This item was submitted to [Loughborough's Research Repository](#) by the author.
Items in Figshare are protected by copyright, with all rights reserved, unless otherwise indicated.

Parallel load-balancing for combustion with spray for large-scale simulation

PLEASE CITE THE PUBLISHED VERSION

<https://doi.org/10.1016/j.jcp.2021.110187>

PUBLISHER

Elsevier

VERSION

AM (Accepted Manuscript)

PUBLISHER STATEMENT

This paper was accepted for publication in the journal Journal of Computational Physics and the definitive published version is available at <https://doi.org/10.1016/j.jcp.2021.110187>

LICENCE

CC BY-NC-ND 4.0

REPOSITORY RECORD

Thari, Ali, Nicholas Treleaven, Max Staufer, and Gary Page. 2021. "Parallel Load-balancing for Combustion with Spray for Large-scale Simulation". Loughborough University. <https://hdl.handle.net/2134/14034824.v1>.

Parallel Load-Balancing for Combustion with Spray for Large-Scale Simulation

A. Thari^{a,*}, N.C.W. Treleaven^b, M. Staufer^c, G.J. Page^a

^a*Department of Aeronautical and Automotive Engineering, Loughborough University,
Loughborough, LE11 3TU, United Kingdom*

^b*CERFACS, CFD Team, 42 Av. Gaspard Coriolis, 31057, Toulouse, France*

^c*Rolls-Royce Deutschland Ltd & Co KG, 15827 Blankenfelde-Mahlow, Germany*

Abstract

An asynchronous task-based Eulerian-Lagrangian approach for efficient parallel multi-physics simulations that can scale for arbitrary large number of particles and non-uniformly distributed particles is presented. The parallel methodology is based on a task-based partitioning of the multi-physics problem, where each single-physics problem is considered as a task and carried out using its own set of processes. This allows the two problems to solve their governing equations concurrently; therefore, hiding the computational cost incurred of solving an additional physical solver. Applications to complex breakup mechanism leading to highly dynamic computational loading and three-dimensional swirl combustion chamber with reacting flow/spray with extremely uneven particle distribution demonstrate the improved parallel efficiency and great potential of the presented approach.

Keywords: Task-based parallelism, Asynchronous multi-physics, shared-memory communication, Eulerian-Lagrangian, Combustion with spray

1. Introduction

Large scale computational problems are commonplace in a wide range of application areas such as the aerospace, automotive and oil industries. Par-

*Corresponding author

Email address: Ali.Thari@lboro.ac.uk (A. Thari)

4 allel computing is a vital tool to ensure that realistic and accurate computa-
5 tional simulation may be conducted within a reasonable time frame. Parallel
6 computing for numerical simulation such as Computational Fluid Dynam-
7 ics (CFD) is usually based on the domain decomposition approach. That
8 is, the global problem is divided into a number of local problems (or parti-
9 tions) using a spatial decomposition and each are solved semi-independently
10 on a computing process. The efficiency of this approach is impacted by any
11 uneven work-load across the partitions and delays caused by data transfer
12 between partitions. For CFD of aerodynamic problems using a finite vol-
13 ume discretisation, domain decomposition is well established and can scale
14 to thousands of processing units. However, for multi-physics problems which
15 solve a variety of equations – some which may not be in finite volume form –
16 then efficient parallelisation is more challenging (Jofre et al., 2015; Vázquez
17 et al., 2016) and is the focus of this work.

18 The general model of large-scale computer systems consists of several in-
19 dependent computer nodes each with ownership of local memory. The nodes
20 within a computer system are connected to a network that provides the abil-
21 ity to communicate among the computer nodes. Each node can either be
22 a single processor or multiprocessor systems where each processor contains
23 several cores. All cores have equal access to its node’s memory. Nowadays,
24 large-scale clusters are moving towards larger core counts per node¹ which
25 implies more cores sharing same memory and less memory- and communi-
26 cation bandwidth per core. The Message Passing Interface (MPI) library
27 is commonly used to communicate among cores irrespectively whether they
28 share memory or sit at a different node. A simulation program is launched
29 by a number of MPI processes that is usually equal to the total number of

¹<https://www.archer2.ac.uk/about/hardware.html>

30 cores considered: a practice used in this work. The MPI rank is frequently
31 used as shorthand for the MPI process ID.

32 A good multi-physics example is CFD modelling of gas turbine combus-
33 tion. The flow-field in a combustor is highly turbulent, chemically reacting
34 and consists of multiple fluid phases and has significant effects from radiative
35 heat transfer. Liquid fuelled engines work by using a fuel injector to create a
36 spray of fuel droplets, whilst the aerodynamic design creates a high swirl to
37 provide a stable combustion zone for gaseous fuel generated from evapora-
38 tion. From the fuel injector, there is primary and secondary break-up of fuel
39 ligaments into droplets (Fig. 1), and these may then experience evaporation,
40 condensation, collision and coalescence, before finally feeding the gaseous
41 phase and are combusted. Whilst the initial fuel ligaments may be conceiv-
42 ably be captured by an Eulerian grid based approach using techniques such
43 as Volume of Fluid, the number and size of droplets are impractical to be
44 resolved by an Eulerian grid, and so a Lagrangian particle tracking method is
45 normally adopted. With ever tightening legislation regarding NOX and soot
46 emissions from aero engine gas turbines, then it is critical to have fast and
47 accurate simulation processes to assess candidate designs. This can only be
48 achieved if the methods exhibit good parallel scaling for very large numbers
49 of fuel droplets.

50 This multi-physics combustion problem is frequently solved by means of
51 a coupled Eulerian-Lagrangian approach (Apte et al., 2003a; Kahila et al.,
52 2018; Aguerre and Nigro, 2019). The reacting gas flow is solved using a
53 finite-volume method on an Eulerian grid while the fuel droplets/particles
54 are tracked using a Lagrangian approach. Although such methodologies over-
55 come issues related to extreme mesh resolutions, the droplet count near the
56 injector is of the order of 50 million (Apte et al., 2003a, 2009), while further

Figure 1: An Air-blast atomiser sketching the breakup process from the liquid film into tiny dispersed droplets which eventually evaporate.

57 downstream they evaporate into vapour. This is one of the problems for
 58 parallel computing load balancing using a geometric partitioning: within a
 59 given region the droplet density may be very different to the finite-volume
 60 density(e.g. Sitaraman and Grout (2016)). Although it is possible for a par-
 61 titioning algorithm to use a locally varying weighting to compensate, it is
 62 challenging to determine these a priori. And, in our case with a dynam-
 63 ically changing loading where droplets constantly are injected, atomised,
 64 moved from partition to another and eventually eliminated due to evap-
 65 oration would require a dynamic repartitioning technique which can have
 66 unacceptable overheads (Rettinger and Rde, 2019).

67 The Eulerian-Lagrangian algorithm is generally implemented within a
 68 single program and the Eulerian and Lagrangian equations are solved se-
 69 quentially (Houzeaux et al., 2016) (i.e. within a time-step, the flow-field
 70 update is computed, then the new flow-field passed to the particle tracking

71 and the particle update is computed). A straightforward parallelisation can
 72 be achieved by assigning all particles within one Eulerian sub-domain to its
 73 corresponding computing unit. This method is also referred as the spatial
 74 partitioning approach. Data transfer for the Lagrangian part only occurs
 75 when particles move between sub-domains. This has been a standard ap-
 76 proach for combustion applications (Pankajakshan et al., 2011) and as noted
 77 earlier can result in unbalanced loading and poor parallel efficiency in ap-
 78 plications where the particles are clustered within only a few number of sub
 79 domains (e.g. near the fuel injector) (Apte et al., 2003a,b). The sequential
 80 approach ignores the inherent parallelism of the problem where both flow
 81 and particle updates could be computed simultaneously using the conditions
 82 at the beginning of the time step. This is a key element of the current work.

83 A second parallel approach divides the droplets equally among all com-
 84 puter cores ignoring their spatial location (Sbalzarini et al., 2006; Shigeto and
 85 Sakai, 2011; Kafui et al., 2011; Sitaraman and Grout, 2016): this is referred
 86 to as a particle sharing algorithm. Although it ensures load balancing, this
 87 method requires collective communication in order to keep track of which
 88 Eulerian domain partition the droplets reside. On computations with large
 89 core counts, this can have an adverse effect on scalability. Some improve-
 90 ments have been made by maximising the ownership fraction of the droplets
 91 and processor ranks using an equalization algorithm (Sitaraman and Grout,
 92 2016). Fig. 2 illustrates the two parallel algorithms for particle tracking,
 93 where Fig. 2b illustrates the particle sharing algorithm. In this figure the
 94 coloured boxes represent the geometric partition of the grid and each colour
 95 represents a rank, whilst the circles represent the particles which use the
 96 same colour coding to show in which rank they reside. Overwhelmingly the
 97 method of Fig. 2a is the most common, due to its conceptual and practical

simplicity. However, neither of the methods in the figure are suitable for large scale computations, especially moving towards exascale.

Figure 2: (a) Spatial partitioning approach where each rank tracks the particles that are located within the same sub domain. (b) Particle sharing algorithm would only ensure an equal amount of particles of each rank regardless of their spatial location.

As mentioned earlier, the benefit of a spatial based algorithm is that it reduces communication by having the local flow information and particles on the same processor. When the physics of the problem does not impose a severe non-uniform distribution of particles, this approach is very efficient. Work by Capecelatro and Desjardins (2013) showed good scale-up for cases involving 382 million particles with 4096 CPUs. On the other hand, the work presented by Kaludercic (2004); Pankajakshan et al. (2011) have demonstrated its significant limitation when considering situations with uneven distribution of particles across the domain. One way to overcome this is by the mirror domain decomposition by Darmana et al. (2006). All cores receive a copy of the data, but each processor is only responsible for computing updates on a subset of particles - after each time step the updates are then synchronised. Unfortunately, the method was shown to be efficient only up to 32 processors due to the increased communications in the synchronisation stage. In addition, for large numbers of particles and large core counts,

115 each processor is requiring a significant amount of memory to store infor-
 116 mation on particles that it is not computing. Another approach suggested
 117 by Frank et al. (2001) is to have a servicing core for each calculation core.
 118 The servicing cores make sure to distribute an even amount of workload to
 119 each calculation core. It was demonstrated to be viable up to 64 proces-
 120 sors, although the parallel efficiency had reduced down to 40 %. Zhao et al.
 121 (2004) presented a dynamic load balancing approach to alleviate the imbal-
 122 ance issue. The performance gain was limited after 8 processors due to the
 123 increased overhead of the repartitioning process. Another critical element to
 124 this approach is the required collective communication. A bookkeeping pro-
 125 cedure must be carried out before each data exchange among all cores. This
 126 is to provide each CPU information on how many particles have entered its
 127 grid domain and from which ranks. A large number of these communications
 128 will return a null value because no particles have passed to its corresponding
 129 partition. Sitaraman and Grout (2016) outlines this issue and presents a
 130 different method employing one-sided remote memory access (RMA) com-
 131 munication; an approach which is introduced in the MPI 2 standard (Gropp
 132 et al., 1999).

133 A similar communication strategy has been suggested by Buaria and Ye-
 134 ung (2017) & Fanfarillo and Del Vento (2018) which employ the Partitioned
 135 Global Address Space (PGAS). This is simply a form of Remote Direct Mem-
 136 ory where the memory address space is logically partitioned. Each processor
 137 has its own local portion of the global memory. A processor can retrieve
 138 information from another processor by knowing the data locality. Commu-
 139 nication is reduced to a minimum through the utilisation of PGAS. This
 140 has been showcased for DNS problems incorporating 300 million particles
 141 and thousands of cores. However, the work only examines cases with uni-

142 form particle distribution meaning that the issue with work imbalance is not
143 considered.

