Computer Methods in Applied Mechanics and Engineering 418 (2024) 116571

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

A voxel-based machine-learning framework for thermo-fluidic identification of unknown objects

T.I. Zohdi

Department of Mechanical Engineering, 6117 Etcheverry Hall, University of California, Berkeley, CA, 94720-1740, USA

ARTICLE INFO

Keywords:
Unidentified object
Thermo-fluidic signature
Machine-learning

ABSTRACT

The rapid identification of unknown objects by their thermo-fluid flow field signature is becoming increasingly more important. In this work, a machine-learning framework is developed that efficiently simulates and adapts object geometries in order to match the thermo-flow field signature generated by an unknown object, across a time series of voxel-frames. In order to achieve this, a thermo-fluid model is developed, based on the Navier–Stokes equations and the first law of thermodynamics, using a voxel rendering of the system, which is rapidly solved with a voxel-tailored, temporally-adaptive, iterative solution scheme. This voxel-framework is then combined with a genomic-based machine-learning algorithm to develop a digital-twin (digital-replica) of the system that can run in real-time or faster than the actual physical system. Numerical examples are provided to illustrate the framework.

1. Introduction

Since their inception, cameras have been continuously used for medical, industrial and military purposes. In the last 20 years, there has been a dramatic evolution incorporating (a) multispectral cameras, based on technologies that extend the classical visible wavelength paradigms (380–720 nm), to thermographic/infrared regimes (1000–14000 nm) to create an image and (b) 3D (time-of-flight) cameras, using LIDAR, radio-based imaging and tomography. They have fundamentally changed the ability to extract information from complex events that would have been unthinkable until recently. In particular, the ability to extract thermo-fluid flow data in three dimensions, utilizing real-time tomographically-based imaging, has now brought forth many possibilities (Fig. 1). For example, tomography, which uses multidirectional penetrating waves and the splicing of sections with tomographic reconstruction, has become a critical component for 3D data extraction. Several commercial products now exist that enable the instantaneous measurement within a 3D measurement volume of all three velocity components, in addition to thermal fields (Thermographic Particle Image Velocimetry (PIV)). This immediately allows for voxelization of a 3D space, which yields an image comprised of 3D cubes (volume pixels=voxels), each containing velocity and thermal field data.[1] Thus, it is now possible to rapidly extract, frame-by-frame, voxel fields of dynamic thermo-fluidic events. We refer the interested reader to Elsinga et al. [1], Herman [2], Herman and Lent et al. [3], Schroder et al. [4], Wienke [5], Aguirre-Pablo et al. [6], Atkinson and Soria [7], Discetti et al. [8], Geoghegan et al. [9], Willert et al. [10], Buchmann et al. [11], Casey et al. [12], Tien et al. [13], Xiong et al. [14], Watamura et al. [15], Klinner and Willert [16], McPhail et al. [17], Cierpka et al. [18], Scarano et al. [19], McPhail et al. [20], Hicham et al. [21], Zhu et al. [22], Lynch and Wagner [23], Liu et al. [24], He et al. [25], Liu and Mason [26], Chilton [27], Tariq

E-mail address: zohdi@berkeley.edu.

[1] Tomographic-PIV utilizes tomographic reconstruction (Computed Tomography, CT) of voxel intensities, as is the norm in medical applications. For example, one of most advanced systems in the world is the FlowMaster Tomographic PIV https://www.lavision.de/en/products/flowmaster/tomographic-piv/, which allows for instantaneous measurement of all three components of the velocity in a 3D volume and uses CT to reconstruct voxel intensities. Other products such as Microvec https://piv.com.sg/piv-products/tomographic-piv/ and HSI: https://hsi.ca/product/tomo-piv/ have similar approaches.

https://doi.org/10.1016/j.cma.2023.116571
Received 24 September 2023; Received in revised form 24 October 2023; Accepted 25 October 2023
Available online 10 November 2023
0045-7825/

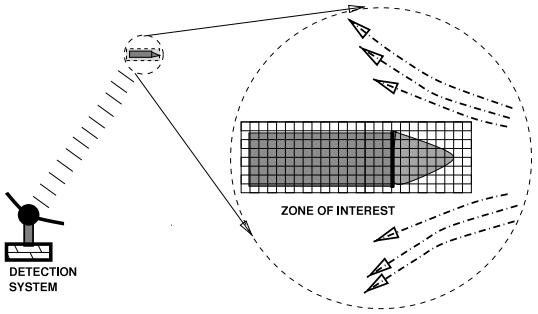

Fig. 1. A model problem of object detection.

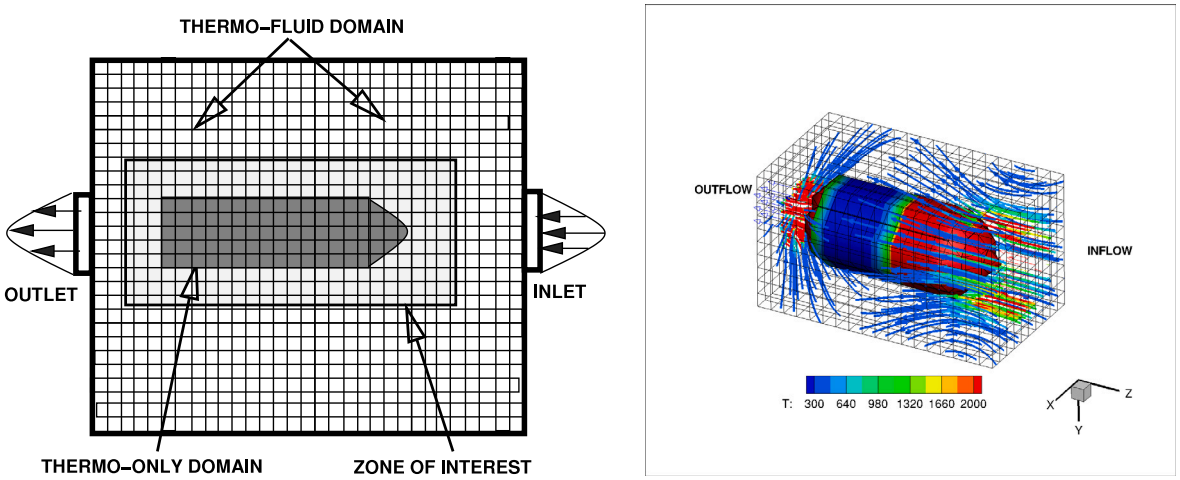

Fig. 2. Model problem: A cross-section of the schematic for the system with the two types of domains: (a) A thermo-fluid domain (ambient domain) and a (solid) thermo-only domain(unidentified object). (b) The domain of interest where the object is detected. Fig. 7 illustrates the results (evolution of flow streamlines).

et al. [28] for details on the wide-range of topics discussed. This is particularly critical due to the increase of unidentified flying objects, heat seeking weaponry, drones, high-altitude remote sensing, satellite constellations, etc., in a progressively more crowded airspace. Accordingly, it is critical to perform inverse problem calculations to ascertain what is producing a detected signature-*which is the subject of the current work.*

In this work, a machine-learning framework is developed that rapidly simulates and adapts object geometries in order to match the thermo-flow field signature generated by an unknown object, across a time series of voxel-frames. In order to achieve this, a thermo-fluid model is developed, based on the Navier–Stokes equations and the first law of thermodynamics, using a voxel rendering of the system, which is rapidly solved with a voxel-tailored, temporally-adaptive, iterative solution scheme. This voxel-framework is then combined with a genomic-based machine-learning algorithm to develop a digital-twin (digital-replica) of the system that can run in real-time or faster than the actual physical system. Numerical examples are provided to illustrate the framework on the model problem shown in Fig. 3.

2. Camera to PDE: *Voxel-based computing for voxel-based data*

Tomographic-PIV has evolved over the last 20 years to extend PIV to measuring 3D vector fields, and is based on computing the velocity vector field in a flow from the displacement of imaged tracer particles, from two subsequently captured images of the region of interest. This employs a multicamera stereoscopic set up, focusing on a 3D volume. Two subsequent time steps are illuminated and imaged in order to compute velocities. This employs tomographic reconstruction of an instantaneous particle distribution based on the projections of this distribution onto several cameras where tiny tracer particles are added to the flow of interest and illuminated with a short laser light pulse. The light scattered by the particles is captured by several cameras (typically 3–6), which are arranged to maximize stereographic processing. The process also uses MART (Multiplicative Algebraic Reconstruction Technique), which was introduced by Herman and Lent [3] that creates a digital voxel representation of the volume, where the intensity values correlate to the values represented by the particles at those locations. We refer the interested reader to Elsinga et al. [1]-Tariq et al. [28].

Relatively recently, Aguirre-Pablo, et al. [6] demonstrated the viability of using four low-cost smartphone cameras to perform Tomographic PIV using colored shadows to imprint two or three different time-steps on the same image. In that work, they used

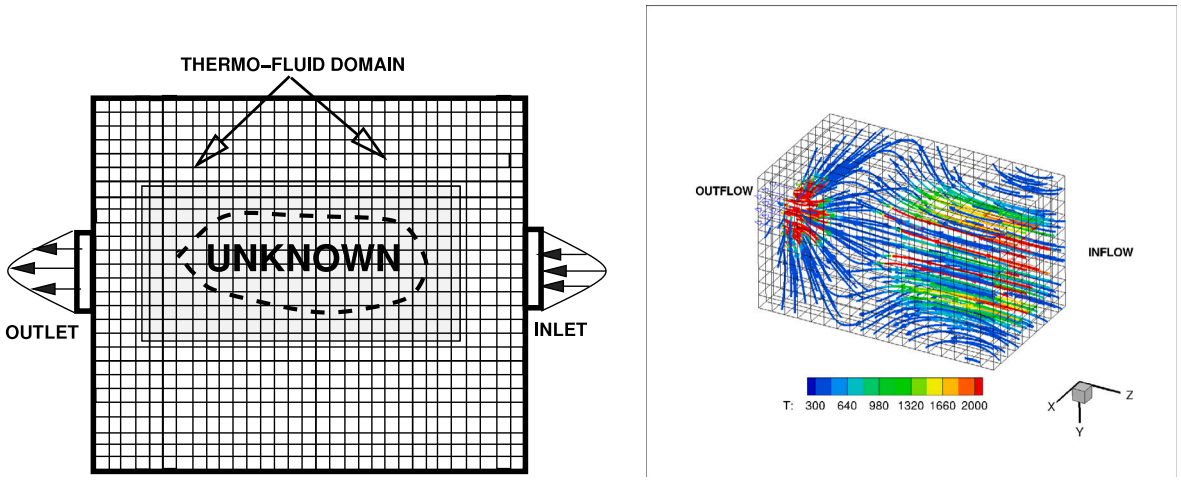

Fig. 3. Model problem: A cross-section of the schematic for the system with the desired domain identified. The calculation is first made without the object present, which then provides the objective function for the system with the object present.

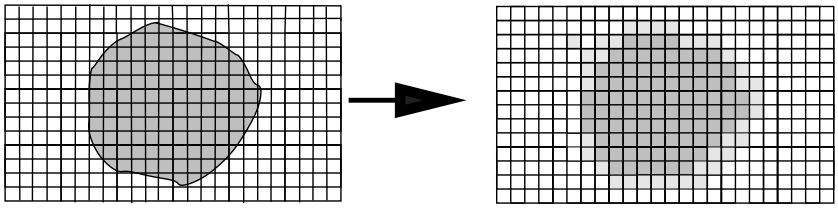

Fig. 4. Left: an actual structure and Right: a voxel representation.

commercially available Tomographic PIV software for the calibration, 3-D particle reconstruction and particle–field correlations, to obtain all three velocity components in a volume. Their system was compared to a commercial stereoscopic PIV system for error estimates and provided a proof of concept that can make Tomographic-PIV cost a fraction of traditional approaches, in particular due to the massive economies of scale associated with the worldwide spread of smartphone camera technologies, similar to the ubiquitous spread of LIDAR, which is driven more by consumer demand than scientific need. Thus, as the authors indicate, taking advantage of smartphone computing power to process data and enabling smartphone Tomographic-PIV is, as with so many previous technologies, not a matter of "if", it is a matter of "when".

