A Spatial Arrangement Visualisation Strategy for Learning from Large Ensembles of Unsteady Flow-field Data

Aljaz Kotnik*, Graham Pullan†
Whittle Laboratory, Department of Engineering, University of Cambridge, Cambridge, UK

Discovering cause-and-effect relationships, the links between input parameters to an experiment or simulation and the observed outcome, is fundamental to the aerodynamic design process. An understanding of the underlying flow mechanisms responsible for these connections is the key transferable learning to be derived. The engineer's domain knowledge is crucial for flow-field data interpretation and, in design studies containing large numbers of cases, the necessary human-data interaction becomes the bottleneck of the process. We demonstrate a conceptual framework that accelerates this step by reducing the number of detailed flow-field visualisation analyses and comparisons required to gain a full understanding of cause-and-effect relationships. Our approach allows engineers to visually encode their interpretation of flow-field visualisations by interactively adjusting the on-screen location of the data plots. To identify potential cause-and-effect relationships, this spatial arrangement is then correlated with the input and output metadata associated with each flow-field case. This approach is suitable for any visualisation that includes aspects of the underlying physical processes, and a web browser-based implementation is demonstrated. We introduce a freely available example dataset consisting of unsteady CFD solutions of 2D turbine designs, illustrate the utility of the spatial arrangement framework by applying it to the dataset, and demonstrate the advantages and limitations of visualising unsteady data in the web-browser.

I. Introduction

In aerospace engineering, the analysis of flow-field data visualisations to uncover or explain behavior is a key part of daily workflow. The NASA CFD Vision 2030 [1] report identified that,

> "the CFD capability in 2030 must provide the analyst with a more intuitive and natural interface into the flow solution to better understand complex flow physics and data trends ...", and
>
> "A single engineer/scientist must be able to conceive, create, analyze, and interpret a large ensemble of related simulations in a time-critical period."

Modern computational fluid dynamics tools are capable of executing hundreds of simulations in a day. Individually analysing large ensembles of flow-fields is time consuming, and a practical approach to data analysis is to work primarily with metadata; high-level metrics (e.g. coefficients of lift, drag, loss,...) which summarise the data. However, this means that large numbers of flow-fields are never analysed in detail, which in turn reduces the engineer's understanding of the design space. To address this, we propose an approach that allows the user to interact dynamically with the flow visualisations when exploring the design space, while at the same time retaining a connection to high-level metrics to answer questions the user may have during design space exploration.

Pullan [2] introduced a hierarchical approach to large flow-field ensemble exploration. The hierarchical approach aims to reduce the amount of data the user is exposed to by trading quality for quantity - the user can be exposed to many (thousands) of simulations represented only as high level metrics, called 'metadata', few (tens) simulations in detail, and available levels in between. The trade-off between quality and quantity is achieved by first creating a series of interactive plots with filtering capability. The user starts with a set of metadata plots, and can interactively select a subset of cases of interest for which more detailed plots can be requested. With the new set of plots the user can repeat the filtering and then request more data as needed. Thus at every step some quantity is traded for more detail. The hierarchical approach by Pullan [2] discusses steady data, but can be extended to cover unsteady data also, as shown in Figure 1.

*Research student
†Professor of Computational Aerothermal Design, AIAA Senior Member

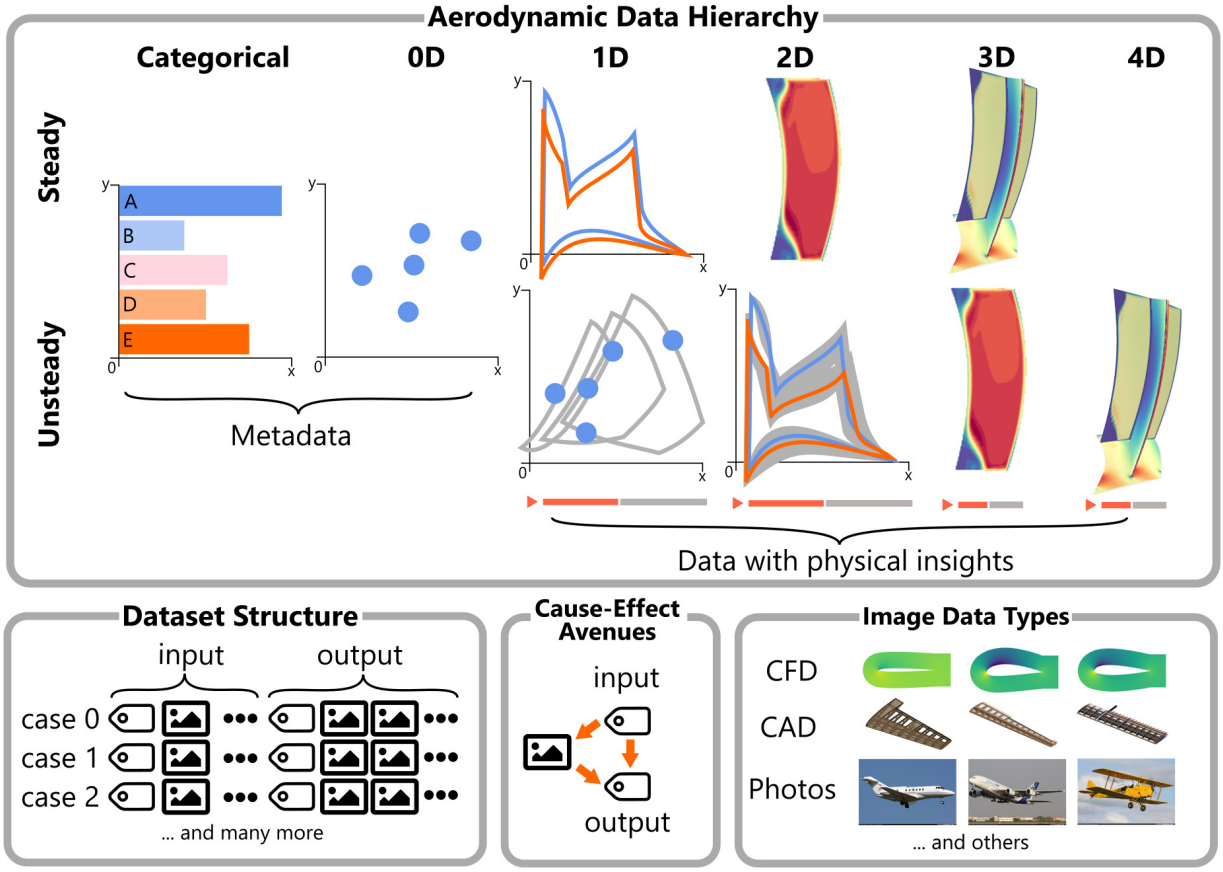

Fig. 1 The data hierarchy adapted from Pullan [2], the structure of the dataset containing several cases, and the available cause-and-effect relationships. The data level is based on the number of dimensions required to position individual values on screen (e.g. each contour data point value is positioned using two dimensions: x and y). Adding unsteady data moves all data types one level higher. Each case can associate input & output metadata with several images. The output images capture the domain-specific mechanism behavior. The most valuable relationships to discover are between metadata and output images. Image data could be visualisations of CFD results, CAD geometries, or photos of the hardware itself.

