JAX-based Aeroelastic Framework for Nonlinear Analysis of Large Aircraft Models

Alvaro Cea and Rafael Palacios

March 18, 2024

Abstract

This paper presents a new implementation for time-domain nonlinear aeroelastic simulations that has been built for performance and robustness. Leveraging on the numerical library JAX, a highly vectorised codebase is written that achieves two orders of magnitude accelerations compare to conventional implementations. This brings full-vehicle simulations to run close to if not in real-time, thus opening new possibilities for aircraft aeroelastic analysis which have traditionally been constrained to either linear, frequency domain solutions, or to their nonlinear counterparts but narrower in scope. Moreover, the approach seamlessly integrates with conventional aeroelastic load packages which facilitates the analysis of complex aircraft configurations. An extensive verification has been carried out on representative aircraft models and compared with MSC Nastran linear and nonlinear solutions. Furthermore, we demonstrate the suitability of the methodology to accurately reconstruct the full 3D solution.

1 Introduction

The ever-growing need for performance and operating costs reduction, together with the current push for sustainability in aviation, are driving new aircraft designs outside the conventional envelop. A particular feature are very high aspect ratio wings to minimise induced drag, which when combined with advancements in lighter materials to reduced vehicle weight, can significantly increase wing flexibility. In this scenario, aeroelastic analysis are expected to become critical in the very early phases of the wing design process: while the field was more important in post-design stages to ensure in-flight integrity, it now becomes paramount to capture the cross-couplings between disciplines. In this more nonlinear landscape, the overall aerodynamic performance needs to be calculated around a flight shape with large deformations [9]; the input for efficient control laws account for the steady state and nonlinear couplings [1]; and the loads ultimately sizing the wings are atmospheric disturbances computed in the time-domain [7]. This is also the case for more radical configurations that may or may not exhibit high flexibility but whose aeroelastic behaviour is more uncertain. A more holistic approach to the design also increases the complexity of the processes exponentially, and the trade-offs and cost-benefit analysis may not be possible until robust computational tools are in-place to simulate the different assumptions. Certification of new air vehicles is another important aspect that requires 100,000s of load cases simulations [10], as it considers manoeuvres and gust loads at different velocities and altitudes, and for a range of mass cases and configurations. This poses another challenge for new methods that aim to include new physics since they normally incur in prohibitly expensive computational times. Lastly, the mathematical representation of the airframe, embodied in the complex Finite-Element Models (FEMs) built by organizations, encompasses a level of knowledge that is to be preserved when including the new physics mentioned above. Those previous considerations set the goals for the current effort:

1) to be able to perform geometrically nonlinear aeroelastic analysis, 2) to work with generic FE models in a non-intrusive manner, and 3) to achieve a computational efficiency that is equivalent to present linear methods (if not faster). Grounded on previous developments where the first two points where demonstrated [14], [?], [5] we tackle the third point herein with a new implementation that achieves remarkable computational performance. The numerical library JAX [3] was leveraged to produce highly vectorised, automatically differentiated routines that are managed by a modular, object-oriented approach in Python. The power of JAX for scientific computation has been proved recently in fluid dynamics [2] and solid mechanics [17] applications. We add to those an aeroelastic solution to enhance already built models for linear loads analysis. This aligns with current efforts to build robust methods that incorporate nonlinear effects to complex 3-D FEMs, via stick models [15] or other modal-based methods [8].