The huge increase in camera capabilities has led to "digital/voxel" based computing methods, which utilize camera-generated voxels (Figs. 4) as computational units of a very regular "voxel-grid". In order to solve PDE's posed over such domains, extremely fast methods can then be used to construct the various derivatives needed in a differential operator, circumventing meshing, mapping, volume integration and stiffness matrix generation (needed for example in Finite Element Methods), as well as matrix-based solution methods, since the voxel structure allows for incredibly efficient matrix-free iterative solvers. In short, voxel-based camera data is ideally-suited to voxel-based computation. The use of voxels (Foley et al. [29]), is widespread in the visualization and analysis of medical and scientific data (Chmielewski et al. [30]) and in the video-gaming industry. The well-known video-game "Outcast", and others in the 1990s employed this graphics technique for effects such as reflection and bump-mapping and usually for terrain rendering, although other techniques have overtaken it as the method of choice. However, the most widely used application of a voxel is to represent solid and fluid structures possessing heterogeneous material properties. For example, in CT scans, so-called Hounsfield units are used which measure the opacity of material to X-rays Novelline [31]. There are approximately 30 different types of values acquired from MRI or ultrasound.

Additionally, LIDAR-based technologies can acquire more data. LIDAR is a technique by which a target is illuminated with a laser and the reflected light is analyzed. LIDAR was developed in the 1960's and combines laser focusing with radar-like technology for calculating distances by measuring the time for a signal to return. It enjoys a wide range of uses (see [32–37]) in the motion capture community and is a relatively standard tool in the atmospheric sciences, ranging from remote sensing, airborne laser mapping and cloud measurement, and has been extended to a variety of applications in engineering and science. Typically, LIDAR uses high-frequency ultraviolet, visible of near infrared light. For reviews, we refer the reader to Ring [38], Cracknell and Hayes [39], Goyer and Watson [40], Medina et al. [41] and Trickey et al. [42]. LIDAR can be considered as one of a family of methods classified as "time-of-flight" cameras, whereby a pulse (optical) energy is released and the time it takes to reach the target and to return, coupled with knowledge of the propagation speed, determines the relative distance. Such devices have steadily evolved for the last

20 years ([43–49]). In particular, due to the large number of UAV's and aircraft available, LIDAR has become quite popular as a surveying method that measures distance by illuminating the target-essentially as a 3D laser scanner. LIDAR became popular in 1971 to map the moon's surface and also as a lunar altimeter. The most common use of time-of-flight methods is as a Digital Elevation Model (DEM) using the concept of a point cloud. The wavelengths used are most commonly 600–1000 nm. However, 1550 nm devices are "eye safe" (for large surface areas), but less accurate (this requires the use of gallium-arsenide, which is costly). Typical airborne topographic lasers operate at 1069 nm. Historically, they enjoy widespread use in agriculture, archaeology, etc. coupled with unmanned autonomous vehicles. This ranges from (1) Profiling LIDAR, where an individual pulse is placed in a single line (2) Small footprint LIDAR: scans at 20 degrees back and forth (3) Large-footprint LIDAR, which pulses an entire area (4) Topographical LIDAR using near infrared (5) Bathymetric LIDAR, which is water penetrating green light to measure seafloor and riverbed elevations and (6) Ground-based LIDAR, where a tripod unit scans a hemisphere. The components can entail (1) LIDAR sensors which scan from side to side: green light or infrared, (2) GPS receivers: tracks the altitude of the plane or UAV, (3) Inertial measurements (IMU) tracks the tilt of the plane or UAV and (4) Computers/data recorders. There are usually two forms of signal storage: (1) Discrete LIDAR stores peak return signals and (2) Full wave LIDAR, which stores everything. Voxelization of LIDAR data can be performed by dividing the entire scanned volume into a collection of 3D regular cubes (voxels) and voxel values are assigned according to the attribute values of the LIDAR point(s) within the corresponding voxels. In summary, regardless of the types of camera technologies combined, in many cases, the voxels are already supplied, and it makes little sense to employ, for example, Finite Element Methods, involving meshing, mapping, volume integration, stiffness matrix generation, etc.

3. Objective function construction: matching voxel-by-voxel and frame by frame

We develop an objective function based on the sum of a frame by frame, voxel by voxel difference (each located at $\boldsymbol{x}_i^v, i = 1,2,3...N_v$) between the trial simulation and the observed data-set quantities of interest

$$\Pi(\Lambda_1,\dots\Lambda_N) \propto \sum_{j=1}^{W}\sum_{f=1}^{N_f}\sum_{i=1}^{N_v} w_j \|A^{sim,w}(\boldsymbol{x}_i^v) - A^{data,w}(\boldsymbol{x}_i^v)\|_{L_2(\Omega_f)}, \tag{3.1}$$

where the number of frames are $f = 1,2,\dots,N_f$, the number of voxels are $i = 1,2,\dots,N_v$, w_j are the number of weighted different objectives, $L_2(\Omega_f)$ is the norm over each domain frame volume Ω_f and the structural design variables are $\boldsymbol{\Lambda} = \{\Lambda_1,\dots\Lambda_N\}$. Following Zohdi [50–54], we formulate the objective as a cost-error function minimization problem that seeks geometrical designs that produce a match. We specifically focus minimizing a thermo-fluid flow objective comprised of

$$\begin{aligned}\pi(\Lambda_1,\dots\Lambda_N) \;=\; & w_\theta \underbrace{\frac{1}{N_v N_f}\frac{\sum_{f=1}^{N_f}\sum_{i=1}^{N_v}\|\theta^{sim}(\boldsymbol{x}_i^v) - \theta^{data}(\boldsymbol{x}_i^v)\|_{L_2(\Omega_f)}}{\sum_{f=1}^{N_f}\sum_{i=1}^{N_v}\|\theta^{data}(\boldsymbol{x}_i^v)\|_{L_2(\Omega_f)}}}_{\Pi^{(\theta)}} \\ & + w_{\boldsymbol{v}} \underbrace{\frac{1}{N_v N_f}\frac{\sum_{f=1}^{N_f}\sum_{i=1}^{N_v}\|\boldsymbol{v}^{sim}(\boldsymbol{x}_i^v) - \boldsymbol{v}^{data}(\boldsymbol{x}_i^v)\|_{L_2(\Omega_f)}}{\sum_{f=1}^{N_f}\sum_{i=1}^{N_v}\|\boldsymbol{v}^{data}(\boldsymbol{x}_i^v)\|_{L_2(\Omega_f)}}}_{\Pi^{(\boldsymbol{v})}} \\ & + w_s \underbrace{\frac{1}{N_v N_f}\frac{\sum_{f=1}^{N_f}\sum_{i=1}^{N_v}\|S^{sim}(\boldsymbol{x}_i^v) - S^{data}(\boldsymbol{x}_i^v)\|_{L_2(\Omega_f)}}{\sum_{f=1}^{N_f}\sum_{i=1}^{N_v}\|S^{data}(\boldsymbol{x}_i^v)\|_{L_2(\Omega_f)}}}_{\Pi^{(Voxel)}},\end{aligned} \tag{3.2}$$

where θ is the temperature field AND $\boldsymbol{v}$ is the velocity field. $S(\boldsymbol{x}_i^v)$ indicates that the voxel is considered to be a solid. We further normalize the cost function to yield

$$\Pi(\Lambda_1,\dots\Lambda_N) = \frac{\pi(\Lambda_1,\dots\Lambda_N)}{w_\theta + w_{\boldsymbol{v}} + w_s}. \tag{3.3}$$

4. Governing equations

For completeness, in the appendix we start from first principles, proceeding by developing a coupled thermo-fluid model for the fluid surrounding the unidentified object and the heat-generated by the fluid flow, resulting in the following for the balance of linear momentum,

$$\frac{\partial \boldsymbol{v}}{\partial t} = \frac{1}{\rho}\left(\nabla_x \cdot \boldsymbol{\sigma} + \boldsymbol{f}\right) - \nabla_x \boldsymbol{v}\cdot\boldsymbol{v}, \tag{4.1}$$

where ρ is the density, $\boldsymbol{v}$ is the velocity field, $\boldsymbol{\sigma}$ is the stress field and $\boldsymbol{f}$ are body forces and for the first law of thermodynamics we have

$$\frac{\partial \theta}{\partial t} = \frac{1}{\rho C}\left(\boldsymbol{\sigma} : \nabla_x \boldsymbol{v} - \nabla_x \cdot \boldsymbol{q} + \rho z\right) - \nabla_x \theta \cdot \boldsymbol{v}, \tag{4.2}$$

where θ is the temperature field, C is the heat capacity, $\boldsymbol{q}$ is the thermal flux and $\boldsymbol{z}$ are the heat sources (see Figs. 5, 6 and 8).

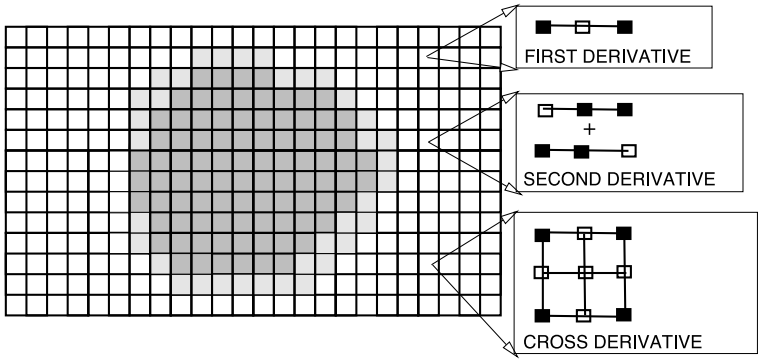

Fig. 5. Voxel-stencils for the (a) derivative, $\frac{\partial()}{\partial x_1}$, (b) the second derivative $\frac{\partial^2()}{\partial x_1^2}$ and (c) the cross derivative, $\frac{\partial^2()}{\partial x_1 \partial x_2}$ Eqs. (6.1)–(6.8) provide the details. Laplacian smoothing is also applied to the assumed solid–fluid interface, using Eqs. (6.9) and (6.10).

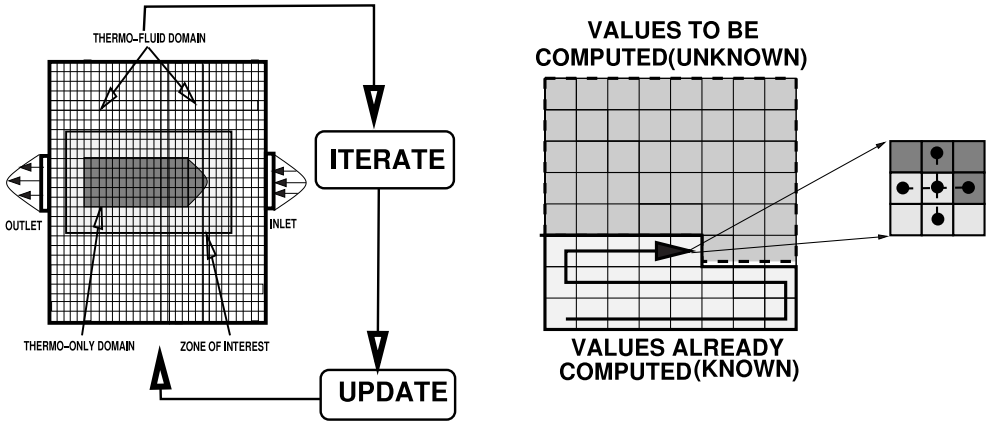

Fig. 6. The overall iterative (left) solution and the matrix-free approach using a moving front through the voxels(right) During the iterative solution process, the most current value of a voxel is used in any calculation, for example a construction of the Laplacian, or any other term in the governing differential equations.

5. Temporal discretization

For the fluid, we write

$$\frac{\partial \boldsymbol{v}}{\partial t} = \frac{1}{\rho}\left(\nabla_x \cdot \boldsymbol{\sigma} + \boldsymbol{f}\right) - \nabla_x \boldsymbol{v} \cdot \boldsymbol{v} \stackrel{\text{def}}{=} \boldsymbol{L}. \tag{5.1}$$

We discretize for time=$t + \phi\Delta t$, and using a trapezoidal "$\phi - scheme$" ($0 \le \phi \le 1$)

$$\frac{\partial \boldsymbol{v}}{\partial t} \approx \frac{\boldsymbol{v}(t+\Delta t) - \boldsymbol{v}(t)}{\Delta t} \approx \boldsymbol{L}(t + \phi\Delta t) \approx \phi \boldsymbol{L}(t+\Delta t) + (1-\phi)\boldsymbol{L}(t). \tag{5.2}$$

Rearranging yields

$$\boldsymbol{v}(t+\Delta t) \approx \boldsymbol{v}(t) + \Delta t\left(\phi \boldsymbol{L}(t+\Delta t) + (1-\phi)\boldsymbol{L}(t)\right) \tag{5.3}$$

where the voxel-based spatial discretization, which will be applied to the derivative terms (such as $\nabla_x \cdot \boldsymbol{\sigma}$), are contained in $\boldsymbol{L}$. The discretized system is formulated next as an implicit time-stepping scheme within each time step L.