Figure 1 shows an abstract view of datasets containing metadata and images, depicting process inputs and outputs. Analysis of cause-and-effect relationships is possible based on pairs of input and output data. Potential causality relationships need to be supported by a valid explanation of the behavior of the underlying mechanisms. In many cases, these mechanisms are captured in images, for example flow-field contour plots. The input and output data can contain several images and/or metadata variables. In datasets with many metadata variables, there may be several underlying cause-and-effect relationships to discover, and systematically verifying them can be time-consuming. To effectively analyze and verify cause-and-effect relationships, an intuitive, interactive user interface, allowing the user to match the changes of physical features shown by the images with the corresponding metadata trends, is required.

Interactive positioning of images on-screen allows the user to create clusters of what they perceive are similar images. By meaningfully arranging the images relative to one another, the user effectively encodes their interpretation of the images into the on-screen position of the image. This 'spatial encoding' is the core interactive idea that is leveraged to accelerate the exploration session.

Correlation is not causation: the system can identify correlation, but only the user can verify whether the correlation helps explain the behavior. Our approach is to allow the user to interpret the data, encode their understanding, and search for correlations with the metadata. If the user successfully verifies a correlation using their domain knowledge, then they have found a cause-and-effect relationship. As the proposed process begins with the interpretation of the data

the onus is on the user to formulate a hypothesis, and then check it using the data, as opposed to initially searching for most promising correlations, and then trying to formulate hypothesis to fit them.

The first contribution of this paper is a proposed generic two-part approach for accelerated identification of cause-and-effect relationships through interactive on-screen spatial encoding:

1) **Divide and explore.** Image-based datasets typically include subset groups whose members are related. Identifying these groups is a key step in the discovery of cause-effect relationships in large image-based datasets. Based on the interactivity features demonstrated by Lekschas et al. [3], our approach enables the user to analyse the entire dataset in an efficient manner.
2) **Connecting spatial arrangement to input-output metadata.** Inspired by investigators finding patterns in data by arranging physical pictures on tabletops, interactive on-screen spatial arrangement of images is used to encode the user's domain knowledge. The resulting on-screen arrangement can be used to find correlated metadata variables and recommend them to the user for consideration.

The second contribution of the paper is an unsteady data visualisation approach that allows interactive cause-and-effect exploration of unsteady flow-field ensembles. An example unsteady ensemble dataset that is representative of a task faced by an engineer is presented. Visualisations of unsteady data that allow the spatial arrangement approach to be used are presented. Finally, the memory and data transfer limitations, and data storage approaches to tackle these limitations are discussed.

Section II shows how the proposed approach to find metadata correlations with physical features works in practice, and sketches how the two individual parts listed above would be used. These two parts are subsequently explained in detail: Section III - Grouping images to divide and explore; Section IV - Connecting spatial arrangement of images to input-output meta data. Section V shows how this approach can be applied to unsteady data at different levels of detail. Interactive visualisation of unsteady data using a web-based approach is limited by the client memory, and the connection speed which delivers the data to the client. This is most important for large unsteady contour plot data. Section VI discusses which datasets can be smoothly viewed in the browser, the loading times the user can expect, and data preparation strategies to increase the size of the grids that can be interactively visualised. Finally, an example unsteady turbine design ensemble dataset is introduced in Section VII, and the approaches discussed throughout the paper are applied to the example dataset in Section VIII.

II. Conceptual overview of proposed workflow

To illustrate the benefit of the proposed approach, we use the example of an engineer studying the appearance and behavior of corner separations (a type of three-dimensional boundary layer separation prone to form in the suction-surface/endwall corner of a compressor blade row), in an example dataset featuring 590 compressor stator design candidates [4]. The dataset consists of designs that result in three flow field 'types': with a corner separation at the inner radius endwall, both inner and outer radius endwall corner separation, or no corner separation at all. The dataset thus contains subsets featuring different configurations of corner separations, as well as differences in separation sizes. The differences in size and type are driven by the blade lean angle, the number of blades in the row, and their inlet angle. The basis for the exploration are the flow visualisations, in this case stator exit loss contours.

At the beginning of the exploration the engineer is presented with a 'wall' of 590 contour plots, as shown in Figure 2 a). To organise the session, the contours can be arranged on-screen by similarity by dragging and dropping. The contours can be piled on-top of one another to create groups of related contours, such as contours showing separation both at the hub and the casing, as seen in Figure 2 b). To create a group of contours, the engineer can use a lasso tool to select several contours, and select the 'group' option from the pop-up menu. Groups can be 'entered', which allows the engineer to focus only on the members of the group, while temporarily hiding all other contours. The advantages of creating groups during exploration are discussed in Section III.

The engineer is now interested to know aspects of blade design encourage the formation of corner separations, and why some cases show two corner separations, and others only one. The engineer can interactively arrange the three groups that represent individual flow field 'types' on screen, with the group showing no corner separation on the left, and two corner separations on the right. The links between the on-screen position of the contours and the metadata are used to identify 'lean type' as an important geometrical parameter, as seen in Figure 2 c). The same approach can be used to investigate the design parameters linked to the size of the corner separations. Section IV discusses how spatial encoding is used to find metadata related to the on-screen arrangement of contours.

