Advances in Engineering Software 173 (2022) 103257

Contents lists available at ScienceDirect

Advances in Engineering Software

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

Research paper

Automated classification of linear bifurcation buckling eigenmodes in thin-walled cylindrical shell structures

Adam J. Sadowski

Department of Civil and Environmental Engineering, Imperial College London, UK

ARTICLE INFO

Keywords:
Shell structures
Buckling
Automated parameter sweeps
Convolutional neural network
Image classification
Chromatic signatures

ABSTRACT

Computational problems in structural engineering are growing ever larger and solutions must increasingly be based on correspondingly large datasets obtained from detailed parametric sweeps. However, the acquisition of computational datasets of useful size is also becoming increasingly unfeasible without extensive use of automation. In computational shell buckling studies, particularly those of thin-walled shells under complex loading conditions, an important qualitative piece of information is the class of buckling mode which reveals the dominant destabilising membrane stress components. Unfortunately, the diversity of geometries that can be encountered in computational shell buckling studies is truly vast, and there is currently no way to rapidly assess the buckling mode without laborious direct human observation of the model output.

This paper presents an automated classification tool for linear bifurcation buckling eigenmodes in cylindrical shells such as those found as wind turbine support towers, chimneys, silos, tanks, piles and pipelines. It is based on a convolutional neural network implemented using the `PyTorch` machine learning framework. The adopted network architecture is based on those widely adopted for image classification and recognition tasks, chosen based on a stratified five-fold cross-validation exercise. The network is trained on a purposefully generated basic dataset of 13,392 linear bifurcation buckling eigenmodes modes encoded as chromatic signatures in `.jpg` images (enhanced to 25,726 by transformations). An example parametric sweep of a cylindrical shell under unsymmetrical wind loading illustrates the performance of the classifier. A `GitHub` repository offers `Python` scripts and instructions on how to download the dataset and trained network.

1. Introduction

Thin-walled metal civil engineering shell structures such as silos, tanks, wind turbine support towers and long tubulars are known to exhibit a particularly rich diversity of possible responses with an ultimate limit state typically governed by buckling. The buckling limit state is widely known to be influenced by the geometric form, material response, relative slenderness, loading and restraint conditions and, in particular, the inevitable systematic and random construction or manufacturing imperfections. A truly vast body of literature has accumulated on the mechanics, testing, analysis and design of shells which the reader may begin to consult in any of the following sources [1–6].

A significant advance in transforming shell buckling research into modern design and construction practice has been made in the past decade through the evolution of the EN 1993-1-6 Eurocode on the strength and stability of shell structures. Though published in 2007 [7] it received an amendment in 2013 [8] and has just now in 2022 undergone significant transformation into a new prEN 1993-1-6 [9] due for publication in the next few years. An important recent innovation has been the very general Reference Resistance Design (RRD) framework invented by Rotter [10] which allows the ultimate limit state buckling resistance of a structural system to be characterised in a compact algebraic form via the abstraction of a 'capacity curve' (Fig. 1). The formulation captures the main mechanics governing the slenderness-dependent buckling resistance through a number of physically meaningful algebraic parameters, which may be established in isolation by purposefully designed computational parametric sweeps. Establishing these important parameters for metal (and even cementitious) shells has been the focus of significant research activity in the past decade [5,11–14], as they are additionally used as a basis for two other major computational design methods used in EN 1993-1-6 (design by LBA-MNA [15] and by GMNIA [4]). A detailed discussion of the framework, together with a rigorous computational strategy to design automated parametric sweeps for the purposes of obtaining high-quality algebraic RRD characterisations of any metal shell system, may be found in the Author's previous open-access publication [6] which readers are warmly invited to consult. The EN 1993-1-6 computational analysis taxonomy (LA, LBA, (G)(M)N(I)A etc.) referred to here is summarised in the Appendix.

E-mail address: a.sadowski@imperial.ac.uk.

https://doi.org/10.1016/j.advengsoft.2022.103257
Received 17 June 2022; Received in revised form 28 July 2022; Accepted 20 August 2022
Available online 7 September 2022

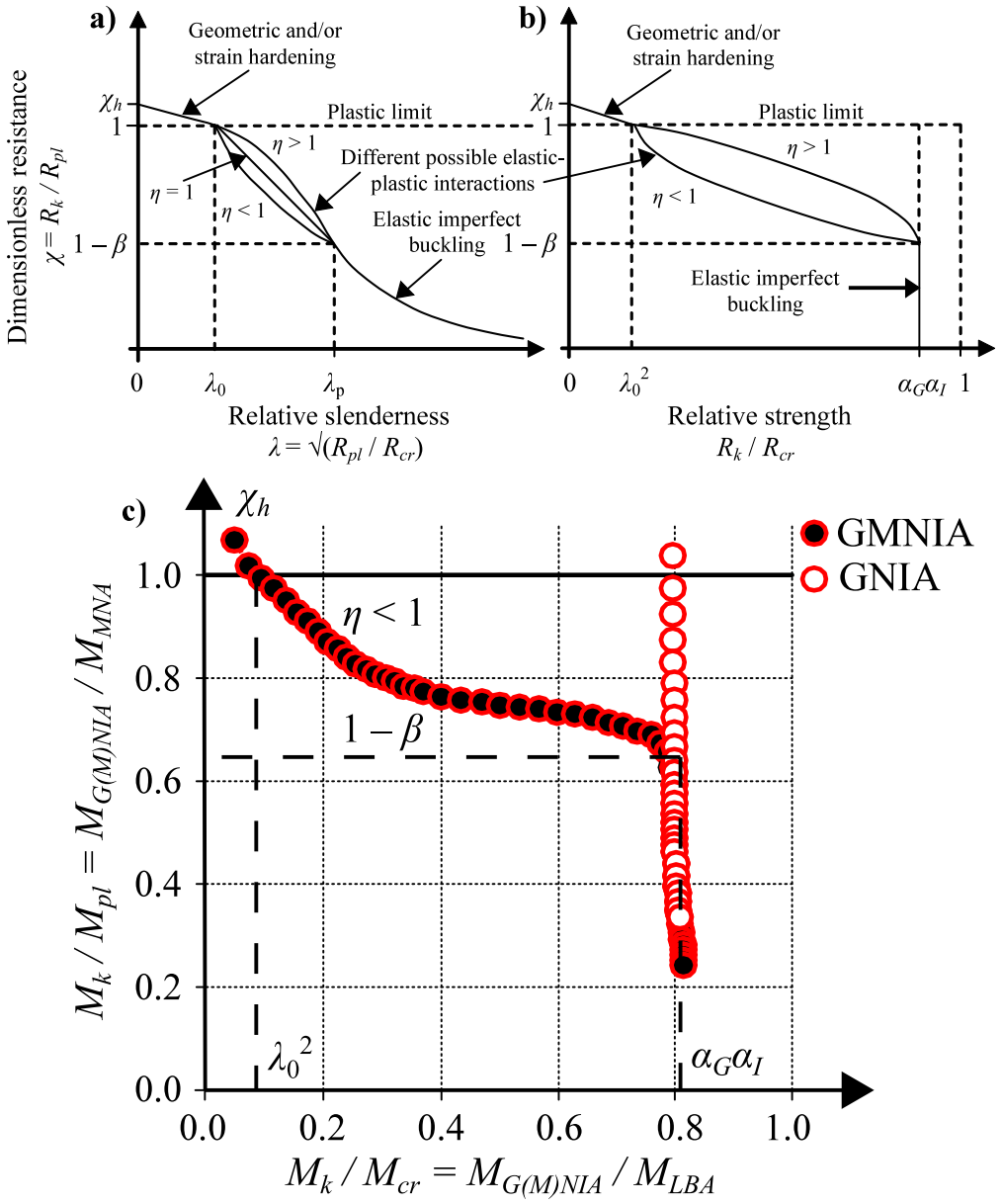

Fig. 1. The capacity curve framework used by Reference Resistance Design (RRD), illustrated by an idealised schematic of the (a) 'traditional' and (b) 'modified' or Rotter forms, and (c) showing actual computed G(M)NIA curves.

The most comprehensive RRD characterisation to date was undertaken by Wang et al. [14] who performed a total of 28,250 nonlinear equilibrium path-tracing MNIAs, GNIAs and GMNIAs on the structural system of cylindrical shells under uniform bending. This characterisation has now been formally adopted as a resistance model in the upcoming prEN 1993-1-6 [9], and others are under development. The system of a cylinder under uniform bending is 'simple' in the sense that the nominal pre-buckling stress state is obvious and there is no ambiguity as to the location of failure by buckling or nonlinear collapse. Yet if a relatively 'simple' system like this required such an intensive parametric sweep in order to be fully characterised for the purposes of design, more complex systems with larger parameter hyperspaces (due to more complex geometry, material, loading or boundary condition combinations) will require significantly more computational attention.

Central to a correct RRD characterisation of any new system is an understanding of the mechanics of the response which is used to interrogate and verify the computed capacity curves, a task that is unachievable without considering information ulterior to the computed buckling resistance such as the critical buckling mode. For example, the relative simplicity of the nominal stress state in cylinders under uniform bending implies that the critical buckling mode will always be a well-defined localised meridional compression buckle concentrated along the most compressed meridian. The same cannot be said *a priori* for more complex systems for which the destabilising membrane stress component is far from obvious, and which with current methods would require direct visual observation of the buckling mode by the analyst for classification purposes. For parametric sweeps aiming to exceed the current record of 28,250 nonlinear analyses performed for the 'simple' system of a cylinder under uniform bending, classification of every individual result by a human analyst is not a feasible way forward.

2. Aims and scope

The rich diversity of shell systems and their responses precludes the straightforward creation of an automated buckling mode classifier. This paper proposes to treat the problem as one of image classification and to exploit the deep learning technique of a convolutional neural

network (CNN). In their seminal 1998 paper, LeCun et al. [16] made the case for multi-layer CNNs as powerful general classifiers due to their ability to automatically 'learn' higher-level feature extractors directly from raw data (e.g. vectors of image pixels) where they would have previously had to be hand-crafted through a laborious and error-prone process. In contrast to simpler fully-connected network architectures in wider use prior to then, CNNs additionally achieve some degree of insensitivity to shift, scale and distortion of the extracted features [16] when combined with max-pooling layers [17,18] which made them particularly effective at noisy image recognition tasks.

