SUNSET code: Scalable Unstructred Node-SET code for high fidelity fluid simulations in complex geometries

J. R. C. King*
Department of Mechanical, Aerospace and Civil Engineering,
The University of Manchester, Manchester, M13 9PL, UK
(Dated: September 4, 2022)

This document contains a brief description and guide to the SUNSET code. Numerical methods in this code follow [1] (also available at arXiv:2102.02019).

I. DIRECTORY STRUCTURE

II. GOVERNING EQUATIONS

The code solves the compressible Navier-Stokes equations in the form

$$\frac{\partial \ln \rho}{\partial t} + \boldsymbol{u} \cdot \nabla \ln \rho = -\nabla \cdot \boldsymbol{u} \tag{1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\frac{1}{\rho}\nabla p + \frac{1}{\rho}\nabla \cdot \boldsymbol{\tau} + \boldsymbol{f} \tag{2}$$

$$\frac{\partial \rho E}{\partial t} + \boldsymbol{u} \cdot \nabla (\rho E) = -\nabla \cdot (p\boldsymbol{u}) - \nabla \cdot \boldsymbol{q} + \nabla \cdot (\boldsymbol{\tau}\boldsymbol{u}) + \boldsymbol{u} \cdot \boldsymbol{f} \tag{3}$$

and

$$\frac{\partial Y_\alpha}{\partial t} + \boldsymbol{u} \cdot \nabla Y_\alpha = \frac{\dot{\omega}_\alpha}{\rho} - \frac{1}{\rho}\nabla (\rho \boldsymbol{V}_\alpha Y_\alpha) \qquad \forall \alpha \in [1, N_S] \tag{4}$$

where ρ is the density, $\boldsymbol{u}$ the velocity, p the pressure, E the total energy, $\boldsymbol{\tau}$ the viscous stress, $\boldsymbol{f}$ a body force, $\boldsymbol{q}$ the heat flux, Y_α the concentration of species $\alpha \in [1, N_S]$, and $\boldsymbol{V}_\alpha$ is the molecular diffusion velocity of species α. $\dot{\omega}_\alpha$ is the production rate of species α.

The viscous stress is defined as

$$\boldsymbol{\tau} = \mu \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T - \frac{2}{3}\boldsymbol{I}\nabla \cdot \boldsymbol{u} \right), \tag{5}$$

where μ is the dynamic viscosity and $\boldsymbol{I}$ is the identity tensor. The energy is

$$E = \sum_\alpha h_\alpha Y_\alpha - \frac{p}{\rho} + \frac{1}{2}\boldsymbol{u} \cdot \boldsymbol{u} \tag{6}$$

where the sum over α is over all $\alpha \in [1, N_S]$, and h_α is the enthalpy of species α, given by

$$h_\alpha = \int_{T_{ref}}^{T} c_{p,\alpha} dT + h_{\alpha,ref} \tag{7}$$

with $h_{\alpha,ref}$ the enthalpy of species α at the reference temperature T_{ref}, and $c_{p,\alpha}$ the specific heat capacity at constant pressure of species α, which in general may be a function of temperature T. The molecular diffusion term in (4) is modelled as a Fickian diffusion process with

$$-\rho \boldsymbol{V}_\alpha Y_\alpha = \rho D_\alpha \nabla Y_\alpha - Y_\alpha \sum_{\beta \in [1, N_S]} \rho D_\beta \nabla Y_\beta, \tag{8}$$

* jack.king@manchester.ac.uk

in which D_α is the molecular diffusivity of species α, and the final term is a correction to ensure that

$$\sum_\alpha Y_\alpha = 1. \tag{9}$$

The heat flux vector $\boldsymbol{q}$ is then given by

$$\boldsymbol{q} = -\lambda \nabla T + \sum_\alpha \rho \boldsymbol{V}_\alpha Y_\alpha h_\alpha, \tag{10}$$

where λ is the thermal conductivity of the mixture. The system is closed with an ideal gas equation of state

$$p = \rho R_{mix} T = \rho R_0 T \sum_\alpha \frac{Y_\alpha}{W_\alpha} \tag{11}$$

where R_{mix} is the gas constant for the mixture, R_0 is the universal gas constant, and W_α is the molar mass of species α.

A. Evaluation of temperature and heat capacity

TBC.
TBC for cp independent of T.