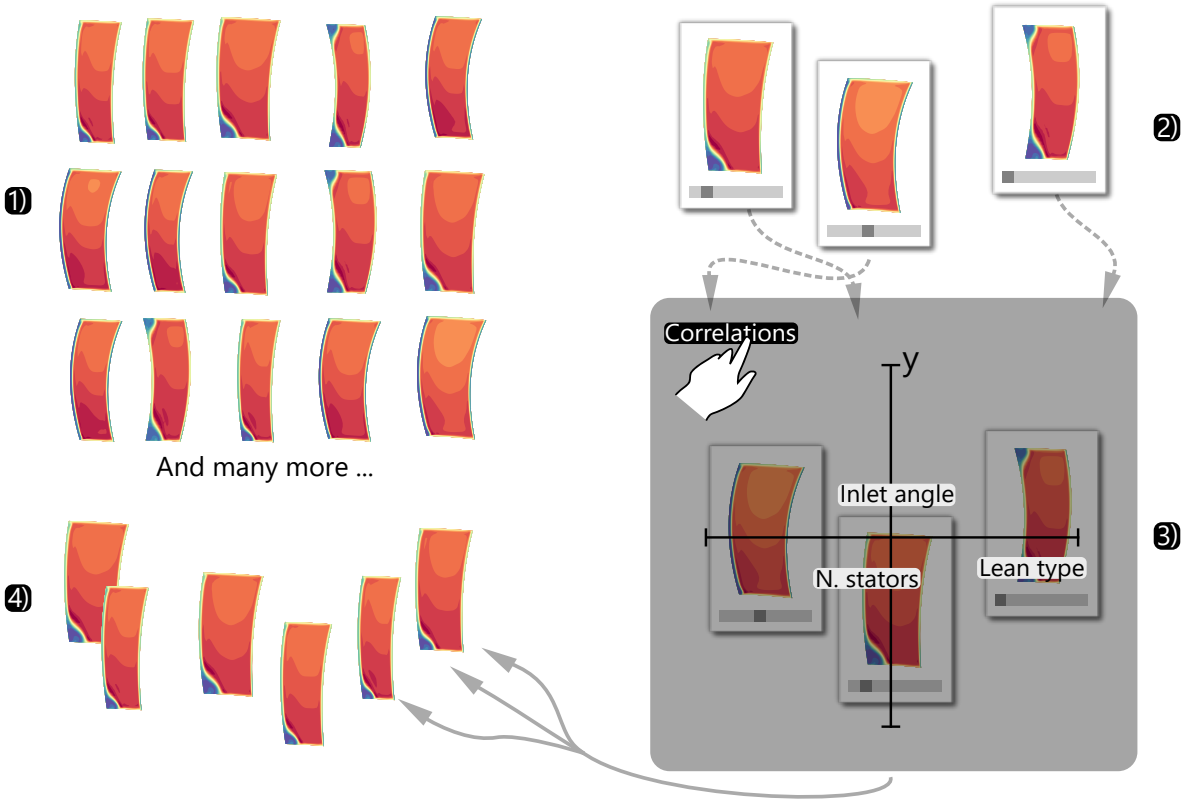

Fig. 2 1) The starting point is a '*wall of contours*'. 2) The user can organise the contours by *grouping* related contours together. 3) The on-screen position of contours or contour groups can be used to *encode* trends and find correlated metadata variables. 4) Groups can be *entered* to continue a more detailed exploration within the context of the group.

III. Divide and explore

The first step in the workflow of Figure 2 is to accelerate the analysis by separating images into meaningful groups (e.g. by grouping all flow-fields with distinct flow features together, for instance by shock position, or separation point). This aids the cause-effect discovery process in two ways: first, representatives of individual subset groups can be correlated with input-output metadata to determine why the images show sets of different flow features, rather than having to analyse each image separately; second, the smaller variations within a subset group can also be correlated with the metadata to ease the identification of the mechanisms driving the detailed variations of the flow features. The focus is on allowing the practitioner to use their domain knowledge to aid the process as much as possible.

One approach is to compare individual dataset members, and progressively improve the engineer's understanding of the dataset. When following this approach, all 590 images must be analyzed in detail to get a complete understanding of the data. Alternatively, the engineer could first group the images by the type of corner-separation they exhibit, thus allowing separate analysis of what causes different corner separations to arise, and differences between their sizes. Instead of analyzing 590 images, the user can then analyse the causes of different corner separations using just 3 images, and then analyse groups of 185 (hub and casing), 280 (hub), and 125 (no separation) images separately, within the context of the individual flow-field types. The division of the analysis process into two steps, and the questions answered at the different levels are shown in Figure 3. The separate analysis of smaller groups already requires fewer pairwise comparisons of members to be made, thus accelerating the analysis. Furthermore, the similarity of taxonomy group members allows subtle differences between members to be observed faster.

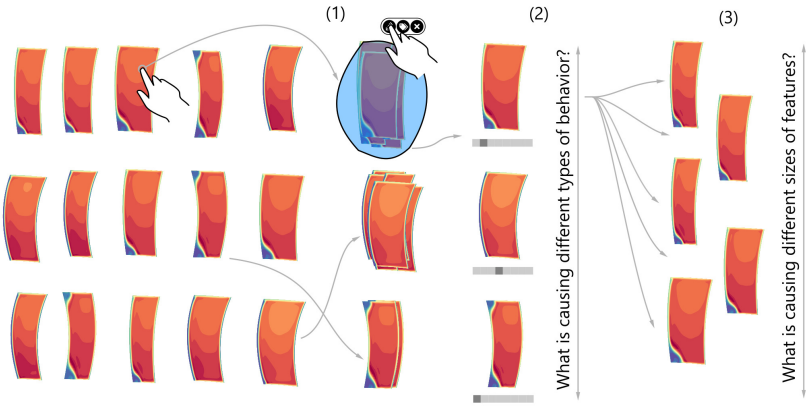

Fig. 3 Instead of comparing all contours at once, 1.) form meaningful groups, 2) compare them to understand the fundamental differences, and 3.) analyse the groups in detail separately.

IV. Connecting spatial arrangement to input-output metadata

Section III discussed how the analysis of an ensemble flow-field dataset can be split into two parts: identifying flow-fields with different sets of features, and identifying the cause of the differences between comparable features. This section describes an approach to aid the discovery of these correlations.

Trends can be found by separately analyzing either the metadata or the images in isolation, but working with both simultaneously allows potential cause-and-effect relationships to be identified. The user can ask questions of the data in two ways: by mapping from the metadata to the images, or *vice versa*.

The user may want to know how a metadata parameter impacts the features shown by the images (question mapping from metadata to the images). By rearranging the images on-screen using associated metadata, the user can directly compare the images to identify any trends associated with the metadata variable used for the arrangement [3, 5]. For datasets with hundreds of metadata variables, the process of sequentially arranging the images by one metadata parameter, then the next, could be time-consuming.

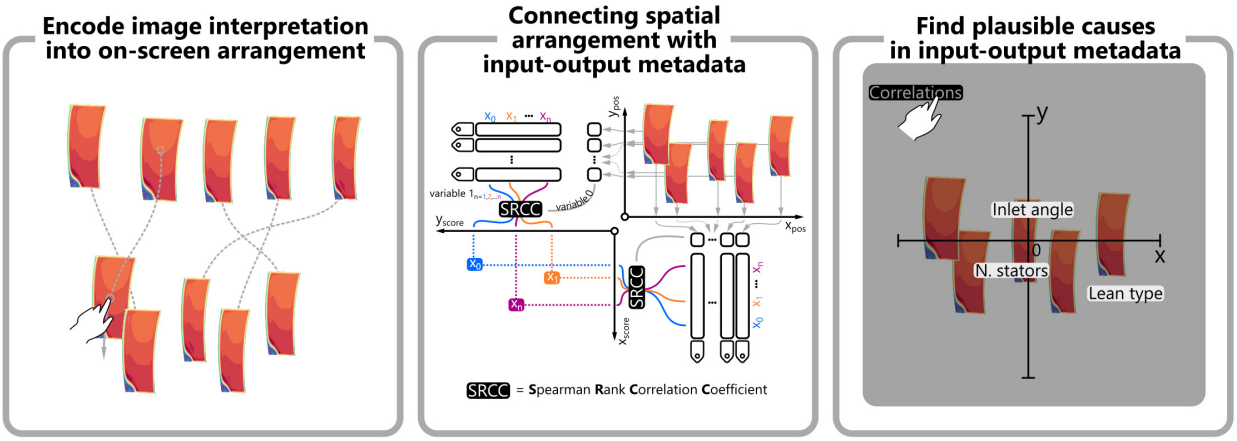

Fig. 4 The user spatially encodes their interpretation of the images by arranging them via dragging. The Spearman rank-correlation coefficient is calculated to identify input-output metadata variables correlated with the arrangement. A pop-up menu is used to present the calculated correlations.