Remark 1. The same process is applied to the thermal field

$$\frac{\partial \theta}{\partial t} = \frac{1}{\rho C}\left(\boldsymbol{\sigma} : \nabla_x \boldsymbol{v} - \nabla_x \cdot \boldsymbol{q} + \rho z\right) - \nabla_x \theta \cdot \boldsymbol{v} \stackrel{\text{def}}{=} Z, \tag{5.4}$$

yielding

$$\theta(t+\Delta t) \approx \theta(t) + \Delta t\left(\phi Z(t+\Delta t) + (1-\phi)Z(t)\right) \tag{5.5}$$

6. Spatial voxel-based discretization

Referring to Fig. 2, the following voxel-based approximations are used:

1. VOXEL GRADIENT: For the first derivative of a primal variable v at (x_1, x_2, x_3):

$$\frac{\partial v}{\partial x_1} \approx \frac{v(x_1 + \Delta x_1, x_2, x_3) - v(x_1 - \Delta x_1, x_2, x_3)}{2\Delta x_1} \tag{6.1}$$

2. VOXEL LAPLACIAN: For the derivative of a flux at (x_1, x_2, x_3):

$$\begin{aligned} \frac{\partial}{\partial x_1}\left(A\frac{\partial v}{\partial x_1}\right) &\approx \frac{\left(A\frac{\partial v}{\partial x_1}\right)|_{x_1+\frac{\Delta x_1}{2},x_2,x_3} - \left(A\frac{\partial v}{\partial x_1}\right)|_{x_1-\frac{\Delta x_1}{2},x_2,x_3}}{\Delta x_1} \\ &= \frac{1}{\Delta x_1}\left[A(x_1+\frac{\Delta x_1}{2},x_2,x_3)\left(\frac{v(x_1+\Delta x_1,x_2,x_3)-v(x_1,x_2,x_3)}{\Delta x_1}\right)\right] \\ &- \frac{1}{\Delta x_1}\left[A(x_1-\frac{\Delta x_1}{2},x_2,x_3)\left(\frac{v(x_1,x_2,x_3)-v(x_1-\Delta x_1,x_2,x_3)}{\Delta x_1}\right)\right], \end{aligned} \tag{6.2}$$

where we have used

$$A(x_1 + \frac{\Delta x_1}{2}, x_2, x_3) \approx \frac{1}{2}\left(A(x_1 + \Delta x_1, x_2, x_3) + A(x_1, x_2, x_3)\right) \tag{6.3}$$

and

$$A(x_1 - \frac{\Delta x_1}{2}, x_2, x_3) \approx \frac{1}{2}\left(A(x_1, x_2, x_3) + A(x_1 - \Delta x_1, x_2, x_3)\right) \tag{6.4}$$

3. VOXEL CROSS-DERIVATIVE: For the cross-derivative of a flux at (x_1, x_2, x_3):

$$\begin{aligned} \frac{\partial}{\partial x_2}\left(A\frac{\partial v}{\partial x_1}\right) &\approx \frac{\partial}{\partial x_2}\left(A(x_1,x_2,x_3)\left(\frac{v(x_1+\Delta x_1,x_2,x_3)-v(x_1-\Delta x_1,x_2,x_3)}{2\Delta x_1}\right)\right) \\ &\approx \frac{1}{4\Delta x_1 \Delta x_2}(A(x_1,x_2+\Delta x_2,x_3)\left[v(x_1+\Delta x_1,x_2+\Delta x_2,x_3)-v(x_1-\Delta x_1,x_2+\Delta x_2,x_3)\right] \\ &- A(x_1,x_2-\Delta x_2,x_3)\left[v(x_1+\Delta x_1,x_2-\Delta x_2,x_3)-v(x_1-\Delta x_1,x_2-\Delta x_2,x_3)\right]). \end{aligned} \tag{6.5}$$

Remark 2. To illustrate second-order accuracy, consider a Taylor series expansion for an arbitrary function w

$$w(x+\Delta x) = w(x) + \frac{\partial w}{\partial x}|_x \Delta x + \frac{1}{2}\frac{\partial^2 w}{\partial x^2}|_x (\Delta x)^2 + \frac{1}{6}\frac{\partial^3 w}{\partial x^3}|_x (\Delta x)^3 + \mathcal{O}((\Delta x)^4) \tag{6.6}$$

and

$$w(x-\Delta x) = w(x) - \frac{\partial w}{\partial x}|_x \Delta x + \frac{1}{2}\frac{\partial^2 w}{\partial x^2}|_x (\Delta x)^2 - \frac{1}{6}\frac{\partial^3 w}{\partial x^3}|_x (\Delta x)^3 + \mathcal{O}((\Delta x)^4) \tag{6.7}$$

Subtracting the two expressions yields

$$\frac{\partial w}{\partial x}|_x = \frac{w(x+\Delta x) - w(x-\Delta x)}{2\Delta x} + \mathcal{O}((\Delta x)^2). \tag{6.8}$$

All other derivatives follow from this basic process, which is relatively standard in the Finite Difference community.

Remark 3. At the length-scales of interest, it is questionable whether the ideas of a sharp material interface are justified. Accordingly, later, we simulated the system with and without Laplacian smoothing, whereby one smooths the material data by post-processing the original material data, voxel by voxel, to produce a smoother material representation, for example, for the density, $\hat{\rho}$ (using the previous voxel approximations and nodal subscript notation):

$$\begin{aligned} \nabla^2 \rho &= \frac{1}{(\Delta x_i)^2}\left(\rho_{i+1,j,k} - 2\rho_{i,j,k} + \rho_{i-1,j,k}\right) \\ &+ \frac{1}{(\Delta x_j)^2}\left(\rho_{i,j+1,k} - 2\rho_{i,j,k} + \rho_{i,j-1,k}\right) \\ &+ \frac{1}{(\Delta x_k)^2}\left(\rho_{i,j,k+1} - 2\rho_{i,j,k} + \rho_{i,j,k-1}\right) = 0 \end{aligned} \tag{6.9}$$

which yields a smoother value of $\rho_{i,j,k}$, denoted $\hat{\rho}_{i,j,k}$, given by

$$\nabla^2 \rho = 0 \Rightarrow \hat{\rho}_{i,j,k} = \frac{1}{6}\left(\rho_{i+1,j,k} + \rho_{i-1,j,k} + \rho_{i,j+1,k} + \rho_{i,j-1,k} + \rho_{i,j,k+1} + \rho_{i,j,k-1}\right). \tag{6.10}$$

The same process was applied to the other parameters, generically denoted, $A(\boldsymbol{x})$, by enforcing $\nabla_x^2 A = 0$, as well as for any other material data. The simulations were run with and without data smoothing, with the results being negligibly different for sufficiently fine voxel-meshes.

7. Overall iterative (implicit) solution method

Following the basic framework in Zohdi [55–58], let us consider the finite difference nodes (i) for the velocity field:

$$\boldsymbol{v}_i^{L+1,K} = \boldsymbol{v}_i^{L} + \Delta t \left(\phi \boldsymbol{L}_i^{L+1,K-1} + (1-\phi)\boldsymbol{L}_i^{L} \right), \tag{7.1}$$

where i is the node counter, which is of the form

$$\boldsymbol{v}_i^{L+1,K} = \mathcal{G}(\boldsymbol{v}_i^{L+1,K-1}) + R_i, \tag{7.2}$$

where $K = 1, 2, 3, \ldots$ is the index of iteration within time step $L+1$ and

- $\mathcal{G}(\boldsymbol{v}_i^{L+1,K-1}) = \phi \Delta t \boldsymbol{L}_i^{L+1,K-1}$ and
- $R_i = \boldsymbol{v}_i^{L} + \Delta t(1-\phi)\boldsymbol{L}_i^{L}$.

The term R_i is a remainder term that does not depend on the current solution (only on the previous time step's solution). The convergence of such a scheme is dependent on the behavior of $\mathcal{G}$. Namely, a sufficient condition for convergence is that $\mathcal{G}$ is a contraction mapping for all $\boldsymbol{v}_i^{L+1,K}$, $K = 1, 2, 3, \ldots$ In order to investigate this further, we define the iteration error as

$$\epsilon_i^{L+1,K} \stackrel{\text{def}}{=} \boldsymbol{v}_i^{L+1,K} - \boldsymbol{v}_i^{L+1}. \tag{7.3}$$

A necessary restriction for convergence is iterative self-consistency, i.e. the "exact" (discretized) solution must be represented by the scheme, $\boldsymbol{v}_i^{L+1} = \mathcal{G}(\boldsymbol{v}_i^{L+1}) + R_i$. Enforcing this restriction, a sufficient condition for convergence is the existence of a contraction mapping

$$\begin{aligned} \| \underbrace{\boldsymbol{v}_i^{L+1,K} - \boldsymbol{v}_i^{L+1}}_{\epsilon_i^{L+1,K}} \| &= \| \mathcal{G}(\boldsymbol{v}_i^{L+1,K-1}) - \mathcal{G}(\boldsymbol{v}_i^{L+1}) \| \\ &\leq \eta^{L+1,K} \| \boldsymbol{v}_i^{L+1,K-1} - \boldsymbol{v}_i^{L+1} \|, \end{aligned} \tag{7.4}$$

where, if $0 \leq \eta^{L+1,K} < 1$ for each iteration K, then $\epsilon_i^{L+1,K} \rightarrow \boldsymbol{0}$ for any arbitrary starting value $\boldsymbol{v}_i^{L+1,K=0}$, as $K \rightarrow \infty$, which is a contraction condition that is sufficient, but not necessary, for convergence. The convergence of Eq. (7.1) is scaled by $\eta \propto \phi \Delta t$. Therefore, we see that the contraction constant of $\mathcal{G}$ is:

- directly dependent on the magnitude of the interaction forces ($\|\boldsymbol{L}\|$),
- directly proportional to Δt.

Thus, decreasing the time step size improves the convergence. *In order to maximize the time-step sizes (to decrease overall computing time) and still meet an error tolerance on the numerical solution's accuracy*, we build on an approach originally developed for continuum thermo-chemical multifield problems (Zohdi [55–58]), where one assumes: (1) $\eta^{L+1,K} \approx S(\Delta t)^p$, ($S$ is a constant) and (2) the error within an iteration behaves according to $(S(\Delta t)^p)^K \epsilon^{L+1,K} = \epsilon^{L+1,K}$, $K = 1, 2, \ldots$, where $\epsilon^{L+1,0} = \boldsymbol{v}_i^{L+1,K=1} - \boldsymbol{v}_i^{L}$ is the initial norm of the iterative (relative) error and S is intrinsic to the system. For example, for second-order problems, due to the quadratic dependency on Δt, $p \approx 1$. The objective is to meet an error tolerance in exactly a preset (the analyst sets this) number of iterations. To this end, one writes $(S(\Delta t_{\text{tol}})^p)^{K_d} \epsilon^{L+1,0} = TOL$, where TOL is a tolerance and where K_d is the number of desired iterations. If the error tolerance is not met in the desired number of iterations, the contraction constant $\eta^{L+1,K}$ is too large. Accordingly, one can solve for a new smaller step size, under the assumption that S is constant,

$$\Delta t_{\text{tol}} = \Delta t \underbrace{\left(\frac{(\frac{TOL}{\epsilon^{L+1,0}})^{\frac{1}{pK_d}}}{(\frac{\epsilon^{L+1,K}}{\epsilon^{L+1,0}})^{\frac{1}{pK}}} \right)}_{\stackrel{\text{def}}{=} \Lambda_K}. \tag{7.5}$$

The assumption that S is constant is not critical, since the time steps are to be recursively refined and unrefined throughout the simulation. Clearly, the expression in Eq. (7.5) can also be used for time step enlargement, if convergence is met in less than K_d iterations (typically chosen to be between five to ten iterations). Specifically, the solution steps are, for a multiphysics problem ($\boldsymbol{v}$ and θ) within a time-step:

- (1): Start a global fixed iteration (set $i = 1, \ldots, N_n$ (node counter) and $K = 0$ (iteration counter))