144 A few papers have examined a hybrid message-passing/shared memory
145 approach (e.g. MPI-OpenMP) (Yakubov et al., 2013). Here, the Eulerian
146 solver benefits from the distributed memory through MPI and the Lagrangian
147 solver takes advantage from the shared memory multiprocessing. The basic
148 idea is to launch a specific number of OpenMP threads to solve the La-
149 grangian equations once the MPI processes have advanced the Eulerian solver
150 one time step. The parallelism of the Lagrangian solver is limited to declared
151 OpenMP constructs, which scales well for problems with a much small num-
152 ber of particles as demonstrated by (Yakubov et al., 2013) with applications
153 involving up to 100,000 particles.

154 Amritkar et al. (2014) presents a parallel framework based solely on
155 shared memory using OpenMP. This study combines the different parallel
156 modes, i.e. domain decomposition and particle sharing algorithms, by ex-
157 ploiting the flexibility offered by OpenMP. That is, the threads alternate
158 between running across the blocks and particle numbers. Changing modes
159 of parallelism is relatively cost-free in OpenMP where it would have involved
160 large overheads with MPI. The work demonstrated appreciable improvements
161 in performance compared to a pure MPI implementation. However, this
162 approach is limited to systems with shared-memory configuration implying
163 that the approach is not suitable for simulation run across multiple com-
164 puter nodes. Houzeaux et al. (2016) employs a hybrid approach to solve
165 the Eulerian-Lagrangian problem. Instead of having a single code execution,
166 a multi-code approach where each solver/code acquires separate computer
167 nodes, is applied to enable overlapping between the Eulerian and Lagrangian
168 calculations. OpenMP is used for the main loops in the Lagrangian Parti-

169 cle Tracking (LPT) solver within each MPI subdomain. Also, it applies a
 170 Dynamic Load Balancing library (DLB) which ensures a reasonable work-
 171 load across the threads at run time. That is, DLB tracks every MPI process
 172 and when it detects that some are waiting or blocked, it will lend their re-
 173 sources/cores to spawn more OpenMP threads within the same node. The
 174 approach has been tested for large scale HPC application including up to 64
 175 computational nodes with 16 cores per node. While applying DLB showed
 176 distinct improvement in all the cases, the computational gain between single-
 177 code and multi-code was not as definite. As aforementioned, parallelism of
 178 OpenMP applications is limited to selected sections in the code and can there-
 179 fore become significantly less efficient than using a pure MPI implementation.
 180 Also, the single-code approach benefits from better data locality with respect
 181 to exchanging data between the Eulerian and Lagrangian solvers, while its
 182 counterpart requires a send-receive communication pair each time-step.

183 The combination of large droplet count and the poor parallel efficiency
 184 for Eulerian-Lagrangian combustion applications lead the community to use
 185 an approximation by bundling particles into parcels. That is, each parcel
 186 (represented by a single computational particle) represents a number of real
 187 particles which share similar features, such as mass, radius and tempera-
 188 ture. These will be generated by preserving mass conservation and repre-
 189 sent a considerably lower count compared to the actual number of droplets
 190 (Apte et al., 2003a,b; Kahila et al., 2018). This obviously has a large ben-
 191 efit in terms of computation time, but at the cost of simulation accuracy.
 192 Tracking each individual droplet in a parallel computing framework with
 193 a domain-decomposition and spatial particle algorithm is not possible for
 194 realistic problems as the uneven load balancing will stop the parallel scal-
 195 ing. However, the diffusion of flames is controlled by their stretching that

196 is directly influenced through the mixing (Veynante and Vervisch, 2002; van
197 Oijen et al., 2016). Large-Eddy-Simulations (LES) is generally considered
198 essential to model such turbulence mixing and combustion dynamics even
199 for industrial applications (Pierce and Moin, 1998; Apte et al., 2003b; Kahila
200 et al., 2018). Hence, a parcel approach for LES simulations is less suitable
201 since such a method provide an averaged particle trajectory contrary to the
202 instantaneous field provided by the LES. Apte et al. (2003a) presented a
203 hybrid particle-parcel approach to overcome this and towards a more LES
204 suited particle tracking methodology.

205 The novel contribution of this work is a paradigm change for parallelisa-
206 tion of multi-physics problems such as Eulerian-Lagrangian spray modelling
207 in combustion applications. We will present a task parallel framework ex-
208 ploiting shared memory communication and large core counts per node. The
209 novelty is to split the Eulerian-Lagrangian solvers into two separate tasks,
210 which allows the Eulerian and Lagrangian parts to carry out their respec-
211 tive operations concurrently. The interchange of data between the solvers
212 is conducted using one-sided shared memory communication and avoids the
213 normal send-receive communication pair. The work presented here currently
214 focusses on single-node performance, which represents a crucial first step
215 towards achieving efficient parallel computing across multiple nodes. The
216 aim of this work is to present a parallel framework that can achieve almost
217 perfect single-node parallel efficiency for complex multi-physics problems by
218 maintaining an even load balancing for arbitrary non-uniform particle dis-
219 tribution. This can then be expanded to multiple node machines that will
220 enable the community to conduct cost-efficient large-scale parallel simula-
221 tions for liquid fuelled aeroengine combustors.

222 In this work, two different parallelisation of an industrial CFD code will be

223 evaluated to validate and verify the novel parallelisation concept. The first is
224 the baseline code which has been extensively used by industry for simulating
225 reacting flows and designing gas turbine aero engines: **PRECISE-UNS** (Cha
226 et al., 2005; Anand et al., 2013; Zhu et al., 2013). It features the common
227 coupling methodology between the Eulerian and Lagrangian phase where
228 the code advances selectively one time-step with the Eulerian gas field and
229 subsequently a time-step with the Lagrangian liquid phase. The parallel
230 methodology is also based on the common spatial particle algorithm which
231 suffers excessively from parallel scalability for large scale simulations. The
232 second version is strongly based on the latter in terms of physical modelling.
233 However, the Eulerian and Lagrangian parts have been split into two separate
234 processes that communicate together over the course of a simulation. This
235 version will also feature the new parallel methodology presented in this work,
236 namely a two-way coupled Asynchronous Task-based Eulerian-Lagrangian
237 solver (ATEL). This work will be validated by comparing two different test
238 cases to the baseline code, but also to available experimental and CFD data.

239 The paper is structured as follows. Section 2 provides a description of
240 the mathematical models and the two-way coupling methodology between
241 the Eulerian and Lagrangian phase. In Section 3, the parallel framework
242 of ATEL is explained in great detail which begins the asynchronous and
243 parallel task-based coupling approach. This is followed by a description of
244 the shared memory method employed for the communication between the
245 parallel tasks. Finally, the mapping and synchronisation between the flow
246 and spray ranks are explained. This is followed by Section 4 where the
247 first study-case is carried out on two-dimensional nitrogen chamber with a
248 complex breakup mechanism. Section 5 presents the second case of three-
249 dimensional combustor with reacting flow. Both test-cases are validated,

rigorously tested and analysed for parallel performances. Concluding remarks
are given in Section 6.

2. Mathematical Models

The section describes the governing equations of both the gaseous and
liquid phases. A Large-Eddy-Simulation (LES) approach is used with unre-
solved turbulence modelled by the Smagorinsky Subgrid Scale (SGS) model.
The two-way coupling between the two phases is included. The relative ve-
locity between the gas and droplet is incorporated to calculate the forces
on the droplets, while each droplet is treated as a point-source through the
mass, momentum and energy gas equations.

2.1. Gas phase equations

The Eulerian flow field is governed by the three-dimensional governing
LES Favre-filtered Navier-Stokes equations which represents the mass, mo-
mentum and energy equations,

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial (\bar{\rho} \tilde{u}_i)}{\partial x_i} = \dot{m}_d \quad (1)$$

$$\frac{\partial \bar{\rho} \tilde{u}_i}{\partial t} + \frac{\partial}{\partial x_j} (\bar{\rho} \tilde{u}_i \tilde{u}_j) + \frac{\partial \bar{p}}{\partial x_j} = \frac{\partial}{\partial x_i} [\bar{\rho} (\widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j) + \tau_{ij}] + S_{i,d}, \quad (2)$$

$$\begin{aligned} \frac{\partial \bar{\rho} \tilde{h}}{\partial t} + \frac{\partial}{\partial x_i} (\bar{\rho} \tilde{u}_i \tilde{h}) = & \frac{\partial \bar{p}}{\partial t} + \overline{u_i \frac{\partial p}{\partial x_i}} - \frac{\partial}{\partial x_i} [\bar{q}_i + \bar{\rho} (\widetilde{u_i h} - \tilde{u}_i \tilde{h})] \\ & + \overline{\tau_{ij} \frac{\partial u_i}{\partial x_j}} + Q_d \end{aligned} \quad (3)$$

where u_i , p , ρ , q_i and τ_{ij} are the gas velocity, pressure, density, energy
flux and viscous tensor, respectively. The bar ($-$) and tilde ($\sim$) represent

the unweighted and density weighted average, respectively. The energy Eq. (3) solves for the absolute enthalpy which includes the chemical formation enthalpy of species. The mass, momentum and energy source terms ($\dot{m}_p$, $\dot{S}_p$, Q_p) due to the fuel droplets will be discussed further below. Further details regarding the governing equations and turbulence modelling is given in Appendix A.

2.1.1. Combustion Model

Combustion simulations with complex fuel like kerosene or diesel can involve hundreds of different species which is computationally impractical to solve for each of their respective transport equations. Therefore, chemical reduction and combustion modeling are often employed to compute the combustion processes in a more efficient manner (van Oijen et al., 2016; Poinot and Veynante, 2005). In this work, a Flamelet-Generated-Manifold (FGM) with a presumed beta-Probability Density Function (β -PDF) combustion model is used, which employs a pre-computed tabulation of various thermodynamic, transport and turbulence-chemistry interaction properties of different premixed flames. The mixture fraction Z and progress variable ϕ and their respective variances are used to construct the FGM look-up table. The mixture fraction Z ranges in the interval from pure air to pure fuel $[0, 1]$. The progress variable ϕ represents the reaction progress within the flame and is defined as zero at the unburnt region. There are different ways to define the progress variable (van Oijen et al., 2016) and the one used here is based on the mass fraction of CO_2 , H_2O and H_2 (See Appendix A). Additionally, the variances of each of the two control variables are solved as transport equations to incorporate the turbulence chemistry interaction. Effectively, this means that the combustion model adds four additional transport equations and are presented in Appendix A. When a spray model (See Section 2.2)

is incorporated to the LES filtered control variable of the mixture fraction must be modified to include a source term $\dot{\omega}_Z$ that represents the amount of fuel spray that has been evaporated in the last timestep as calculated by the Lagrangian solver. The modified transport equation of mixture fraction becomes:

$$\frac{\partial \bar{\rho} \tilde{Z}}{\partial t} + \frac{\partial}{\partial x_i} (\bar{\rho} \tilde{u}_i \tilde{Z}) = \frac{\partial}{\partial x_i} \left[\nu_t \frac{\partial \tilde{Z}}{\partial x_i} \right] + \dot{\omega}_Z \quad (4)$$

2.2. Spray Model

The Spray model is included in the Lagrangian spray where each droplet's trajectory is tracked using

$$m_d \frac{du_{i,d}}{dt} = F_{i,d} + G_{i,d} + H_{i,d}, \quad (5)$$

where m_d and $u_{i,d}$ are the mass and velocity of droplet d . The drag force $F_{i,d}$ is calculated by

$$F_{i,d} = C_d (Re_d) \frac{1}{2} \rho |\mathbf{u}_{relv}| u_{i,relv} S_d, \quad (6)$$

where $\mathbf{u}_{relv}$ is the relative velocity between the gas and liquid phase. The frontal area of the fuel droplet is denoted as S_d and Re_d is the Reynolds number based on the droplet diameter D_d and relative velocity

$$Re_d = \frac{\rho |\mathbf{u}_{relv}| D_d}{\mu}, \quad (7)$$

The drag coefficient C_d is a function of droplet Reynolds number Re_d (Liu et al., 1993)

$$C_d = \begin{cases} 24 (1 + 0.15 Re_d^{0.687}) / Re_d & Re_d \leq 1000 \\ 0.424 & Re_d > 1000 \end{cases} \quad (8)$$

The virtual force $G_{i,d}$ accounts for the additional fluid surrounding a droplet which needs to be accelerated when the droplet accelerates

$$G_{i,d} = -C_{vm} \rho \Omega \frac{d(u_{i,relv})}{dt}, \quad (9)$$

where the virtual mass constant, $C_{vm} = 0.5$. The body forces on the droplet $H_{i,d}$ is neglected in this work.