The process can be accelerated by first identifying the changes that are visible in the images, and then finding the associated correlations in the metadata (question mapping from the images to the metadata). This approach relies on the

analyst interacting with the data to encode their domain knowledge, such that the correlations in the metadata can be found. We suggest that the encoding is done via on-screen arrangement of the flow-field images.

The discovery of metadata related to the on-screen arrangement pattern consists of two steps. The first step is to interactively arrange the images on-screen into a meaningful pattern, and the second is finding variables that correlate well with the arrangement in the x and y coordinate directions. The first step allows domain knowledge to be used to interpret the images and encode the domain knowledge.

In the second step, shown in Figure 4, the correlation between the on-screen arrangement and the metadata variables is calculated using the SRCC (Spearman's Rank Correlation Coefficient), which assesses how well the relationship between two variables can be described using a monotonic function. The use of SRCC requires the user to arrange the images such that the changes either decrease or increase monotonically in the vertical and horizontal directions. One input variable of the SRCC calculation is a metadata variable, and the other is the vertical or horizontal position of the small multiple on-screen. Therefore, the SRCC is calculated twice for every metadata variable.

As well as finding potential cause-and-effect relationships between the metadata and individual group members, the spatial arrangement and correlation approach can also be used to study the connections between groups (piles) of images and the metadata. After the piles are created, the task for the analyst is to explain the cause of inherent differences between the piles. The approach discussed for analyzing individual images can also be applied to piles. The analyst can spatially arrange the piles to form a meaningful trend or pattern, metadata variables correlated with the spatial arrangement are automatically detected, and the analyst can use those as clues to generate a domain-specific explanation for the observed pattern.

The cases corresponding to the images can have several images associated with them, each showing a different aspect of the domain specific behavior. By creating piles based on one set of images, and then switching the view to another, the user can see how insights created when analyzing the first set correlate with the second. In this way, on-screen arrangement can be seen as a lens through which to look for cause-and-effect relationships between image-based data.

V. Unsteady data visualisatons

Sharing unsteady data is often done by first converting individual time snapshots into 2D images (regardless if the time snapshot data is 2D or 3D), and combining the images into an appropriate file format for sharing, e.g. a MPEG movie file. The rendering of 2D images in this way limits the opportunities for interaction. This is particularly restrictive for 3D unsteady simulations, where the user may wish to adjust the view of the simulated geometry on-the-fly. For example, commonly used sharing file formats do not support a reorientation of the domain, and the user is forced to make a new file to view the domain from the newly desired viewpoint. Similarly, a new file has to be created for every new iso-surface value, color scheme change, or color domain limits change.

Ideally, sharing unsteady data would preserve interactivity, be accessible without the need of specialised software, and be quick and easy to view. We propose pursuing this vision using a web-based approach. By dynamically visualising data in the browser the interactivity is preserved, sharing is done by sharing the link to the visualisations, and is accessible from any machine with a browser and an internet connection.

Unsteady point and line data can be shown as the current timestep in the foreground, and an outline of all other timesteps in the background, as shown in Figure 1. For scatterplots, the paths the points take with respect to time may be an informative representation. By separating the unsteady scatter and line plots into multiple individual plots, each showing the unsteady data of a particular case, the user can create groups of similar designs, as outlined in Section III. Figure 5 shows examples of unsteady scatter and line plots and how they can be separated into individual axes.

An interactive visualisation of 2D contour snapshots from a LES simulation of a turbine blade trailing edge, from Pullan [6], is shown in Figure 6. The demo features a central view showing the currently selected snapshot, and two views on either side connected to it. The left view enables navigation between different snapshots by hovering the mouse cursor over individual points, while the right view allows the user to extract and visualise data from a particular vertical line in the domain. Furthermore, this demo demonstrates the ease with which unsteady data visualisations can be shared over the internet.

A challenge of the approach shown in Figure 6 is that hovering over the points requires the user to look away from the contour plot, which makes tracking motion of flow features harder. Adding a playbar, such as those found in video players to navigate through time, removes the dependency on another plot, and allows the user to navigate the data without taking their eyes off the contour. Figure 5 c) shows an example of such an unsteady contour plot.

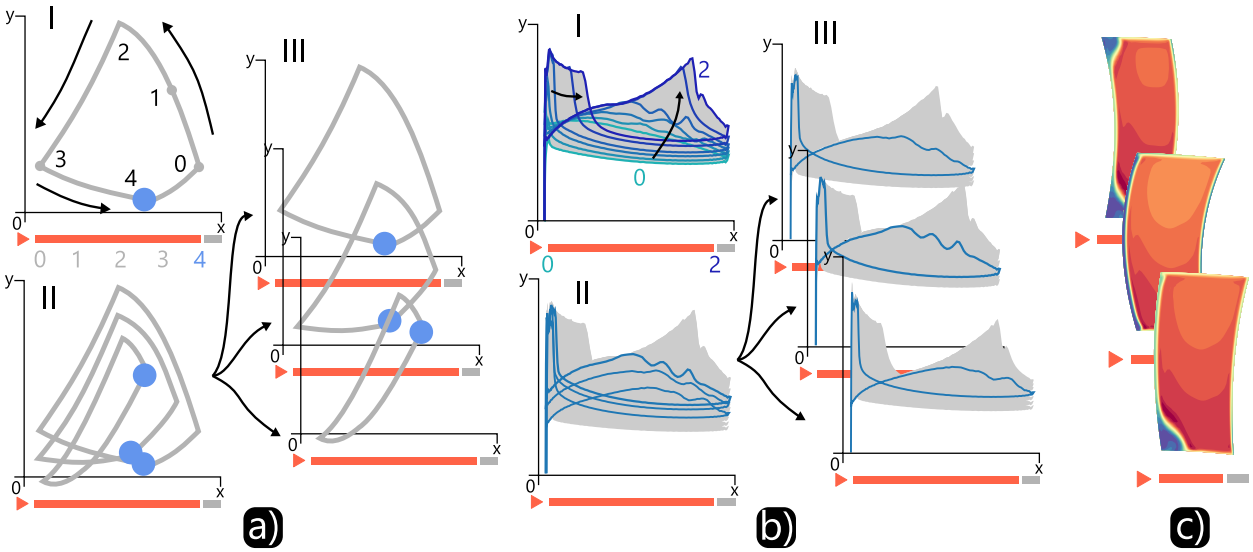

Fig. 5 a) Unsteady $0D$ data. b) Unsteady $1D$ data. c) Unsteady $2D$ data. I.) A schematic representation of a single unsteady task visualisation. The arrows indicate the change when the 'play' button is activated. II.) Several unsteady $0D$ and $1D$ tasks at once. III.) Optionally splitting the plot into several individual plots. These can be rearranged to look for correlations with metadata, and navigated along the play bar independently.