Our proposed method has two main inputs for the analysis: a linear (arbitrarily complex) FE model, and aerodynamic influence coefficient matrices that provide the mapping between FE states and the corresponding aerodynamic forces (either in modal or in physical coordinates). The latter are obtained herein from the Doublet Lattice Method (DLM) and a rational function approximation (RFA) [16] to transform to the time domain. We have also presented a more efficient data-driven approach that circumvents the additional states added by the RFA in [?] and the approach would also be suitable for more accurate Computational Fluids Aerodynamics (CFD). Using the 3D FE model, a skeleton-like substructure along the main load paths is derived, on which modal shapes and nonlinear couplings are evaluated in intrinsic variables (velocities and strains). They conform a basis of a Galerkin-projection of the geometrically-nonlinear 1D domain after which the projected equations are solved in time-domain. Advantages of the approach are its direct and accurate map between the 3D and 1D domains, as it only requires of a modal condensation that is already available in many industrial aeroelastic models to link the structural model to the aerodynamic loading. This is unlike stick models which need of various post-processing steps to build the equivalent stiffness and mass models. Furthermore, we show how the full 3D solution using the nonlinear 1D solution is computed to a good accuracy by reconstructing the cross-sectional elements and applying a Radial Basis Function (RBF) interpolation to the remaining nodes in the domain. A well established formulation effectively applied to industrial-scale aeroelastic models and now combined with a highly vectorised implementation in JAX results in an extremely efficient nonlinear aeroelastic solver. The overall procedure has been implemented in what we have named as *Nonlinear Modal Reduced Order Model* (NMROM).

The structure of the rest of the paper is as follows. Sec. 2 presents a summary of the mathematical description that conforms the backbone behind the computational implementation of `FEM4INAS` (Finite-Element-Models for Intrinsic Nonlinear Aeroelastic Simulations), the high performance software for aeroelasticity we have built. Sec. 3 shows the verification cases that cover the static and dynamic structural response of of a simplified aircraft model, and the aeroelastic response to gusts of a full aircraft configuration. The improvements in performance are highlighted in all of the examples. Lastly, sec. 4 summarises the the achievements and further developments planned for future work.

2 Theory and implementation

In this section we briefly describe the backbone theory of the proposed methods for nonlinear aeroelasticity modelling. For further details, see [?], [5]. First we present the main formulation that leads to a geometrically nonlinear enhancement of generic aircraft models; subsequently we show how the main aeroelastic system is built using modal-based aerodynamics, and finally some implementation details are highlighted.

2.1 Airframe idealisation

An illustration of the overall solution process is presented in Fig. 1. We start with a representative FE model of the airframe. It is common practice for large-scale aeroelastic models to feature lumped masses along a load path axis that are attached to their corresponding cross-sectional nodes via interpolation elements. With those characteristics a reduced model can be obtained that captures well the stiffness and inertia properties in the condensed matrices, $\boldsymbol{K}_a$ and $\boldsymbol{M}_a$, via a static or dynamic condensation. A database of Aerodynamic Influence Coefficient matrices (AICs) is also needed as input; this may be given in the time-domain, as in the UVLM [11], or in frequency domain, as in the DLM, in which case a RFA in used to amend the data for time-domain simulations. The modal-based aerodynamics and the condensed matrices of the structure are input into a our analysis framework that builds the nonlinear dynamics of the condensed model without having to call back the stiffness and mass matrices. To achieve this a formulation based on velocities and strains in the material frame of reference is employed.

After the state of the condensed model has been solved for, the full 3D state can be reconstructed: firstly the displacements of the cross-sectional nodes linked to the reduced model via the interpolation elements are computed using the positions and rotations of the latter; secondly, Radial Basis Functions (RBFs) kernels are placed on those cross-sections, thus building an intermediate model that is utilised to extrapolate the positions of the remaining nodes in the full model. It is demonstrated with representative aircraft structures that this process yields accurate results when compared to simulations with the full FEM, which paves the way for potentially a more detailed analysis: first with the addition of CFD as the aerodynamic solver, and second with the transfer of the full deformed state back to the original FE solver to study other phenomena such as buckling.