CNNs had been proposed in the 1970s, but their full potential has only begun to be exploited in recent decades thanks to the ready availability of large datasets necessary for deep learning (with arose in tandem with the development of a high-bandwidth internet) and powerful personal computing machines necessary to perform the linear algebra steps of the backpropagation algorithm (in particular, graphics processing units or other accelerators [19]). The `LeNet` family of CNNs created by LeCun et al. in the late 1980s were the first to be used successfully and commercially for handwritten digit recognition [20], and CNNs in increasingly sophisticated form have since been trained for image classification [21,22], human action recognition [23] and language processing [24], among many other applications. Further relevant details of CNNs are introduced where appropriate in this paper.

This paper develops an automated classification tool for linear bifurcation buckling eigenmodes based on a convolutional neural network implemented in the `PyTorch` machine learning framework [25,26]. The technique is currently limited to linear buckling eigenmodes as these may be computed very efficiently by an LBA (see the Appendix for definition) and it is possible to rapidly create an arbitrarily large training dataset. Additionally, linear buckling eigenmodes are sufficiently visually similar to nonlinear incremental buckling modes (e.g. compare Figs. 3 and 6 with Figs. 7 and 8 in [12]) that a CNN classifier trained on the former will likely also perform well on the latter. The classifier can then be readily retrained in future once appropriate data has been gathered using transfer learning techniques [27].

A procedure for encoding buckling modes as RGB-channel `.jpg` images is first developed, a necessary step to take advantage of the 'tensor' architecture underpinning the `PyTorch` framework. A basic training dataset of 13,392 buckling eigenmodes across eleven classes is then generated, artificially enhanced to 25,726 during training with additional probabilistic image transformations. A feasible architecture of the CNN is determined in a 'greedy' way by a five-fold stratified cross-validation exercise after which the final network is trained for release using the enhanced training dataset. The performance of the CNN classifier is then illustrated on a cylindrical shell system of industrial importance, and avenues for future development are proposed.

The methodology presented in what follows has been developed with the aid of the ABAQUS finite element software because of its prevalence in computational shell buckling studies and its integration with a `Python 2` scripting language interpreter, a crucial consideration for the automation of otherwise laborious repetitive tasks. However, there is no reason an analogous procedure could not be established using other popular solvers. All scripts referenced in what follows may be downloaded from the accompanying `GitHub` repository (see link at the end).

3. Encoding linear buckling eigenmodes as images

The proposed procedure for encoding linear buckling eigenmode deformations originally stored as nodal displacements on a potentially irregular finite element mesh in the ABAQUS output `.odb` file into a regularised grid of 3 × 8-bit RGB pixel triplets stored in an image file such as `.jpg` is an indirect one consisting of two distinct steps. To manipulate the ABAQUS `.odb` files it is necessary to use its built-in `Python` interpreter which as of 2022 is at version 2.7.15. Although it now supports many of the popular `Python` libraries such as `numpy` or `scipy`, it was apparently never intended as a general-purpose `Python` interpreter and it does not appear to be possible to install additional libraries to add functionality. Consequently, it is currently not possible to directly import the `PIL` or `opencv` imaging libraries via the ABAQUS `Python` interpreter, necessary to create image files, nor of course the `PyTorch` machine learning framework which requires a `Python` interpreter at version 3.x.

3.1. First step: intermediate binary representation

In the first step, the nodal buckling deformations of the curved shell wall stored in the `.odb` file are transformed into a planar 'canvas' via the ABAQUS `Python 2` interpreter and written to an intermediate file, most efficiently in binary format (implemented here as a `.bin` file). The transformation is illustrated in Fig. 2, where 1,2,3 is a generic right-handed Cartesian coordinate system for nodal coordinates and degrees of freedom with axis 3 assumed here to always be the axis of revolution. For a 3D shell part generated by ABAQUS as a 'shell of revolution', $1 = z$, $2 = x$ and $3 = y$ with x, y, z representing a Cartesian coordinate system. Where the part is generated as a 'shell of extrusion', $1 = x$, $2 = y$ and $3 = z$. An additional valid possibility is $1 = y$, $2 = z$ and $3 = x$.

The planar canvas is defined in an s, m plane which at its simplest can be defined using only the original nodal coordinates of the undeformed shell wall. The out-of-plane buckling displacements are then placed on a w axis perpendicular to the s, m plane. The transformation to be performed at each node position is

$$s = \left(\sqrt{c_1^2 + c_2^2}\right) \cdot \tan^{-1}\left(\frac{c_2}{c_1}\right) \tag{1}$$

$$m = c_3 \tag{2}$$

$$w = \sqrt{(c_1 + u_1)^2 + (c_2 + u_2)^2} - \sqrt{c_1^2 + c_2^2} \tag{3}$$

where c_i are the original nodal coordinates of the undeformed shell in any of $i = 1, 2, 3$ (ABAQUS `COORD` field output), u_i are the corresponding nodal buckling displacements (`U` field output) and $\tan^{-1}()$ must be implemented as the quadrant-correcting `atan2()` function. It may be noted that the (usually negligible) in-plane component of the buckling mode displacements u_3 is omitted from the transformation which currently considers only the more dominant out-of-plane displacements. The illustration in Fig. 2 illustrates the procedure on a single cylindrical shell segment.

During this transformation, the minimum and maximum of each of the s, m, w axes are identified to determine a 'bounding box' on the canvas needed for the second step of the encoding. The byte structure of the `.bin` file generated at this stage is summarised in Table 1. This transformation step is implemented in the supporting `APy2_ABQDeformations_to_Binary.py`.

Table 1
Byte structure of the `.bin` file encoding a cylindrical shell linear buckling eigenmode on an intermediate planar canvas.

Type	Bytes	Value
`unsigned int`	4	N_{nodes} (no. of nodes in the shell element mesh)
`double`	8	s_{min}
`double`	8	s_{max}
`double`	8	m_{min}
`double`	8	m_{max}
`double`	8	$max(\lvert w_{min}\rvert, w_{max})$
`double`	8	s
`double`	8	m
`double`	8	w

} × N_{nodes} times

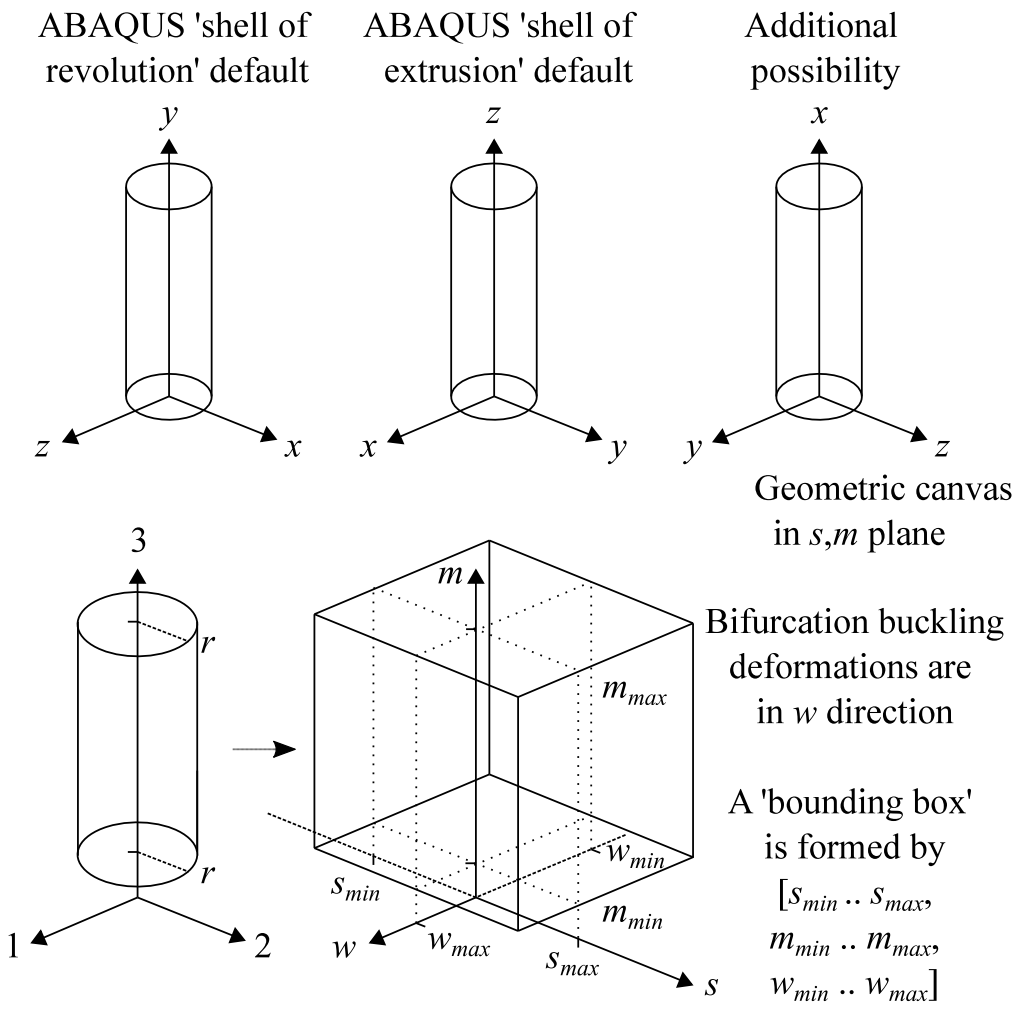

Fig. 2. Illustration of coordinate systems and the first step of the transformation from nodal displacements on the curved shell to an intermediate planar canvas.

3.2. Second step: generation of chromatic signatures

The binary `.bin` file may now read by a `Python 3` interpreter to transform the continuous planar representation of the buckling deformations in s, m, w space into a regularised grid of discrete 3×8-bit RGB pixel triplets that are then written to an image file (here a `.jpg`) with the aid of the `opencv` imaging library. Crucially, the `Python 3` interpreter may be invoked dynamically by the ABAQUS `Python 2` interpreter via an indirect system call using `os.command()` with the name of the binary file passed as a command-line argument. The transformation is illustrated in Fig. 3 where zero-based indexing is used to determine the pixel position $P(Row, Col)$.