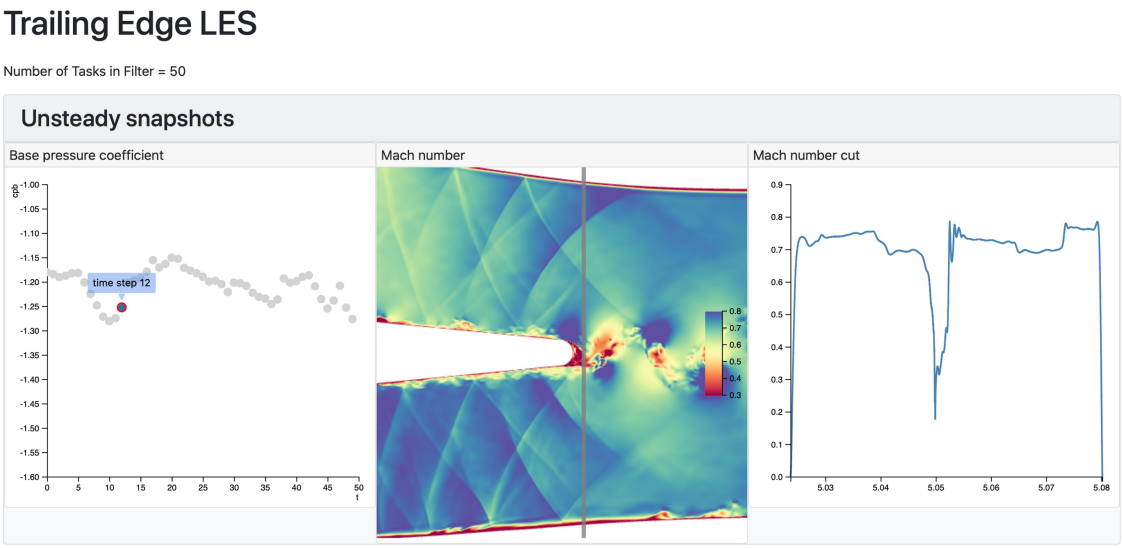

Fig. 6 An unsteady data visualisation approach taken from Pullan [6]. The left view allows the user to navigate between timestep snapshots by hovering over specific points. When a point is hovered over the central view updates. The vertical line in the central view can be moved horizontally. Data from the central view along this line is extracted on-the-fly and visualised in the right view.

VI. Data storage and memory handling

The data visualisations discussed in the previous sections are rendered in the web-browser using JavaScript and WebGL. The amount of data that can handled using JavaScript depends, in part, on the particular JavaScript engine version used, but a typical value for the V8 JavaScript engine used in Chrome and other Chromium based web browsers is 4GB. Furthermore, this data needs to be transferred from the server to the client. The connection speed therefore determines how quickly the user can start engaging with the interactive visualisations. Ideally, all of the requested data would be presented simultaneously, and be ready in fractions of a second. These limitations are especially important to

consider when attempting to analyse flow-field data, which is typically larger than 1GB. For example, an unsteady $2D$ flow field with 10^6 nodes and 100 timesteps is 1.2GB large if uncompressed (10^6 nodes ×(2 spatial variables + 1 flow variable)×4 bytes per value). Simultaneously visualising 16 such cases is 4.8 times larger than the memory available. Putting aside the browser memory limitation, and assuming a connection speed of $10MB/s$ it would take 32 minutes for all the data to be fully transferred.

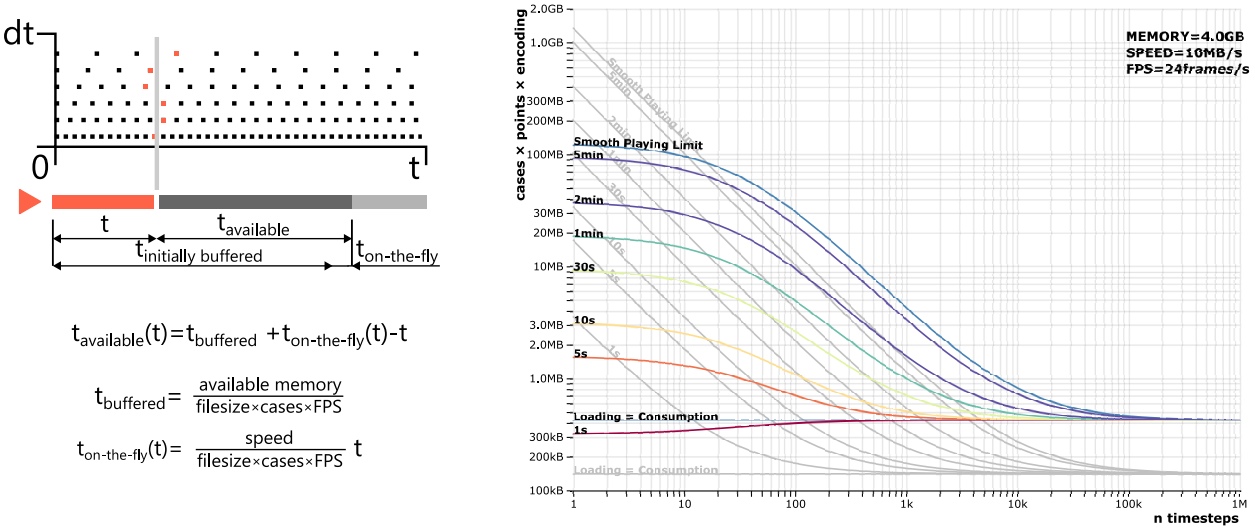

Fig. 7 The player displays the timestep snapshot closest to a given time, which allows datasets with different timestep sizes to be played together. To be able to smoothly play through the entire domain in the browser either all the data must be loaded in, the connection speed must handle loading the files on-the-fly, or a combination of both. The chart can be used to determine whether a particular set of cases can simultaneously be played smoothly, and how much time it will take to buffer the required initial data. The gray lines represent the loading behavior when raw grid data, split into timestep snapshot files, is used for visualisation. The colored lines represent the loading behavior for data transformed into a format aimed at rendering. The coloured lines flatten out at low timestep numbers as an additional file must be loaded in to allow correct interpretation of the data prepared for rendering. An interactive version of chart is available [7].

A simple way of splitting the data into smaller chunks is to separate the individual time-step snapshots into separate files, allowing partial data to be loaded in for each of the desired cases. Continuing the example from the previous paragraph, for 16 cases roughly 20% of the entire data can be loaded. Furthermore, at $10MB/s$, the first time-step data is available in roughly $19.2s$, and the user can interact with it while the rest of the data is being loaded. The time to first interaction is slow, and approaches to accelerate it are discussed below. The time to download the maximum data possible is 6.7 minutes, and depends only on the data size limit and the connection speed.

Another consideration is the data consumption rate; the number of time-step snapshots per second at which the data is shown, and therefore the amount of data that the user moves through per second. When synchronously playing cases with different time-steps, the cases with larger time-steps would move through their time domain faster. To account for different time-steps the snapshots shown are selected by selecting the ones closest to a given time, as shown in Figure 7.

In cases where the entire data cannot be loaded at once, the consumption rate will determine whether the playing will be interrupted to load the necessary data. If the data consumption rate balances the loading speed, shown as the $Loading = Consumption$ line in Figure 7, then the data can be played smoothly for any number of time-steps. If the loading speed is lower than the consumption rate, but still sufficiently high (area below the $SmoothPlayingLimit$ in Figure 7), the required data can be loaded on-the-fly while the already loaded data is being consumed. Otherwise the playing will be interrupted.