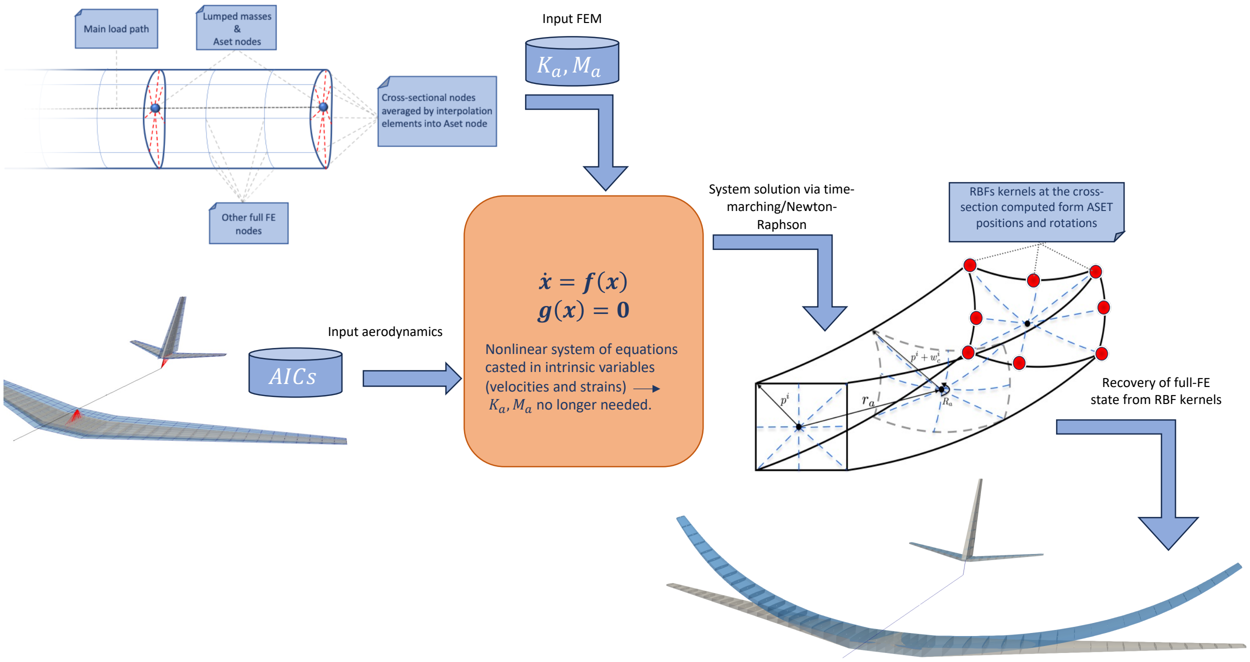
Main load path
Lumped masses & Aset nodes
Cross-sectional nodes averaged by interpolation elements into Aset node
Other full FE nodes
Input FEM
K_a, M_a
Input aerodynamics
$AICs$
$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$
$\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$
Nonlinear system of equations casted in intrinsic variables (velocities and strains) → K_a, M_a no longer needed.
System solution via time-marching/Newton-Raphson
RBFs kernels at the cross-section computed form ASET positions and rotations
p^i
$p^i + u^i_r$
r_a
R_a
Recovery of full-FE state from RBF kernels

Figure 1: Workflow of the solution process

This process of enhancing the linear 3D model with geometric nonlinearities along the slender dimension relies on the main assumption that cross-sectional deformations of the solid body in the reference configuration are not coupled to the this dimension as moving through configurations in time. As a result, distributed internal stresses act only through the normal of the cross-sections in the undeformed configuration [4]. Applying the appropriate integration over the cross sectional reference area of the distributed traction forces, a Cosserat rod model is built, where the deformed state on the full domain is approximated by a deformable space curve Γ – identified with the aircraft major load-paths. The primary variables are the local inertial (linear and angular) velocities, grouped in the variable $\boldsymbol{x}_1(s,t)$, and internal force and moments combined in $\boldsymbol{x}_2(s,t)$. They are function of the 1D spatial dimension, s, and time, t. Applied forces and moments per unit length, $\boldsymbol{f}_1$, come naturally as follower forces and moments respectively. The equations are well described in [12, Ch. 8]. Constitutive properties are given by the compliance matrix, $\boldsymbol{\mathcal{C}}$, relating sectional forces and moments to strains and curvatures and the sectional mass matrix, $\boldsymbol{\mathcal{M}}$, linking velocities and momenta. Finding a good approximation to these matrices is a common challenge in formulations that aim to build nonlinear 1D models from full FE models. This work circumvents having to calculate explicit expressions of $\boldsymbol{\mathcal{C}}$ and $\boldsymbol{\mathcal{M}}$ by solving the equations in modal space and linking them to the modal shapes and their derivatives as first described in [13].