2.2.1. Evaporation model

The evaporation model employed in this work is proposed by Chin and Lefebvre (1983), where the mass of each particle is controlled by the evaporation rate

$$\frac{dm_d}{dt} = 2\pi D_d \left(\frac{\lambda}{C_p} \right)_g \ln(1 + B_m), \quad (10)$$

where λ and C_p are the mean thermal conductivity and specific heat of the gas, respectively. The mass transfer number B_M is defined as

$$B_M = \frac{Y_{F_s}}{1 - Y_{F_s}}, \quad (11)$$

where the mass fraction of fuel vapour at the droplet surface is given by

$$Y_{F_s} = \left(1 + \left(\frac{p}{p_{F_s}} - 1 \right) \frac{M_A}{M_F} \right)^{-1}, \quad (12)$$

where p , p_{F_s} , M_A and M_F are the air pressure, fuel vapour pressure based on the Clausius-Clapeyron relation and molecular weight of air and fuel, respectively. As it is assumed that the droplet is spherical and the effect of convective cooling due to relative velocity is negligible, the fuel droplet temperature can be assumed to be constant and equal to its surface temperature.

$$\frac{dT_d}{dt} = \frac{Q - Q_c}{C_{p,f} m_d} \quad (13)$$

Eq. (13) describes the temperature change of the droplet's surface over a time duration dt . The heat transferred from air to fuel and the heat absorbed through evaporation are denoted as Q and Q_c , respectively.

$$Q = 2\pi D_p (T - T_d) \ln(1 + B_M) / B_M \quad (14)$$

$$Q_c = \frac{dm}{dt} L \quad (15)$$

328 2.2.2. Spray Breakup Modelling

329 This work employs a stochastic atomisation model based on the Fokker-
 330 Planck differential equation. The solution to such equation leads to a distri-
 331 bution function for $x = \ln(r)$, where r is the droplet radius (Gorokhovski,
 332 2001; Apte et al., 2003a).

$$T(x, t + 1) = \frac{1}{2} \left[1 + \operatorname{erf} \left(\frac{x - x_d - \langle \xi \rangle}{\sqrt{2\langle \xi^2 \rangle}} \right) \right], \quad (16)$$

The first ξ and second moments ξ^2 need closure and the expression derived
 by (Apte et al., 2003a, 2009) are employed

$$\langle \xi \rangle = K \ln \left(\frac{\operatorname{We}_{crit}}{\operatorname{We}_d} \right) \quad (17)$$

$$\langle \xi^2 \rangle = -\langle \xi \rangle \operatorname{We}_d, \quad (18)$$

333 where We_{crit} and We_d are the critical and droplet Weber numbers. The
 334 constant K is set to 0.6. The breakup of a parent droplet will take place
 335 when $\operatorname{We}_d > \operatorname{We}_{crit}$ and $t > t_{bu}$, where t is the elapsed time of the droplet
 336 and breakup frequency $1/t_{bu}$

$$t_{bu} = \sqrt{1/3} \sqrt{\frac{\rho_d}{\rho}} \frac{r_j}{|\mathbf{u}_{relv}|} \quad (19)$$

337 The distribution function spans over a wide range, which implies that newly
 338 formed droplets have a wide range of droplet sizes which makes this atomi-
 339 sation model sufficient for both primary and secondary breakup. A detailed
 340 derivation and discussion of this model are provided by (Gorokhovski, 2001;
 341 Apte et al., 2003a). This method is generally very expensive because the
 342 droplet count can quickly explode. To overcome this problem, the parcel
 343 approach can be considered where each parcel contains a number density N_d
 344 which provides the number of droplets it actually represents. With breakup,

345 the parcel count can still become excessive, and therefore, another approx-
 346 imation would be to sample the diameter of the parcels and update their
 347 number densities without actually creating additional droplets to track. How-
 348 ever, this averaged based approach is in contrast to LES applications (Apte
 349 et al., 2003a). An alternative solution is to consider the droplets partly
 350 atomised before being injected to the domain. A Rosin-Rammler distribu-
 351 tion can be employed to provide a distribution of droplet diameters using a
 352 mean diameter d_{32} , assuming that such droplet sizes are more influenced by
 353 the evaporation rate rather than atomisation (Cha et al., 2005; Jones et al.,
 354 2014).

355 2.3. Spray Injection

356 The spray injection is carried out by defining a number of injection loca-
 tions N_{sl} distributed on a fuel nozzle ring as shown in Fig. 3. The number of

Figure 3: A sketch of the fuel nozzle ring from which the droplets will be released at each time-step.

357
 358 fuel nozzle rings, their locations, thicknesses and orientations are defined at
 359 the beginning of a simulation, and the injection of all droplets will take place
 360 at each time step. The thickness of the ring is used to randomise the droplet's
 361 location in the radial and azimuthal directions. Each injection location $N_{i,sl}$

will inject droplets according to a defined distribution of droplet diameters,
 for example, a Rosin-Rammler distribution. That is, if the distribution has
 N_{bin} bin sizes along with N_{sl} injection locations and N_{fnzo} fuel nozzle rings,
 the total number of parcels injected per time step is

$$N_{d,\Delta t} = N_{\text{fnzo}} N_{\text{sl}} N_{\text{bin}} \quad (20)$$

The inlet spray velocity is defined using a cylindrical coordinate system with
 reference to the fuel nozzles. That is, for each nozzle an axial V_x , tangential
 V_θ and radial V_r velocity components need to be defined.

2.4. Two-way coupling Eulerian-Lagrangian model

A two-way coupled framework is relevant for vaporizing fuel spray where
 a portion of the mass liquid fuel (See Eq. (10)) converts to vapour at each
 timestep. This fuel vapour needs to be added as a source term in the mass
 conservation equation. The mass source term in Eq. (1) is defined as

$$\dot{m}_d = \sum_{i=1}^{N_d} \left(\frac{dm_d}{dt} \right), \quad (21)$$

where N_d is the total number of droplets. The same quantity must be added
 to the mixture fraction equation (See Eq. (4)) where the source term is taken
 to be $\dot{\omega}_Z = \dot{m}_d$. In turn, the momentum and energy source terms can be
 expressed as

$$S_{i,d} = \sum_{i=1}^{N_d} \left(\frac{d(m_d u_{i,d})}{dt} \right) \quad (22)$$

$$Q_d = \sum_{i=1}^{N_d} Q_{c,i} - Q_i, \quad (23)$$

The flow coupling onto the spray is incorporated due to the use of the relative
 velocity when computing the force on each droplet.

376 2.5. Numerical approach

377 The governing equations are solved with a Finite Volume Method (FVM)
378 using a second-order central difference scheme for the convective and diffu-
379 sive terms with a pressure based solver using the SIMPLE (Patankar, 1980)
380 scheme adapted to transient problems. The time marching is conducted em-
381 ploying a second order backward Euler scheme. Further details of the flow
382 solver has been presented by Anand et al. (2013); Zhu et al. (2013).

383 The gas velocity imposed on the droplets is interpolated by using a near-
384 est neighbour method, which means the cell centre velocity is directly pre-
385 scribed to every droplet that is located into that respective cell. Similarly,
386 each particle-induced source terms is directly prescribed to its associated cell
387 centre. The effect of such approximations will not be assessed in this work.
388 Further details of the spray model and the coupled methodology with the
389 gas phase have been presented and validated by Cha et al. (2005).

390 3. Parallel algorithm

391 As stated, the aim of this work is to develop a parallel strategy for
392 Eulerian-Lagrangian that overcome the bottleneck associated with load bal-
393 ancing and poor parallelism. For large-scale combustion application, the
394 parallel algorithm is based on domain decomposition with a spatial parti-
395 cle algorithm for the spray part as explained in Section 1. However, this
396 can quickly lead to unacceptable parallel efficiency due to load imbalancing.
397 Fig. 4a illustrates the distribution of particles for a case which is running
398 on 160 cores using 10 nodes. Fig. 4b shows the CPU consumption for two
399 different MPI processes, one (Rank: 0) which is extremely loaded with par-
400 ticles and another (Rank 46) which barely has any. It is observed that Rank
401 0 spends around 40% of the time waiting while other processes are carrying

Figure 4: The distribution of particles over the MPI processes illustrates a major critical performance problem where nearly 90 % of the particles are assigned to 10% of the total processes.

402 out the Lagrangian tracking, which is exactly what is observed for the 90%
 403 percent of ranks which have a similarly low number of droplets. This shows
 404 clearly the current parallel method's weakness in coping with large scale HPC
 405 simulation. The core count per node is generally increasing for modern HPC
 406 implying that scientific codes will have to employ more parallelism to ex-
 407 ploit current and the future HPC architectures. In this work, we consider
 408 each MPI rank belongs to a unique core. Hence, the number of MPI ranks
 409 corresponds to the number of cores used.

410 The parallel approach presented here will help to overcome the current
 411 bottleneck and can cope very efficiently with future HPC systems. This work
 412 will specifically concentrate on single-node performance, as this is believed
 413 to be a key step before multi-node HPC applications can become efficient.

414 3.1. An Asynchronous Task-Based Eulerian-Lagrangian parallelism

415 Studying the mathematical and numerical models of both Eulerian and
 416 Lagrangian approaches reveals that the dependence between one another

Figure 5: Flowchart of the asynchronous coupling procedure.

417 can be limited to one step. That is, both methodologies only require a single
 418 synchronisation, namely prior to a new time step, to advance both solvers in
 419 time. In other words, most of the mathematical operations that take place
 420 individually on both solvers can be carried out concurrently. One of the main
 421 problems with the current parallel methodology is the sequential nature of
 422 the execution which turns into a severe issue in the case of a load imbalance
 423 (See Fig. 4). It implies that the vast majority of the computational resources
 424 are waiting most of their CPU time so a minority can complete their respec-
 425 tive tasks. This degenerates the parallel efficiency and ultimately increases
 426 the computational cost considerably. This paper proposes an Asynchronous

Figure 6: An example with a single computer node split into between 6 and 2 flow and spray ranks, respectively. Each flow rank is assigned to one partition whereas the spray ranks track the particles across the entire domain.

427 Task-based Eulerian-Lagrangian (ATEL) framework to ensure every CPU is
 428 occupied most of the time. The flow chart of ATEL is illustrated in Fig. 5.
 429 This framework does not assume a specific engineering application, however,
 430 we will concentrate on gas turbine combustors where the spray plays a vital
 431 role in determining the engine's efficiency, emission and durability. Hence,
 432 the Lagrangian particle tracking solver is in this work also the spray solver,
 433 but in principle any Lagrangian solver can be employed within this suggested
 434 framework.