First, a variable named MPD is set specifying the number of pixels along the longest dimension of the canvas, taken as 1000 to accommodate the fact that the pixel positions are related to nodal positions and thus to the resolution of the finite element mesh which can have many hundreds of elements along a dimension. Deformation information is then only lost where a mesh dimension has more than 1000 linear elements, thought to be a reasonable upper bound, or where nodal spacing is highly irregular. The aspect ratio of the canvas is then calculated as

$$AR = \frac{m_{max} - m_{min}}{s_{max} - s_{min}} \tag{4}$$

where $AR > 1$ represents a slender canvas where the number of pixel rows n_R and pixel columns n_C are respectively given by

$$n_R = MPD \tag{5}$$

$$n_C = \left\lceil \frac{MPD}{AR} \right\rceil \tag{6}$$

such that $n_R > n_C$. A scale factor

$$SF = \frac{MPD}{m_{max} - m_{min}} \tag{7}$$

must also be defined. Similarly, an aspect ratio $AR \leq 1$ represents a squat canvas where

$$n_R = \lceil MPD \times AR \rceil \tag{8}$$

$$n_C = MPD \tag{9}$$

$$SF = \frac{MPD}{s_{max} - s_{min}} \tag{10}$$

such that $n_R \leq n_C$. The bounding box information is now used to translate the canvas so that all data is situated in the $s \geq 0$ and $m \geq 0$ quadrant

$$m' = (m - m_{min}) \times SF \tag{11}$$

$$s' = (s - s_{min}) \times SF \tag{12}$$

which implies $0 \leq m' \leq MPD$ and $0 \leq s' \leq \frac{MPD}{AR} < MPD$ for a slender canvas, and $0 \leq m' \leq MPD \times AR < MPD$ and $0 \leq s' \leq MPD$ for a squat canvas. The w surface defined on continuous (s', m') canvas coordinates, which hitherto did *not* yet need to follow a regular grid, may now be

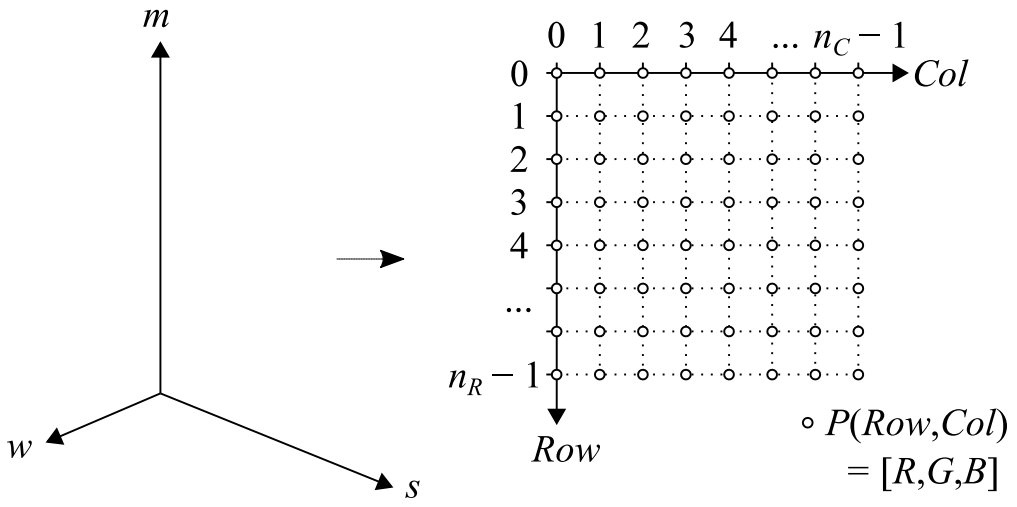

Fig. 3. Illustration of the second step of the transformation from a continuous geometric canvas in s, m, w space to discrete pixel array.

mapped to a discretised W surface defined on a regular rectangular array of discrete pixel coordinates (Row, Col) using the `griddata` function imported from the `scipy.interpolate` library and saved to a regular `.jpg` image file. The individual pixel values area defined by a RGB triplet with each 8-bit colour channel computed as

$$R = \left\lceil \frac{1}{2}(1 + \mathrm{sgn}(W_{norm})) \times |W_{norm}| \times 255 \right\rceil \tag{13}$$

$$G = 0 \tag{14}$$

$$B = \left\lceil \frac{1}{2}(1 - \mathrm{sgn}(W_{norm})) \times |W_{norm}| \times 255 \right\rceil \tag{15}$$

where $W_{norm} = \frac{W}{max(|w_{min}|, w_{max})}$ is the normalised out-of-plane buckling displacement. These equations are designed to produce a 'chromatic signature' characterising the eigenmode: outward buckling displacements (acting away from the original wall position, $W_{norm} > 0$) produce only red pixels of varying intensity while inward buckling displacements produce only blue pixels ($W_{norm} < 0$), with an $(R, G, B) = (0, 0, 0)$ triplet producing a black pixel corresponding to no local buckling deformation ($W_{norm} = 0$). A selection of different linear buckling eigenmodes encoded as images in this manner are presented in the next section. This transformation step is implemented in `Py3_Binary_to_Image.py`.

In the above, the G channel is currently unused and will be ignored by the input layer of the subsequent neural network (with the choice of not using the G channel specifically being entirely arbitrary). However, it is free to be used in future formulations to store additional information, for example if the encoding is extended to include the in-plane buckling displacement component u_3, to treat non-cylindrical shells of revolution or to handle other more complex conditions. If more channels are necessary still, the encoding can be modified to adopt the 4-channel RGBA or CMYK formats, or channels can be distributed across multiple image files. The structured storage of encoded buckling modes in an image-friendly format, as opposed to simple arrays, permits rapid visualisation using widely-available photo editing software and aids in interpretation.

4. Classification of linear buckling eigenmodes

In establishing classification categories, only those eigenmodes are considered that are likely to be most representative of practically ubiquitous 'medium-length' civil engineering cylindrical shells with restrained radial displacements at both edges (BC1r, BC1f, BC2r or BC2f type boundary conditions [9], see also the Appendix). The critical eigenmodes of such cylinders are dominated by out-of-plane deformations of the shell midsurface with negligible in-plane deformation and no net displacement of the centroidal axis. A cylinder with a free unrestrained edge (BC3r or BC3f) may buckle into eigenmodes which bear little resemblance to those in a cylinder with both edges radially restrained even when under the same nominal membrane stress state, which may prevent the CNN layers from learning meaningful feature maps for certain classes. Additionally, very short or very long cylinders lead to chromatic signatures whose aspect ratios are likely too distorted for meaningful image classification. The class names used in what follows are the same as the instance labels used during training for which alphabetical ordering is important and which determines their zero-indexed position in the CNN's output layer. They are not necessarily introduced in alphabetical order for reasons of narrative.

4.1. Meridional membrane compression

Medium-length cylinders under uniform meridional compression exhibit a particularly rich diversity of possible eigenmodes obtained by LBA. The shapes of possible eigenmodes are governed by the Koiter circle [28–30] which establishes a relationship between the critical numbers of n circumferential full waves and m meridional half-waves of a doubly-periodic eigenmode in terms of the cylinder radius r, thickness t, length L, Young's modulus E and Poisson ratio ν. The most visually characteristic types include the 'axisymmetric' mode where $n = 0$ (`MerCompAxi` in Fig. 4), the 'chequerboard' mode where n and m are such that the eigenmode half-wavelengths along the two orthogonal axes are comparable (`MerCompChequer`), and then any of the nearly inexhaustible number of combinations of n and m which are permitted by the Koiter circle (`MerCompOtherKoiter`). Further visual illustrations of these modes may be found in e.g. [31,32].

Short cylinders under uniform meridional compression are characterised by axisymmetric modes ($n = 0$) with a single meridional half-wave ($m = 1$) [30,33] which are included in the present classifier under `MerCompAxi`. However, very long cylinders buckle as Euler columns which are characterised by a net displacement of the centroidal axis and are presently not included in the classifier. Non-uniform meridional compression such as that caused by concentrated edge loading [34] or net centroidal bending moments [5,35] creates localised eigenmodes (`MerCompLocal`) whose individual half-wavelengths are similarly governed by the Koiter circle.

4.2. Circumferential membrane compression

Linear buckling eigenmodes in medium-length cylinders under uniform circumferential compression are characterised by large global

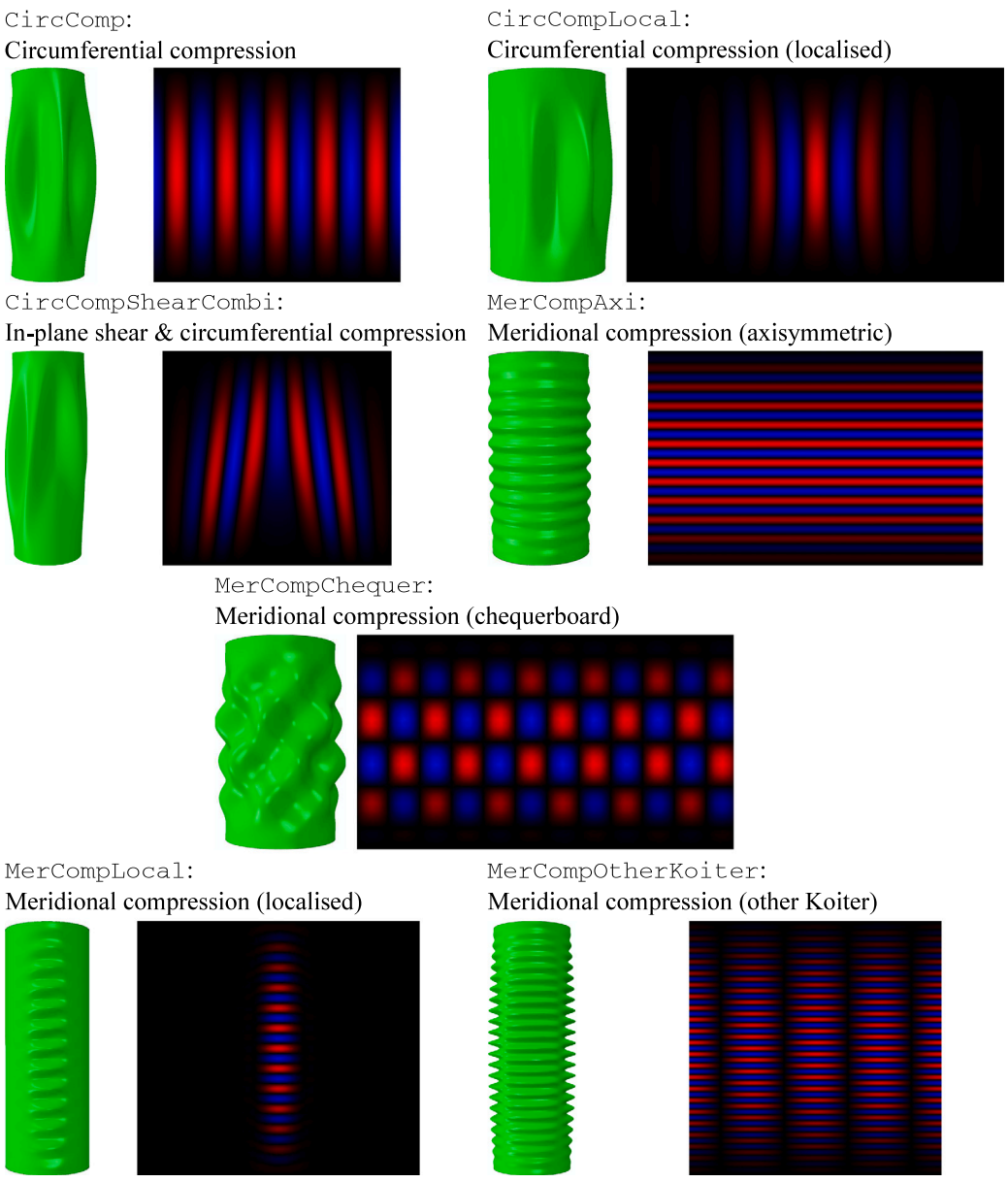

Fig. 4. Examples of a first group of buckling eigenmode classes involving meridional and circumferential compression for medium-length cylindrical shells together with their chromatic signatures.

buckles with a length and slenderness-dependent number of n circumferential full-waves but only a single meridional half-wave ($m = 1$) regardless of length [36] (`CircComp` in Fig. 4). Non-uniform circumferential compression leads to similarly-shaped eigenmodes, though with deformations localised on the side of highest circumferential compression (`CircCompLocal`) [37,38].

