Differentiable Gust and Maneuver Loads at Scale on Very Flexible Aircraft

Rafael Palacios, Alvaro Cea, Lucian Iorga

1 Introduction

Our goal is a computational framework that includes nonlinear effects for aeroelastic simulation at industrial scale. We seek the following features [CP25]: 1) to be able to perform geometrically nonlinear aeroelastic analysis, 2) to work with existing generic FEMs in a non-intrusive manner, 3) to achieve a computational efficiency that is equivalent to present linear methods (if not faster), and 4) compute derivatives of the aeroelastic response via Algorithm Differentiation (AD).

In this work we explore the latest advances on accelerator's parallelization in the JAX framework [BFH+18], and how to integrate them into our solution process to enable intensive aeroelastic simulations under geometrically nonlinear assumptions. Specifically, we set out to characterize the dynamics of highly flexible aircraft in response to the large envelope of simulations required to capture in-flight loads encountered in the certification process – while introducing new physics that account for the large displacements and rotations ultra-high-aspect-ratio wings are expected to undergo. For this, a Single Program Multiple Data (SPMD) paradigm is employed with the main computation spanning as many devices as available in the cluster and performing collective operations to communicate between devices. By addressing in one run multiple scenarios (maneuvres and gust responses at different velocities and altitudes, and for a range of mass cases and configurations), we are able to produce the critical loading characteristics of the aircraft in very short simulation times. Moreover, we can differentiate the boundaries of the critical cases using the already demonstrated capabilities of AD within JAX [CP25], but now extending them to produce derivatives across those design envelopes. This implies computing gradients across concurrent simulations and collective operations, which we show it is well managed by the chosen library, that is actively developed to solved similar problems in the realm of machine learning. This is expected to be highly applicable in providing designers with additional insights about sensitivities and in extending gradient-based optimization analysis with load-sizing constraints.

2 Methodology

The proposed solution has been named the Forager Pattern and is depicted in Fig. 1. Many simulations are launched concurrently with the predefined load cases. The solutions of all these simulations are collected (hundreds of cases, hundreds of nodes, thousands of time steps make for a single field of interest like the stress to have a size of the order of $10^7 - 10^8$). A filtering step consists of a selection of monitoring points of interest (nodes in the FEM), and then a double reduction operation in both time and load cases (for example, the maximum of the selected field in time and across cases). The output is a selection of the most problematic load cases according to the predefined metric in the input file. For these critical points, the program builds the inputs for the cases previously run in parallel but now with AD and on a much smaller basis, and finally more FENIAX process are spawn for the AD computations. In this way we have created a meta-program that can automatically create subprograms based on the results.

3 Indicative results

The proposed approach will be demonstrated on an aircraft model undergoing very large nonlinear displacements. The University of Bristol Ultra-Green (BUG) aircraft model [SCN+18] is the chosen platform, as it is not based on proprietary data and it showcases high-aspect ratio wings and a global FEM based on beam and shell elements in MSC Nastran. The main components of the aeroelastic model have been presented in [ARI26]. Here, we build