B. Temperature dependent transport properties

The temperature dependence of the transport properties is modelled by the relationship

$$\mu = \mu_{ref} \left(\frac{T}{T_{ref}} \right)^{r_T}, \tag{12}$$

where μ_{ref} is the viscosity at the reference temperature T_{ref}, and $r_T = 0.7$ is a constant exponent. The mixture thermal conductivity is given by

$$\lambda = \frac{\mu}{c_p Pr}, \tag{13}$$

in which c_p is the heat capacity of the mixture, and Pr is the Prandtl number, taken as constant. The molecular diffusivity of species α is given by

$$D_\alpha = \frac{\lambda}{\rho c_p Le_\alpha}, \tag{14}$$

where Le_α is the Lewis number for species α, which is assumed constant for each species.

C. Evaluation of reaction rates

TBC.

III. NUMERICAL IMPLEMENTATION

A. Basic framework and principles

The domain is discretised with an unstructured node-set. Each node (or collocation point) i has position $\boldsymbol{r}_i$, discretisation length-scale s_i (the average local inter-node spacing), and holds the primary variables $\ln \rho_i$, $\boldsymbol{u}_i$, ρE_i. The governing equations are solved on the set of nodes. Spatial derivatives are evaluated using LABFM [1] in the

first two dimensions (x, y) and high-order finite differences (FD) in the third dimension (z). Technically, the node-set is semi-unstructured...

Time integration is by third-order Runge-Kutta scheme. The solution is de-aliased with a high-order LABFM+FD filter after each time step [1]. The accuracy of the spatial discretisation can be changed through compiler flags, between 4^{th} and 12^{th} order accurate in space. In typical usage, 8^{th} order provides a good compromise between accuracy and cost.

At wall, inflow and outflow boundaries, the node-set is structured, with a band of 5 rows of uniformly spaced nodes. Wall boundaries may be curved, but *at present,* must have uniform resolution along their length. On boundaries (and within the first 2 rows of uniform nodes) the derivatives and filters in the $x - y$ plane are calculated through combinations of FD (normal to boundaries), one-dimensional LABFM (tangential to boundaries), and 2D LABFM. Details in [1].

Aside from the complexities of indexing and derivative evaluation associated with the unstructured node-set approach, the framework is similar to a high-order (central) finite difference scheme.

B. Evaluation of derivatives

Derivatives of the primary variables $\ln \rho$, $\boldsymbol{u}$, ρE and Y_α are evaluated with the LABFM method [1]. Temperature gradients ∇T are also evaluated using LABFM, as are Laplacians where required (i.e. $\nabla^2 \boldsymbol{u}$, $\nabla^2 T$, $\nabla^2 Y_\alpha$).

To avoid the explicit evaluation of cross-second derivatives of velocity (which appear in the terms $\nabla \cdot \boldsymbol{\tau}$ and $\nabla \cdot (\boldsymbol{\tau u})$), these terms are constructed by evaluating the gradient of the velocity divergence $\nabla \cdot \boldsymbol{u}$.

The evaluation of derivatives using LABFM constitutes the bulk of the computational cost, and it is hence desirable avoid where possible. Gradients of secondary properties may be evaluated from analytic expressions as follows. The pressure gradient is calculated from the temperature gradient as

$$\nabla p = p \left(\frac{R_0}{R_{mix}} \sum_\alpha \frac{\nabla Y_\alpha}{W_\alpha} + \frac{1}{T} \nabla T + \nabla \ln \rho \right). \tag{15}$$

Gradients of mixture specific heat capacity and enthalpy are also evaluated through analytic expressions in terms of temperature gradients TBC. Similarly, the gradients of the transport properties, which have analytic relations with the thermodynamic quantities, are evaluated as

$$\nabla \mu = \frac{r_T \mu}{T} \nabla T \tag{16}$$

$$\nabla \lambda = \frac{r_T \lambda}{T} \nabla T + \frac{\lambda}{c_p} \nabla c_p \tag{17}$$

$$\nabla D_\alpha = \frac{r_T D_\alpha}{T} \nabla T - D_\alpha \nabla \ln \rho \tag{18}$$

C. Boundaries

Symmetry and periodic boundary conditions are available. The only boundary condition available in the third dimension is periodic. Wall, inflow and outflow boundaries are imposed through the Navier-Stokes characteristic boundary condition formalism of [citeXX]. The basic framework follows that described for isothermal flows in [1].

D. Parallelisation

The code is accelerated with a shared-distributed framework, using both OpenMP and MPI. It should be noted that the parallel efficiency of the OpenMP is low, due to memory-access issues. Provided sufficient memory is available, it is therefore recommended to run the code with very few threads per MPI rank: ideally 1, but any more than ≈ 5 and resources are being used inefficiently.