4.3. In-plane membrane shear

Uniform in-plane membrane shear can be caused by torsional moments applied about the centroidal axis causing inclined global eigenmodes spanning the full cylinder length (`ShearTorsion` in Fig. 5), while similarly full-length eigenmodes arise under non-uniform membrane shear distributions caused by a global transverse force (`ShearTransverse`). In-plane membrane shear caused, for example, by a combined global torsion and global transverse shear loading creates localised eigenmodes (`ShearLocal`).

4.4. Combinations

Loading that causes multiaxial states of destabilising membrane stresses poses a particular challenge for classification, owing to the many possible loading combinations generating many possible eigenmode geometries. One combination is that of in-plane meridional compression and shear such as that caused by a global transverse shear force on the cylinder. Longer cylinders loaded transversely as cantilevers are dominated by local meridional compression near one boundary and exhibit eigenmodes of class `MerCompLocal` (Fig. 4), while shorter ones exhibit shear eigenmodes of class `ShearTransverse` (Fig. 5). A transitional length domain exists where the eigenmode exhibits characteristic features of both [12] (`MerCompShearCombi` in Fig. 5).

Other combinations are more difficult to reliably identify based on visual assessment alone, even for human observers. For example, eigenmodes caused by both in-plane shear (owing to torsion or a transverse shear force) and circumferential compression are difficult to distinguish from those caused by shear alone since they all exhibit

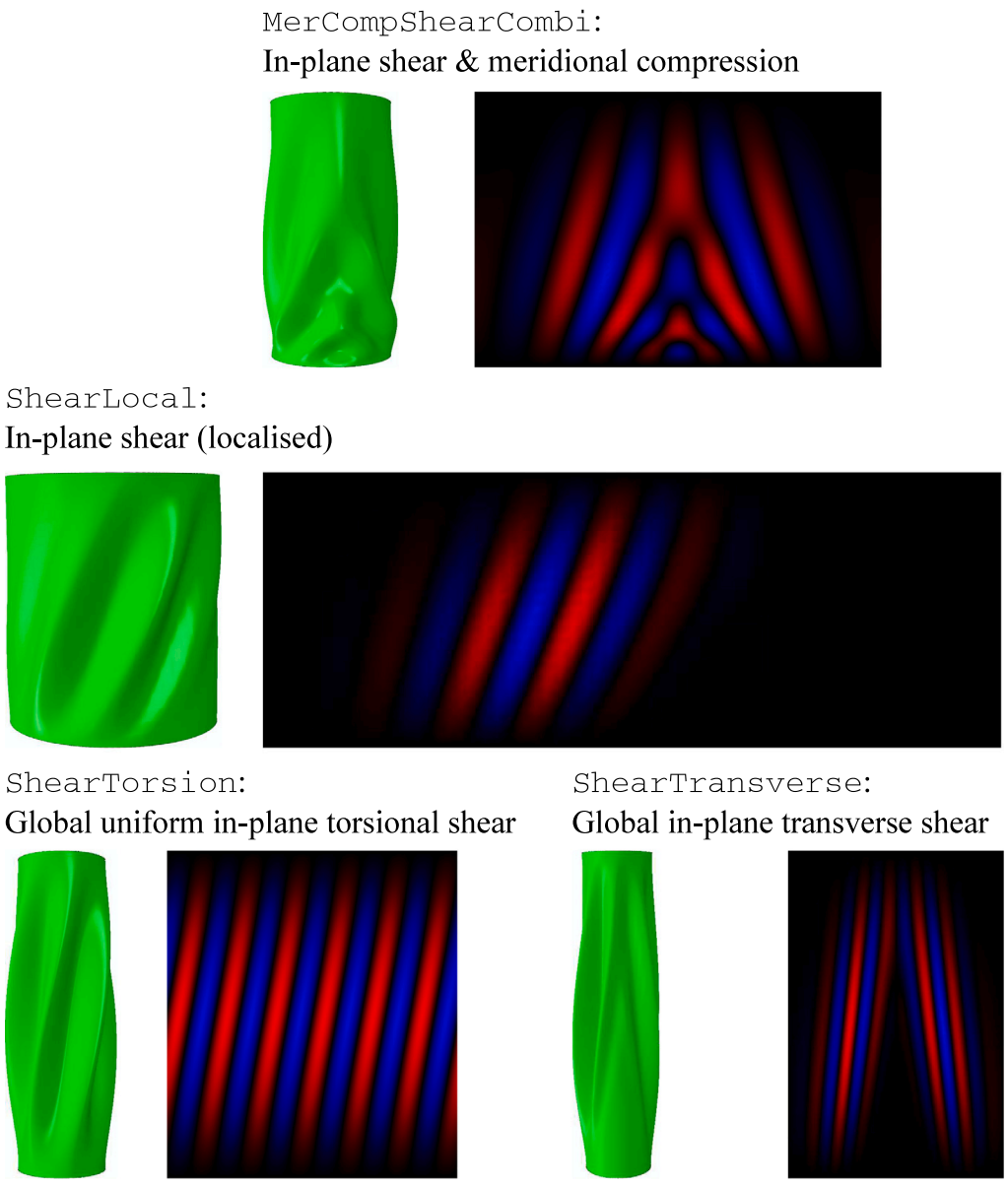

Fig. 5. Examples of a second group of buckling eigenmode classes involving in-plane shear for medium-length cylindrical shells together with their chromatic signatures.

characteristically inclined buckles (compare `CircCompShearCombi` in Fig. 4 with `ShearTransverse` in Fig. 5). It is for this reason that a class combining meridional and circumferential compression, or all three destabilising membrane stress components, has not been created at this time.

5. Training of the convolutional neural network

In what follows, reference is made to the dimensionless length groups

$$\omega = \frac{L}{\sqrt{rt}} \tag{16}$$

which governs the behaviour of short cylinders and

$$\Omega = \frac{L}{r}\sqrt{\frac{t}{r}} = \omega\frac{t}{r} \tag{17}$$

which governs the behaviour of long cylinders. Together these have been widely used to characterise the length-dependent buckling response of cylindrical shells under various loads [5,6,33,35].

5.1. Generation of basic training dataset

A generic cylindrical shell system was devised to generate a potentially inexhaustible supply of prototype eigenmodes from LBAs for the purposes of establishing a basic training dataset of chromatic signatures (Fig. 6). Implemented in `APy2_Generate_Training_Set.py` to be called from the ABAQUS `Python 2` interpreter, the system consists of a complete cylindrical shell with one edge fully clamped (BC1r) and the other kinematically constrained to a centroidal reference point (effectively BC2r). The surface of the cylinder is loaded by an inwards-acting 'external' pressure p to contribute uniform circumferential compression. The reference point is additionally loaded by:

- A force F acting along the centroidal axis to contribute uniform meridional compression;
- A force Q acting transversely to the cylinder axis to contribute a meridionally and circumferentially-varying meridional compression as well as circumferentially-varying in-plane shear;
- A moment M acting in the direction of Q to contribute circumferentially-varying meridional compression;

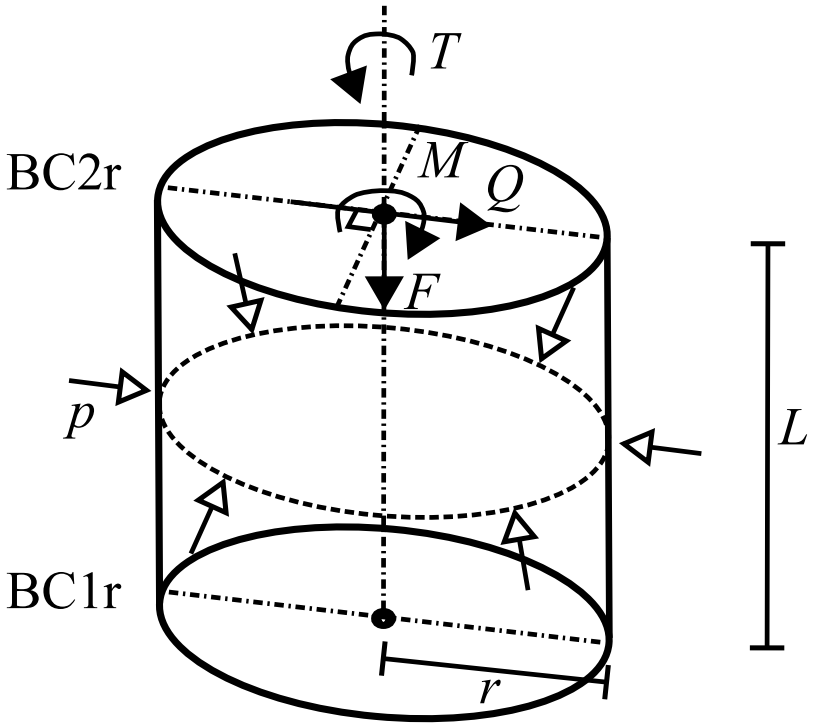

Fig. 6. Illustration of a generical cylindrical shell system under a number of basic loads.