- (2): If $i > N_n$ then go to (4)
- (3): If $i \leq N_n$ then:
  - (a) Compute the velocity $\boldsymbol{v}_i^{L+1,K}$ and $\theta_i^{L+1,K}$
  - (b) Go to (2) for the next node ($i = i + 1$)
- (4): Measure weighted error (normalized) quantities
  - (a) $\epsilon_K \stackrel{\text{def}}{=} \gamma_v \dfrac{\sum_{i=1}^{N_n} \|\boldsymbol{v}_i^{L+1,K} - \boldsymbol{v}_i^{L+1,K-1}\|}{\sum_{i=1}^{N_n} \|\boldsymbol{v}_i^{L+1,K}\|} + \gamma_\theta \dfrac{\sum_{i=1}^{N_n} \|\theta_i^{L+1,K} - \theta_i^{L+1,K-1}\|}{\sum_{i=1}^{N_n} \|\theta_i^{L+1,K}\|}$
  - (b) $E_K \stackrel{\text{def}}{=} \dfrac{\epsilon_K}{TOL_r}$
  - (c) $\Lambda_K \stackrel{\text{def}}{=} \left(\dfrac{(\frac{TOL}{\epsilon^0})^{\frac{1}{pK_d}}}{(\frac{\epsilon^K}{\epsilon^0})^{\frac{1}{pK}}} \right)$.
- (5): If the tolerance is met: $E_K \leq 1$ and $K < K_d$ then
  - (a) Increment time: $t = t + \Delta t$
  - (b) Construct the next time step: $\Delta t^{new} = \Lambda_K \Delta t^{old}$,
  - (c) Select the minimum size: $\Delta t = min(\Delta t^{lim}, \Delta t^{new})$ and go to (1)
- (6): If the tolerance is not met: $E_K > 1$ and $K < K_d$ then
  - (a) Update the iteration counter: $K = K + 1$
  - (b) Reset the node counter: $i = 1$
  - (c) Go to (2)
- (7): If the tolerance is not met ($E_K > 1$) and $K = K_d$ then
  - (a) Construct a new time step: $\Delta t^{new} = \Lambda_K \Delta t^{old}$
  - (b) Restart at time t and go to (1)

Time-step size adaptivity is critical, since the system's dynamics and configuration can dramatically change over the course of time, possibly requiring quite different time step sizes to control the iterative error. However, to maintain the accuracy of the time-stepping scheme, one must respect an upper bound dictated by the discretization error, i.e., $\Delta t \leq \Delta t^{lim}$. Note that in step (5), Λ_K may enlarge the time-step if the error is lower than the preset tolerance. At a given time, once the process is complete, then the time is incremented forward and the process is repeated. The overall goal is to deliver solutions where the iterative error is controlled and the temporal discretization accuracy dictates the upper limit on the time step size (Δt^{lim}). Clearly, there are various combinations of solution methods that one can choose from. For example, for the overall field coupling, one may choose implicit or explicit staggering and within the staggering process, either implicit ($0 < \phi \leq 1$) or explicit time-stepping ($\phi = 0$), and, in the case of implicit time-stepping, iterative or direct solvers for the Navier–Stokes equations and the first law of thermodynamics. Furthermore, one could employ internal iterations for each field equation, then update more sophisticated metrics for certain components of the error.

Remark 4. The cost of constructing an array for the temporal update using a voxel calculation is primarily associated with summing up the voxel derivative terms needed to construct $\boldsymbol{L}$ and Z in Eqs. (5.2) and (5.5), multiplied by the number of internal iterations needed for convergence within a time step (since it is a temporally implicit method), times the number of time steps. There is no matrix inversion or linearization needed, simply vector-array updates for the voxel values. This is in contrast to the Finite Element Method, with costs associated with (1) conformally meshing the domain (mappings, etc.) (2) numerically integrating the weak form to generate the stiffness matrix and (3) solving a system of simultaneous equations (see Zohdi [59]). The operation counts of the voxel approach, relative to FEM, is dramatically less however, the FEM can potentially be more accurate with adaptive mesh refinement. However, the comparison is somewhat moot, since the voxels are the only data available in this model problem scenario.

7.1. Model problem and numerical example

As an example, we consider the direct numerical simulation of the fluid flow using the Navier–Stokes equations and first law of thermodynamics (streamlines shown) with two side vents. Fig. 2 illustrates a cross-section of the schematic for the system with the two types of domains: (a) A thermo-fluid domain (ambient domain) and (b) a thermo-only domain (unidentified object), where the velocity field is set to zero ($\boldsymbol{v} = \boldsymbol{0}$). Fig. 7 illustrates the results (evolution of flow streamlines). In the model problem, we have made the vent sizes 0.25 that of the wall. A $20 \times 20 \times 20$ stencil grid was used. A standard MacBook Pro was used for all calculations using a code written by the author. A simulation of this type takes a fraction of a second (the rest of the thermo-flow parameters were those immediately following Eq. (8.9)).

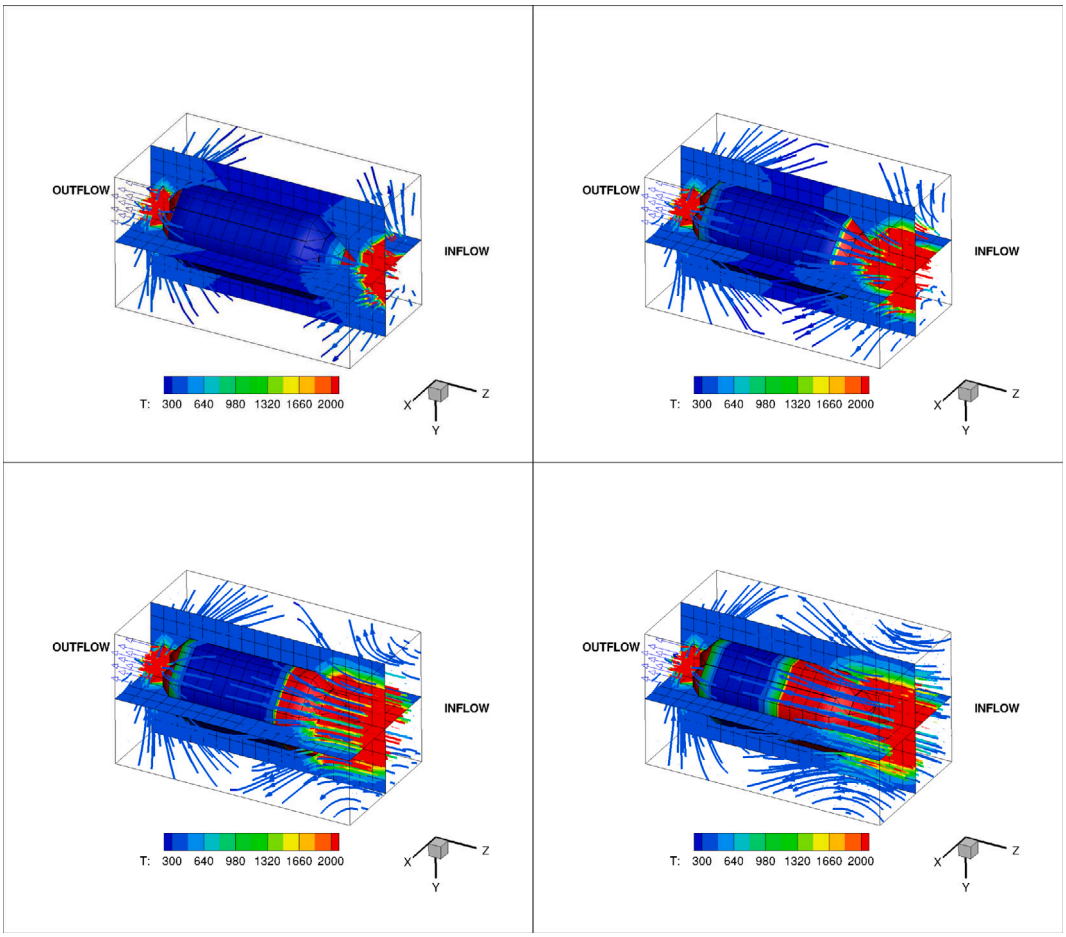

Fig. 7. Successive frames of flow using a direct numerical simulation of the fluid flow using the Navier–Stokes equations (streamlines shown) with two side vents. The evolution of flow streamlines are shown, as well as slices through the orthogonal planes.

8. Genomic machine-learning thermal signature optimization

The rapid rate at which these simulations can be completed enables the ability to explore inverse problems seeking to determine what parameter combinations can deliver a desired result (Fig. 9). In order to cast the objective mathematically, we set the problem up as a Machine Learning Algorithm (MLA); specifically a Genetic Algorithm (GA) variant, which is well-suited for nonconvex optimization. Following Zohdi [50–54], we formulate the objective as a cost function minimization problem that seeks system parameters that match the detected fields, characterized by the cost-error function in Π. *The weights can be adjusted according to what is deemed more important the (1) thermal, (2) flow-field and/or (3) volume measurement.* We systematically minimize $min_{\Lambda}\Pi$, by varying the design parameters: $\boldsymbol{\Lambda}^i \stackrel{\text{def}}{=} \{\Lambda_1^i, \Lambda_2^i, \Lambda_3^i, \ldots, \Lambda_N^i\}$. The system parameter search is conducted within the constrained ranges of $\Lambda_1^{(-)} \leq \Lambda_1 \leq \Lambda_1^{(+)}$, $\Lambda_2^{(-)} \leq \Lambda_2 \leq \Lambda_2^{(+)}$, $\Lambda_3^{(-)} \leq \Lambda_3 \leq \Lambda_3^{(+)}$, etc. These upper and lower limits are dictated by what is physically feasible.

8.1. Machine-Learning Algorithm (MLA)

Cost functions such as Π are nonconvex in design parameter space and often nonsmooth. Their minimization is usually difficult with direct application of gradient methods. This motivates nonderivative search methods, for example those found in Machine-Learning Algorithms (MLA's). One of the most basic subsets of MLAs are so-called Genetic Algorithms (GAs). For a review of GAs, see the pioneering work of John Holland ([60], [61]), as well as Goldberg [62], Davis [63], Onwubiko [64] and Goldberg and Deb [65]. A description of the algorithm will be described next, following Zohdi [50–54].

8.2. Algorithmic structure

The MLA/GA approach is extremely well-suited for nonconvex, nonsmooth, multicomponent, multistage systems and, broadly speaking, involves the following essential concepts (Fig. 9):

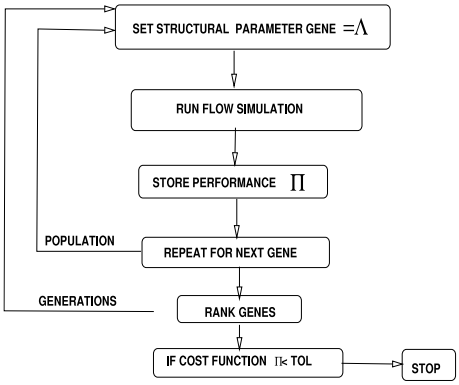

Fig. 8. The computational flow of the MLA/GA-Machine Learning Algorithm/Genetic Algorithm (Zohdi [50–54]).