Domain decomposition is three-dimensional, and *currently* static. The total number of MPI ranks is $N_p = N_{px} N_{py} N_{pz}$, where the subscripts x, y and z indicate the number of ranks in each direction. The physical sizes

of the subdomains are adjusted such that each MPI rank contains the same number of collocation points, resulting in approximate load balancing. On simple domains the parallel efficiency is $> 96\%$ at 1024 MPI ranks (relative to 4 MPI ranks).

The domain-decomposition is likely to be restructured in the coming year, so I'll leave further details for now.

IV. GENERATING INPUT FILES

THIS SECTION TBC. CURRENTLY BASED ON OLD BUBBLY CODE

Input files are generated using the program `gen2d`, found in `source/gen/`. Navigate here then

```
make
./gen2d
cp I* ../../.
```

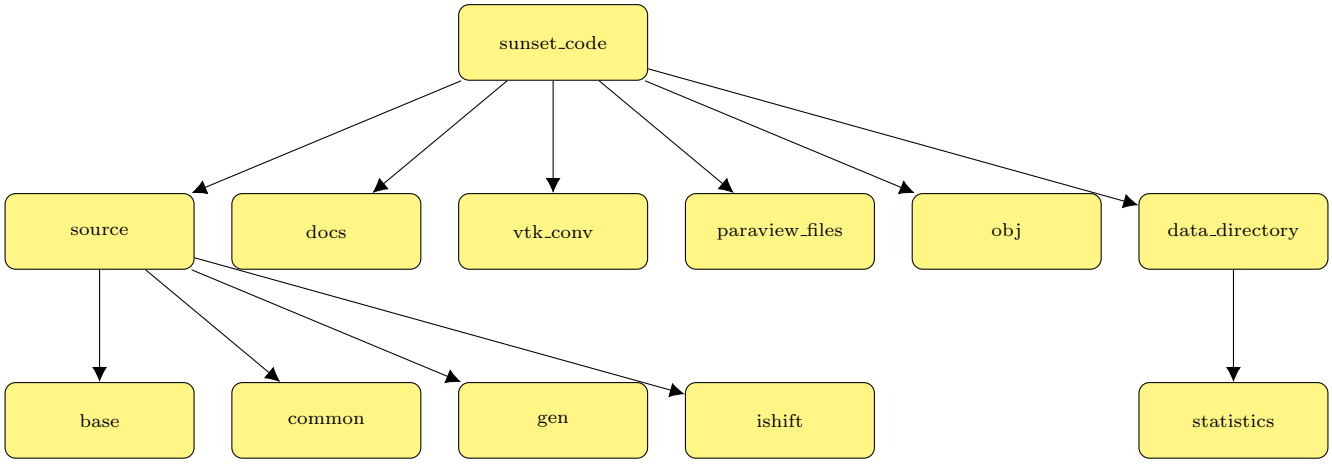
When `gen2d` is run, a list of cases is displayed. Input a number to specify a case. Once generated, you can use the command `cp I* ../../.` to copy the case files `INDAT`, `IPART`, and `IBOUND` to the main directory. Cases are hard-coded in the program `gen2d`. To modify them, or create new cases, you need to modify the file `datclass.F90`, and recompile with `make`.

Within each case in `datclass.F90`, domain sizes (`xl`, `yl`, and `h0`) are specified, along with the resolution. Then dimensionless parameters are specified. Followed by boundary conditions (described below), and then the specification of the initial particle distribution. Note that entire problem is non-dimensionalised, so the domain specification and intitial velocity fields should have characteristic scales of unity. Also note that `gen2d` generates two-dimensional case files, and a parameter `nz` which specifies the number of particles in the third dimension (if a three-dimensional simulation is run).

The following cases are currently included in `datclass.F90`:

1. Still water/bubble column
2. Dam Break (Lobovsky)
3. Taylor-Green
4. Poiseuille/channel flow
5. Third order Stokes wave

A. Boundary framework



- `source` contains source files for the code:
  - `base` contains main modules for `sunset_code`
  - `para` contains parameters and common variables

  - `gen` contains code to generate casefiles and initial conditions

- `vtk_conv` contains a program to convert output files into `.vtu` files, which can be read by Paraview.
- `paraview_files` the program in `vtk_conv` creates `.vtu` files here.
- `obj` contains `.o` and `.mod` files created during compilation.
- `docs` contains this document and associated files.
- `data_directory` contains output files produced by the code. `statistics` contains files describing global, time-varying measures (e.g. total K.E.).

---

[1] J. King, S. Lind, High-order simulations of isothermal flows using the local anisotropic basis function method (labfm), Journal of Computational Physics 449 (2022) 110760. doi:doi:https://doi.org/10.1016/j.jcp.2021.110760.