The chart in Figure 7 can be used to determine whether a particular dataset can be played smoothly, and how long it takes for the required buffering data to be loaded. The y-axis represents the data size that needs to be transferred to advance by one time-step snapshot. The y-axis value is a combination of the number of points in the domain, the number of bytes required to encode the position and value of each point (for example (2 positional values + 1 flow value)×4 bytes for $float32$ number representations), and the number of cases considered at the same time. The x-axis

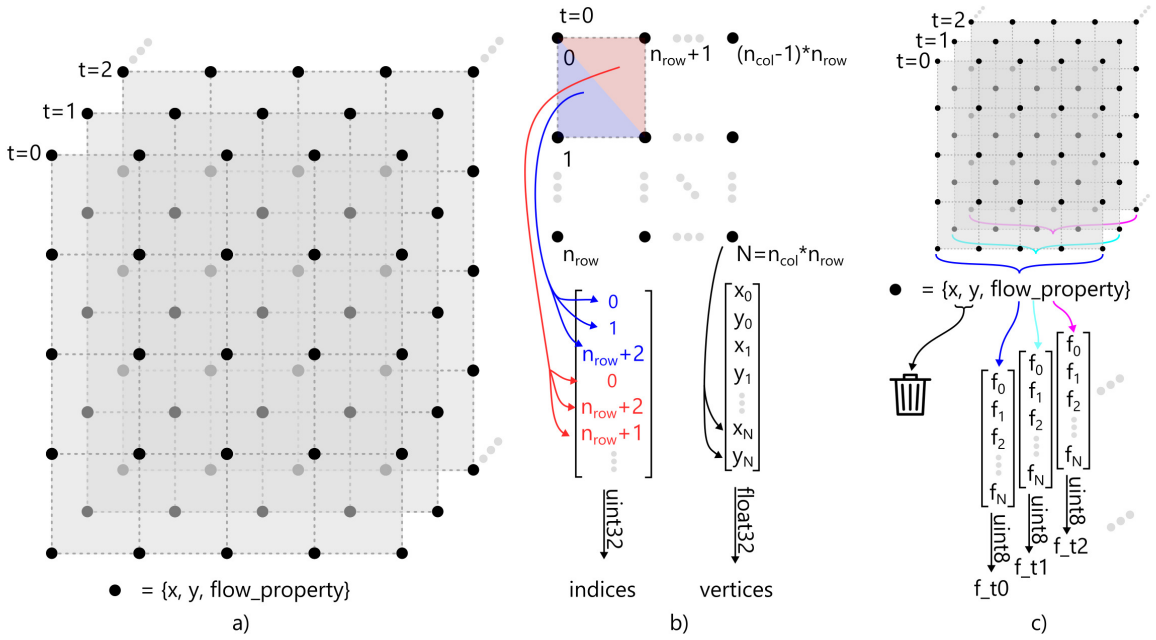

Fig. 8 a) A separate set of grid data is stored for every individual timestep. b) Assuming the grid geometry does not change with time, a single set of x and y locations can be used for drawing all timesteps. An additional 'indices' array that prescribes how to combine the vertices to form triangles for drawing is created. c) The flow property values of a single timestep are separated from the grid position data, which reduces the amount of data that has to be loaded on-the-fly. A further reduction is obtained by encoding these value using only 8 bits.

is the number of time-steps in the dataset.

The equations shown in Figure 7 calculate the amount of available playing time (time used to move through all the snapshots) as a function of the time already played are the basis for the chart. The intent is to have no more playing time available just as the last snapshot is shown. For example, if consumption balances loading, then $t_{available} > 0$, and datasets with any number of timesteps can be played smoothly. To calculate the maximum size of the data for a single time-step snapshot, we can assume that the entire memory has been used to buffer the required data, and set $t_{available} = 0$ at $t = n_{timesteps}/FPS$, where FPS is the number of timesteps, or frames, played per second. By varying $n_{timesteps}$ the smooth playing limit can be found. Any datasets that fall above this limit cannot be played smoothly in the browser unless the memory limit is relaxed, the connection speed is improved, or the consumption rate is lowered.

As per the chart in Figure 7, 16 cases with 10^6 nodes ($y = 16$ individual timestep data of $12MB = 192MB$ and $x = 100$ timesteps) cannot simultaneously be played smoothly, unless the timestep snapshot data is reduced. Figure 8 shows a two step approach to reducing the data, and preparing it for rendering on the GPU. Figure 8 a) shows the individual timestep snapshot data. Each snapshot contains the x and y positions for each grid node, alongside the flow variables of interest. Assuming that the grid node positions don't change between timestep snapshots, the x and y grid node positions can be stored separately from the snapshot data, reducing the snapshot size. Splitting the grid node positions from the flow field values also allows this data to be prepared for rendering by the GPU before loading it into the browser, thus saving the processing time. Figure 8 b) shows two new files that are created. The grid position data is stored in *vertices*. The contour is drawn by the GPU as triangles. The data on which vertices together form triangles that should be drawn is stored in the *indices* file.

The snapshot data can be reduced further by considering a different number encoding. By default the numbers are stored as $float32$ using 4 bytes each. By remapping the values of a single snapshot to the range $[0, 255]$ the values can be saved using just 1Byte. The minimum and maximum values can be saved in the metadata file, and passed to the GPU alongside the data. The GPU then maps the $uint8$ values back to the correct range. This compression technique reduces the snapshot size by a factor of 4. Other compression techniques are available ([8], is a lossy compression scheme capable of compressing the data up to 16 times without producing any perceptible errors in the visualisation). Including such powerful methods of data compression could further improve loading performance, and thus allow cases with

larger domains to be visualised in a web-browser.

VII. Example dataset of 2D unsteady turbine simulations

Figure 9 shows different aspects of the example dataset consisting of unsteady CFD solutions of 69 2D turbine designs, all with a degree of reaction of 0.5, and exit Mach number of 0.65. The CFD evaluation has been done using Turbostream 3, a GPU accelerated unsteady RANS code [9]. The dataset is open [7], and contains the original flow fields in npz compressed files, some extracted metadata, and files for rendering.

All designs were created using the same methodology, and the design was specified with only the flow coefficient, and stage loading. The methodology did not attempt to optimise the designs, and it is likely that adjusting the aerofoil shapes would increase their performance. The number of stators and rotors varies between individual designs. The domain is created by stacking together grid blocks, one per each passage, as shown in Figure 9. Each block has 65 nodes per passage width, and a 282 and 343 nodes in the axial direction for the stator and rotor blocks respectively. Alltogether, the domain has on the order of 3×10^5 nodes, depending on the number of rotor and stator blocks. The unsteadiness comes from rotors passing the stators. 72 timestep snapshots are available, which represent the time needed for the rotor blades to pass a single stator passage height.