Using the intrinsic modes and the projection of the state variables, a Galerkin projection is carried out such that $\boldsymbol{x}_1 = \boldsymbol{\phi}_1\boldsymbol{q}_1$ and $\boldsymbol{x}_2 = \boldsymbol{\phi}_2\boldsymbol{q}_2$ and the equations of motion take the following form:

$$\begin{aligned}\dot{\boldsymbol{q}}_1 &= \boldsymbol{\omega}\odot\boldsymbol{q}_2 - \boldsymbol{\Gamma}_1 : (\boldsymbol{q}_1\otimes\boldsymbol{q}_1) - \boldsymbol{\Gamma}_2 : (\boldsymbol{q}_2\otimes\boldsymbol{q}_2) + \boldsymbol{\eta}\\ \dot{\boldsymbol{q}}_2 &= -\boldsymbol{\omega}\odot\boldsymbol{q}_1 + \boldsymbol{\Gamma}_2^\top : (\boldsymbol{q}_2\otimes\boldsymbol{q}_1)\end{aligned} \tag{1}$$

where $\odot$ is the Hadamard product (element-wise multiplication), $\otimes$ is the tensor product operation and $:$ is the double dot product[1]. The new form of the equations in compact tensorial notation is in fact the way they have been implemented and vectorised. This description is geometrically-exact with quadratic nonlinearities only, encapsulated in the nonlinear modal couplings of the third-order tensors $\boldsymbol{\Gamma}_1$ and $\boldsymbol{\Gamma}_2$ (the former introduces the gyroscopic terms in the dynamics and the latter introduces the strain-force nonlinear relation). $\boldsymbol{\eta}$ is the modal projection of the external forcing terms. They are computed as integrals along the load-path 1D domain as an inner product: $\langle\boldsymbol{u},\boldsymbol{v}\rangle = \int_\Gamma \boldsymbol{u}^\top\boldsymbol{v}ds$, for any $\boldsymbol{u}\in\mathbb{R}^6$ and $\boldsymbol{v}\in\mathbb{R}^6$:

$$\begin{aligned}\Gamma_1^{ijk} &= \langle\boldsymbol{\phi}_{1i}, \mathcal{L}_1(\boldsymbol{\phi}_{1j})\boldsymbol{\psi}_{1k}\rangle,\\ \Gamma_2^{ijk} &= \langle\boldsymbol{\phi}_{1i}, \mathcal{L}_2(\boldsymbol{\phi}_{2j})\boldsymbol{\psi}_{2k}\rangle,\\ \eta_i &= \langle\boldsymbol{\phi}_{1i}, \boldsymbol{f}_1\rangle\end{aligned} \tag{2}$$

where $\mathcal{L}_1$ and $\mathcal{L}_2$ are linear operators, $\boldsymbol{\psi}_1 = \boldsymbol{\mathcal{M}}\boldsymbol{\phi}_1$ and $\boldsymbol{\psi}_2 = \boldsymbol{\mathcal{C}}\boldsymbol{\phi}_2$ are also cast as momentum and strain mode shapes and approximated using the Linear Normal Modes of the FE model. In other words, each natural vibration mode can be uniquely expressed in terms of velocity, force/moment, momentum, or strain variables. While those would be redundant in a conventional linear vibration analysis, they will enable to identify all the coefficients in Eqs. (8).

[1] The double dot product, commonly found in solid mechanics descriptions, represents a contraction of the last two indexes of the first tensor with the first two indexes of the second one; it however needs further specification as two alternative definitions can be adopted and here we opt for the following: $\boldsymbol{a} : \boldsymbol{b} = a_{..ij}b_{ij..}$. This has implications on the definition of the transpose of $\boldsymbol{\Gamma}_2$ in the second equation since for high order tensors multiple transpose operators can be defined. Consistency is achieved by ensuring the dot product operation satisfies the following: $\boldsymbol{x}\cdot(\boldsymbol{\Gamma} : (\boldsymbol{y}\otimes\boldsymbol{z})) = \boldsymbol{y}\cdot\left(\boldsymbol{\Gamma}^\top : (\boldsymbol{z}\otimes\boldsymbol{x})\right)$, which leads to the transpose of the third order tensor, $\boldsymbol{\Gamma} = \Gamma^{ijk}$, as $\boldsymbol{\Gamma}^\top = \Gamma^{jki}$.