- A moment T acting about the centroidal axis to contribute uniform in-plane shear.

A set of up to 31 LBAs may be performed for every investigated combination of r, t and L. First, LBAs can be performed under each of the ${}_5C_1 = 5$ (where ${}_nC_r = n!/(r!(n-r))$ is the combination formula) basic loads p, F, Q, M and T individually (each initially set to a unit magnitude) to compute their critical eigenvalues. The magnitudes of each of the basic loads can then be set to the respective critical positive eigenvalues, and LBAs performed for all ${}_5C_2 = 10$ combinations of pairs of loads, then all ${}_5C_3 = 10$ combinations of trios of loads, then all ${}_5C_4 = 5$ combinations of quartets of loads and finally the ${}_5C_5 = 1$ possible way of applying them all together. Each combined-load LBA can thus be performed under loading where each basic load is at its most individually critical value, encouraging wherever possible the generation of 'combined' eigenmodes where two or more load components are simultaneously critical.

The scientific literature on shell buckling was consulted to design purposeful geometric ranges to ensure that the basic training dataset contained a sufficient number of eigenmodes for each of the eleven classes considered:

- Cylinders with an edge rotational restraint and $\omega < 5$ are considered 'short' and may be used to generate 'axisymmetric' eigenmodes with one meridional half-wave ($m = 1$) of class `MerCompAxi` (Fig. 4) when under uniform meridional compression [39] or class `MerCompLocal` when under uniform bending [35].
- The generic system employed here exhibits an effective Euler column buckling length of $2L$ which can be shown to be critical for $\Omega > 1.43$ [9,13]. For the generation of Koiter-like eigenmodes (`MerCompAxi`, `MerCompChequer`, `MerCompOtherKoiter`), cylinders under uniform meridional compression may be restricted to the 'medium-length' domain $0.1 < \Omega < 1$. This range is also appropriate for the generation of shear modes (`ShearLocal`, `ShearTorsion` and `ShearTransverse`; Fig. 5) and uniform circumferential compression modes (`CircComp`; Fig. 4).
- Cylinders under transverse shear loading with $0.45 < \Omega < 0.65$ buckle into intricate eigenmodes with combined features of both in-plane shear and meridional compression (`MerCompShearCombi`; Fig. 5; [12]). Shorter cylinders with $\Omega < 0.45$ exhibit an in-plane shear eigenmode of type `ShearTransverse` (Fig. 5) while longer ones with $\Omega > 0.65$ exhibit a localised meridional compression eigenmode of type `MerCompLocal` (Fig. 4).

For each LBA, the eigenmodes corresponding to the lowest five positive eigenvalues were automatically processed into chromatic signatures and saved as `.jpg` files (giving up to $5 \times 31 = 155$ per $\{r, t, L\}$ triplet, though many were discarded to remove overly similar or unrealistic entries) using the procedure described in Section 3. Even higher-order eigenmodes were not retained so as not to bias the training dataset towards increasingly exotic (if mathematically permissible) eigenmodes. It is a fortuitous consequence of the present focus on the generation of buckling eigenmode shapes for the purposes of training a neural network for image recognition that the modelling procedures may be significantly laxer than they would otherwise have been had the focus been on the careful computation of safety-critical characteristic buckling loads. The former is known to require exceptional care and expertise for correct modelling which the reader may find evidence of in e.g. [32,40,41]. Similarly, the actual values of elastic modulus or Poisson ratio used in this process are not important, though values for isotropic steel were adopted out of tradition ($E = 200$ GPa and $\nu = 0.3$), as well as a unit wall thickness. LBAs were performed until a satisfactory number of human-labelled chromatic signatures were gathered for each of the eleven classes defined here, resulting in a labelled (but class-imbalanced) basic dataset of 13,392 `.jpg` files stored across eleven directories at approximately 850 MB in size.

5.2. Dataset enhancement through image transformation

In the `Pytorch` framework, data instances are prepared for the training pipeline with the aid of a `Dataset` object which establishes the file directory structure, loads the contents of `.jpg` image files into

tensors, optionally performs transformations on the tensors and finally returns tensor-label pairs corresponding to any given index $i \in [1, n]$ where $n = 13{,}392$ is the nominal size of the basic dataset. These pairs are requested by a `DataLoader` object which conveniently groups them into batches for training according to a sampling scheme defined by an optional `Sampler` object. After the chromatic signatures are loaded from `.jpg` files into tensors by the custom `Dataset` object (see `Py3_LBANet_Dataset.py`), the following transformations are potentially applied depending on the eigenmode class to artificially enhance the dataset (inspired by similar techniques applied to the MNIST handwritten digit benchmarking dataset [19] and others):

- `SquarePadding`: Every image tensor of initial size $3 \times H \times W$ (where H and W respectively are the image height and width in pixels) is padded with black pixels $(R, G, B) = (0, 0, 0)$ along the appropriate dimension to achieve a square image tensor where $H = W = MPD = 1000$.
- `SplitPad`: If the image tensor represents an eigenmode that is symmetric about the mid-width (but neither axisymmetric nor periodic), either the left-hand side or the right-hand side of the tensor (i.e. along the W dimension) is chosen at random and translated to the other half, with its original region overwritten by black pixels. This is to encourage the network to recognise the eigenmode class even if circumferential symmetry conditions are used during modelling and only half of the chromatic signature is available.
- `ResizeMove`: If the image tensor represents a 'localised' eigenmode (i.e. one with `Local` in the class name), the image is resized to one of an equally probable 10%, 20%, 25%, 50% or 100% (no change) of its original 1000×1000 pixel dimension and translated to a random position on an otherwise black pixel tensor. This is to encourage the network to recognise local eigenmodes regardless of their position on the shell wall.
- `Flip`: For every image tensor, there is a probability that it will be additionally flipped vertically (along the H dimension with $p = 0.5$) and perhaps also horizontally (along the W dimension with $p = 0.5$). Most classes are unaffected by this transformation, which is applied primarily to encourage the network to recognise shear-related eigenmodes regardless of loading direction.

Classes requiring the `SplitPad` or `ResizeMove` transformations were weighted by a factor of three over other classes to ensure that a sufficient number of their eigenmodes are sampled during training. For each weighted class to overall have an equal probability of $\frac{1}{11}$ of being sampled from, every data instance was assigned an individual class-specific 'balanced weight' w_c obtained as $\frac{1}{11}$ divided by the enhanced class size (Table 2). This was then supplied to a `WeightedRandomSampler` object used by the `DataLoader` object which sampled an artificially enhanced and probabilistically class-balanced dataset of 25,726 instances with replacement from the original class-imbalanced basic dataset of 13,392 instances.

5.3. Network design and cross-validation

Four trial CNN architectures were implemented in the `PyTorch` framework. Each network consists of a 2-channel 1000×1000 input layer, the unused G channel (Eqs. (13)–(15)) being first stripped out of the incoming image tensor. The first layer is always a convolutional layer which trains a series of m square kernels of size k^2 (here either 10^2 or 20^2 pixels in size) which map the 2×1000^2 input layer into m features maps of size $(1000-(k-1))^2$ assuming a unit stride. This is then followed by a max-pooling layer which passes a square kernel of size w^2 (here either 2^2 or 4^2) with unit stride over each of the m feature maps to downsample its size to $(\lfloor \frac{1}{w}(1000-(k-1)) \rfloor)^2$ by taking the maximum of all inputs in its w^2 field. Subsequent convolutional-maxpool layer pairs work on the same principle, and up to four such pairs with varying settings were investigated. These layer pairs act as automated feature extractors from the raw input data, whose outputs are then passed to a number of fully-connected layers which act as the classifier of these features [16].

Rectified linear units (which have an activation function $max(0, z)$ where z is the weighted input to the unit) were used for both convolutional (but not max-pool) and fully-connected layers. The `ReLU` activation function is variously suggested to perform well for image classification tasks as it does not saturate (constant gradient for $z > 0$) and cannot produce negative outputs (since pixels cannot have negative RGB values) [42]. Training used a stochastic gradient descent optimiser on a cross-entropy loss function with a learning rate of 10^{-3}, a batch size of 32 and no momentum. These hyperparameters were determined heuristically in a 'greedy' way on preliminary investigations given that it is not feasible to exhaustively explore the performance of all possible values with all possible network architectures. All training was done on an Nvidia Jetson AGX Xavier Development Kit, an embedded Linux platform with a hardware and software stack dedicated to machine learning.

Five-fold stratified cross-validation was used on the basic dataset to determine an 'optimal' network architecture, based on the `StratifiedKFold` capability of `sklearn` which preserves the class proportions across each of the folds. Here, every distinct fifth of the 13,392-strong basic dataset was in turn held out as a 'test' dataset with the trial architecture trained for 20 epochs on the remaining four-fifths. Illustrated in Fig. 7, the statistics of the classification accuracies across all five folds are compared for each architecture evaluated on either the training or test datasets. With an increasing number of convolutional/max-pool layer pairs, the network gains the ability to abstract more and higher-level spatial features from the raw pixel data, while additional fully-connected layers enable the network to classify these features more effectively. Architectures 1 to 3 suffer from underfitting and do not yet adequately classify the image data, although accuracy improves with additional convolutional/max-pool layers which fortuitously also decrease the number of trainable parameters in the network, showing the advantage of convolution kernel with shared weights over wide classical fully-connected layers. Importantly, there is only a small difference in classification accuracy between the training and test datasets, suggesting that the trained networks are good at generalising to unseen data and are unlikely to suffer from overfitting. For this reason, and because of the probabilistic image transformations applied to greatly enhance the dataset during final training, it did not at this time seem necessary to employ regularisation techniques such as dropout layers [43], early termination or weight decay [42]. The training code is implemented in `Py3_LBANet_Train.py`, with the X'th trialled architecture importable from the corresponding `Py3_LBAnet_ArchX.py` file. A summary of this exercise performed on four trial architectures is reported in Table 3.

5.4. Training `LBANet`

The optimal network architecture thus determined is illustrated in Fig. 8, here termed `LBANet` in homage to the `LeNet` family of convolutional neural networks on which it is directly based [16,20]. Its state dictionary contains just over 150 million trainable parameters and occupies about 600 MB. It was trained on the 25,726-strong enhanced dataset, with the best-performing network retained out of 80 epochs achieving a human-equivalent classification accuracy of 99.74% on the enhanced dataset (66 misclassifications) and 99.44% on the basic dataset (75 misclassifications). The misclassifications were inspected and determined to be largely benign, relating to eigenmodes that could plausibly have been humanly classified as determined by the CNN based on the destabilising membrane stress conditions present, the misclassification reflecting the occasional ambiguity in the labelling process. The current record of 99.79% classification accuracy on the

Table 2
Summary of the basic and enhanced datasets, with different options for individual instance weights.