435 The original framework contains both the Eulerian-Lagrangian methods
 436 where each rank is responsible for a subdomain of the flow and spray models.
 437 The novel framework presented here implies that a subset of MPI ranks will
 438 be tasked to only carry out the Lagrangian tracking, while the remaining
 439 processes will be allocated to the Eulerian domain. Fig. 6 illustrates an

440 example with 5 MPI ranks allocated to the flow solver, where each has a
 441 partition or block of the domain. Additionally, there are 2 MPI ranks for the
 442 spray solver and they will both track their respective particles throughout
 443 the domain. Therefore, they will have access to all the 5 blocks corresponding
 444 to the other MPI ranks allocated to the flow solver. Note that the algorithm
 445 can work with arbitrary splits between the flow and spray solvers, and the
 446 rank split should be chosen such that both solvers reach the synchronisation
 447 step around the same time. It is rather straight forward to ensure an almost
 448 perfect load balancing of particle tracking among cases with multiple MPI
 449 spray ranks. In this work, the injection locations N_{sl} for each fuel nozzle is
 450 split evenly among all ranks. This ensures that each spray process will inject
 451 the same number of particles per time step.

452 At the initialisation stage the MPI ranks are either assigned to run the
 453 flow or spray solver. During the synchronisation step, the flow field and
 454 source terms need to be communicated to the spray and flow ranks, respec-
 455 tively. Subsequently, each spray and flow rank carries out the various tasks
 456 concurrently until the next time step. It is worth mentioning that the asyn-
 457 chronous approach has a slight change in the way the system is solved. For
 458 the sequential approach the spray solver is always advanced using the flow
 459 field information from the beginning of the time-step, while the flow field is
 460 advanced using the spray information from the end of the time-step. The
 461 presented asynchronous approach consistently employs the respective field
 462 information from the beginning of the time-step for both solvers. The im-
 463 plications of this will be studied in Section 5, where a sequential version
 464 has been performed to assess accurately the impact of such an approach.
 465 The sequential case mimics the baseline approach in terms of the sequence
 466 in updating the flow and spray field; therefore, providing an opportunity to

467 perform a two-step verification study of ATEL with respect to the baseline
468 solver. Firstly, by using the sequential case study ATEL can be verified based
469 solely on its parallel methodology, and the outcome is expected to be identical
470 to the baseline solver since the underlaying physics is equivalent. Next,
471 a concurrent ATEL study will be carried out to assess any potential impacts
472 such synchronisation methodology causes with respect to the sequential approach.
473

474 *3.2. Shared Memory & Interprocess One-Sided Communication*

475 The two main problems with the presented task based approach is memory
476 consumption and communication. The spray solver does not know a priori
477 the particle's trajectory, nor the blocks it will pass through. One approach
478 would be to transfer the mesh associated to all the blocks during
479 initialisation, and similarly the entire flow field at each synchronisation step.
480 In this way, the spray solver can locally keep track of where each particle
481 is located and very efficiently access the information needed. However, this
482 method is memory intensive and does not suit the HPC trend of increasing
483 core count per node and less memory per core. It implies that each MPI
484 spray rank would have a copy of the entire grid and flow field. Additionally,
485 transferring the flow variables for each partition to all spray ranks at each
486 time-step can become a bottleneck itself and potentially spoil the speed-up
487 gain from using the asynchronous method.

488 The increasing core count per node of modern HPC systems motivates
489 the use of shared memory. In conventional MPI applications, the memory of
490 each process is private and can be transferred through message passing. The
491 MPI standard 3.0 (Message Passing Interface Forum, 2015) introduces the
492 concept of one-sided shared memory communication, where memory within
493 a node can be shared among multiple MPI processes, which can avoid data

Figure 7: The MPI shared memory concept where each node has an internal communicator and a corresponding local ID rank.

494 replication. The higher core count per node will inevitably increase the data
 495 exchange within a node, which can be performed more efficiently through
 496 direct shared memory access.

497 Fig. 7 illustrates the general concepts of this feature, where the MPI
 498 processes have two IDs; a local ID within a node, and a global ID associated
 499 to the global communicator `MPI_COMM_WORLD`. It is the former type of ID
 500 in conjunction with a shared memory communicator `MPI_COMM_SHM` that are
 501 introduced with this novel concept. Each node has its own `MPI_COMM_SHM`
 502 with its local MPI ranks. Each MPI process can allocate shared memory of a
 503 certain size by means of the MPI function `MPI_WIN_ALLOCATE_SHARED`. This
 504 memory can be accessed by another MPI rank through a pointer which points
 505 to the memory address of this window. The pointer can be set up by means
 506 of the MPI function `MPI_WIN_SHARED_QUERY`. Following this, the pointer holds
 507 the address of the shared window and can access the information it contains.
 508 It is the responsibility of the developer to ensure that when remotely accessing
 509 such memory, they are not being modified and the data corresponds to the
 510 correct time step.

511 To mimic the suggested concept of the coupled Eulerian-Lagrangian frame-
 512 work, a bandwidth analysis is carried out using the partitioning strategy in-
 513 troduced in Fig. 6. That is, there is one process which is receiving data

Figure 8: Communication time comparison between P2P and SHM communication technique for varying message sizes. All analyses are tested with one receiving rank and $N_r - 1$ sending ranks.

514 from a number of processes. This is equivalent to having a number of
 515 flow processes sending their local flow field to a single MPI spray rank.
 516 The reverse data exchange has also been investigated and shows a similar
 517 trend and is not presented here. The size of the data exchanged is var-
 518 ied from around 0.1 to 100 Mb per process. The test case was run on a
 519 single node with 40 cores in the form of two Intel(R) Xeon(R) Gold 6248
 520 at 2.50 GHz. The C++ code related to this analysis can be found in:
 521 <https://github.com/akjt/SharedMemoryParallelisation.git>.

522 Fig. 8 shows the communication cost comparison between the conven-
 523 tional Point to Point (P2P) and SHared Memory (SHM) communication
 524 techniques using up to $N_r = 40$ ranks, with all tests having one receiving
 525 rank and the remaining sending. Each sending MPI process communicates
 526 with the single receiving MPI process with varying message sizes and the test
 527 is repeated 100 times to obtain an average timing for each sending MPI pro-
 528 cess. Finally, the speed-up figures have been averaged over the total number

529 of sending MPI ranks. The consumed time includes the time to fill up the
 530 buffers and until it reaches to the receiver at the synchronisation step. In
 531 the case of SHM the latter contribution is negligible since there is no actual
 532 transferring of data, and the real cost is only the time to fill up the buffer.

533 It is clear that with more ranks involved the communication cost rises
 534 since the single rank needs to receive data from up to 39 ranks. Also, the
 535 speed-up using the SHM method seems to be consistently the same for all
 536 message sizes peaking at 3x faster for the case with 40 ranks.

537 3.3. Mapping between Flow and Spray

538 By exploiting the shared memory feature, each spray rank can be given
 539 access to all blocks of the domain without consuming any additional memory.
 540 Each spray rank defines an Array of Structure data type where each element
 541 of the array corresponds to a block of the flow domain (See Fig. 6),

$$\mathbf{flowvar}(i_b = 1 : n_b) \rightarrow \left\{ \begin{array}{c} u_{ptr}(1 : n_c) \\ v_{ptr}(1 : n_c) \\ w_{ptr}(1 : n_c) \\ \vdots \end{array} \right\},$$

542 where n_b is the total number of blocks. The local cell count associated to
 543 a block i_b is denoted n_c . The members contained in each element of the
 544 structure corresponds to pointers that are pointing to where the various flow
 545 variables are stored for the corresponding block i_b . Hence, the allocation and
 546 write-access of $\mathbf{flowvar}(i_b)$ is carried out by the flow rank that is associated
 547 to block i_b , and the spray ranks treat them as read-only data. For the case
 548 with the conventional point-to-point communication, the members would
 549 effectively be locally allocated arrays which are transferred among MPI ranks
 550 in a conventional manner. Similarly, each spray rank has an associated data

551 structure of source terms, which corresponds to each flow block.

$$\text{sprayvar}(i_b = 1 : n_b) \rightarrow \begin{Bmatrix} mass_{ptr}(1 : n_c) \\ momx_{ptr}(1 : n_c) \\ momy_{ptr}(1 : n_c) \\ \vdots \end{Bmatrix}$$

552 Notice that these source terms have already been mapped to the flow do-
 553 main, as shown in the exemplified pseudo code for the mass source term
 554 in Algorithm 1. Unlike the data structure of **flowvar**, each spray rank al-
 555 locates its own **sprayvar**($i_b = 1 : n_b$) and writes directly the source terms
 556 with the appropriate indexing, as observed in Algorithm 1 which illustrates
 557 the case for the mass source terms of each droplet. Since each spray rank
 558 covers all blocks, the source term contribution per time step requires an in-
 559 ternal reduction step of each member of **sprayvar**(i_b) across all spray ranks
 560 $i_{sp} = 1 \dots N_{sp}$. Subsequently, only one spray rank needs to transfer the
 561 source terms for each block i_b to the associated flow ranks (See Algorithm 2
 562 & 3) .

563 From the flow rank's perspective it only requires one element for the data
 564 structures **flowvar**(1) & **sprayvar**(1) which correspond to its own block.
 565 That is assuming each flow rank has only a single designated block. Although,
 566 the choice of this data structure allows for the possibility to have several
 567 blocks per flow rank. As mentioned above, the flow writes directly the flow
 568 variables into **flowvar** which is shared with the spray ranks, while it receives
 569 the source terms **sprayvar** from one spray rank following a reduction step
 570 of **sprayvar** among the spray ranks.

Algorithm 1:

Result: Mass source terms mapped to the flow domain.

```
foreach particle do
   $i_b \leftarrow \text{particle.block\_id}$  // block id ;
571   $i_c \leftarrow \text{particle.cell\_id}$  // local cell id corresponding to the
      block ;
   $\text{spraysource}(i_b).\text{mass}(i_c) \leftarrow \text{ADD particle.mass\_source}$  ;
end
```

572 To illustrate how this compares with the synchronisation step defined in
573 the flow chart Fig. 5, Algorithms 2 & 3 have been presented. In the case of
574 the point-to-point synchronisation each flow rank would transfer its flow field
575 at each time step to each spray rank. From the spray rank's perspective, it
576 would have to receive the flow field associated to each block. With a shared
577 memory communication technique, the synchronisation becomes very simple
578 since most of the work has been carried out during the initialisation of the
579 shared memory windows.

Algorithm 2: Synchronisation using shared memory method.

```
if Flow Rank then
  // Wait for all spray ranks to complete the time-step.
   $i_b \leftarrow \text{myblock\_id}$  ;
  BARRIER with all spray ranks  $\in i_{sp} = 1 \dots N_{sp}$ .;
  COPY  $\text{sprayvar}(i_b)$  to a local buffer;
end
if Spray Rank then
580  foreach block do
      // Sum sources to Spray root rank's shared buffer.
       $\text{sprayvar}(i_b)_{i_{sp}=1} \leftarrow \text{REDUCE sprayvar}(i_b)_{i_{sp}=1 \dots N_{sp}}$ 
    end
    // Wait for all flow ranks to complete the time-step.
    BARRIER with all flow ranks  $\in i_f = 1 \dots N_f$ .;
    COPY  $\text{flowvar}(i_b = 1 : n_b)$  to a local buffer;
  end
  Advance to next time-step.
```

581 To ensure safe and consistent parallel execution there are two vital points
 582 to consider with shared memory communication. Firstly, before starting a
 583 new time-step, both the flow and spray must ensure their respective part-
 584 ners have completed their time-step and we can read the new information,
 585 which is also referred as the synchronisation step. For the point-to-point
 586 communication method this is implicitly ensured, however, a shared memory
 587 communication approach requires an explicit action. There are several ways
 588 to ensure synchronisation (see Message Passing Interface Forum (2015)). In
 589 this work, we complete this with a `MPI_BARRIER` as also indicated in Algo-
 590 rithm 2.