2.2 formulas

$$\mathbf{\Gamma}_1 = einsum('isn, jskn- > ijk', \boldsymbol{\phi}_1, \sqsubseteq())$$

2.3 placeholder

The corresponding distribution of linear and rotational momenta at the master nodes can be obtained using the condensed inertia matrix, $\mathbf{\Psi}_{1j} = \boldsymbol{M}_a\mathbf{\Phi}_{1j} = \boldsymbol{M}_a\mathbf{\Phi}_{0j}$, expressed in their components in the global frame of reference. The introduction of this momentum mode has allowed calculations to be performed on distributed mass models and the use of any type of condensation technique. Because the mass matrix is already calculated as an integral along the 3D domain and then condensed to a set of master nodes, the continuous momentum mode shapes, $\boldsymbol{\psi}_1$, are considered lumped and defined as, %

$$\boldsymbol{\psi}_{1j}(s) = \Xi_0(s_i)\mathbf{\Psi}_{1j,i}\delta(s - s_i) \tag{3}$$

where δ is Dirac's delta function. Each displacement mode also generates a corresponding internal stress state. This defines discrete force/moment modes, $\mathbf{\Phi}_2$, which are obtained from the displacement modes and the condensed stiffness matrix using a summation-of-forces approach from aircraft load analysis, [**?**, Ch. 18]. If N_n is the total number of nodes, the internal node forces are for each mode j computed as $\mathfrak{f}_{(j)} = \boldsymbol{K}_a\mathbf{\Phi}_{0j}^{1-3}$, with $\mathfrak{f}_{(j)} \in \mathbb{R}^{3N_n}$; and analogously the internal moments as $\mathfrak{m}_{\mathfrak{f}(j)} = \boldsymbol{K}_a\mathbf{\Phi}_{0j}^{4-6}$. Next, a function $\mathcal{S}(\mathfrak{h}, s_i)$ is defined so that it sums the value of $\mathfrak{h}$ at each node in the path from a free end to the node s_i in the structure. $\mathbf{\Phi}_2$ is calculated as %

$$\mathbf{\Phi}_{2j,i+\frac{1}{2}} = \begin{bmatrix} \mathcal{S}(\mathfrak{f}_{(j)}, s_i) \\ \mathcal{S}\left(\mathfrak{m}_{\mathfrak{f}(j)} + (\boldsymbol{r}_i - \boldsymbol{r}_{i+\frac{1}{2}}) \times \mathfrak{f}_{(j)}, s_i\right) \end{bmatrix} \tag{4}$$

% where $\boldsymbol{r}_i$ is the position vector of the nodes summed by $\mathcal{S}$, and $\boldsymbol{r}_{i+\frac{1}{2}}$ the mid position between nodes s_i and s_{i+1}. The first term is the sum of forces due to modal displacements and the second one the sum of moments due to modal rotations and the cross product of the position vector and the previous calculated force. This equation represents an equilibrium of internal forces and moments – between node $i$ and $i+1$. Results are presented in the mid-point $i+\frac{1}{2}$ because more information cannot be extracted in terms of linear stresses from one node to the other. %Note, however, $i+\frac{1}{2}$ corresponds to the $i$th raw in the matrix $\mathbf{\Phi}_2$ and free-end points of the structure have a value of zero. The sum $\mathcal{S}$ is taken as positive in the direction of increasing s, and negative otherwise, so that the addition of internal forces and moments from all nodes in the model equals zero. An algorithm to calculate the optimal path for this sum was designed. The continuous force modes, $\boldsymbol{\phi}_2$, are now interpolated as piece-wise constant between nodes,