1. **POPULATION GENERATION:** Generate a parameter population of genetic strings: $\boldsymbol{\Lambda}^i$
2. **PERFORMANCE EVALUATION:** Compute performance of each genetic string: $\Pi(\boldsymbol{\Lambda}^i)$
3. **RANK STRINGS:** Rank them $\boldsymbol{\Lambda}^i$, $i = 1, \ldots, S$ from best to worst
4. **MATING PROCESS:** Mate pairs/produce offspring
5. **GENE ELIMINATION:** Eliminate poorly performing genetic strings
6. **POPULATION REGENERATION:** Repeat process with updated gene pool and new *random* genetic strings
7. **SOLUTION POST-PROCESSING:** Employ gradient-based methods afterwards in local "valleys"-*if smooth enough*

8.3. Specifics

Following Zohdi [50–54], the algorithm is as follows:

- **STEP 1:** Randomly generate a population of S starting genetic strings, $\boldsymbol{\Lambda}^i, (i = 1, 2, 3, \ldots, S)$:

$$\boldsymbol{\Lambda}^i \stackrel{\text{def}}{=} \left\{ \begin{array}{c} \Lambda_1^i \\ \Lambda_2^i \\ \Lambda_3^i \\ \ldots \\ \Lambda_N^i \end{array} \right\} \tag{8.1}$$

- **STEP 2:** Compute fitness of each string $\Pi(\boldsymbol{\Lambda}^i)$, (i=1, …, S)
- **STEP 3:** Rank genetic strings: $\boldsymbol{\Lambda}^i$, (i=1, …, S) from best to worst
- **STEP 4:** Mate nearest pairs and produce two offspring, (i=1, …, S):

$$\lambda^i \stackrel{\text{def}}{=} \boldsymbol{\Phi} \circ \boldsymbol{\Lambda}^i + (\mathbf{1} - \boldsymbol{\Phi}) \circ \boldsymbol{\Lambda}^{i+1} \stackrel{\text{def}}{=} \left\{ \begin{array}{c} \phi_1 \Lambda_1^i \\ \phi_2 \Lambda_2^i \\ \phi_3 \Lambda_3^i \\ \ldots \\ \phi_N \Lambda_N^i \end{array} \right\} + \left\{ \begin{array}{c} (1-\phi_1)\Lambda_1^{i+1} \\ (1-\phi_2)\Lambda_2^{i+1} \\ (1-\phi_3)\Lambda_3^{i+1} \\ \ldots \\ (1-\phi_N)\Lambda_N^{i+1} \end{array} \right\} \tag{8.2}$$

and

$$\lambda^{i+1} \stackrel{\text{def}}{=} \boldsymbol{\Psi} \circ \boldsymbol{\Lambda}^i + (\mathbf{1} - \boldsymbol{\Psi}) \circ \boldsymbol{\Lambda}^{i+1} \stackrel{\text{def}}{=} \left\{ \begin{array}{c} \psi_1 \Lambda_1^i \\ \psi_2 \Lambda_2^i \\ \psi_3 \Lambda_3^i \\ \ldots \\ \psi_N \Lambda_N^i \end{array} \right\} + \left\{ \begin{array}{c} (1-\psi_1)\Lambda_1^{i+1} \\ (1-\psi_2)\Lambda_2^{i+1} \\ (1-\psi_3)\Lambda_3^{i+1} \\ \ldots \\ (1-\psi_N)\Lambda_N^{i+1} \end{array} \right\} \tag{8.3}$$

where for this operation, the ϕ_i and ψ_i are random numbers, such that $0 \leq \phi_i \leq 1$, $0 \leq \psi_i \leq 1$, which are different for each component of each genetic string

- **STEP 5:** Eliminate the bottom M strings and keep top K parents and their K offspring (K offspring+K parents+$M = S$)
- **STEP 6:** Repeat STEPS 1–5 with top gene pool (K offspring and K parents), plus M new, randomly generated, strings

- **OPTION:** One can rescale and restart search around best performing parameter set every few generations, thus refocussing the computation effort around the most promising (optimal) areas of design space.

Remark 5. If one selects the mating parameters $\phi's$ and $\psi's$ to be greater than one and/or less than zero, one can induce "mutations", i.e. characteristics that neither parent possesses. However, this is somewhat redundant with introduction of new random members of the population in the current algorithm. If one does not retain the parents in the algorithm above, it is possible that inferior performing offspring may replace superior parents. Thus, top parents should be kept for the next generation. This guarantees a monotone reduction in the cost function. Furthermore, retained parents do not need to be reevaluated, making the algorithm less computationally expensive, since these parameter sets do not have to be reevaluated (or ranked) in the next generation. Numerous studies of the author (Zohdi [50–54]) have shown that the advantages of parent retention outweighs inbreeding, for sufficiently large population sizes. Finally, we observe that this algorithm is easy to parallelize. After application of such a global search algorithm, one can apply a gradient-based method, if the objective function is sufficiently smooth in that region of the parameter space. In other words, if one has located a convex portion of the parameter space with a global genetic search, one can employ gradient-based procedures locally to minimize the objective function further, since they are generally much more efficient for convex optimization of smooth functions. An exhaustive review of these methods can be found in the texts of Luenberger [66] and Gill, Murray and Wright [67].

8.4. Algorithmic settings

In the upcoming example, the design parameters $\boldsymbol{\Lambda} = \{\Lambda_1, \Lambda_2 ... \Lambda_N\}$ are optimized over the search intervals (12 variables): $\boldsymbol{\Lambda}_i^- \leq \boldsymbol{\Lambda}_i \leq \boldsymbol{\Lambda}_i^+$, $i = 1, 2, \ldots, 12$. Specifically, we varied the 12 parameters associated with the body and used the following MLA settings[2]:

- Number of design variables: 12,
- Population size per generation: 24,
- Number of parents to keep in each generation: 6,
- Number of children created in each generation: 6,
- Number of completely new genes created in each generation: 12,
- Number of generations for re-adaptation around a new search interval: 20 and
- Number of generations: 100.

8.5. Parameter search ranges and results

In order to drive the process of searching for object shapes that could possibly produce the observed thermo-flow field, one needs some sort of general shape representation that can be morphed and combined with other shapes, in order to make complex objects. For illustration purposes, we will use generalized hyper-ellipsoids. We considered a 12 parameter system design representing (a) 3 nosecone length parameters (L_1, L_2, L_3) and 3 shape parameters (q_1, q_2, q_3)

$$\left(\frac{\|x_1 - x_{10}\|}{L_1}\right)^{q_1} + \left(\frac{\|x_2 - x_{20}\|}{L_2}\right)^{q_2} + \left(\frac{\|x_3 - x_{30}\|}{L_3}\right)^{q_3} = 1. \tag{8.4}$$

and (b) 3 fuselage length parameters (R_1, R_2, R_3) and 3 fuselage shape parameters (p_1, p_2, p_3)

$$\left(\frac{\|x_1 - x_{10}\|}{R_1}\right)^{p_1} + \left(\frac{\|x_2 - x_{20}\|}{R_2}\right)^{p_2} + \left(\frac{\|x_3 - x_{30}\|}{R_3}\right)^{p_3} = 1. \tag{8.5}$$

The design vector is

$$\boldsymbol{\Lambda} = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_N\} = \{L_1, L_2, L_3, q_1, q_2, q_3, R_1, R_2, R_3, p_1, p_2, p_3\}. \tag{8.6}$$

We note that we can select a subset of parameters to generate simple shapes, such as if we wanted a smooth geometrical transition from the nosecone and the fuselage, then we enforce

$$L_1 = R_1,\ L_2 = R_2,\ q_1 = p_1 \text{ and } q_2 = p_2, \tag{8.7}$$

or using only one shape and

$$L_1 = L_2 = L_3 \text{ and } q_1 = q_2 = q_3 = 2, \tag{8.8}$$

to generate a sphere.

The domain size was $(-1/2 \leq x_1 \leq 1/2, -1/2 \leq x_2 \leq 1/2, -1/2 \leq x_3 \leq 1/2)$ The following search parameter ranges were used (with $w_1 = w_2 = w_3 = 1$):

[2] As in the previous example, a $20 \times 20 \times 20$ stencil grid was used along with a standard MacBook Pro laptop for all calculations using a voxel code written by the author.

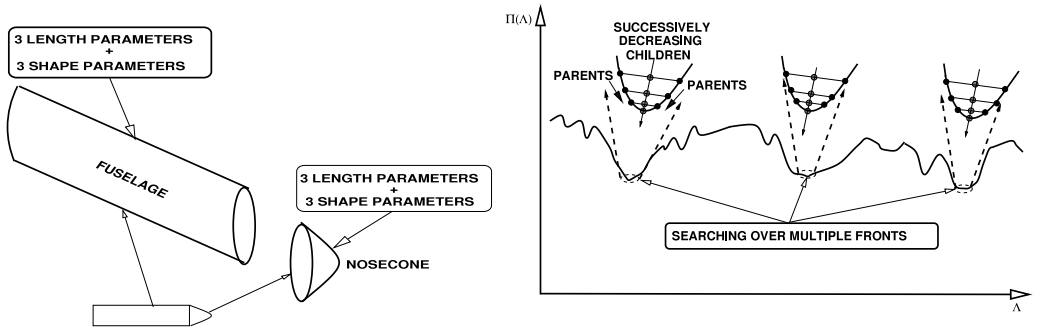

Fig. 9. LEFT: The structural parameters of the system. 12 in total: (a) 3 nosecone shape parameters and 3 length parameters and (a) 3 fuselage shape parameters and 3 length parameters. We note that if we want a smooth geometrical transition from the nosecone and the fuselage, then we enforce $L_1 = R_1$, $L_2 = R_2$, $q_1 = p_1$ and $q_2 = p_2$. RIGHT: The basic action of a MLA/GA-Machine Learning Algorithm/Genetic Algorithm, which searches over multiple locations in parameter space simultaneously (Zohdi [50–54]).

- $\Lambda_{i=1-3}$ = Nosecone length ratios (relative to the box size $\pm 1/2$): $\Lambda_i^- = 0.2 \leq \Lambda_i \leq \Lambda_i^+ = 1$, $i = 1, 2, 3$
- $\Lambda_{i=4}$ = Nosecone shape: $\Lambda_4^- = 1 \leq \Lambda_4 \leq \Lambda_4^+ = 20$,
- $\Lambda_{i=5}$ = Nosecone shape: $\Lambda_5^- = 1 \leq \Lambda_5 \leq \Lambda_5^+ = 20$,
- $\Lambda_{i=6}$ = Nosecone shape: $\Lambda_6^- = 1 \leq \Lambda_6 \leq \Lambda_6^+ = 20$,
- $\Lambda_{i=7-9}$ = Fuselage length ratios (relative to the box size $\pm 1/2$): $\Lambda_i^- = 0.2 \leq \Lambda_i \leq \Lambda_i^+ = 1$,
- $\Lambda_{i=10}$ = Fuselage shape: $\Lambda_i^- = 1 \leq \Lambda_i \leq \Lambda_i^+ = 20$,
- $\Lambda_{i=11}$ = Fuselage shape: $\Lambda_i^- = 1 \leq \Lambda_i \leq \Lambda_i^+ = 20$,
- $\Lambda_{i=12}$ = Fuselage shape: $\Lambda_i^- = 1 \leq \Lambda_i \leq \Lambda_i^+ = 20$,

As an example, we consider a combination of two objects. This problem is quite nonconvex, due to the many possibilities of combined objects that could produce similar thermo-flow fields. We considered a 12 parameter system design, however, imposing that the nosecone shape parameters equal one another, and only taking half of the shape, with a test set "detected" object generated by a random set between the search interval:

$$\boldsymbol{\Lambda}^{det} = \{0.412, 0.412, 0.528, 12.295, 12.295, 9.8960.594, 0.594, 0.450, 8.732, 8.732, 14.908\}. \tag{8.9}$$

We used 50 frames for the cost function and the following fluid–solid parameters:

- Initial temperature of 300°,
- Fluid viscosity of $\mu = 0.1$ Pa/s,
- Fluid density $\rho_f = 1.228$ kg/m^3,
- Solid density $\rho_f = 5000$ kg/m^3,
- Fluid heat capacity = 700 J/K,
- Solid heat capacity = 100 J/K,
- Fluid thermal conductivity = 0.03 W/m K,
- Solid thermal conductivity = 10 W/m K,
- Inlet velocity = outlet velocity (parabolic profile) = 5 m/s,
- Optimization subweights: $w_\theta = w_{\boldsymbol{v}} = w_s = 1$.