Algorithm 3: Synchronisation using point-to-point method.

```

if Flow Rank then
  // Flow rank broadcasts its local flowfield to all
  // spray ranks.
  BROADCAST flowvar to all spray ranks.;
  // Receive sources associated to the local block.
   $i_b \leftarrow \text{my\_block\_id}$ ;
  RECEIVE sprayvar( $i_b$ ) $_{i_{sp}=1}$  from root spray rank.
end
if Spray Rank then
  // Each spray rank receives all local flowfields.
  foreach block do
591    $i_b \leftarrow \text{block\_id}$ ;
    $i_d \leftarrow \text{rank\_id}(i_b)$ ;
   RECEIVE  $\leftarrow \text{flowvar}(i_b)$  from rank  $i_d$ ;
   // Sum sources & send from root to the associated
   // block.
   sprayvar( $i_b$ ) $_{i_{sp}=1} \leftarrow \text{REDUCE sprayvar}(i_b)_{i_{sp}=1 \dots N_{sp}}$ ;
   if Spray Root then
     | SEND  $\leftarrow \text{sprayvar}(i_b)_{i_{sp}=1}$  to flow rank  $i_d$ 
   end
  end
end
Advance to next time-step.
```

Secondly, only the rank that owns the shared buffer window has write access while its associated partner(s) has read-only access. In fact, the read-only permission is restricted to be used once per time-step namely during the synchronisation among the owner rank and its partner(s). This single read-only permission is used to make a safe copy of the received data into a local buffer that can be used throughout the calculations of next iteration. The reason for this is, for example, if the spray rank is accessing the flow field for a particular cell to compute drag, the flow rank responsible for this cell could potentially be updating that flow variable for the next time-step, which can lead to data-race issues. That is, since the flow or spray rank uses its sharing buffers directly during the iterations, it is never safe to allow its partners to read such buffer during its calculations. In practical terms it means that each flow rank i_f needs a single copy of `sprayvar` (i_b) where i_b is the block associated to i_f , and each spray rank requires a copy from all flow ranks ($i_f = 1 \dots N_f$) the flow field associated to each block `flowvar` ($i_b = 1 : n_b$).

4. Atomising spray in a diesel engine configuration

Injection conditions	Case 1	Case 2	Case 3
P_{gas} [MPa]	1.1	3.0	5.0
$\dot{m}_d$ [g/s]	6.06	5.36	5.13
D_d [μm]	300	300	300
u_d [m/s]	102	90.3	86.4
t [ms]	4.0	8.0	8.0

Table 1: The parameters of the three study-cases.

The first case-study to validate the parallel method is based on a diesel engine configuration which has been tested experimentally by Hiroyasu and Kadota (1974). Additionally, a CFD study has been conducted by Apte et al. (2003a). The closed cylindrical chamber has a length of 138 mm and a

Figure 9: A comparison of the jet penetration for all three cases. It is evident that the base line code and the ATEL are nearly identical. The stochastic breakup method will inevitably cause deviations due to different random number sequences when using different number of ranks. • Hiroyasu and Kadota (1974) ■ Apte et al. (2003a) (—) Base line (- - -) ATEL

612 diameter of 56 mm. The diesel fuel of $\rho = 840 \text{ kg/m}^3$ is injected by a single
 613 hole nozzle which is equivalent to the size of the injected droplet diameter,
 614 namely $D_d = 300 \mu\text{m}$. Hence, this case has only one injection location with
 615 a nozzle ring diameter of 0 m. The liquid fuel is injected into a constant
 616 pressure nitrogen chamber at $T = 294.15 \text{ K}$. The flowfield is at rest and is
 617 only disturbed by the momentum exchange once the droplets are injected
 618 into the domain. This case has been tested under three different conditions
 619 summarised in Table 3. The injection velocity has been obtained using the
 620 fixed size of the injected droplets together with the mass flow rate of the
 621 liquid $\dot{m}_d$ and assuming spherical shaped droplets,

$$\dot{m}_d = \pi \frac{D_d^2}{4} \rho_d u_d \quad (24)$$

Ranks	CPUh	Wall clock (h)	Flow CPUh	Spray CPUh	Efficiency (%)
40	48.00	1.20 (0.43)	27.10	19.80	35.63
35	48.40	1.38 (0.49)	24.05	23.39	35.33
30	39.50	1.32 (0.57)	21.30	17.38	43.29
25	34.30	1.37 (0.68)	19.60	14.03	49.85
20	34.00	1.70 (0.86)	17.19	16.27	50.29
15	22.10	1.47 (1.14)	16.10	5.60	77.38
10	28.40	2.84 (1.71)	15.50	12.67	60.21
5	17.10	3.42 (3.42)	13.18	3.81	100.00

Table 2: The scaling of the base line code using Case 1. The parallel efficiency is calculated with respect to the case with 5 ranks. The ideal wall clock times are given in brackets.

622 The computational domain has been discretized with $200 \times 96 \times 96$ finite
623 volume cells and a time-step of $\Delta t = 15.0 \mu s$ has been employed.

624 Fig. 9 shows the jet penetration for all three cases including the exper-
625 imental results obtained by Hiroyasu and Kadota (1974) and the numerical
626 findings available by Apte et al. (2003a). The comparison between the base
627 line code and ATEL is almost identical. There two elements which inevitably
628 will result in small deviations. Firstly, in the base-line code the spray part
629 is advanced in time using the flow information from the previous time-step,
630 while the flow step was based on spray information obtained from the same
631 time-step. In ATEL, both the spray and the flow advance in time using the
632 results from the start of the time-step. Although this source of deviation
633 is small, it is observable when a one-to-one comparison is made. Secondly,
634 the stochastic breakup model employed in this case-study relies on a ran-
635 dom number generator. While each MPI rank has its own private stream,
636 altering the number of ranks will ultimately cause different random number
637 sequences. In comparison to the experimental results, it is evident that both
638 the base line code and ATEL fail to capture the penetration depth during the
639 start but do so at a later stage. Collision and coalescence are not incorpo-

N_{fl}	N_{sp}	CPUh	Flow CPUh	Spray CPUh	Wall clock [h]	Efficiency [%]	Reduction CPUh	Efficiency Change [pp]
<i>40</i>	<i>0</i>	<i>48.0</i>	<i>27.1</i>	<i>19.8</i>	<i>1.20 (0.43)</i>	<i>35.63</i>	<i>0.00</i>	<i>0.00</i>
39	1	78.1	76.1	1.95	1.95 (0.43)	21.9	-30.1	-13.7
38	2	52.1	49.5	2.61	1.30 (0.43)	32.8	-4.10	-2.80
37	3	39	36.1	2.93	0.98 (0.43)	43.8	9.00	8.22
36	4	37.3	33.6	3.73	0.93 (0.43)	45.8	10.7	10.2
35	5	34	29.8	4.25	0.85 (0.43)	50.3	14.0	14.7
34	6	33.7	28.7	5.06	0.84 (0.43)	50.7	14.30	15.1
33	7	34	28.1	5.95	0.85 (0.43)	50.3	14.0	14.7

Table 3: Performance data for the new developed parallel code ATEL using Case 1. The total MPI ranks is maintained at 40, but the split has been varied. The ideal wall clock is indicated in brackets and the baseline performance is highlighted with italic for comparison.

rated in the spray model, which is most likely the reason for this observation, as the droplets at the early stage are clustered together (Apte et al., 2003a).

All performance test cases have been conducted on a node with 2 processors of Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz with 20 cores each. All cases have been consistently tested up to 40 MPI ranks. Table 1 summarises the performance of the baseline code. The figures show that the spray solver performance scales with 1.5x against 3.9x for the flow solver when comparing the runs for $N_{fl} = 5$ and $N_{fl} = 40$. The performance of the spray solver also fluctuates with respect to the number of flow ranks $N_{fl} = 40$ (or number of partitions). The load balancing of the droplets is dependent on the distribution of droplets in the domain in relation to the domain decomposition, which means fewer partitions would lead to a more favourable load distribution of the droplets. Ultimately, the CPU cost of the spray solver is strongly case dependent, and therefore implies no apparent relation between increasing processing units and speed-up. As a consequence, the overall parallel efficiency degenerates considerably. This is also due to the sequential

Figure 10: The performance of the ATEL showing both solvers CPU time and their respective synchronisation time. The abscissa only indicates the number of MPI ranks for both the solvers. The base line performance data is provided for reference. (●) Wall clock base line (■) Wall clock Flow base line (▲) Wall clock Spray base line (●) Wall clock ATEL (■) Wall clock Flow ATEL (▲) Wall clock Spray ATEL (■) Waiting time for flow ATEL (▲) Waiting time for spray ATEL

656 nature where the significantly less loaded processes are awaiting the remain-
657 ing to complete the spray step. Notice that the presented total CPUh also
658 includes any overhead and housekeeping incurred during the simulation such
659 as pre/post-processing and synchronisation; hence, it is slightly larger than
660 the combined flow and spray CPUh.

661 Table 2 provides the performance data of the proposed ATEL method-
662 ology using a total of 40 MPI ranks where split among flow N_{fl} and spray
663 N_{sp} ranks has been varied. It is clear that using fewer spray resources than
664 required might have an adverse effect on performance, however, its potential

665 is huge. Comparing to the baseline code, it is possible to achieve a reduction
 666 of 14 CPUh resulting in an increase of 25 percentage point efficiency and 30%
 667 reduction in wall clock time. The table clearly shows significant reductions
 668 in CPUh for the spray solver compared to the baseline case, while the CPUh
 669 for the flow solver remains unaffected. However, there is one outcome that
 670 requires more attention: the spike in CPUh for the flow solver is exclusively
 671 due to its significant idling time for the case with $N_{fl} = 39$ and $N_{sp} = 1$ (See
 672 also Fig. 10). This case does require at least two spray ranks to avoid having
 673 such a load imbalance between the solvers. Fig. 10 shows the actual com-
 674 pute time for both the flow and spray solvers in addition to their associated
 675 waiting time for a total number of 40 MPI ranks. The waiting time is when
 676 the flow or spray solver has been waiting for the other solver's completion of
 677 a time-step. From the flow solver's perspective, the idling time drops signif-
 678 icantly when the number of spray ranks is increased. On the other hand, it
 679 increases for the spray modestly with spray ranks (or decreasing flow ranks).
 680 A pivotal outcome that can be observed from the ATEL performance figures
 681 is that while both solvers might have an approximately similar wall clock
 682 time independently, the total clock time does not deviate much from this, in
 683 contrast to the base line case which is the accumulation of both solvers' CPU
 684 time. This shows a potential benefit of asynchronous algorithms for future
 685 multiphysics simulations.

686 As a final note in regards to this test case and what has been learned, it is
 687 important to mention that this case does impose challenges to the suggested
 688 framework despite its apparent simplicity. All runs start from no droplets
 689 present in the domain. This means that all sprays ranks are doing nothing or
 690 very little during the first part of the simulation, which ultimately increases
 691 the final idling time for the spray solver. However, as the simulation pro-

Inlet air Pressure	Inlet Air Temperature	Air-Fuel ratio	Fuel mass flow rate	Inlet Fuel Temperature	SMD
4 bar	550 K	20	2.98 g/s	350 K	$6.5\mu\text{m}$

Table 4: A summary of the air and fuel inlet conditions.

	Nozzle 1	Nozzle 2	Nozzle 3
Diameter	0.015 m	0.014 m	0.013 m
(V_x, V_θ, V_r)	(30,0,0) m/s	(30,0,-15) m/s	(30,0,-30) m/s
Fuel rate	0.993 g/s	0.993 g/s	0.993 g/s

Table 5: A summary of the fuel nozzle details.

gresses the flow field begins to develop, and droplet count rises quickly up to a maximum of around 14M due to the injection and breakup process. In turn, the flow ranks might experience considerable idling time depending on the number of spray ranks employed. That is the reason why an appreciable waiting time can be observed for both solver for this case-study (See Fig. 10). This implies that the load balancing is varying strongly over the course of the simulation. Nonetheless, the ATEL framework showed its robustness in dealing with such less frequent scenarios. For applied large scale LES simulations of combustion flow, these scenarios are less likely to occur since a common practice is to begin from a developed flow field with a distributed spray field, where load balancing is not as dynamic as in fact observed in this test case.