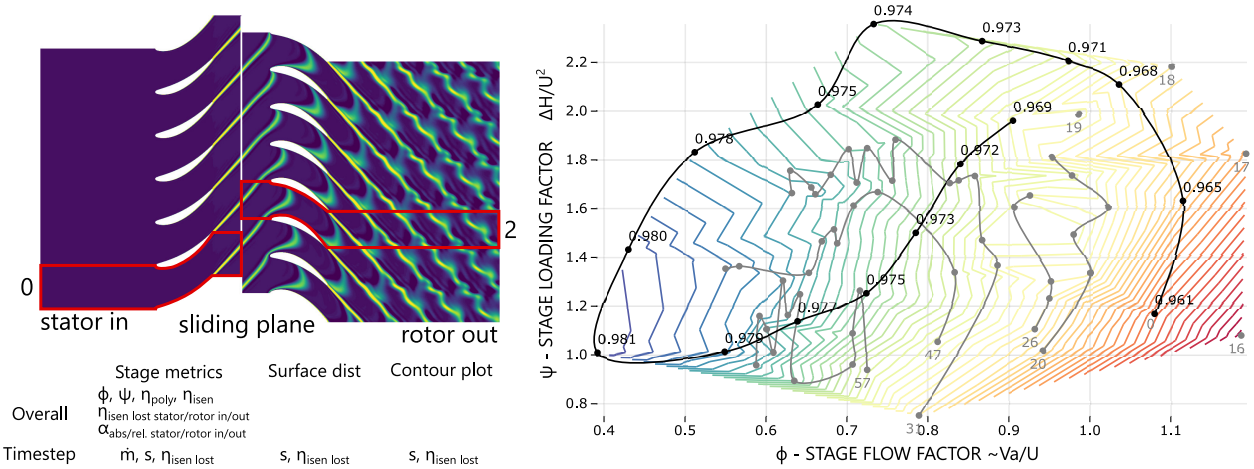

Fig. 9 A 2D contour showing the stator and rotor blocks. The sliding plane shows where the change of reference frame happens. Metadata such as flow angles, entropies and efficiencies shown in the table, have already been extracted from the unsteady flow-fields, both for the overall time-averaged solution and individual timesteps. The design flow coefficients and stage loading were taken from the Smith chart [10], and points denote individual designs. The coloured lines show the isentropic efficiency contours based on the overall isentropic efficiencies of the example designs. The black and gray lines show the naming convention of the cases.

The black and grey lines in Figure 9 show the naming convention. The design names are sequential numbers from 0 to 68, and some are drawn in gray next to the points in Figure 9. The black outline shows the first 16 cases, which were selected to bound the design space of interest. Points along the black line are labelled with their isentropic efficiencies. The gray lines connect the remaining designs, with each gray line connecting all designs in an efficiency interval, for example the area between 93% and 94% lines, as determined by the Smith chart [10]. The gray lines have the case names for their starting design point printed below them, and show in which order the case names increase from the starting point. The colorful contour lines represent iso-efficiency lines based on the design CFD evaluations, with the higher efficiencies in blue, and lower efficiencies in red.

VIII. Demonstration visualisation of the unsteady dataset

In this section we demonstrate the merits of a web-browser based approach for visualisation and analysis of unsteady ensemble data. The demo focuses on unsteady contour visualisation, as the data involved is larger than the scatterplot or line plot data, and the visualisation and its sharing are therefore more challenging.

The cases in the example dataset have domains with roughly 3×10^5 nodes, and the data for a single timestep for a

single case is therefore roughly 0.3MB. Figure 7 shows that a single case can be smoothly played within 1 second. A set of 16 cases has a timestep data size of 4.8MB, and can be ready for simultaneous playing within roughly 30 seconds. This demonstration is available online [7].