$$\boldsymbol{\phi}_{2j}(s) = \Xi_0(s_i)\mathbf{\Phi}_{2j,i+\frac{1}{2}} \tag{5}$$

with $s \in [s_i, s_{i+1}]$, as before. Finally, the strain modes $\boldsymbol{\psi}_2$ are obtained from spatial derivatives of the displacement modes along Γ, following Eq. (**??**), and interpolated as piece-wise constant too,

$$\boldsymbol{\psi}_{2j}(s) = -\frac{\boldsymbol{\phi}_{1j}(s_{i+1}) - \boldsymbol{\phi}_{1j}(s_i)}{\Delta s_i} + \boldsymbol{E}^\top\frac{\boldsymbol{\phi}_{1j}(s_{i+1}) + \boldsymbol{\phi}_{1j}(s_i)}{2} \tag{6}$$

2.4 Nonlinear aeroelastic system

The full aeroelastic solution is described extending Eq. (8) with gravity forces, $\boldsymbol{\eta}_g$, aerodynamic forces and gust disturbances, $\boldsymbol{v}_g$. Control states can also be included [4], but they are not necessary for this work. For a set of reduced frequencies and a given Mach number, the DLM (or a higher fidelity aerodynamic method) yields the Generalised Aerodynamic Forces (GAFs). The current implementation uses Roger's rational function approximation to those GAFs, which results in the follower modal forces:

$$\begin{aligned}\boldsymbol{\eta}_a = \tfrac{1}{2}\rho_\infty U_\infty^2 \Bigg(& \boldsymbol{\mathcal{A}}_0 \boldsymbol{q}_0 + \frac{c}{2U_\infty}\boldsymbol{\mathcal{A}}_1 \boldsymbol{q}_1 + \left(\frac{c}{2U_\infty}\right)^2 \boldsymbol{\mathcal{A}}_2 \dot{\boldsymbol{q}}_1 \\ & + \boldsymbol{\mathcal{A}}_{g0} \boldsymbol{v}_g + \frac{c}{2U_\infty}\boldsymbol{\mathcal{A}}_{g1} \dot{\boldsymbol{v}}_g + \left(\frac{c}{2U_\infty}\right)^2 \boldsymbol{\mathcal{A}}_{g2} \ddot{\boldsymbol{v}}_g + \sum_{p=1}^{N_p} \boldsymbol{\lambda}_p \Bigg)\end{aligned} \tag{7}$$

The coupling of the structure and aerodynamic equations combined with the aerodynamic lags gives the final ODE system:

$$\begin{aligned} \dot{\boldsymbol{q}}_1 &= \hat{\boldsymbol{\Omega}} \boldsymbol{q}_2 - \hat{\boldsymbol{\Gamma}}_1 : (\boldsymbol{q}_1 \otimes \boldsymbol{q}_1) - \hat{\boldsymbol{\Gamma}}_2 : (\boldsymbol{q}_2 \otimes \boldsymbol{q}_2) + \hat{\boldsymbol{\eta}} \\ \dot{\boldsymbol{q}}_2 &= -\boldsymbol{\omega} \odot \boldsymbol{q}_1 + \boldsymbol{\Gamma}_2^\top : (\boldsymbol{q}_2 \otimes \boldsymbol{q}_1) \\ \dot{\boldsymbol{\lambda}}_p &= \hat{\boldsymbol{\mathcal{A}}}_{p+2} \boldsymbol{q}_1 + \hat{\boldsymbol{\mathcal{A}}}_{p+2} \dot{\boldsymbol{v}}_g - \frac{2U_\infty \gamma_p}{c} \boldsymbol{\lambda}_p \end{aligned} \tag{8}$$