5. Reacting Spray in a Gas Turbine Combustor

This case-study examines a reacting LES simulation of a gas turbine combustor. The burner’s geometry is based on the DLR Generic Single Sector Combustor (GENRIG), which has been described in detail by Freitag et al. (2010) and Meier et al. (2011). This has further been examined by a reacting LES simulation carried out by Jones et al. (2014). The numerical model was

Figure 11: A sketch of discretized domains of the combustor and burner.

	Sequential		Concurrent	
	$z = 7 \text{ mm}$	$z = 30 \text{ mm}$	$z = 7 \text{ mm}$	$z = 30 \text{ mm}$
$ \bar{u} _2$	0.016%	0.012%	0.899%	0.671%
$ u_{RMS} _2$	0.034%	0.058%	1.504%	2.97%
$ \bar{T} _2$	0.009%	0.001%	0.221%	0.379%
$ T_{RMS} _2$	0.018%	0.047%	0.054%	0.549%

Table 6: The L2 norm error with respect to the baseline framework. It is evident that the sequential approach is nearly identical to the baseline, while the concurrent one does lead to a slightly larger discrepancy.

710 validated with experimental results using a cold-case (non-reactive), and sub-
711 sequently carried out an analysis of a reactive LES simulation (See Jones et al.
712 (2014)). In this study, a reactive LES simulation with evaporating droplets
713 will be considered only. Such cases that commonly suffer from highly non-
714 uniform distribution of particles, the great majority of droplets are clustered
715 near the injector making such a simulation difficult to scale for large particle
716 count as explained in Sections 1 and 3. Firstly, the base-line and ATEL will
717 be compared with results obtained by Jones et al. (2014). Then, a paral-

Figure 12: The radial mean and RMS velocity profile at $z = 7$ mm (left column) and $z = 30$ mm (right column) are compared against the baseline code and Jones et al. (2014). A synchronised version of ATEL has also been shown to assess the impact of the asynchronous approach. — Sequential ATEL • Concurrent ATEL ■ baseline • Jones et al. (2014)

718 lel performance analyses will be carried out to investigate the novel parallel
 719 methodology presented in this work by comparing the speed-up of the base-
 720 line code and ATEL. All performance test cases have been conducted on a
 721 node with 2 processors of Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz with
 722 20 cores each.

723 The computational domain consists of a plenum where the flow develops
 724 into a turbulent and representative inlet condition for the two radial swirlers.
 725 The domain consists of 7 million computational cells with the smallest cell

Figure 13: The radial mean and RMS temperature profile at $z = 7$ mm (left column) and $z = 30$ mm (right column) are compared against the baseline code and Jones et al. (2014). A synchronised version of ATEL has also been shown to assess the impact of the asynchronous approach. — Sequential ATEL • Concurrent ATEL ■ baseline • Jones et al. (2014)

size $\Delta x \approx \Delta y \approx \Delta z \approx 0.5$ mm (See Fig. 11). The inlet air pressure is 4 bar and preheated to an inlet temperature of 550 K. The spray is injected at $T_D = 350$ K with a Sauter Mean Diameter $SMD = 6.5 \mu\text{m}$ using 10 bin sizes for the Rosin-Rammler distribution. Three fuel nozzle rings each with 60 injection locations have been applied with a centre point at $(x, y, z) = (0, 0, 0)$. The rings' diameters are respectively set to 0.015 m, 0.014 m and 0.013 m. This ensures a fair distribution of the inlet fuel injection. In this case, the droplets are assumed atomised and are mostly affected by the evaporation

Ranks	CPUh	Wall clock (h)	Flow CPUh	Spray CPUh	Efficiency (%)
40	1919	47.96 (16.43)	1071	832.5	34.25
35	1481	42.30 (18.77)	1006	461.1	44.38
30	1241	41.35 (21.90)	841.0	388.6	52.96
25	1070	42.79 (26.28)	766.3	293.8	61.41
20	925.1	46.25 (32.85)	682.9	233.8	71.03
15	852.6	56.84 (43.80)	680.0	164.8	77.06
10	771.4	77.14 (65.70)	600.0	165.1	85.17
5	657.0	131.4 (131.4)	575.1	76.44	100.00

Table 7: The scaling of the baseline code using Case 1. The parallel efficiency is calculated with respect to the case with 5 ranks. The ideal wall clock times are given in brackets.

N_{fl}	N_{sp}	CPUh	Flow CPUh	Spray CPUh	Wall clock (h)	Efficiency (%)	Reduction CPUh	Efficiency Change [pp]
<i>40</i>	<i>0</i>	<i>1919</i>	<i>1071</i>	<i>832.5</i>	<i>47.96 (16.43)</i>	<i>34.25</i>	<i>0.00</i>	<i>0.00</i>
39	1	1435	1386	35.60	35.88 (16.43)	45.78	482.9	11.52
38	2	1183	1081	41.60	29.58 (16.43)	55.52	735.2	21.28
37	3	1182	1069	41.08	29.56 (16.43)	55.57	736.2	21.32
<i>20</i>	<i>0</i>	<i>925.1</i>	<i>682.9</i>	<i>233.8</i>	<i>46.25 (32.85)</i>	<i>71.03</i>	<i>0.00</i>	<i>0.00</i>
19	1	772.2	694.2	31.56	38.61 (32.85)	85.10	152.9	14.06
18	2	770.4	681.7	76.95	38.52 (32.85)	85.29	154.7	14.26

Table 8: Performance data for the new developed parallel code ATEL using Case 1. The total MPI ranks is maintained at 40, but the split has been varied. The ideal wall clock for the corresponding total number ranks is indicated in brackets and the baseline performance is highlighted with italic for comparison.

rate. Hence, no breakup model has been employed in this study. The total mass fuel rate is set to $\dot{m}_f = 2.98 \text{ g/s}$ and is split evenly among the three nozzles. Further details of the setup has been summarised in Table 4 & 5. The time-step $\Delta t = 10^{-6}$ is applied for all simulations. Prior to conducting any LES sampling, a steady-state RANS simulation was conducted to provide an initial condition for the LES calculation. Then, the simulation was run for 20,000 time-steps which is around 4 flow-through times before initiating the statistical sampling. For validation purposes, the simulation was

run for 10,000 time-steps accounting for around 2 flow-through times, while for the parallel performance study only 1000 time-steps were necessary to obtain representative figures for the computational cost. For the validation and performance studies, the total number of droplets stays approximately constant around 25 million active with some evaporating and others being injected at each time-step.

Figs. 12 and 13 compare the mean and Root Mean Square (RMS) of the velocity and temperature fields along the radial direction at two different planes $z = 7$ mm (left column) and $z = 30$ mm (right column). The results obtained by the ATEL framework compares excellently with the baseline code as expected since both are based on the same physics and modelling techniques. In comparison to Jones et al. (2014) there are measurable deviations particularly for the temperature profiles, which is sensitive to the combustion model employed (Veynante and Vervisch, 2002). The combustion model employed by Jones et al. (2014) is based on a stochastic PDF approach while the tabulated FGM approach is used in this work (See Section 2.1.1). Additionally, we have sampled over 10,000 time-steps using $\Delta t = 10^{-6}$ against the 300,000 time-steps with $\Delta t = 10^{-7}$ used by Jones et al. (2014). However, the objective and essence of this work is to ensure that the novel parallel methodology ATEL does not deviate from the baseline framework, which itself has been extensively tested, validated and used by industry for the past many decades Cha et al. (2005); Zhu et al. (2013); Anand et al. (2013). Therefore, the scope here is to firstly verify that the task based parallel approach produce exact results with respect to the baseline framework, and assess the implications of the asynchronous approach where the flow and spray solvers are solved consecutively, which as a result lead to slightly different synchronisation procedure (See Section 3.1). By sequential ATEL we refer to the

769 same sequence of updating the flow and spray field; therefore, this is ex-
 770 pected to produce the same solution compared to the baseline framework.
 771 From Figs. 12 and 13 the difference obtained from the synchronous ATEL
 772 and the baseline solver is hardly visible, while minor differences are observed
 773 for the asynchronous ATEL.

774 Table 6 shows the L2 norm error with respect the baseline solver which
 775 is calculated for any field variable ψ using following relations

$$\epsilon = \frac{|\psi_{\text{baseline}}|^2 - |\psi_{\text{ATEL}}|^2}{|\psi_{\text{baseline}}|^2} \times 100 \quad (25)$$

776

$$|\psi|^2 = \sqrt{\sum_{k=1}^{n=1} |\psi_k|^2}. \quad (26)$$

777 It is evident from Table 6 that the general comparison between ATEL and
 778 the baseline is adequate, and nearly identical for the sequential version, which
 779 mimics the synchronisation procedure in the baseline solver. Expectedly, it
 780 does indicate an increased, but acceptable, discrepancy with respect to the
 781 concurrent ATEL due to the more explicit nature of synchronising between
 782 the flow and spray field (See Section 3.1).

783 Table 7 shows the parallel scaling figures using the baseline code, while
 784 Table 8 shows the performance data for the ATEL framework. The figures
 785 associated to CPUh covers the entire simulation including pre- and post-
 786 processing, while the flow and spray CPUh are the costs of their respective
 787 time-marching, which in turn is 95% of the total computational cost. Com-
 788 paring the baseline performance to those presented in Table 8, the ATEL
 789 framework improves the parallel efficiency by 21 percentage point and a to-
 790 tal cost reduction of 735 CPUh. For a smaller number of ranks the baseline
 791 code obtains higher parallel efficiencies that can be attributed to the relative
 792 even load distribution of droplets among the decreasing number of blocks,

793 and therefore more particles per block. Additionally, the flow-to-spray cost
 794 ratio is larger due to the increased number of cells per block. Comparing the
 795 case with 20 ranks ($N_{fl} = 19$ and $N_{sp} = 1$) shows an even higher parallel effi-
 796 ciency of 85% obtained by the ATEL framework against 71% by the baseline
 797 code with $N_{fl} = 20$ (See Table 8) indicating its robustness to increase the
 798 parallel performance even in cases where the parallel performance is already
 799 acceptable. Table 8 also compares the flow and spray CPUh showing a drop
 800 of nearly 800 CPUh with respect to the spray solver for the baseline case
 801 against the ATEL case on 40 ranks. Also, it is observed that the spray cost
 802 doubles going from 19-1 to 18-2 rank split. That is primarily due to the syn-
 803 chronisation step: The flow-to-spray cost ratio is larger with only 19 or 18
 804 ranks, and the spray ranks consume most of their time waiting for the flow
 805 ranks to complete their time-step. It is therefore sufficient to have one spray
 806 rank when the total number of ranks reduces to 20. However, for the 40-1
 807 rank split the flow CPUh is higher compared to the baseline case, which is
 808 attributed to the additional synchronisation time where the flow ranks wait
 809 for the spray rank to complete its time-step. By adding an additional spray
 810 rank (38-2 rank split) we observe the flow CPUh to drop back to the base-
 811 line figure. The choice of how to split among the flow and spray solvers will
 812 become relevant for further studies when larger scale multi-node simulations
 813 are conducted.