Output layer position[a]	Class name	Primary transformation set[b]	Basic size	Enhanced size n_c	Basic individual weight	Enhanced individual weight	Balanced individual weight w_c
0	`CircComp`	{}	1,921	(×1) 1,921	7.47×10^{-5}	3.89×10^{-5}	4.73×10^{-5}
1	`CircCompLocal`	{ResizeMove,SplitPad}	1,005	(×3) 3,015	7.47×10^{-5}	3.89×10^{-5}	3.02×10^{-5}
2	`CircCompShearCombi`	{Flip,SplitPad}	724	(×3) 2,172	7.47×10^{-5}	3.89×10^{-5}	4.19×10^{-5}
3	`MerCompAxi`	{}	1,008	(×1) 1,008	7.47×10^{-5}	3.89×10^{-5}	9.02×10^{-5}
4	`MerCompChequer`	{}	1,493	(×1) 1,493	7.47×10^{-5}	3.89×10^{-5}	6.09×10^{-5}
5	`MerCompLocal`	{ResizeMove, SplitPad}	2,022	(×3) 6,066	7.47×10^{-5}	3.89×10^{-5}	1.50×10^{-5}
6	`MerCompOtherKoiter`	{}	1,701	(×1) 1,701	7.47×10^{-5}	3.89×10^{-5}	5.34×10^{-5}
7	`MerCompShearCombi`	{Flip,SplitPad}	1,031	(×3) 3,093	7.47×10^{-5}	3.89×10^{-5}	2.94×10^{-5}
8	`ShearLocal`	{Flip,ResizeMove}	1,005	(×3) 3,015	7.47×10^{-5}	3.89×10^{-5}	3.02×10^{-5}
9	`ShearTorsion`	{Flip}	1,102	(×1) 1,102	7.47×10^{-5}	3.89×10^{-5}	8.25×10^{-5}
10	`ShearTransverse`	{Flip,SplitPad}	380	(×3) 1,140	7.47×10^{-5}	3.89×10^{-5}	7.97×10^{-5}
			$\sum = 13{,}392$	$\sum = 25{,}726$	$7.47 \times 10^{-5} \times 13{,}392 = 1$	$3.89 \times 10^{-5} \times 25{,}726 = 1$	$\sum_{c \in classes} (w_c \times n_c) = 1$

[a]Valid at the time of writing. May change if new classes are added in future. Please note the zero-based `Python` indexing.
[b]Where the class is essentially unaffected by the flipping transformations, this is not included in the set.

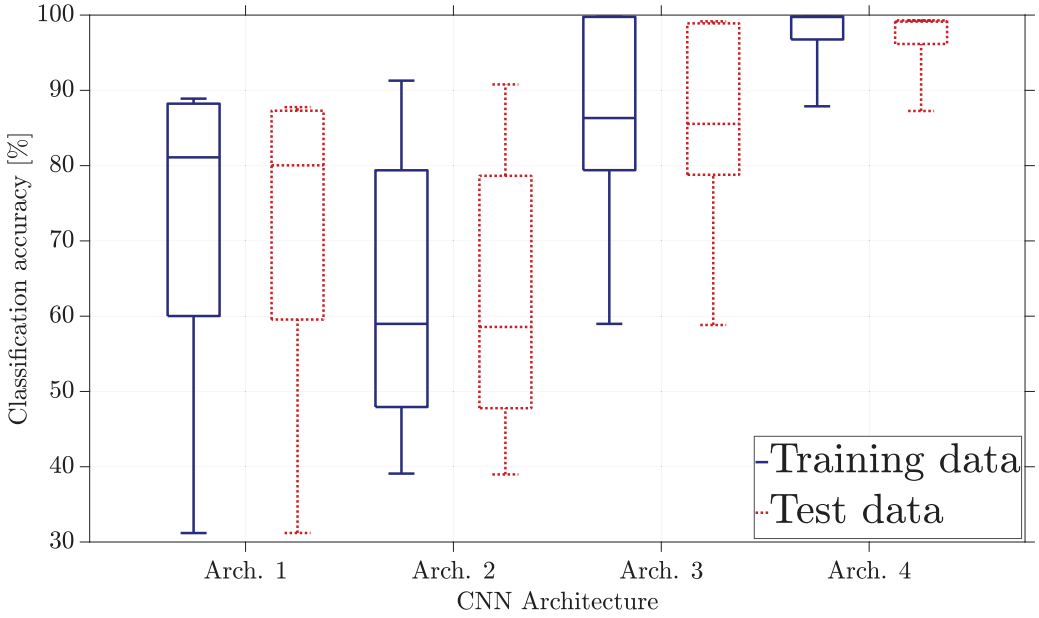

Fig. 7. Classification accuracy of four trial CNN architectures established on the training (blue) and test (red) datasets on the basis of a five-fold stratified cross-validation exercise. The boxplots illustrate the 0th (min.), 25th, 50th (median), 75th and 100th (max.) percentiles as standard.).

Table 3
Summary of the four CNN architectures investigated in this study.

Property	Arch. 1	Arch. 2	Arch. 3	Arch. 4
No. of parameters	615,195,477	262,128,617	421,268,116	150,191,289
Size on disk [GB]	2.46	1.05	1.69	0.60
Mean time per epoch [s]	1171	1765	1513	1567
No. convolutional layers	1	2	3	4
... kernel size & stride [pixels]	20, 1	20, 1	10, 1	10, 1
... feature maps[a]	(10)	(10,20)	(10,20,30)	(10,20,30,50)
... pooling size & stride [pixels]	4, 1	4, 1	2, 1	2, 1
No. fully-connected layers	3	3	4	4
... output features[b]	(1024,512,9)	(1024,512,9)	(1024,512,256,9)	(1024,512,256,9)

[a]e.g. (10,20,30) means the first, second and third convolutional layers have 10, 20 and 30 feature maps respectively.
[b]e.g. (1024,512,9) means the first, second and third fully-connected layers have 1024, 512 and 9 output features respectively.

popular MNIST dataset [44] has similarly likely reached the upper bound on accuracy set by human data labelling ambiguity.

The trained network may be downloaded according to the instructions given in the accompanying `GitHub` repository as a `pickled` serialised dictionary stored in the customary `.pt` file which can be loaded for further training or inference with the `PyTorch` framework or exported to other formats. Inference with `PyTorch` takes $O(10^{-2})$ seconds per signature on a GPU (not including loading the network to GPU which takes $O(10^{0})$ seconds), and $O(10^{-1})$ seconds on a CPU with a slightly faster network loading time than to GPU (but still $O(10^{0})$ seconds). It is stressed that the version of `LBANet` released at the time of writing is still a 'seed' CNN, and its performance on more complex load conditions will be gradually improved with time as more training data is accumulated for specific problems (see next discussion).

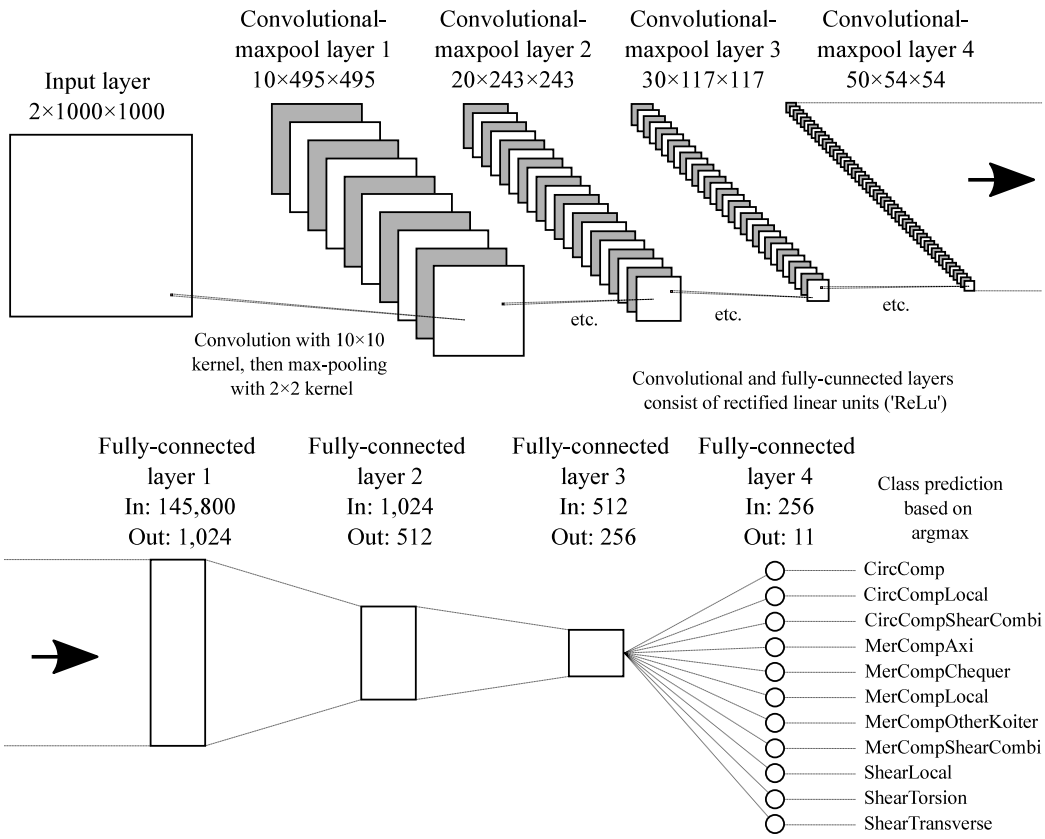

Fig. 8. Architecture of `LBANet`.