in this system the aerodynamic added-mass effect has been moved to the left hand side such that $\mathbf{A}_2 = (\boldsymbol{I} - \frac{\rho c^2}{8}\boldsymbol{\mathcal{A}}_2)^{-1}$, and it couples all DoF in $\boldsymbol{q}_1$. Thus the natural frequency terms become $\hat{\boldsymbol{\Omega}} = \mathbf{A}_2 diag(\boldsymbol{\omega})$ and the nonlinear terms $\hat{\boldsymbol{\Gamma}} = \mathbf{A}_2 \boldsymbol{\Gamma}$. The effect of all external forces, aero, $\boldsymbol{\eta}_a$, gravity, $\boldsymbol{\eta}_g$, and others, $\boldsymbol{\eta}_f$, are combined in such that $\hat{\boldsymbol{\eta}} = \mathbf{A}_2 \left(\left(\boldsymbol{\eta}_a - \frac{\rho c^2}{8} \boldsymbol{\mathcal{A}}_2 \dot{\boldsymbol{q}}_1 \right) + \boldsymbol{\eta}_g + \boldsymbol{\eta}_f \right)$. Note that the structure of the formulation in modal space naturally leads to reduced order models and easily caters for vectorised operations. For instance, the contraction of $\boldsymbol{\Gamma}_2^T\top$ in the second equation of Eqs. 8 is implemented as an Einstein summation in JAX as `einsum(ijk,ki->j, gamma2, tensordot(q2, q1, axes=0)`. This will be briefly discussed next.

2.5 Computational implementation

The main contribution of this work is a new computational implementation that achieves accelerations of over 2 orders of magnitude with respect to its predecessor[2]. In addition, a highly modular, flexible architecture based on software design patterns has been put in place, which was further described in [6]. Moreover, the resulting nonlinear aeroelastic framework is suitable for modern hardware architectures and the computation of sensitivities via algorithmic differentiation (AD) has been implemented as a crucial part for design optimisation, both of which will be exemplified in future work.

The key enabler was moving from standard Python to a highly vectorised, JAX-based numerical implementation. JAX is a Python library designed for high-performance numerical computing with focus on machine learning activities [3]. This library is developed and maintained by Google research team. It combines XLA (accelerated linear algebra) and Autograd, the former being a compiler that optimises models for different hardware platforms, the latter is an Automatic Differentiation (AD) tool in Python. Moreover, its extensible system of composable function transformations provides

[2]:Both the new implementation and the examples of this paper can be found at `https://github.com/ACea15/FEM4INAS`.

a set of important features for Computational Science as illustrated in Fig. 2. For instance, the vmap function allows for complex **vectorisation** operations and the pmap function for Single-Program Multiple-Data (SPMD) **parallelisation**. Both forward and reverse mode **automatic differentiation** are supported. Finally the **just-in-time compilation** (jit) relies on the XLA engine to compile and execute functions on CPUs but also on accelerators such as GPUs and TPUs, offering a versatile solution for seamlessly connecting the software to various types of hardware without requiring extra CUDA code, or a Domain Specific Language (DSL).

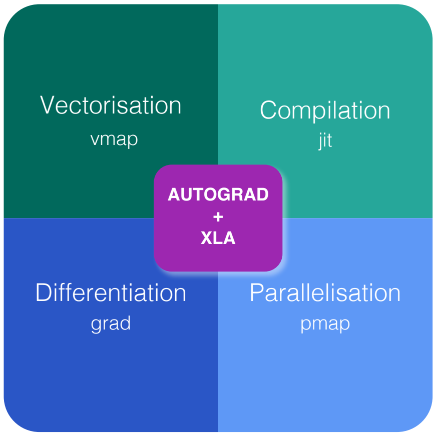

Figure 2: JAX capabilities for modern scientific computing

The new capabilities come at the expense of a higher restriction in the way the code is written. Compilation and transformations in JAX only work for functionally pure programs, which pushes the software to comply with a nonconventional functional paradigm. Some of these characteristics are **pure functions**, i.e. functions that have no side effects, **input/output** stream management needs to be placed outside the numerical algorithms, **inmutability** of arrays, and **function composition**, or the ability to create functions by chaining other callables.

These very constraints allow to achieve the capabilities describe above via the many abstractions implemented internally in the library.