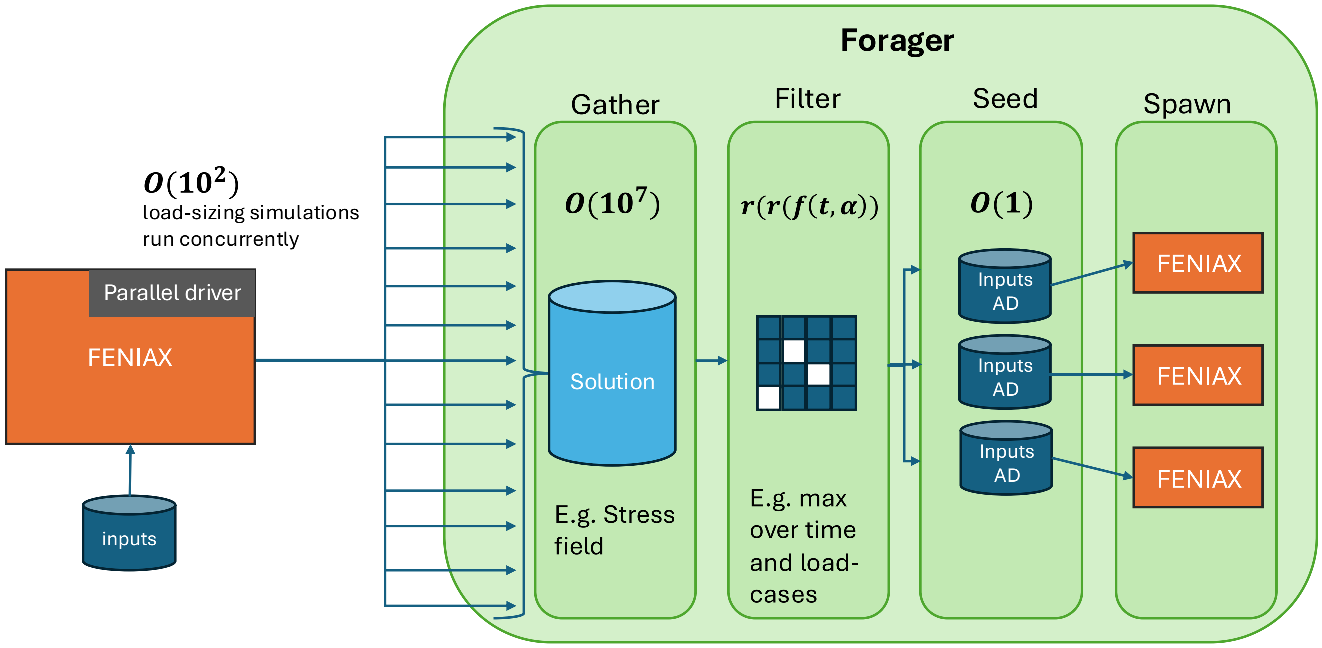

Figure 1: Forager pattern for differentiable-parallel simulations

load envelopes of static and dynamic aeroelastic simulations and differentiate across the concurrent simulations to obtain sensitivities of dynamic loads as well as moment statistics.

3.1 Gust loads

Dynamic aeroelastic analyses will study the response of the aircraft to multiple 1-cos gusts for varying length, intensity and the density of the airflow. The Mach number is kept constant at 0.7. A Runge-Kutta solver is employed to march in time the equations with a time step of 10^{-3} s and the total number of modes used was 100. A total of 512 gusts cases are run concurrently for all possible combinations of 8 gust lengths between 50 and 200 meters, 8 gust intensities between 5 and 25 m/s, and 8 airflow densities between 0.34 and 0.48 $\mathrm{Kg/m^3}$. This means that over 360,000 equations are being marched in time, in this case for 2 seconds which is enough to capture peak loads. We have verified the concurrent implementation by satisfactory comparing single-point simulations to the same points within the parallel results. Table 1 contains the simulation times of the calculation, which shows one order of magnitude increase in performance when running in parallel in the CPU versus a complete single simulation running sequentially, and another order of magnitude when moving from the CPU to a modern GPU. This exemplifies the power of modern hardware for scientific computation. Finally, Fig. 2 shows the 3D reconstructed flight shape of the airframe for a gust of 150 m length, intensity of 20 m/s and flow density of 0.41 $\mathrm{Kg/m^3}$.