814 6. Concluding remarks

815 This paper has presented a novel parallel approach for a two-way coupled
 816 Eulerian-Lagrangian methods based on an asynchronous task-based (ATEL)
 817 methodology that overcomes the load balancing problem encountered in con-
 818 ventional parallel Eulerian-Lagrangian frameworks. The novel approach has
 819 been verified to near machine precision accuracy with respect to the baseline

820 solver, and validated with experimental and numerical findings from other
821 sources. The presented work has focussed on ATEL's single node perfor-
822 mance and the impact on using an asynchronous based coupling. The parallel
823 performance was documented for both the baseline and ATEL frameworks,
824 and has been investigated using a non-reacting 2D cylindrical nitrogen cham-
825 ber with a complex breakup mechanism of droplets, and a complex three-
826 dimensional combustion chamber with reacting flow and evaporating fuel
827 droplets.

828 The novel approach demonstrated a significant improve in the overall
829 parallel efficiency for both test cases. ATEL provided huge cost saving by
830 reducing the total CPUh with 62%, and improves the parallel efficiency of
831 the coupled Eulerian-Lagrangian as a whole by 21 percentage points com-
832 pared to the baseline methodology. Additionally, ATEL has been shown to
833 even further improve test cases which were not severely impacted by parallel
834 performance by increasing the parallel efficiency from 71% to 85% indicating
835 its robustness in coping with a wide range of scenarios.

836 The results show that the load balancing issue can be completely elimi-
837 nated, and furthermore, the additional cost of solving the Lagrangian field in
838 addition to the Eulerian field, can be almost entirely hidden by means of the
839 task based asynchronous nature. The Eulerian solver is hardly affected by
840 removing a single or two processing ranks out of a total of 40 – this is likely
841 because the flow solver is memory bandwidth limited. However, these two
842 ranks when re-allocated to solve the Lagrangian field are sufficient to cover
843 its computational requirements. That is, the cost of adding an extra physical
844 solver to solve a multi-physical problem can be obtained with almost no com-
845 putational overhead. However, the results are for single compute node cases
846 only, and in future work we will investigate as to whether this performance

benefit to multi-node computations on large problems.

Acknowledgements

This work is financially supported by Rolls-Royce and the EPSRC (Engineering and Physical Sciences Research Council) through the Centre for Doctoral Training in Gas Turbine Aerodynamics (grant ref. EP/L015943/1). The authors are also grateful to Dr. Christopher Goddard from Rolls-Royce for providing the GENRIG test case and Dr. Ruud Eggels from Rolls-Royce Deutschland for valuable discussions.

Declaration of Competing Interest

The authors declare that they have no know competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Mathematical Model

The viscous stress tensor and energy flux introduced in Eq. (2) and Eq. (3) read the following

$$\tau_{ij} = \mu \left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i} \right) - \frac{2}{3} \mu \frac{\partial \tilde{u}_k}{\partial x_k} \delta_{ij}, \quad (\text{A.1})$$

$$q_i = \bar{\lambda} \frac{\partial \tilde{T}}{\partial x_i} + \bar{\rho} \sum_{k=1}^N \tilde{h}_k \bar{Y}_k \bar{V}_{k,i} \quad (\text{A.2})$$

where μ is the dynamic viscosity and δ_{ij} is the Kronecker delta. The energy flux q consists of a heat flux due to temperature difference following the Fourier's law of diffusion, where λ is the conductivity of heat, and the diffusion of specie k with enthalpies of h_k , mass fraction Y_k and diffusion velocities $V_{k,i}$ of the k th specie in the i th direction. This diffusion term is

provided by the FGM look-up table and where the diffusion velocity was found using the Fick's Law (van Oijen et al., 2016). In this work, the energy Eq. (3) solves for the absolute specific enthalpy h , which is derived from the following relation,

$$h = h_s + \sum_{k=1}^N \Delta h_{f,k}^o, \quad (\text{A.3})$$

where $\Delta h_{f,k}^o$ is the formation enthalpy for specie k , and h_s is the sensible enthalpy which is given by following relation

$$h_{s,k} = \int_{T_0}^T C_{p,k} dT, \quad (\text{A.4})$$

where $C_{p,k}$ is the heat capacity at constant pressure for the k th specie and T is temperature.

Appendix A.1. LES model

In a LES formulation it is sought to resolve largest turbulence eddies while the unresolved scales are modelled with a sub-grid model. The filtered Navier-Stokes equations (See Eqs. (1) to (3)) are not closed due to unresolved sub-grid stresses ($\widetilde{u_i u_j} - \widetilde{u_i} \widetilde{u_j}$) and enthalpy fluxes ($\widetilde{u_i h_s} - \widetilde{u_i} \widetilde{h_s}$). In this work, a standard Smagorinsky model is employed for the sub-grid scales following the procedure presented by Anand et al. (2013) and Poinso and Veynante (2005). Using the Boussinesq approximation it can be shown that the unresolved sub-grid stresses can be approximated as

$$\widetilde{u_i u_j} - \widetilde{u_i} \widetilde{u_j} \approx -2\nu_t \left(\widetilde{S_{ij}} - \frac{1}{3} \widetilde{S_{kk}} \delta_{ij} \right) + \frac{2}{3} \bar{\rho} k \delta_{ij} \quad (\text{A.5})$$

$$\widetilde{S_{ij}} = \frac{1}{2} \left(\frac{\partial \widetilde{u_i}}{\partial \widetilde{x_j}} + \frac{\partial \widetilde{u_j}}{\partial \widetilde{x_i}} \right) \quad (\text{A.6})$$

where the subgrid scale viscosity ν_t is expressed as

$$\nu_t = C_s^2 \Delta^{4/3} l_t^{2/3} \left(2 \widetilde{S_{ij}} \widetilde{S_{ij}} \right)^{1/2}, \quad (\text{A.7})$$

where Δ is the grid width, l_t is the turbulence length scale and $C_s = 1.2$ is the Smagorinsky model constant. The unresolved enthalpy diffusion fluxes is modelled by the gradient approach (Poinsot and Veynante, 2005).

$$\widetilde{u_i h} - \widetilde{u_i} \widetilde{h} \approx -\frac{\nu_t}{\text{Pr}_t} \frac{\partial \widetilde{h}}{\partial x_i} \quad (\text{A.8})$$

where Pr_t is the turbulent Prandtl number equal to 0.9. More information on subgrid scale using the Smagorinsky model for reacting flows is referred to Poinsot and Veynante (2005); Chrighi et al. (2012); Anand et al. (2013); Zhu et al. (2013)

Appendix A.2. Combustion model

The FGM combustion model employed in this work adds additional four transport equations, namely transport of mixture fraction, progress variable and their respective variances. These read:

$$\frac{\partial \widetilde{\rho Z}}{\partial t} + \frac{\partial}{\partial x_i} \left(\widetilde{\rho u_i Z} \right) = \frac{\partial}{\partial x_i} \left[\nu_t \frac{\partial \widetilde{Z}}{\partial x_i} \right] + \dot{\omega}_Z \quad (\text{A.9})$$

$$\frac{\partial \widetilde{\rho \phi}}{\partial t} + \frac{\partial}{\partial x_i} \left(\widetilde{\rho u_i \phi} \right) = \frac{\partial}{\partial x_i} \left[\nu_t \frac{\partial \widetilde{\phi}}{\partial x_i} \right] + \widetilde{\dot{\omega}_c}, \quad (\text{A.10})$$

$$\frac{\partial \widetilde{\rho Z'^2}}{\partial t} + \frac{\partial}{\partial x_i} \left(\widetilde{\rho u_i Z'^2} \right) = \frac{\partial}{\partial x_i} \left[\nu_t \frac{\partial \widetilde{Z'^2}}{\partial x_i} \right] + C_1 \widetilde{\rho \nu_t} \left(\frac{\partial \widetilde{Z}}{\partial x_i} \right) \Delta^3 - C_2 \widetilde{\rho \nu_t Z'^2} \Delta \quad (\text{A.11})$$

$$\begin{aligned} \frac{\partial \widetilde{\rho \phi'^2}}{\partial t} + \frac{\partial}{\partial x_i} \left(\widetilde{\rho u_i \phi'^2} \right) &= \frac{\partial}{\partial x_i} \left[\nu_t \frac{\partial \widetilde{\phi'^2}}{\partial x_i} \right] + C_3 \widetilde{\rho \nu_t} \left(\frac{\partial \widetilde{\phi'^2}}{\partial x_i} \right) \Delta^3 \\ &+ C_4 \widetilde{\phi'^2 \dot{\omega}_c} - C_5 \widetilde{\rho \nu_t \phi'^2} \Delta, \end{aligned} \quad (\text{A.12})$$

where model constant $C_1 = 1.215$ and the remaining $C_3 = C_3 = C_4 = C_5 = 1.0$. The progress variable and progress variable variance source terms $\widetilde{\dot{\omega}_\phi}$ and

891 $\widetilde{\phi'^2 \dot{\omega}_\phi}$ are provided by the FGM look-up table and depends on the definition
 892 of ϕ . The following definition is used in this work

$$\phi = \frac{Y_{\text{CO}_2}}{M_{\text{CO}_2}} + \frac{Y_{\text{H}_2\text{O}}}{M_{\text{H}_2\text{O}}} + \frac{Y_{\text{H}_2}}{M_{\text{H}_2}}, \quad (\text{A.13})$$

893 where M is the molecular weight of each specie. More information on how
 894 to construct the FGM look-up table and determine the unclosure terms in
 895 the variance transport equations are described in great detail by van Oijen
 896 et al. (2016) & Poinso and Veynante (2005), respectively.

897 References

- 898 Aguerre, H.J., Nigro, N.M., 2019. Implementation and validation of a La-
 899 grangian spray model using experimental data of the ECN Spray G injec-
 900 tor. *Computers and Fluids* 190, 30–48. URL: [https://doi.org/10.1016/](https://doi.org/10.1016/j.compfluid.2019.06.004)
 901 [j.compfluid.2019.06.004](https://doi.org/10.1016/j.compfluid.2019.06.004), doi:10.1016/j.compfluid.2019.06.004.
- 902 Amritkar, A., Deb, S., Tafti, D., 2014. Efficient parallel CFD-DEM sim-
 903 ulations using OpenMP. *Journal of Computational Physics* 256, 501–
 904 519. URL: <http://dx.doi.org/10.1016/j.jcp.2013.09.007>, doi:10.
 905 1016/j.jcp.2013.09.007.
- 906 Anand, M.S., Eggels, R., Staufer, M., Zedda, M., Zhu, J., 2013. An Ad-
 907 vanced Unstructured-Grid Finite-Volume Design System for Gas Turbine
 908 Combustion Analysis, pp. 1–12.
- 909 Apte, S.V., Gorokhovski, M., Moin, P., 2003a. LES of atomizing spray
 910 with stochastic modeling of secondary breakup. *International Journal of*
 911 *Multiphase Flow* 29, 1503–1522. doi:10.1016/S0301-9322(03)00111-3.
- 912 Apte, S.V., Mahesh, K., Gorokhovski, M., Moin, P., 2009. Stochas-
 913 tic modeling of atomizing spray in a complex swirl injector using

- 914 large eddy simulation. *Proceedings of the Combustion Institute* 32 II,
915 2257–2266. URL: <http://dx.doi.org/10.1016/j.proci.2008.06.156>,
916 doi:10.1016/j.proci.2008.06.156.
- 917 Apte, S.V., Mahesh, K., Moin, P., Oefelein, J.C., 2003b. Large-eddy sim-
918 ulation of swirling particle-laden flows in a coaxial-jet combustor. *In-*
919 *ternational Journal of Multiphase Flow* 29, 1311–1331. doi:10.1016/
920 S0301-9322(03)00104-6.
- 921 Buaria, D., Yeung, P.K., 2017. A highly scalable particle tracking algo-
922 rithm using partitioned global address space (PGAS) programming for
923 extreme-scale turbulence simulations. *Computer Physics Communications*
924 221, 246–258. URL: <http://dx.doi.org/10.1016/j.cpc.2017.08.022>,
925 doi:10.1016/j.cpc.2017.08.022.
- 926 Capecelatro, J., Desjardins, O., 2013. An Euler-Lagrange strategy for sim-
927 ulating particle-laden flows. *Journal of Computational Physics* 238, 1–31.
928 URL: <http://dx.doi.org/10.1016/j.jcp.2012.12.015>, doi:10.1016/
929 j.jcp.2012.12.015.
- 930 Cha, C.M., Zhu, J., Rizk, N.K., Anand, M.S., 2005. A comprehensive liquid
931 fuel injection model for CFD simulations of gas turbine combustors. 43rd
932 AIAA Aerospace Sciences Meeting and Exhibit - Meeting Papers 8, 1809–
933 1825. doi:10.2514/6.2005-349.
- 934 Chin, J.S., Lefebvre, A.H., 1983. Steady-state evaporation characteristics of
935 hydrocarbon fuel drops. *AIAA Journal* 21, 1437–1443. doi:10.2514/3.
936 8264.
- 937 Chrigui, M., Gounder, J., Sadiki, A., Masri, A.R., Janicka, J., 2012. Partially
938 premixed reacting acetone spray using LES and FGM tabulated chem-