For illustration purposes, heating term (ρz) in the first law of thermodynamics was made proportional to the stress-power ($\boldsymbol{\sigma} : \nabla_x \boldsymbol{v}$,)

$$\boldsymbol{\sigma} : \nabla_x \boldsymbol{v} + \rho z = (1 + H)\boldsymbol{\sigma} : \nabla_x \boldsymbol{v}, \tag{8.10}$$

where $H = 10 \times 10^6$. Fig. 10 illustrates the results for successive generations, allowing the MLA/GA to readapt every 20 generations. Often, this action is more efficient than allowing the algorithm not to readapt, since it probes around the current optimum for better local alternatives. The best after generation 100 was $\Pi^{100} = 2.455 \times 10^{-9}$, which started (generation 1) at $\Pi^{1,ave}$=0.695, yielding a reduction of $\frac{\Pi^{1,ave} - \Pi^{100}}{\Pi^{1,ave}} = \frac{0.699 - 2.455 \times 10^{-9}}{0.695} = 99.999\%$. The optimal design vector was:

$$\boldsymbol{\Lambda}^{det} = \{0.430, 0.430, 0.530, 10.085, 10.085, 10.048, 0.593, 0.593, 0.453, 7.461, 7.461, 15.162, \} \tag{8.11}$$

which replicates the geometry (to within a voxel-tolerance size) that captures the flow-fields and thermal states quite accurately. Specifically, the individual and total cost functions were:

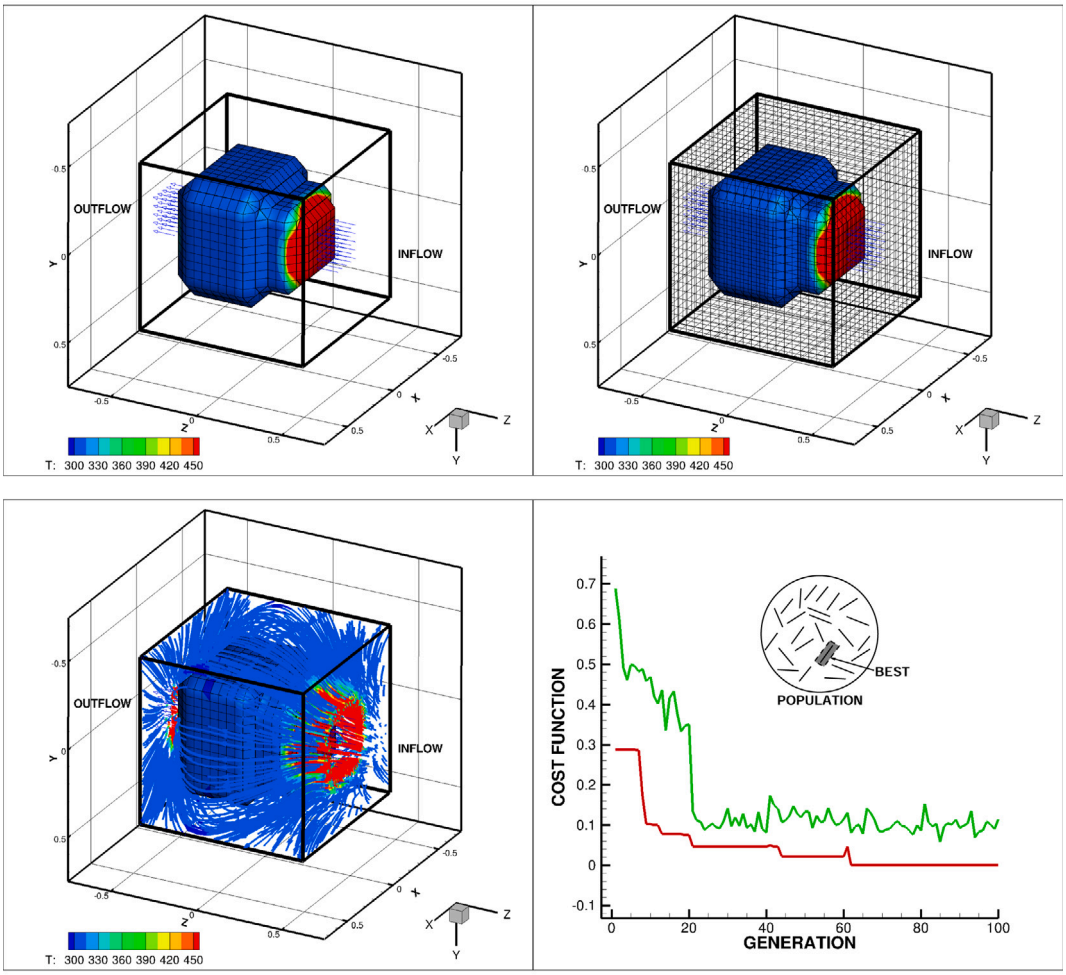

Fig. 10. Shown are the object to be detected, the voxel grid, the flow around the object (colors indicating the temperature) and the cost function for the best performing gene (*red*) as a function of successive generations, as well as the average cost function of the entire population of genes (*green*). We allowed the MLA/GA to readapt every 20 generations. The best after generation 100 was $\Pi^{100} = 2.455 \times 10^{-9}$, which started (generation 1) at $\Pi^{1,ave}$=0.695, yielding a reduction of $\frac{\Pi^{1,ave} - \Pi^{1,100}}{\Pi^{ave}} = \frac{0.699 - 2.455 \times 10^{-9}}{0.695} = 99.999\%$.

- $\Pi^{(\theta)} = 6.975 \times 10^{-11}$,
- $\Pi^{(v)} = 7.297 \times 10^{-9}$,
- $\Pi^{voxel} = 0.0$ (exact, to machine precision),
- $\Pi(\Lambda_1, \ldots \Lambda_N) = \frac{\pi(\Lambda_1, \ldots \Lambda_N)}{w_\theta + w_v + w_s} = 2.455 \times 10^{-9}$.

The entire 100 generation simulation, with 24 genes per evaluation (2400 total designs) took a few minutes on a laptop, *making it ideal as a design tool.* We note that, for a given set of parameters, a complete simulation takes less than one second, thus thousands of parameter sets can be evaluated in an hour, *without even exploiting the inherent parallelism of the MLA/GA*. The speed at which the overall process can be completed makes it a suitable digital-twin of the system that can run in real-time or faster than the actual physical system.

9. Summary and extensions

In summary, this work developed a combined voxel-based machine-learning framework for the rapid identification of unknown objects by their thermo-fluid flow field signature. Specifically, a machine-learning framework was developed that rapidly simulates and adapts object geometries in order to match the thermo-flow field signature generated by an unknown object, across a time series of voxel-frames. In order to achieve this, a thermo-fluid model is developed, based on the Navier–Stokes equations and the first law of thermodynamics, using a voxel rendering of the system, which was efficiently solved with a voxel-tailored, temporally-adaptive,

iterative solution scheme. This voxel-framework was then combined with a genomic-based machine-learning algorithm to develop a digital-twin of the system that can run in real-time or faster than the actual physical system.

As discussed earlier in the work, in the last 20 years, there has been a dramatic evolution incorporating (a) multispectral cameras, based on technologies that extend the classical visible wavelength paradigms (380–720 nm), to thermographic/infrared regimes (1000–14000 nm) to create an image and (b) 3D (time-of-flight) cameras, using LIDAR, radio-based imaging and tomography. They have fundamentally changed the ability to extract information from complex events that would have been unthinkable until recently. In particular, the ability to extract thermo-fluid flow data in three dimensions, utilizing real-time tomographically-based imaging, has now brought forth many possibilities. In particular, tomography, which uses multidirectional penetrating waves and the splicing of sections with tomographic reconstruction, has become a critical component for 3D data extraction. Several commercial products now exist that enable the instantaneous measurement within a 3D volume of all three velocity components, in addition to thermal fields. This immediately allows for voxelization of a 3D space, which yields an image comprised of voxels, each containing velocity and thermal field data. In order to solve PDE's posed over such domains, extremely fast methods can then be used to construct the various derivatives needed in a differential operator, circumventing meshing, mapping, volume integration and stiffness matrix generation (needed for example in Finite Element Methods), as well as matrix-based solution methods, since the voxel structure allows for incredibly efficient matrix-free iterative solvers. In short, voxel-based camera data is ideally-suited to voxel-based computation. Ultimately, for ultrafast split-second decision making, for example to react to the object, the use of such a paradigms may need the inclusion of simplified reduced-order models that can be trained on the data generated by more complex models, such as the one introduced in the body of this work, such as Artificial Neural Networks (ANN). ANN have received huge attention in the scientific community over the last decade and are based on layered input–output type frameworks that are essentially adaptive nonlinear regressions of the form $\mathcal{O} = \mathcal{B}(\boldsymbol{I}, \boldsymbol{w})$, where $\mathcal{O}$ is a desired output and $\mathcal{B}$ is the ANN comprised of (1) **synapses**, which multiply inputs (I_i, $i = 1, 2, \dots, M$) by weights (w_i, $i = 1, 2, \dots, N$) that represent the input relevance to the desired output, (2) **neurons**, which aggregate outputs from all incoming synapses and apply activation functions to process the data and (3) **training**, which calibrates the weights to match a desired overall output. Blending of these various paradigms (complex models, simplified reduced-order models and neural nets) is the subject of current work of the author (Zohdi [68]), using genetic-based methods for auto-calibration of the ANN weights.

Declaration of competing interest

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

Data availability

No data was used for the research described in the article.

Acknowledgments

This work has been partially supported by the UC Berkeley College of Engineering, USA and Sandia National Labs, USA.

Appendix

A.1. Fluid flow model

For a hydrostatic fluid the stress can be written as

$$\boldsymbol{\sigma} = -P_o \mathbf{1}, \tag{A.1}$$

where $P_o = \frac{tr\boldsymbol{\sigma}}{3}$ is the hydrostatic pressure. In other words, there are no shear stresses in a fluid at rest. In the dynamic case, the pressure, denoted the "thermodynamic pressure", is related to the temperature and the fluid density by an equation of state

$$\mathcal{Z}(P, \rho, \theta) = 0. \tag{A.2}$$

For a fluid in motion

$$\boldsymbol{\sigma} = -P\mathbf{1} + \boldsymbol{\tau}^{vs} \tag{A.3}$$

where $\boldsymbol{\tau}^{vs}$ is a so-called viscous stress tensor, needed in a balance of linear momentum[3]:

$$\nabla_x \cdot \boldsymbol{\sigma} + \boldsymbol{f} = \rho \frac{d\boldsymbol{v}}{dt}, \tag{A.4}$$

[3] An inviscid or "perfect" fluid is one where $\boldsymbol{\tau}^{vs}$ is taken to be zero, even when motion is present.

where $\boldsymbol{v}$ is the fluid velocity at point $\boldsymbol{x}$ and $\boldsymbol{f}$ are the body forces. Thus, for a compressible fluid in motion:

$$\frac{tr\boldsymbol{\sigma}}{3} = -P + \frac{tr\boldsymbol{\tau}^{vs}}{3}. \tag{A.5}$$

In general, for a fluid we have

$$\boldsymbol{\tau}^{vs} = \mathcal{G}(\boldsymbol{D}) \qquad \text{and} \qquad \boldsymbol{D} \overset{\text{def}}{=} \frac{1}{2}(\nabla_x \boldsymbol{v} + (\nabla_x \boldsymbol{v})^T), \tag{A.6}$$

where $\boldsymbol{v}$ is the velocity and $\boldsymbol{D}$ is the symmetric part of the velocity gradient. For a Newtonian fluid, where a linear relation exists between the viscous stresses $\boldsymbol{\tau}^{vs}$ and $\boldsymbol{D}$

$$\boldsymbol{\tau}^{vs} = \mathcal{G}(\boldsymbol{D}) = \boldsymbol{C} : \boldsymbol{D} \tag{A.7}$$

where $\boldsymbol{C}$ is a symmetric positive definite (fourth-order) viscosity tensor. For an isotropic (standard) Newtonian fluid we have

$$\boldsymbol{\sigma} = -P\mathbf{1} + \lambda tr\boldsymbol{D}\mathbf{1} + 2\mu\boldsymbol{D} = -P\mathbf{1} + 3\kappa\frac{tr\boldsymbol{D}}{3}\mathbf{1} + 2\mu\boldsymbol{D}', \tag{A.8}$$

where κ is called the bulk viscosity, λ is a viscosity constant, μ the shear viscosity and $\boldsymbol{D}' = \boldsymbol{D} - \frac{tr\boldsymbol{D}}{3}\mathbf{1}$. Explicitly, with an (x_1, x_2, x_3) Cartesian triad

$$\underbrace{\begin{Bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{12} \\ \sigma_{23} \\ \sigma_{31} \end{Bmatrix}}_{\overset{\text{def}}{=}\{\boldsymbol{\sigma}\}} = \underbrace{\begin{Bmatrix} -P \\ -P \\ -P \\ 0 \\ 0 \\ 0 \end{Bmatrix}}_{\overset{\text{def}}{=}\{-\boldsymbol{P}\}} + \underbrace{\begin{bmatrix} c_1 & c_2 & c_2 & 0 & 0 & 0 \\ c_2 & c_1 & c_2 & 0 & 0 & 0 \\ c_2 & c_2 & c_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix}}_{\overset{\text{def}}{=}[\boldsymbol{C}]} \underbrace{\begin{Bmatrix} D_{11} \\ D_{22} \\ D_{33} \\ 2D_{12} \\ 2D_{23} \\ 2D_{31} \end{Bmatrix}}_{\overset{\text{def}}{=}\{\boldsymbol{D}\}}, \tag{A.9}$$

where $c_1 = \kappa + \frac{4}{3}\mu$ and $c_2 = \kappa - \frac{2}{3}\mu$, where $D_{ij} = \frac{1}{2}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right)$. The so-called "Stokes' condition" attempts to force the thermodynamic pressure to collapse to the classical definition of mechanical pressure, i.e.