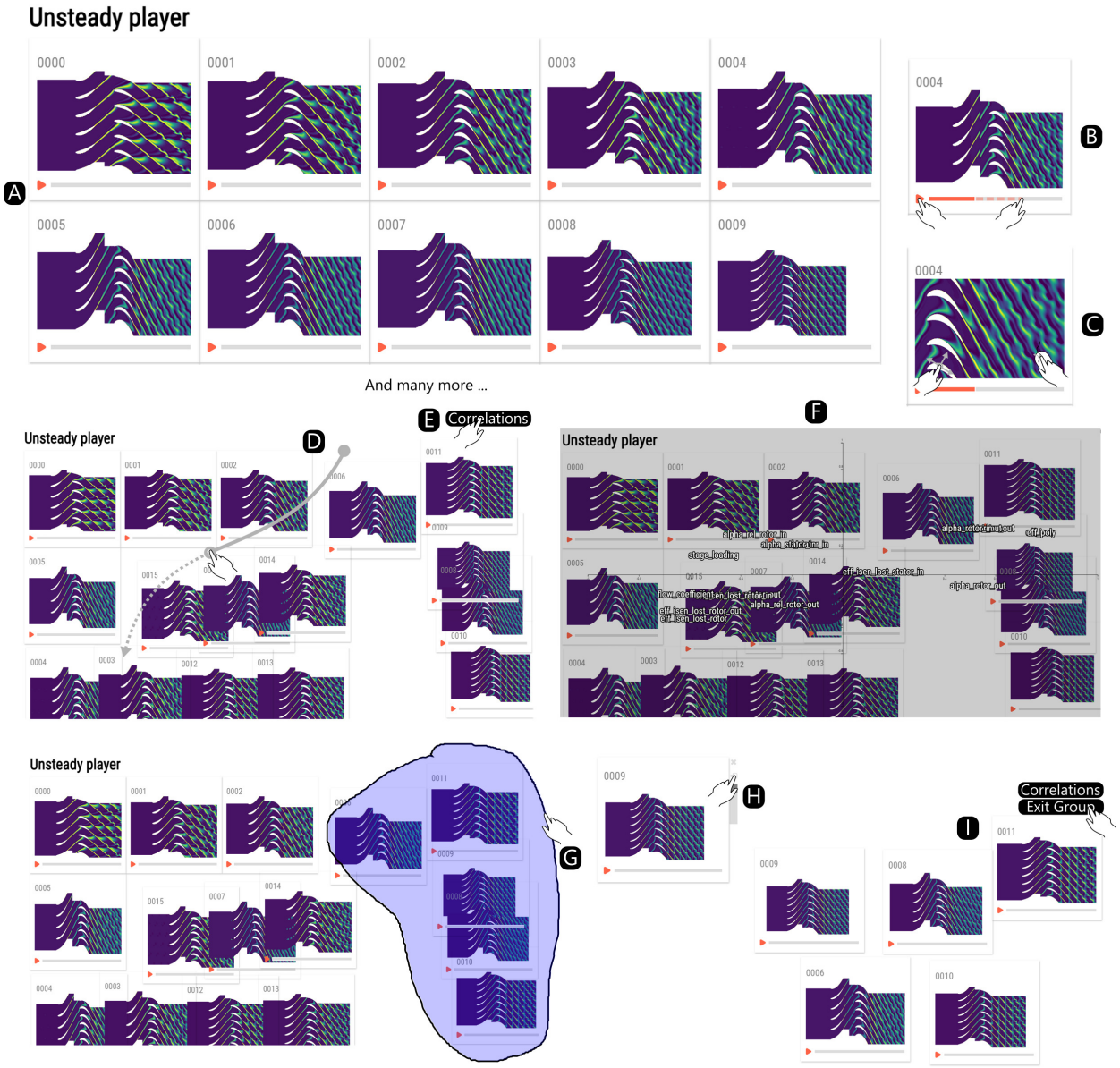

Fig. 10 A) The initial 'wall of contours'. B) Timestep snapshots can be played by pressing the play button, or the user can skip to a particular snapshot by clicking on the play bar. C) The user can use scroll to zoom in, and click and drag to pan the view. D) By dragging the contours, the user can encode their interpretation of the data, for example by placing contours with more mixed out wakes to the left, and less mixed out wakes to the right. E) A button must be clicked to display the correlations. Clicking anywhere on-screen closes the correlations view. F) Correlations are displayed as labels on a 2D axes. Clicking on the labels rearranges the cases in order of the clicked variable either in the x or y dimension. G) Clicking on the whitespace and dragging creates a lasso selection. After selection is complete the relevant cases are automatically piled together into a group. H) A created group has three interactive elements on the side: a button to dissolve the group, a button to enter the group, and bookmarks to switch between the case that is presented as the top of the group pile. I) When within a group a button that allows the group to be left appears.

Figure 10A shows the contours of the first 16 cases of the example dataset. When an example case is played, as shown in Figure 10B, the stator wake entropy signatures can be seen to be cut by the rotor blades. Zooming in shows more clearly how the cut wakes reach the rotor TE faster near the centerline of the passage, distorting its shape. The user notices that for case 9 the stator wake pattern remains visible throughout the outlet, and only mixes out gradually, as opposed to some cases where the mixing is faster, such as case 4. To see if this observation correlates with the calculated isentropic efficiencies the contours can be arranged from left (case 4) to right (case 9) by the perceived mixing out of the wakes. Figure 10F shows the Spearman rank correlation coefficient scores between individual metadata variables and the on-screen position of contour images. The calculated scores are drawn as text labels on a 2D axes. The position of $alpha_{rotorout}$ along the y-axis shows that the arrangement of contours in the vertical direction is not correlated with this parameter. On the other hand, eff_{poly} seems to be reasonably correlated with the horizontal arrangement. This is insufficient to claim that the patterns observed in the wake *cause* improved efficiency, but the correlation is an interesting hint. The correlation score labels can be clicked to arrange the contours by a particular variable. The direction in which the contours will be arranged (x or y) is the one with the currently larger SRCC score. For example, by clicking on eff_{poly} the contours are arranged in the horizontal direction by their polytropic efficiency while their y position remains unchanged.

To continue investigating the correlation between observed wake patterns and efficiency, the user can group a subset of designs with more pronounced wake patterns. A lasso selection is initiated by pressing the left mouse button when over the whitespace and dragging. After a suitable area is selected the lasso is closed by releasing the left mouse button. Any contours within the selected space are automatically piled into a group. Additional contours can be added to the pile by dragging and dropping them over the pile. The bookmarks allow the user to switch between the contours shown on top of the pile. The contours can still be played. By entering the group, the exploration is restricted to the members of that particular group. Only the group members are visible to the user, and only these cases are used to calculate the spatial arrangement correlations with the metadata. When within a group a button that allows the user to exit the group appears in the top right.

If the user is satisfied that the interactive connection between the unsteady contour plots and the associated metadata was sufficiently explained by the rotor outlet wake patterns they can select a different feature of interest to analyse. If they are interested to analyse how the entropy is generated along the rotor chords they can open the separate unsteady line plot demo [7].

IX. Summary

The understanding of cause-and-effect relationships is the main goal of aerodynamic ensemble dataset analysis. To gain this understanding, the flow-fields need to be visualised, interpreted, and compared, which requires the engineer's domain knowledge. To accelerate the human-data interactions required to generate understanding, new approaches to visualise and explore large ensemble datasets are proposed.

The paper first presents a conceptual divide & explore approach to structure image-based dataset exploration, and an interactive approach to spatially encode the engineer's domain knowledge have been presented. These two concepts together allow a faster exploration of image-based datasets, and help the practitioner use their domain knowledge in the search for possible cause-and-effect relationships.

The second part of the paper discusses unsteady data visualisation, and ways in which unsteady data can be analysed using the divide & explore, and spatial arrangement approaches. The discussed web-based approach enables the practitioner to analyse several unsteady simulations simultaneously, preserves interactivity, and enables easy sharing. This can be particularly beneficial for whole flow-field ensemble data, which can be cumbersome to transport and visualise en masse. We discuss approaches to prepare the data, and discuss what sizes of flow field domains can currently be handled by the browser.

A typical workflow using the proposed framework is demonstrated for a new open dataset consisting of 69 unsteady CFD simulations of $2D$ turbines is introduced.

Acknowledgments

The authors would like to express their gratitude to James Brind who has made available his turbine geometry creation and evaluation scripts, and assisted in the creation of the example dataset.

References

[1] Slotnick, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., and Mavriplis, D., "CFD vision 2030 study: a path to revolutionary computational aerosciences," , 2014.

[2] Pullan, G., "A Web-Based Database Approach to CFD Post-Processing," *55th AIAA Aerospace Sciences Meeting*, 2017, p. 0814.

[3] Lekschas, F., Zhou, X., Chen, W., Gehlenborg, N., Bach, B., and Pfister, H., "A Generic Framework and Library for Exploration of Small Multiples through Interactive Piling," *IEEE Transactions on Visualization and Computer Graphics*, 2020. https://doi.org/10.1109/TVCG.2020.3028948, URL https://vcg.seas.harvard.edu/publications/a-generic-framework-and-library-for-exploration-of-small-multiples-through-interactive-piling.

[4] Pullan, G., Chuan, T., Wong, D., and Jasik, F., "Enhancing Web-Based CFD Post-Processing using Machine Learning and Augmented Reality," *AIAA Scitech 2019 Forum*, AIAA, 2019, p. 2223. https://doi.org/10.2514/6.2019-2223.

[5] Lekschas, F., Bach, B., Kerpedjiev, P., Gehlenborg, N., and Pfister, H., "HiPiler: Visual Exploration of Large Genome Interaction Matrices with Interactive Small Multiples," *IEEE Transactions on Visualization and Computer Graphics*, Vol. 24, No. 1, 2018, pp. 522–531. https://doi.org/10.1109/TVCG.2017.2745978.

[6] Pullan, G., "Making use of our data," *Journal of the Global Power and Propulsion Society*, Vol. 2021, No. May, 2021, pp. 1–13.

[7] Kotnik, A., https://aljazkotnik.github.io/unsteadyflowensembles/, December 2021.

[8] Lindstrom, P., "Fixed-rate compressed floating-point arrays," *IEEE transactions on visualization and computer graphics*, Vol. 20, No. 12, 2014, pp. 2674–2683.

[9] Brandvik, T., and Pullan, G., "An accelerated 3D Navier–Stokes solver for flows in turbomachines," *ASME J. of Turbomach.*, 2011.

[10] Smith, S., "A simple correlation of turbine efficiency," *The Aeronautical Journal*, Vol. 69, No. 655, 1965, pp. 467–470.