6. `LBANet` application to cylindrical shells under wind load

A cylindrical shell under an unsymmetrical wind-induced normal surface pressure represents a challenging application of the buckling eigenmode classification tool to a system of industrial importance, being relevant to the analysis and design of silo and tank structures which are at risk of wind-induced buckling when empty [45]. The LBA eigenmodes are known to vary qualitatively with the cylinder length, with shorter cylinders exhibiting localised circumferential compression eigenmodes that eventually transition to localised shear eigenmodes for longer cylinders which in turn transition to localised meridional compression eigenmodes for more slender cylinders. The wind loading is defined in EN 1993-4-1 Annex C [46] as follows, based on wind tunnel tests on an isolated cylinder with a closed roof as interpreted by Rotter [47]:

$$q(\theta) = q_{cl} C_p(\theta) \tag{18}$$

where

$$q_{cl} = 0.92E\left(\frac{t}{r}\right)^2 \frac{\sqrt{rt}}{L} \tag{19}$$

is the classical elastic critical buckling load under uniform external pressure, and

$$C_p(\theta) = \sum_{k=0}^{4}(a_k + b_k \frac{d}{L})\cos(k\theta) \tag{20}$$

is the wind-induced circumferential variation where $(a_0, b_0) = (-0.54, 0.16)$, $(a_1, b_1) = (0.28, 0.04)$, $(a_2, b_2) = (1.04, -0.20)$, $(a_3, b_3) = (0.36, -0.05)$ and $(a_4, b_4) = (-0.14, 0.05)$, $d = 2r$ is the cylinder diameter and a positive pressure is taken to act radially inwards (peaking at $\theta = 0$ in the 'stagnation' zone). A parametric sweep of 400 LBAs was performed in the 2D range $0.05 \leq \Omega \leq 1$ (20 values) and $100 \leq \frac{r}{t} \leq 1000$ (20 values) to compute the critical eigenvalues (load proportionality factors on $q(\theta)$, Eq. (18)) and extract the chromatic signatures as `.jpg` files for classification with `LBANet`.

A 3D surface of the eigenvalues and classified eigenmodes is presented in Fig. 9. The mechanics governing the shape of the surface are described in Chen and Rotter [45] and relate to how different destabilising membrane stress components are differently critical with varying Ω and $\frac{r}{t}$. The circumferential compression localised at the stagnation zone dominates for shorter cylinders, causing the `CircCompLocal` eigenmode. Unsymmetrical normal pressures additionally cause localised meridional compression in cylindrical shells [45] which in turn cause high localised in-plane membrane shear. In the dimensionless length range of approximately $0.4 \leq \Omega \leq 0.5$, the circumferential compression will interact with this shear to cause combined modes of type `CircCompShearCombi`. Longer cylinders are entirely dominated by the destabilising shear component which induces a localised buckle near the base of type `ShearLocal`. Interestingly, as $\frac{r}{t}$ increases and the cylinder becomes more slender this mode turns into a localised meridional compression buckle of type `MerCompLocal`, and there is clearly a range of Ω and $\frac{r}{t}$ where the eigenmode shape is ambiguous and may plausibly be of classified as either localised type or even of a combined `MerCompShearCombi` type. In the regions where eigenmodes exhibit features of multiple destabilising stress components, classification is difficult even for a human observer.

The classifications of `LBANet` are remarkably correct (compare blue outline markers with black outline markers in Eq. (18)) given that the network was trained solely on prototypical LBA eigenmodes generated by the generic system in Fig. 6, and it is stressed that none of the eigenmodes of the cylinder under wind loading were part of the dataset used in initial training. When the network is trained for a further few epochs on chromatic signatures generated under this loading case specifically, the classification improves as expected (compare red outline markers with blue and black ones). The network no longer struggles with identifying the dominant `CircCompLocal` mode in cylinders with

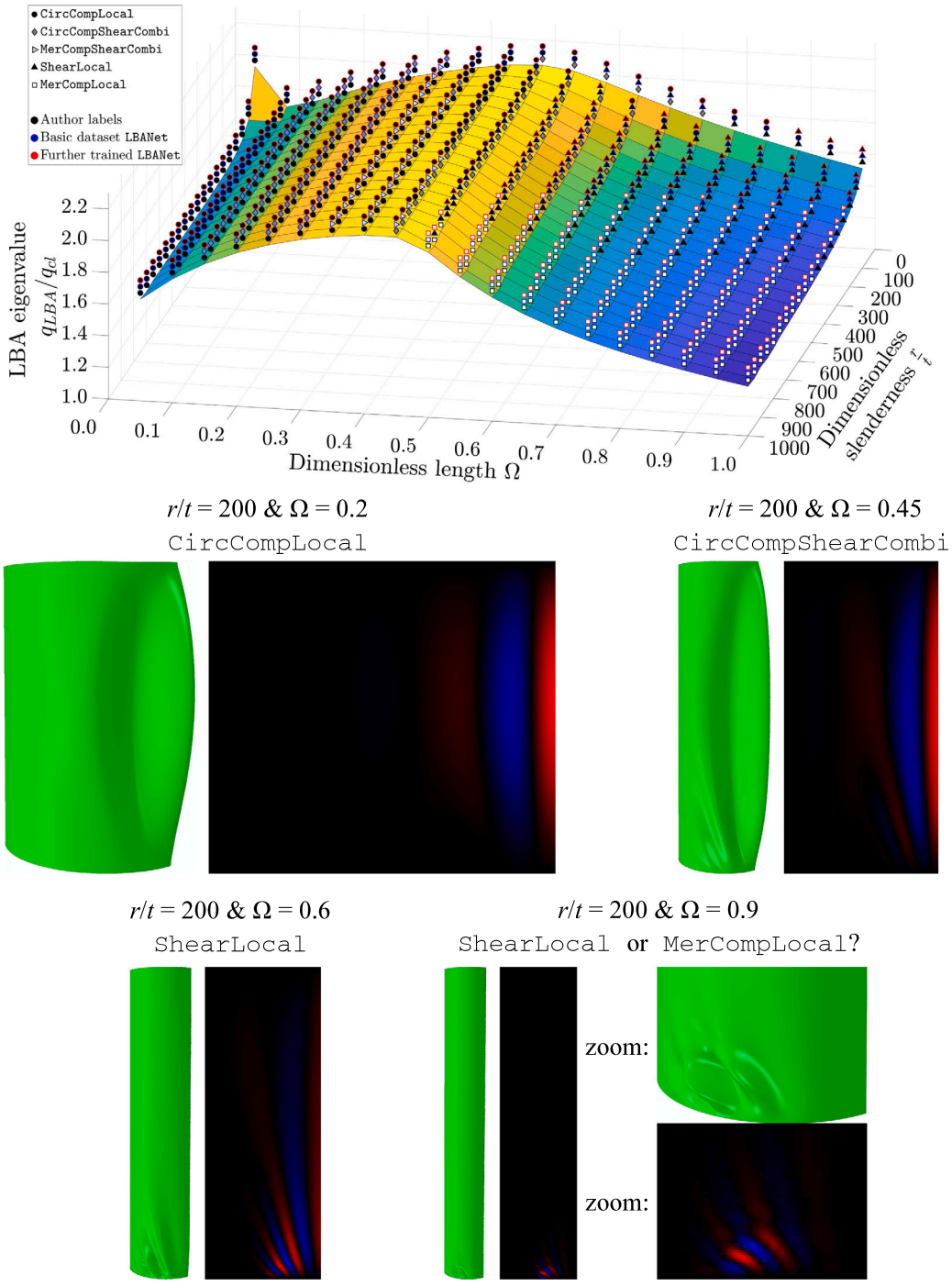

Fig. 9. Application of LBANet to the classification of linear buckling eigenmodes of cylinders under non-symmetric wind pressures.

$\Omega < 0.4$. An ambiguity remains for the approximate region of $0.4 < \Omega < 0.7$ with CircCompShearCombi modes being instead classified as CircCompLocal or ShearLocal only (which at least correctly identifies the presence of one of the constituent destabilising membrane stress components). Similarly, for thinner cylinders with $\Omega > 0.5$ the localised eigenmodes at the base of the cylinder are at times preferentially identified as MerCompLocal rather than ShearLocal, although the Author is admittedly not entirely certain of his own labelling of these eigenmodes given their visual ambiguity (see bottom-right of Fig. 9, with either option being a plausible classification).

7. Conclusions and future work

This paper presents a classification tool for linear bifurcation buckling eigenmodes in cylindrical shell structures based on a convolutional neural network trained on a purposefully generated dataset of prototype eigenmodes encoded as chromatic signatures in .jpg image files. Termed LBANet, the network has been shown to perform at a human-equivalent level in terms of classification accuracy on the prototypical training dataset, artificially enhanced with probabilistic image transformations, as well as an industrially important example of

a cylindrical shell structure of varying length and slenderness under an unsymmetrical wind-induced pressure.

The classification tool is intended to be used to aid in the automated execution of large-scale parametric sweeps involving tens of thousands of parameter combinations so as to free the analyst from visually and laboriously assessing each individual eigenmode. An immediate application is to the ongoing development of the Reference Resistance Design method of EN 1993-1-6 on the strength and stability of metal civil engineering shell structures. The development of this design method is uniquely data-intensive and must be supported by large computationally produced datasets, only possible with automation.

The output of this work may also be treated as a general 'seed' neural network with trained feature maps appropriate for the general recognition of buckling deformations that can be the basis of further training. In future, it is intended to:

- Expand the dataset with purposefully generated chromatic signatures of nonlinear bifurcation buckling modes;
- Enhance the encoding mechanism to non-cylindrical and non-revolution shells, ideally in such a way that the standardised size of the input layer to the network remains unchanged under all conditions;
- Explore using segmentation techniques to enable the network to be used for the detection and classification of individual buckles regardless of position on the structure;
- Extend the application of the classifier to include buckles obtained from photographic images, potentially a valuable tool for forensic investigations of structural failures.

Supplementary materials

Instructions on downloading the training dataset, the trained `LBANet` neural network and supporting `Python` scripts may be found in the following `GitHub` repository:

`https://github.com/SadowskiAJ/LBANet.git`

The Author invites readers to supplement this dataset with additional *labelled* buckling eigenmodes encoded as chromatic signatures in `.jpg` files with a view to training an increasingly robust classifier. The contents of the `GitHub` link above will be updated periodically as more data is acquired.

CRediT authorship contribution statement

Adam J. Sadowski: Conceptualisation, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualisation.

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.

Appendix. EN 1993-1-6 computational analysis taxonomy

The following acronyms define the taxonomy of computational analyses introduced by EN 1993-1-6 [7,9]. Originally intended for metal shells, they have since found widespread adoption for steel structures and are now formally part of the new EN 1993-1-14 [48] on the design of steel structures assisted by finite element analysis:

- LA: linear elastic stress analysis of the perfect structure;
- LBA: linear bifurcation analysis of the perfect structure to obtain a linear bifurcation buckling eigenmode and associated eigenvalue (typically the lowest positive one, which is known as the reference elastic critical buckling resistance);
- MNA: materially nonlinear (with an ideal elastic but rigid plastic material law) but geometrically linear (small displacement theory) analysis of the perfect structure to obtain a nonlinear equilibrium path and the reference plastic resistance;
- GNA: geometrically nonlinear (finite displacement theory) but materially linear analysis of the perfect structure to obtain a nonlinear equilibrium path and the bifurcation or limit point buckling resistance;
- GMNA; geometrically and materially nonlinear (with no limitation on the complexity of the material law) analysis of the perfect structure to obtain a nonlinear equilibrium path and the bifurcation or limit point buckling resistance;
- GMNIA; the same as GMNA except with explicit modelling of imperfections.