$$\frac{tr\boldsymbol{\sigma}}{3} = -P + 3\kappa\frac{tr\boldsymbol{D}}{3} = -P, \tag{A.10}$$

leading to the conclusion that $\kappa = 0$ or $\lambda = -\frac{2}{3}\mu$. Thus, a Newtonian fluid obeying the Stokes' condition has the following constitutive law:

$$\boldsymbol{\sigma} = -P\mathbf{1} - \frac{2}{3}\mu tr\boldsymbol{D}\mathbf{1} + 2\mu\boldsymbol{D} = -P\mathbf{1} + 2\mu\boldsymbol{D}'. \tag{A.11}$$

Note that

$$\dot{J} = \frac{d}{dt}det\boldsymbol{F} = (det\boldsymbol{F})tr(\dot{\boldsymbol{F}} \cdot \boldsymbol{F}^{-1}) = J tr\boldsymbol{L} = J\nabla_x \cdot \boldsymbol{v}, \tag{A.12}$$

where $\boldsymbol{L} = \nabla_x \boldsymbol{v}$ is the velocity gradient. Note that $\nabla_x \cdot \boldsymbol{v} = tr\boldsymbol{L} = tr\boldsymbol{D}$. Therefore, if the fluid is incompressible, $\dot{J} = 0$, then $\nabla_x \cdot \boldsymbol{v} = tr\boldsymbol{L} = tr\boldsymbol{D} = 0$. Therefore,

$$\boldsymbol{\sigma} = -P\mathbf{1} + 2\mu\boldsymbol{D}. \tag{A.13}$$

A conservation of mass dictates

$$\frac{d}{dt}(\rho_o) = \frac{d}{dt}(\rho J) = J\frac{d\rho}{dt} + \rho\frac{dJ}{dt} = 0, \tag{A.14}$$

which leads to

$$\frac{d\rho}{dt} + \frac{\rho}{J}\frac{dJ}{dt} = 0. \tag{A.15}$$

Using Eq. (A.12), Eq. (A.14) becomes

$$\frac{d\rho}{dt} + \rho\nabla_x \cdot \boldsymbol{v} = 0. \tag{A.16}$$

Now write the total temporal ("material") derivative in convective form:

$$\frac{d\rho}{dt} = \frac{\partial\rho}{\partial t} + (\nabla_x \rho) \cdot \frac{d\boldsymbol{x}}{dt} = \frac{\partial\rho}{\partial t} + \nabla_x \rho \cdot \boldsymbol{v}. \tag{A.17}$$

Thus, Eq. (A.16) becomes

$$\frac{\partial\rho}{\partial t} + \nabla_x \rho \cdot \boldsymbol{v} + \rho\nabla_x \cdot \boldsymbol{v} = \frac{\partial\rho}{\partial t} + \nabla_x \cdot (\rho\boldsymbol{v}) = 0. \tag{A.18}$$

Thus, writing the total time derivatives appearing previously as

$$\frac{d\boldsymbol{v}}{dt} = \frac{\partial \boldsymbol{v}}{\partial t}|_{\boldsymbol{x}} + (\nabla_x \boldsymbol{v})|_t \cdot \frac{d\boldsymbol{x}}{dt}, \tag{A.19}$$

the coupled governing equations are (momentarily ignoring thermal effects)

$$\boxed{\begin{aligned} &\frac{\partial \rho}{\partial t} = -\nabla_x \rho \cdot \boldsymbol{v} - \rho \nabla_x \cdot \boldsymbol{v}, \\ &\rho(\frac{\partial \boldsymbol{v}}{\partial t} + (\nabla_x \boldsymbol{v}) \cdot \boldsymbol{v}) = \nabla_x \cdot \boldsymbol{\sigma} + \boldsymbol{f}, \\ &\boldsymbol{\sigma} = -P\mathbf{1} + \lambda tr\boldsymbol{D}\mathbf{1} + 2\mu \boldsymbol{D} = -P\mathbf{1} + 3\kappa \frac{tr\boldsymbol{D}}{3}\mathbf{1} + 2\mu \boldsymbol{D}', \end{aligned}} \tag{A.20}$$

where, for example, P is given by an Equation of State. Collectively, we refer to these equations as the 'Navier–Stokes' equations. There are a total of three variables: $\rho, \boldsymbol{v}$, and P. It is customary to specify $\boldsymbol{v}$ and P on the boundary, and to determine ρ on the boundary through the Equation of State.

Remark 6. In the example provided, we shall consider incompressible fluids. However, for completeness, we briefly illustrate a simple equation of state for fluid compressibility. There are a variety of possible Equations of State that connect the density to the pressure, such as a Boussinesq-like relation, which is adequate to describe dense gases and fluids, derived from[4]

$$\rho \approx \rho_o(P_o) + \frac{\partial \rho}{\partial P} \Delta P, \tag{A.21}$$

where ρ_o and P_o are reference values and $\Delta P = P - P_o$. We define the bulk (compressibility) modulus by $\zeta \stackrel{\text{def}}{=} \rho \frac{\partial P}{\partial \rho}$, yielding

$$\rho \approx \rho_o \left(1 + \frac{1}{\zeta} \Delta P\right) \Rightarrow P \approx P_o + \zeta \left(\frac{\rho}{\rho_o} - 1\right). \tag{A.22}$$

For a constant density case, $\rho = \rho_o$, the Boussinesq-like relation asserts, $P = P_o$.

A.2. Thermophysics model

The interconversions of mechanical, thermal and chemical energy in a system are governed by the first law of thermodynamics. It states that the time rate of change of the total energy, $\mathcal{K} + \mathcal{I}$, is equal to the work rate, $\mathcal{P}$ and the net heat supplied, $\mathcal{H} + \mathcal{Q}$,

$$\frac{d}{dt}(\mathcal{K} + \mathcal{I}) = \mathcal{P} + \mathcal{H} + \mathcal{Q}. \tag{A.23}$$

Here the kinetic energy of a subvolume of material contained in Ω, denoted ω, is $\mathcal{K} \stackrel{\text{def}}{=} \int_\omega \frac{1}{2} \rho \boldsymbol{v} \cdot \boldsymbol{v}\, d\omega$, the rate of work or power of external forces acting on ω is given by $\mathcal{P} \stackrel{\text{def}}{=} \int_\omega \rho \boldsymbol{b} \cdot \boldsymbol{v}\, d\omega + \int_{\partial\omega} \boldsymbol{\sigma} \cdot \boldsymbol{n} \cdot \boldsymbol{v}\, da$, $\boldsymbol{b}$ being the body forces, the heat flow into the volume by conduction is $\mathcal{Q} \stackrel{\text{def}}{=} -\int_{\partial\omega} \boldsymbol{q} \cdot \boldsymbol{n}\, da = -\int_\omega \nabla_x \cdot \boldsymbol{q}\, d\omega$, $\boldsymbol{q}$ being the heat flux, the heat generated due to sources, *such as chemical reactions*, is $\mathcal{H} \stackrel{\text{def}}{=} \int_\omega \rho z\, d\omega$, z are sources, and the stored energy is $\mathcal{I} \stackrel{\text{def}}{=} \int_\omega \rho w\, d\omega$, w being the stored energy. If we make the assumption that the mass in the system is constant, one has,

$$\text{current mass} = \int_\omega \rho\, d\omega = \int_{\omega_0} \rho J\, d\omega_0 \approx \int_{\omega_0} \rho_0\, d\omega_0 = \text{original mass}, \tag{A.24}$$

which implies $\rho J = \rho_0 \Rightarrow \dot{\rho} J + \rho \dot{J} = 0$. Using this and the energy balance leads to

$$\begin{aligned} \frac{d}{dt} \int_\omega \frac{1}{2} \rho \boldsymbol{v} \cdot \boldsymbol{v}\, d\omega &= \int_{\omega_0} \frac{d}{dt} \frac{1}{2} (\rho J \boldsymbol{v} \cdot \boldsymbol{v})\, d\omega_0 \\ &= \int_{\omega_0} (\frac{d}{dt} \rho_0) \frac{1}{2} \boldsymbol{v} \cdot \boldsymbol{v}\, d\omega_0 + \int_\omega \rho \frac{d}{dt} \frac{1}{2} (\boldsymbol{v} \cdot \boldsymbol{v})\, d\omega \\ &= \int_\omega \rho \boldsymbol{v} \cdot \dot{\boldsymbol{v}}\, d\omega. \end{aligned} \tag{A.25}$$

We also have

$$\frac{d}{dt} \int_\omega \rho w\, d\omega = \frac{d}{dt} \int_{\omega_0} \rho J w\, d\omega_0 = \int_{\omega_0} \frac{d}{dt}(\rho_0) w\, d\omega_0 + \int_\omega \rho \dot{w}\, d\omega. \tag{A.26}$$

By using the divergence theorem, we obtain

$$\int_{\partial\omega} \boldsymbol{\sigma} \cdot \boldsymbol{n} \cdot \boldsymbol{v}\, da = \int_\omega \nabla_x \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{v})\, d\omega = \int_\omega (\nabla_x \cdot \boldsymbol{\sigma}) \cdot \boldsymbol{v}\, d\omega + \int_\omega \boldsymbol{\sigma} : \nabla_x \boldsymbol{v}\, d\omega. \tag{A.27}$$

[4] We have ignored thermal effects in this representation.

Combining the results, and enforcing balance of momentum, leads to

$$\int_{\omega} \left(\rho \dot{w} + \boldsymbol{v} \cdot (\rho \dot{\boldsymbol{v}} - \nabla_x \cdot \boldsymbol{\sigma} - \rho \boldsymbol{b}) - \boldsymbol{\sigma} : \nabla_x \boldsymbol{v} + \nabla_x \cdot \boldsymbol{q} - \rho z\right) \, d\omega = \int_{\omega} \left(\rho \dot{w} - \boldsymbol{\sigma} : \nabla_x \boldsymbol{v} + \nabla_x \cdot \boldsymbol{q} - \rho z\right) \, d\omega = 0. \tag{A.28}$$

Since the volume ω is arbitrary, the integrand must hold locally and we have

$$\rho \dot{w} - \boldsymbol{\sigma} : \nabla_x \boldsymbol{v} + \nabla_x \cdot \boldsymbol{q} - \rho z = 0. \tag{A.29}$$

A typical approximation in fluid mechanics is $w \approx \rho C \theta$, where C is the heat capacity and θ is the temperature in Kelvin. As in the Navier–Stokes equations, breaking the thermal rate term into a fixed part and a convective part yields

$$\rho \dot{w} = \rho C (\frac{\partial \theta}{\partial t} + \nabla_x \theta \cdot \boldsymbol{v}) = \boldsymbol{\sigma} : \nabla_x \boldsymbol{v} - \nabla_x \cdot \boldsymbol{q} + \rho z. \tag{A.30}$$

Remark 7. For the remainder of the work, we will assume that the fluid is incompressible, homogeneous and that its properties are thermally-insensitive.