Table 1: Computational times multiple gust problem

Device	Time [sec.]
CPU (single)	$27.8 \times 512 = 14233.6$
CPU (parallel)	922.6
GPU	38.2

3.2 Load envelope differentiation

The quantities being monitored are wing-root shear, torsion and out-of plane bending moments. The parallelization is set for two gust intensities, two flow densities and 16 gust lengths to cover 1-cos gusts from 50 to 200 m/s in increments of 10 m/s. This enables testing the machinery of the forager pattern and verify it can indeed discover critical load cases and automatically compute gradients. The gradient of these critical cases is also calculated with

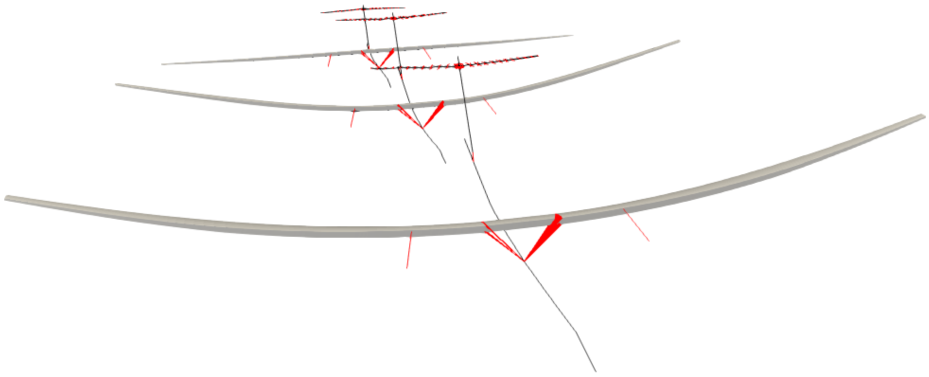

Figure 2: Full aircraft Dynamic response to 1-cos gust excitation

respect to the freestream density, gust length and intensity. Those are not only the parameters for the parallelization but are also chosen to be the input variables of the sensitivity analysis, though any other input parameter could have been chosen. Of the 64 cases analyzed, the forager step picked those with higher intensity and flow density as expected, but only 2 gust lengths of 70 m/s for the shear and torsion, and 110 m/s for the out-of-plane bending. Finite differences are computed with an $\epsilon = 10^{-4}$ and the absolute relative difference with AD is shown as Δ for each of the input parameters ($\partial \rho_{\text{inf}}$ for the flow density derivative, ∂L for the gust length, and ∂w for the gust intensity). The computational wall-time for entire forager loop (concurrent loads, filtering of critical cases and sensitivities on them) was 60.1 and 359.5 seconds on the GPU and CPU respectively.

Table 2: Comparison of peak gust loads at the critical points, AD versus FD

	objective	$\partial \rho_{\text{inf}}$	Δ	∂L	Δ	∂w	Δ
Shear	0.897	1.670	1.6×10^{-5}	-0.0026	4×10^{-6}	0.035	$1. \times 10^{-7}$
Torsion	0.025	-0.018	4.2×10^{-4}	-0.0007	3.9×10^{-7}	0.0018	1.2×10^{-6}
Bending	0.300	0.329	2.9×10^{-5}	0.00003	1.7×10^{-4}	0.012	9.2×10^{-8}

4 Full paper

Further details of implementation, verification and validation will be provided in the full paper. Further results will be included, considering also concurrent simulation of maneuver loads and a demonstration of uncertainty quantification using Monte Carlo analysis on a nonlinear aeroelastic analysis.

References

[ARI26] Alvaro Cea, Rafael Palacios, and Lucian Iorga. Near-Real-Time Nonlinear Analysis of Free-Flying Flexible Aircraft on Modern Hardware Architectures. *AIAA Journal*, 2026.

[BFH+18] 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. https://github.com/google/jax, 2018.

[CP25] Alvaro Cea and Rafael Palacios. JAX-based aeroelastic simulation engine for differentiable aircraft dynamics. *Computer Physics Communications*, 311:109547, June 2025.

[SCN+18] Olivia Stodieck, Jonathan E. Cooper, S. A. Neild, M. H. Lowenberg, and L. Iorga. Slender-Wing Beam Reduction Method for Gradient-Based Aeroelastic Design Optimization. *AIAA Journal*, 56(11):4529–4545, November 2018.