Certain researchers have also purposefully performed LBIA [32] and MNIA [14] which assume the same analysis conditions as LBA and MNA respectively but which include explicitly modelled (so far only geometric) imperfections. Lastly, the boundary condition classification scheme of EN 1993-1-6 for shells of revolution is presented in Table A1 below for reference (finite stiffness conditions have not been included).

Table A1
Summary of the EN 1993-1-6 boundary condition classification scheme for shells of revolution.

Boundary condition code	Simple term	Normal displacements	Meridional displacements	Meridional rotations
BC1r	Clamped	Restrained	Restrained	Restrained
BC1f		Restrained	Restrained	Free
BC2r		Restrained	Free	Restrained
BC2f	Pinned	Restrained	Free	Free
BC3r		Free	Free	Restrained
BC3f	Free edge	Free	Free	Free

References

[1] Elishakoff I, Li W, Starnes Jr JH. On-classical problems in the theory of elastic stability. Cambridge University Press; 2001.

[2] Singer J, Arbocz J, Weller T. Buckling experiments: experimental methods in buckling of thin-walled structures – shells, built-up structures, composites and additional topics – vol. 2. Inc. New York: John Wiley & Sons; 2002.

[3] Rotter JM. Buckling of thin shells: An overview chapter 1. In: Teng JG, Rotter JM, editors. Buckling of thin metal shells. 2004, p. 1–41.

[4] ECCS EDR5#2. Buckling of steel shells - European design recommendations. Brussels: Publication (125) Revised 2nd Impression; 2013,

[5] Rotter JM, Sadowski AJ, Chen L. Nonlinear stability of thin elastic cylinders of different length under global bending. Int J Solids Struct 2014;51:2826–39. http://dx.doi.org/10.1016/j.ijsolstr.2014.04.002.

[6] Sadowski AJ, Fajuyitan OK, Wang J. A computational strategy to establish algebraic parameters for the reference resistance design of metal shell structures. Adv Eng Softw 2017;109:15–30. http://dx.doi.org/10.1016/j.advengsoft.2017.02.012.

[7] EN 1993-1-6. Eurocode 3: Design of steel structures. In: Part 1-6: Strength and stability of shell structures. Brussels: Comité Européen de Normalisation (CEN); 2007.

[8] Rotter JM. New segment in annex e of EN 1993-1-6 on cylindrical shells under global bending and spherical shells under external pressure. 2013, Amendment AM-1-6-2013-13 to EN 1993-1-6, approved by CEN/TC250/SC3 in Zürich.

[9] prEN 1993-1-6. Eurocode 3: Design of steel structures. In: Part 1-6: Strength and stability of shell structures. Brussels: Comité Européen de Normalisation (CEN); 2022, Submitted for CEN Enquiry.

[10] Rotter JM. The new method of reference resistance design for shell structures. In: Proc., int. colloquium on stability and ductility of steel structures. 2016, p. 623–30.

[11] Koukouselis A, Grammatopoulos A, Mistakidis E. Buckling capacity of radially compressed thin-walled reinforced cementitious spheres. Eng Struct 2018;157:63–74. http://dx.doi.org/10.1016/j.engstruct.2017.11.063.

[12] Wang J, Sadowski AJ. Elastic imperfect tip-loaded cantilever cylinders of varying length. Int J Mech Sci 2018;140:200–10. http://dx.doi.org/10.1016/j.ijmecsci.2018.02.027.
[13] Wang J, Sadowski AJ. Elastic imperfect cylinders of varying length under combined axial compression and bending. ASCE J Struct Eng 2020;146(4):04020014. http://dx.doi.org/10.1061/(ASCE)ST.1943-541X.0002560.
[14] Wang J, Fajuyitan OK, Orabi MA, Rotter JM, Sadowski AJ. Cylindrical shells under uniform bending in the framework of reference resistance design. J Construct Steel Res 2020;166:105920. http://dx.doi.org/10.1016/j.jcsr.2019.105920.
[15] Rotter JM. Shell buckling design and assessment and the LBA-MNA methodology. Stahlbau 2011;80(11):791–803. http://dx.doi.org/10.1002/stab.201101491.
[16] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE 1998;86(11):2278–323. http://dx.doi.org/10.1109/5.726791.
[17] Scherer D, Müller A, Behnke S. Evaluation of pooling operations in convolutional architectures for object recognition. In: International conference on artificial neural networks (ICANN 2010), Part III. LNCS, vol. 6354, 2010, p. 92–101. http://dx.doi.org/10.1007/978-3-642-15825-4_10.
[18] Graham B. Fractional max-pooling. 2014, http://dx.doi.org/10.48550/arxiv.1412.6071, arXiv.
[19] Ciresan DC, Meier U, Gambardella LM, J. Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. Neural Comput 2010;22:3207–20. http://dx.doi.org/10.1162/NECO_a_00052.
[20] LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, et al. Backpropagation applied to handwritten zip code recognition. Neural Comput 1989;1(4):541–51. http://dx.doi.org/10.1162/neco.1989.1.4.541.
[21] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, et al. Int J Comput Vis 2022;115:211–52. http://dx.doi.org/10.1007/s11263-015-0816-y.
[22] Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. Commun ACM 2017;60(6):84–90. http://dx.doi.org/10.1145/3065386.
[23] Ji S, Xu W, Yang M, Kai Y. 3D convolutional neural networks for human action recognition. IEEE Trans Pattern Anal Mach Intell 2013;35(1):221–31. http://dx.doi.org/10.1109/TPAMI.2012.59.
[24] Collobert R, Weston J. A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proc. 25th int. conf. on machine learning. 2008, p. 160–7. http://dx.doi.org/10.1145/1390156.1390177.
[25] Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, et al. Automatic differentiation in pytorch. In: 31st conference on neural information processing systems. 2017.
[26] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. PyTorch: An imperative style, high-performance deep learning library. In: 33rd conference on neural information processing systems. 2019.
[27] Goodfellow I, Bengio Y, Courville A. Deep learning. MIT Press; 2016.
[28] Koiter WT. On the stability of elastic equilibrium Ph.D. thesis (in Dutch), Delft University; 1945, (see also Translation AFFDL-TR-70-25 Wright Patterson Air Force Base, 1970).
[29] Koiter WT. Elastic stability and post-buckling behaviour. In: Langer RE, editor. Proceedings of a symposium on nonlinear problems. University of Wisconsin Press.; 1963, p. 257–75.
[30] Calladine CR. Theory of shell structures. Cambridge University Press; 1983.
[31] Kobayashi T, Mihara Y, Fujii F. Path-tracing analyses for post-buckling process of elastic cylindrical shells under axial compression. Thin-Walled Struct 2012;61:180–7. http://dx.doi.org/10.1016/j.tws.2012.05.018.
[32] Sadowski AJ, Pototschnig L, Constantinou P. The 'panel analysis' technique in the computational study of axisymmetric thin-walled shell systems. Finite Elem Anal Des 2018;152:55–68. http://dx.doi.org/10.1016/j.finel.2018.07.004.
[33] Rotter JM, H. Al-Lawati. Length effects in the buckling of imperfect axially compressed cylinders. In: Proc., int. colloquium on stability and ductility of steel structures. 2016, p. 631–8.
[34] Guggenberger W, Greiner R, Rotter JM. The behaviour of locally-supported cylindrical shells: Unstiffened shells. J Construct Steel Res 2000;56(2):175–97. http://dx.doi.org/10.1016/S0143-974X(99)00102-9.
[35] Fajuyitan OK, Sadowski AJ, Wadee MA, Rotter JM. Nonlinear behaviour of short elastic cylindrical shells under global bending. Thin-Walled Struct 2018;124:574–87. http://dx.doi.org/10.1016/j.tws.2017.12.018.
[36] Greiner R. Cylindrical shells under uniform external pressure Chapter 5. In: Teng JG, Rotter JM, editors. Buckling of thin metal shells. 2004, p. 154–74.
[37] Ansourian P. Cylindrical shells under non-uniform external pressure chapter 6. In: Teng JG, Rotter JM, editors. Buckling of thin metal shells. 2004, p. 175–97.
[38] Greiner R, Guggenberger W. Tall cylindrical shells under wind pressure Chapter 7. In: Teng JG, Rotter JM, editors. Buckling of thin metal shells. 2004, p. 198–206.
[39] Yamaki N. Elastic stability of circular cylindrical shells. Elsevier,Science, North Holland; 1984.
[40] Sadowski AJ, Rotter JM. Solid or shell finite elements to model thick cylindrical tubes and shells under global bending. Int J Mech Sci 2012b;74:143–53. http://dx.doi.org/10.1016/j.ijmecsci.2013.05.008.
[41] Sadowski AJ. On the advantages of hybrid beam-shell structural finite element models for the efficient analysis of metal wind turbine support towers. Finite Elem Anal Des 2019;162:19–33. http://dx.doi.org/10.1016/j.finel.2019.05.002.
[42] Nielsen MA. Neural networks and deep learning. Determination Press; 2015.
[43] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. J Mach Learn Res 2014;15:1929–58, https://dl.acm.org/doi/10.5555/2627435.2670313.
[44] Wan L, Zeiler M, Zhang S, LeCun Y, Ferbus R. Regularisation of neural networks using DropConnect. In: Proc. 30th int. conf. on machine learning, vol. 28, no. 3. 2013, p. 1058–66.
[45] Chen L, Rotter JM. Buckling of anchored cylindrical shells of uniform thickness under wind load. Eng Struct 2012;41:199–208. http://dx.doi.org/10.1016/j.engstruct.2012.03.046.
[46] EN 1993-4-1. Eurocode 3: Design of steel structures. In: Part 4-1: Silos. Brussels: Comité Européen de Normalisation (CEN; 2007.
[47] Rotter JM. Wind pressure distribution on squat silos and tanks. Research Report R01-02, University of Edinburgh.; 2001.
[48] EN 1993-1-14. Eurocode 3: Design of steel structures. In: Part 1-14: Design assisted by finite element analysis. Brussels: Comité Européen de Normalisation (CEN); 2022.