- 939 istry. Combustion and Flame 159, 2718–2741. URL: [http://dx.doi.org/](http://dx.doi.org/10.1016/j.combustflame.2012.03.009)
940 10.1016/j.combustflame.2012.03.009, doi:10.1016/j.combustflame.
941 2012.03.009.
- 942 Darmana, D., Deen, N.G., Kuipers, J.A.M., 2006. Parallelization of an Euler-
943 Lagrange model using mixed domain decomposition and a mirror domain
944 technique: Application to dispersed gas-liquid two-phase flow. Journal of
945 Computational Physics 220, 216–248. doi:10.1016/j.jcp.2006.05.011.
- 946 Fanfarillo, A., Del Vento, D., 2018. Notified access in coarray-based hydrody-
947 namics applications on many-core architectures: Design and performance.
948 Parallel Computing 75, 118–129. URL: [https://doi.org/10.1016/j.](https://doi.org/10.1016/j.parco.2018.04.002)
949 parco.2018.04.002, doi:10.1016/j.parco.2018.04.002.
- 950 Frank, T., Bernert, K., Pachler, K., Schneider, H., 2001. Aspects of effi-
951 cient parallelization of disperse gas-particle flow predictions using eulerian-
952 lagrangian approach., in: 4th International Conference on Multiphase
953 Flow, ICMF, New Orleans, Louisiana, USA.
- 954 Freitag, S., Meier, U., Heinze, J., Behrendt, T., Hassa, C., 2010. Measure-
955 ment of initial conditions of a kerosene spray from a generic aeroengine
956 injector at elevated pressure, in: 23rd Annual Conference on Liquid At-
957 omization and Spray Systems. ISBN-978-80-7399-997-1, Brno, Czech Re-
958 public. doi:10.1615/AtomizSpr.2011003457.
- 959 Gorokhovski, B.M., 2001. Stochastic sub-grid modeling of drop breakup for
960 LES of atomizing spray .
- 961 Gropp, W., Thakur, R., Lusk, E., 1999. Using MPI-2: Advanced Features
962 of the Message Passing Interface. 2nd ed., MIT Press.

- 963 Hiroyasu, H., Kadota, T., 1974. Fuel droplet size distribution in diesel com-
964 bustion chamber. SAE Technical Papers , 2615–2624doi:10.4271/740715.
- 965 Houzeaux, G., Garcia, M., Cajas, J.C., Artigues, A., Olivares, E., Labarta,
966 J., Vázquez, M., 2016. Dynamic load balance applied to particle transport
967 in fluids. International Journal of Computational Fluid Dynamics 30, 408–
968 418. doi:10.1080/10618562.2016.1227070.
- 969 Jofre, L., Borrell, R., Lehmkuhl, O., Oliva, A., 2015. Parallel load balancing
970 strategy for Volume-of-Fluid methods on 3-D unstructured meshes. Jour-
971 nal of Computational Physics 282, 269–288. URL: [http://dx.doi.org/](http://dx.doi.org/10.1016/j.jcp.2014.11.009)
972 [10.1016/j.jcp.2014.11.009](http://dx.doi.org/10.1016/j.jcp.2014.11.009), doi:10.1016/j.jcp.2014.11.009.
- 973 Jones, W.P., Marquis, A.J., Vogiatzaki, K., 2014. Large-eddy simulation of
974 spray combustion in a gas turbine combustor. Combustion and Flame 161,
975 222–239. URL: [http://dx.doi.org/10.1016/j.combustflame.2013.](http://dx.doi.org/10.1016/j.combustflame.2013.07.016)
976 [07.016](http://dx.doi.org/10.1016/j.combustflame.2013.07.016), doi:10.1016/j.combustflame.2013.07.016.
- 977 Kafui, D.K., Johnson, S., Thornton, C., Seville, J.P.K., 2011. Parallelization
978 of a Lagrangian-Eulerian DEM/CFD code for application to fluidized beds.
979 Powder Technology 207, 270–278. URL: [http://dx.doi.org/10.1016/j.](http://dx.doi.org/10.1016/j.powtec.2010.11.008)
980 [powtec.2010.11.008](http://dx.doi.org/10.1016/j.powtec.2010.11.008), doi:10.1016/j.powtec.2010.11.008.
- 981 Kahila, H., Wehrfritz, A., Kaario, O., Ghaderi Masouleh, M., Maes, N.,
982 Somers, B., Vuorinen, V., 2018. Large-eddy simulation on the influence
983 of injection pressure in reacting Spray A. Combustion and Flame 191,
984 142–159. doi:10.1016/j.combustflame.2018.01.004.
- 985 Kaludercic, B., 2004. Parallelisation of the Lagrangian model in a mixed
986 Eulerian-Lagrangian CFD algorithm. Journal of Parallel and Distributed
987 Computing 64, 277–284. doi:10.1016/j.jpdc.2003.11.010.

- 988 Liu, A.B., Mather, D., Reitz, R.D., 1993. Modeling the effects of drop drag
989 and breakup on fuel sprays. SAE Technical Papers doi:10.4271/930072.
- 990 Meier, U., Heinze, J., Freitag, S., Hassa, C., 2011. Spray and flame structure
991 of a generic injector at aeroengine conditions, in: Proceedings of ASME
992 Turbo Expo 2011: Power for Land, Sea and Air: GT2011, Vancouver,
993 Canada. doi:10.1115/1.4004262.
- 994 Message Passing Interface Forum, 2015. MPI: A Message-Passing In-
995 terface Standard, Version 3.1. High Performance Computing Center
996 Stuttgart (HLRS) , 868URL: <https://www.open-mpi.org/doc/v3.1/>,
997 doi:10.1021/ac800347j, arXiv:a.
- 998 van Oijen, J.A., Donini, A., Bastiaans, R.J., ten Thije Boonkkamp, J.H.,
999 de Goey, L.P., 2016. State-of-the-art in premixed combustion modeling
1000 using flamelet generated manifolds. Progress in Energy and Combustion
1001 Science 57, 30–74. URL: [http://dx.doi.org/10.1016/j.pecs.2016.07.](http://dx.doi.org/10.1016/j.pecs.2016.07.001)
1002 001, doi:10.1016/j.pecs.2016.07.001.
- 1003 Pankajakshan, R., Mitchell, B.J., Taylor, L.K., 2011. Simu-
1004 lation of unsteady two-phase flows using a parallel Eulerian-
1005 Lagrangian approach. Computers and Fluids 41, 20–26.
1006 URL: [https://www.scopus.com/inward/record.uri?eid=2-s2.](https://www.scopus.com/inward/record.uri?eid=2-s2.0-78649642411&doi=10.1016/j.compfluid.2010.09.020&partnerID=40&md5=cc71eaa2b9b2bfa1b1b18c992245d5ed)
1007 0-78649642411&doi=10.1016/j.compfluid.2010.09.
1008 020&partnerID=40&md5=cc71eaa2b9b2bfa1b1b18c992245d5ed,
1009 doi:10.1016/j.compfluid.2010.09.020.
- 1010 Patankar, S.V., 1980. Numerical heat transfer and fluid flow. Hemisphere
1011 Publishing Corporation (CRC Press, Taylor & Francis Group).

- 1012 Pierce, C.D., Moin, P., 1998. Method for generating equilibrium swirling
1013 inflow conditions. *AIAA Journal* 36, 1325–1327. doi:10.2514/2.518.
- 1014 Poinso, T., Veynante, D., 2005. *Theoretical and Numerical Combustion*.
1015 3rd ed.
- 1016 Rettinger, C., Rüde, U., 2019. Dynamic Load Balancing Techniques
1017 for Particulate Flow Simulations. *Computation* 7, 9. doi:10.3390/
1018 computation7010009, arXiv:1811.12742.
- 1019 Sbalzarini, I.F., Walther, J.H., Bergdorf, M., Hieber, S.E., Kotsalis, E.M.,
1020 Koumoutsakos, P., 2006. PPM - A highly efficient parallel particle-mesh
1021 library for the simulation of continuum systems. *Journal of Computational*
1022 *Physics* 215, 566–588. doi:10.1016/j.jcp.2005.11.017.
- 1023 Shigeto, Y., Sakai, M., 2011. Parallel computing of discrete element method
1024 on multi-core processors. *Particuology* 9, 398–405. URL: <http://dx.doi.org/10.1016/j.partic.2011.04.002>, doi:10.1016/j.partic.2011.04.
1025 002.
1026
- 1027 Sitaraman, H., Grout, R., 2016. Balancing conflicting requirements for
1028 grid and particle decomposition in continuum-Lagrangian solvers. *Par-*
1029 *allel Computing* 52, 1–21. URL: [http://dx.doi.org/10.1016/j.parco.](http://dx.doi.org/10.1016/j.parco.2015.10.010)
1030 2015.10.010, doi:10.1016/j.parco.2015.10.010.
- 1031 Vázquez, M., Houzeaux, G., Koric, S., Artigues, A., Aguado-Sierra, J.,
1032 Arís, R., Mira, D., Calmet, H., Cucchietti, F., Owen, H., Taha, A.,
1033 Burness, E.D., Cela, J.M., Valero, M., 2016. Alya: Multiphysics engi-
1034 neering simulation toward exascale. *Journal of Computational Science*
1035 14, 15–27. URL: <http://dx.doi.org/10.1016/j.jocs.2015.12.007>,
1036 doi:10.1016/j.jocs.2015.12.007.

- 1037 Veynante, D., Vervisch, L., 2002. Turbulent combustion modeling.
1038 Progress in Energy and Combustion Science 28, 193–266. doi:10.1016/
1039 S0360-1285(01)00017-X.
- 1040 Yakubov, S., Cankurt, B., Abdel-Maksoud, M., Rung, T., 2013. Hy-
1041 brid MPI/OpenMP parallelization of an euler-lagrange approach to
1042 cavitation modelling. Computers and Fluids 80, 365–371. URL:
1043 <http://dx.doi.org/10.1016/j.compfluid.2012.01.020>, doi:10.1016/
1044 j.compfluid.2012.01.020.
- 1045 Zhao, X., Wang, J., Zhang, S., 2004. Parallel CFD-DEM for Fluid-Particle
1046 Systems, in: ASME 2004 Heat Transfer/Fluids Engineering Summer Con-
1047 ference Volume 3, pp. 1–10.
- 1048 Zhu, J., Gupta, A., Nastase, C., Anand, M.S., 2013. Development of an LES
1049 approach for compressible reacting flows. 51st AIAA Aerospace Sciences
1050 Meeting including the New Horizons Forum and Aerospace Exposition
1051 2013 doi:10.2514/6.2013-174.

Conflict of Interest

This work has been partly funded by Rolls-Royce and the UK EPSRC under grant number EP/L015943/1 as part of the CDT in Gas Turbine Aerodynamics.