3 Examples

The cases presented are a demonstration of our solution approach to manage geometric nonlinearities, the accuracy of the solvers when compared to full FE simulations, and the computational gains that can be achieved. All computations are carried out on a single core of the same CPU, an i7-6700 with 3.4 GHz clock speed.

4 Conclusions

References

[1] Marc Artola, Norberto Goizueta, Andrew Wynn, and Rafael Palacios. Aeroelastic control and estimation with a minimal nonlinear modal description. *AIAA Journal*, 59(7):2697–2713, 2021.

[2] Deniz A. Bezgin, Aaron B. Buhendwa, and Nikolaus A. Adams. JAX-Fluids: A fully-differentiable high-order computational fluid dynamics solver for compressible two-phase flows. *Computer Physics Communications*, 282:108527, January 2023.

[3] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable transformations of Python+NumPy programs, 2018.

[4] Alvaro Cea. *A Geometrically Nonlinear Approach for the Aeroelastic Analysis of Commercial Transport Aircraft.* PhD thesis, Imperial College London, 2021.

[5] Alvaro Cea and Rafael Palacios. Geometrically Nonlinear Effects on the Aeroelastic Response of a Transport Aircraft Configuration. *Journal of Aircraft*, 60(1):205–220, January 2023.

[6] Alvaro Cea and Rafael Palacios. A Nearly-Real Time Nonlinear Aeroelastic Simulation Architecture Based on JAX. In *AIAA SCITECH 2024 Forum*, Orlando, FL, January 2024. American Institute of Aeronautics and Astronautics.

[7] Carlos E.S. Cesnik, Rafael Palacios, and Eric Y. Reichenbach. Reexamined Structural Design Procedures for Very Flexible Aircraft. *Journal of Aircraft*, 51(5):1580–1591, September 2014.

[8] Ariel Drachinsky and Daniella E. Raveh. Nonlinear Aeroelastic Analysis of Highly Flexible Wings Using the Modal Rotation Method. *AIAA Journal*, 60(5):3122–3134, May 2022.

[9] Alasdair C. Gray and Joaquim R. Martins. Geometrically Nonlinear High-fidelity Aerostructural Optimization for Highly Flexible Wings. In *AIAA Scitech 2021 Forum*, VIRTUAL EVENT, January 2021. American Institute of Aeronautics and Astronautics.

[10] Thiemo M Kier. An integrated model for lateral gust loads analysis and dutch roll flight dynamics using a 3d panel method. International Forum on Aeroelasticity and Structural Dynamics 2013, 2017.

[11] Salvatore Maraniello and Rafael Palacios. State-space realizations and internal balancing in potential-flow aerodynamics with arbitrary kinematics. *AIAA Journal*, 57(6):2308–2321, 2019.

[12] R Palacios and C E S Cesnik. *Dynamics of Flexible Aircraft: Coupled Flight Mechanics, Aeroelasticity, and Control.* Cambridge Aerospace Series. Cambridge University Press, Cambridge, UK, 2023.

[13] R. Palacios, Y. Wang, and M. Karpel. Intrinsic models for nonlinear flexible-aircraft dynamics using industrial finite-element and loads packages. *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, pages 1–18, 2012.

[14] Rafael Palacios and Alvaro Cea. Nonlinear Modal Condensation of Large Finite Element Models: Application of Hodges's Intrinsic Theory. *AIAA Journal*, 57(10):4255–4268, October 2019.

[15] Cristina Riso and Carlos E. S. Cesnik. Impact of Low-Order Modeling on Aeroelastic Predictions for Very Flexible Wings. *Journal of Aircraft*, 60(3):662–687, May 2023.

[16] Kenneth L Roger, Garold E. Hodges, and Larry Felt. Active flutter suppression-A flight test demonstration. *Journal of Aircraft*, 12(6):551–556, 1975.

[17] Tianju Xue, Shuheng Liao, Zhengtao Gan, Chanwook Park, Xiaoyu Xie, Wing Kam Liu, and Jian Cao. JAX-FEM: A differentiable GPU-accelerated 3D finite element solver for automatic inverse design and mechanistic data science. *Computer Physics Communications*, 291:108802, October 2023.