. 8, and we have discussed the relationship between snow distribution and flow field at Line 192-195.