References

[1] G.E. Elsinga, F. Scarano, B. Wieneke, B.W. van Oudheusden, Tomographic particle image velocimetry, Exp. Fluids 41 (15) (2006) 933–947.
[2] G.T. Herman, A. Lent, Iterative reconstruction algorithms, Comput. Biol. Med. 6 (1976) 273–294.
[3] G.T. Herman, Fundamentals of Computerized Tomography: Image Reconstruction from Projections, second ed., Springer, Dordrecht, ISBN: 978-1-84628-723-7, 2009.
[4] A. Schroder, R. Geisler, G.E. Elsinga, F. Scarano, U. Dierksheide, Investigation of a turbulent spot and a tripped turbulent boundary layer flow using time-resolved tomographic PIV, Exp. Fluids 44 (2008) 305–316.
[5] B. Wieneke, Volume self-calibration for 3D particle image velocimetry, Exp. Fluids 45 (2008) 549–556.
[6] A.A. Aguirre-Pablo, M.K. Alarfaj, E.Q. Li, J.F. Hernandez-Sanchez, S.T. Thoroddsen, Tomographic particle image velocimetry using smartphones and colored shadows, Sci. Rep. 7 (2017) 3714, http://dx.doi.org/10.1038/s41598-017-03722-9.
[7] C. Atkinson, J. Soria, An efficient simultaneous reconstruction technique for tomographic particle image velocimetry, Exp. Fluids 47 (2009) 553–568.
[8] S. Discetti, A. Ianiro, T. Astarita, G. Cardone, On a novel low cost high accuracy experimental setup for tomographic particle image velocimetry, Meas. Sci. Technol. 24 (2013) 075302.
[9] P. Geoghegan, N. Buchmann, J. Soria, M. Jermy, Time-resolved PIV measurements of the flow field in a stenosed, compliant arterial model, Exp. Fluids 54 (2013) 1–19.
[10] C. Willert, B. Stasicki, J. Klinner, S. Moessner, Pulsed operation of high-power light emitting diodes for imaging flow velocimetry, Meas. Sci. Technol. 21 (2010) 075402.
[11] N.A. Buchmann, C.E. Willert, J. Soria, Pulsed, high-power LED illumination for tomographic particle image velocimetry, Exp. Fluids 53 (2012) 1545–1560.
[12] T.A. Casey, J. Sakakibara, S.T. Thoroddsen, Scanning tomographic particle image velocimetry applied to a turbulent jet, Phys. Fluids 25 (2013) 025102.
[13] W.-H. Tien, D. Dabiri, J.R. Hove, Color-coded three-dimensional micro particle tracking velocimetry and application to micro backward-facing step flows, Exp. Fluids 55 (2014) 1684.
[14] J. Xiong, R. Idoughi, A.A. Aquirre-Pablo, A.B. Alijedaani, X. Dun, Q. Fu, S.T. Thoroddsen, W. Hedrich, Rainbow particle imaging velocimetry for dense 3D fluid velocity imaging, ACM Trans. Graph. 36 (4) (2017) Article 36.
[15] T. Watamura, Y. Tasaka, Y. Murai, LCD-projector-based 3D color PTV, Exp. Therm Fluid Sci. 47 (2013) 68–80.
[16] J. Klinner, C. Willert, Tomographic shadowgraphy for three-dimensional reconstruction of instantaneous spray distributions, Exp. Fluids 53 (2012) 531–543.
[17] M.J. McPhail, A.A. Fontaine, M.H. Krane, L. Goss, J. Crafton, Correcting for color crosstalk and chromatic aberration in multicolor particle shadow velocimetry, Meas. Sci. Technol. 26 (2015) 025302.
[18] C. Cierpka, R. Hain, N.A. Buchmann, Flow visualization by mobile phone cameras, Exp. Fluids 57 (2016) 1–10.
[19] F. Scarano, Tomographic PIV: principles and practice, Meas. Sci. Technol. 24 (2012) 012001.
[20] M. McPhail, M. Krane, A. Fontaine, L. Goss, J. Crafton, Multicolor particle shadow accelerometry, Meas. Sci. Technol. 26 (2015) 045301.
[21] Hicham Saaid, Jason Voorneveld, Christiaan Schinkel, Jos Westenberg, Frank Gijsen, Patrick Segers, Pascal Verdonck, Nico de Jong, Johan G. Bosch, Sasa Kenjeres, Tom Claessens, Tomographic PIV in a model of the left ventricle: 3D flow past biological and mechanical heart valves, J. Biomech. (ISSN: 0021-9290) 90 (2019) 40–49, http://dx.doi.org/10.1016/j.jbiomech.2019.04.024.
[22] H. Zhu, C. Wang, H. Wang, J. Wang, Tomographic PIV investigation on 3D wake structures for flow over a wall-mounted short cylinder, J. Fluid Mech. 831 (2017) 743–778, http://dx.doi.org/10.1017/jfm.2017.647.
[23] K.P. Lynch, J. Wagner, High-speed tomographic PIV of cylinder wakes in a shock tube using a pulse-burst laser. United States, 2019, https://www.osti.gov/servlets/purl/1640645.
[24] N. Liu, Y. Wu, L. Ma, Quantification of tomographic PIV uncertainty using controlled experimental measurements, Appl. Opt. 57 (2018) 420–427.
[25] C. He, P. Wang, Y. Liu, L. Gan, Flow enhancement of tomographic particle image velocimetry measurements using sequential data assimilation, Phys. Fluids 34 (2022) 035101, http://dx.doi.org/10.1063/5.0082460.
[26] J.G. Liu, P.J. Mason, Essential Image Processing for GIS and Remote Sensing, Wiley-Blackwell, ISBN: 978-0-470-51032-2, 2009, p. 4.
[27] A. Chilton, The working principle and key applications of infrared sensors, 2013, AZoSensors. Retrieved 2020-07-11.
[28] F. Tariq, R. Haswell, P.D. Lee, D.W. McComb, Characterization of hierarchical pore structures in ceramics using multi-scale tomography, Acta Mater. 59 (5) (2011) 2109–2120, http://dx.doi.org/10.1016/j.actamat.2010.12.012.
[29] J.D. Foley, Andries van Dam, J.F. Hughes, S.K. Feiner, Spatial-partitioning representations; surface detail, in: Computer Graphics: Principles and Practice, in: The Systems Programming Series, Addison-Wesley, ISBN: 0-201-12110-7, 1990.
[30] S. Chmielewski, P. Tompalski, Estimating outdoor advertising media visibility with voxel-based approach, Appl. Geogr. 87 (2017) 1–13, http://dx.doi.org/10.1016/j.apgeog.2017.07.007.
[31] R. Novelline, Fundamentals of Radiology, fifth ed., Harvard University Press, ISBN: 0-674-83339-2, 1997.
[32] T.B. Moeslund, E. Granum, A survey of computer vision-based human motion capture, Comput. Vis. Image Underst. 81 (3) (2001) 231–268.
[33] K.K. Biswas, S.K. Basu, Gesture recognition using microsoft kinect®, in: Automation, Robotics and Applications (ICARA), 2011 5th International Conference on, IEEE, 2001.

[34] S. Larsson, J.A.P. Kjellander, Motion control and data capturing for laser scanning with an industrial robot, Robot. Auton. Syst. 54 (6) (2006) 453–460, http://dx.doi.org/10.1016/j.robot.2006.02.002.
[35] K. H. Strobl, E. Mair, T. Bodenmüller, S. Kielhöfer, W. Sepp, M. Suppa, D. Burschka, G. Hirzinger, The self-referenced DLR 3D-modeler, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, St. Louis, MO, USA, 2009, pp. 21–28.
[36] K. H. Strobl, E. Mair, G. Hirzinger, Image-based pose estimation for 3-D modeling in rapid, hand-held motion, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 2011, pp. 2593–2600.
[37] S. Goel, B. Lohani, A motion correction technique for laser scanning of moving objects, IEEE Geosci. Remote Sens. Lett. 22 (2014) 5–228.
[38] J. Ring, The laser in astronomy, New Sci. (1963) 672–673.
[39] A.P. Cracknell, L. Hayes, Introduction to Remote Sensing, second ed., Taylor and Francis, London, ISBN: 0-8493-9255-1, 2007, OCLC 70765252.
[40] G.G. Goyer, R. Watson, The laser and its application to meteorology, Bull. Am. Meteorol. Soc. 44 (9) (1963) 564–575 [568].
[41] A. Medina, F. Gaya, F. Pozo, Compact laser radar and three-dimensional camera, J. Opt. Soc. Amer. A 23 (2006) 800–805.
[42] E. Trickey, P. Church, X. Cao, Characterization of the OPAL obscurant penetrating LIDAR in various degraded visual environments, in: Proc. SPIE 8737, Degraded Visual Environments: Enhanced, Synthetic, and External Vision Solutions 2013, 2013, p. 87370E, http://dx.doi.org/10.1117/12.2015259.
[43] M. Hansard, S. Lee, O. Choi, R. Horaud, Time-of-Flight Cameras: Principles, Methods and Applications, in: SpringerBriefs in Computer Science, ISBN: 978-1-4471-4657-5, 2012, http://dx.doi.org/10.1007/978-1-4471-4658-2.
[44] S. Schuon, C. Theobalt, J. Davis, S. Thrun, High-quality scanning using time-of-flight depth superresolution, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Institute of Electrical and Electronics Engineers, 2008, pp. 1–7.
[45] S.B. Gokturk, H. Yalcin, C. Bamji, A time-of-flight depth sensor - system description, issues and solutions, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Institute of Electrical and Electronics Engineers, 2005, pp. 35–45, http://dx.doi.org/10.1109/CVPR.2004.291.
[46] ASC's 3D Flash LIDAR camera selected for OSIRIS-REx asteroid mission. NASASpaceFlight.com. 2012-05-13.
[47] J. Aue, D. Langer, B. Muller-Bessler, B. Huhnke, Efficient segmentation of 3D LIDAR point clouds handling partial occlusion, in: 2011 IEEE Intelligent Vehicles Symposium (IV), IEEE, Baden-Baden, Germany, ISBN: 978-1-4577-0890-9, 2011, http://dx.doi.org/10.1109/ivs.2011.5940442.
[48] S. Hsu, S. Acharya, A. Rafii, R. New, Performance of a time-of-flight range camera for intelligent vehicle safety applications (PDF), in: Advanced Microsystems for Automotive Applications, in: VDI-Buch, Springer, ISBN: 978-3-540-33410-1, 2006, pp. 205–219, http://dx.doi.org/10.1007/3-540-33410-6-16.
[49] O. Elkhalili, O.M. Schrey, W. Ulfig, W. Brockherde, B.J. Hosticka, P. Mengel, L. Listl, A 64 × 8 pixel 3-D CMOS time-of flight image sensor for car safety applications, in: European Solid State Circuits Conference 2006, ISBN: 978-1-4244-0302-8, 2006, pp. 568–571, http://dx.doi.org/10.1109/ESSCIR.2006.307488.
[50] T.I. Zohdi, The game of drones: rapid agent-based machine-learning models for multi-UAV path planning, Comput. Mech. (2019) http://dx.doi.org/10.1007/s00466-019-01761-9.
[51] T. Zohdi, A machine-learning framework for rapid adaptive digital-twin based fire-propagation simulation in complex environments, Comput. Methods Appl. Mech. Engrg. (2020) http://dx.doi.org/10.1016/j.cma.2020.112907.
[52] T.I. Zohdi, A digital twin framework for machine learning optimization of aerial fire fighting and pilot safety, Comput. Methods Appl. Mech. Engrg. 373 (2021) 113446.
[53] T.I. Zohdi, A digital-twin and machine-learning framework for the design of multiobjective agrophotovoltaic solar farms, Comput. Mech. 68 (2021) 357–370.
[54] T.I. Zohdi, A digital-twin and machine-learning framework for precise heat and energy management of data-centers, Comput. Mech. 69 (2022) 1501–1516.
[55] T.I. Zohdi, Modeling and simulation of a class of coupled thermo-chemo-mechanical processes in multiphase solids, Comput. Methods Appl. Mech. Engrg. (2004) 679–699.
[56] T.I. Zohdi, Computation of strongly coupled multifield interaction in particle-fluid systems, Comput. Methods Appl. Mech. Engrg. 193/6-8 (2007) 3927–3950.
[57] T.I. Zohdi, Simulation of coupled microscale multiphysical-fields in particulate-doped dielectrics with staggered adaptive FDTD, Comput. Methods Appl. Mech. Engrg. 199 (2010) 79–101.
[58] T.I. Zohdi, Embedded electromagnetically sensitive particle motion in functionalized fluids, Comput. Part. Mech. 1 (2014) 27–45.
[59] T.I. Zohdi, A Finite Element Primer for Beginners. The Basics, Springer International Publishing, 2018.
[60] J.H. Holland, Adaptation in Natural & Artificial Systems, University of Michigan Press, Ann Arbor, Mich, 1975.
[61] Holland J.H., J.H. Miller, Artificial adaptive agents in economic theory (PDF), Amer. Econ. Rev. 81 (2) (1991) 365–371, Archived from the original (PDF) on October 27, 2005.
[62] D.E. Goldberg, Genetic Algorithms in Search, Optimization & Machine Learning, Addison-Wesley, 1989.
[63] L. Davis, Handbook of Genetic Algorithms, Thompson Computer Press, 1991.
[64] C. Onwubiko, Introduction to Engineering Design Optimization, Prentice Hall, 2000.
[65] D.E. Goldberg, K. Deb, Special issue on genetic algorithms, Comput. Methods Appl. Mech. Engrg. 186 (2–4) (2000) 121–124.
[66] D. Luenberger, Introduction to Linear & Nonlinear Programming, Addison-Wesley, Menlo Park, 1974.
[67] P. Gill, W. Murray, M. Wright, Practical Optimization, Academic Press, 1995.
[68] T.I. Zohdi, A note on rapid genetic calibration of artificial neural networks, Comput. Mech. 70 (2022) 819–827.