Advanced Genetic Algorithm Optimization for Dynamic Vibration Absorber Design: A Comprehensive Methodology

Master's Thesis Methodology

August 26, 2025

Abstract

DeVana (v0.4.1) is an open-source, first-of-its-kind playground for the design of Dynamic Vibration Absorbers (DVAs) that unifies classical and modern optimization under one reproducible, extensible framework. The platform integrates a rigorous mechanical model (fully coupled 2DOF–3DOF), an Frequency Response Function (FRF)-based fitness pipeline, and a suite of optimizers (Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Simulated Annealing (SA), Covariance Matrix Adaptation Evolution Strategy (CMA-ES), Reinforcement Learning (RL), and Differential Evolution (DE)). Beyond providing many algorithms, DeVana contributes three methodological advances tailored to vibration absorber design: (i) adaptive operator control for GA via ML-bandit and RL controllers that tune crossover, mutation, and population size online; (ii) surrogate-assisted screening using a k-Nearest Neighbors (KNN) predictor to reduce expensive FRF evaluations; and (iii) advanced population seeding (random, Sobol, Latin hypercube, and a NeuralSeeder with Upper Confidence Bound (UCB)/Expected Improvement (EI) acquisition) for improved exploration. The system instruments each run with detailed metrics and provides statistical post-analysis across repeated runs to recommend robust parameter ranges for every DVA variable. We document the mathematical formulation, software architecture, algorithmic controllers, benchmarking protocol, and statistical synthesis pipeline, and we release the full implementation to accelerate research and practice in vibration control.

Keywords: Dynamic vibration absorber, frequency response function, genetic algorithm, CMA-ES, PSO, simulated annealing, differential evolution, reinforcement learning, surrogate modeling, seeding, robustness, reproducibility, open-source.

Software: DeVana v0.4.1 (open-source; available on GitHub and project website).

Contents

1 Introduction and Motivation

This thesis presents DeVana, a comprehensive optimization framework and software platform for designing DVAs with practical applicability in mechanical engineering. The framework addresses the challenges of vast configuration spaces, multi-criteria optimization, and computational efficiency by (i) supporting multiple optimizers within a single configurable playground, (ii) introducing adaptive controllers for GA that learn online, (iii) reducing evaluation cost via surrogate-assisted screening, and (iv) enabling statistical synthesis of recommended parameter ranges across repeated runs.

1.1 Contributions and novelties

1. **Unified optimization playground:** A modular, open-source platform that allows designers to configure and compare GA, PSO, SA, CMA-ES, RL, and DE on identical problems with consistent instrumentation.

2. **Adaptive GA controllers:** Online tuning of crossover, mutation, and population size via an ML-bandit controller and a compact RL agent, overcoming fixed-operator limitations.

3. **Surrogate-assisted screening:** A KNN-based pre-filter that reduces the number of costly FRF evaluations while preserving exploration through novelty.

4. **Advanced seeding:** Random, Sobol, and Latin hypercube Quasi-Monte Carlo (QMC) options plus a NeuralSeeder that proposes individuals using UCB/EI, improving early search quality.

5. **Statistical range synthesis:** A robust pipeline that aggregates multiple independent runs and outputs recommended parameter ranges per DVA variable using complementary criteria (IQR, P5–P95, Tukey, Highest Density Interval (HDI), Top-q, TMAD).

6. **Reproducible engineering workflow:** Versioned release (v0.4.1), open dataset of metrics, and exportable configurations for repeatable studies and fair comparisons.

1.2 Software availability and versioning

DeVana is released under an open-source license at GitHub and the project website. This thesis documents v0.4.1. The repository includes source code, example configurations, and instructions to reproduce all experiments presented herein.

1.3 Thesis roadmap

We begin with the DVA configuration and parameter spaces, followed by the mechanical model and its FRF-based evaluation. We then detail the optimization problem, the traditional GA baseline, and the advanced features. Subsequent sections describe software architecture, the algorithm suite and configuration playground, benchmarking and robustness analysis, statistical range synthesis, and the overall results, before concluding with reproducibility notes and future work.

2 Software architecture and optimization playground

2.1 Design principles

- **Modularity**: separable layers for mechanics, optimization, controllers, seeding, surrogate, Graphical User Interface (GUI), and benchmarking.
- **Reproducibility**: versioned configurations, deterministic seeds on demand, and exportable results.
- **Extensibility**: clean interfaces to add new optimizers, surrogates, and metrics with minimal code changes.
- **Observability**: rich metrics, logs, and artifacts captured per run and per generation.

2.2 System overview

1. **Mechanical core**: builds the mass, damping, stiffness, and force operators and computes FRFs per Sec. **??**.
2. **Fitness pipeline**: evaluates Eq. (22) using modal, inter-modal, and global criteria with user-defined weights.
3. **Algorithm suite**: GA, PSO, SA, CMA-ES, RL, and DE, with consistent bounds, fixed-parameter handling, and termination.
4. **Controllers**: ML-bandit and RL for adaptive GA operator control (Sec. 42–46).
5. **Seeding**: Random, Sobol, LHS, and NeuralSeeder (UCB/EI) as in Sec. 18 onward.
6. **Surrogate screening**: KNN-based pre-filter with novelty (Sec. 8).
7. **Instrumentation**: per-generation and system Key Performance Indicator (KPI)s (Sec. 8).

8. **GUI**: interactive configuration, execution control, visualization (responses, convergence, ranges), and export.

2.3 Optimization playground

Users configure experiments by selecting the optimizer, bounds and fixed masks, objective weights, seeding, controllers, surrogate settings, and stopping criteria. The same mechanical problem can thus be solved by multiple methods, enabling apples-to-apples comparisons with common metrics and visualizations.

2.4 Data, logging, and metrics

At each generation we record time, evaluations, fitness statistics, operator rates, population size, and resource usage, along with artifacts (best solutions, FRFs, convergence traces). Artifacts are exportable as CSV/JSON for downstream analysis.

2.5 Extensibility

New optimizers, surrogates, or controllers implement small interfaces (ask, tell/evaluate, and adapt hooks). New performance metrics can be registered and automatically included in the logging and reporting layers.

2.6 Reproducibility

All experiments store configuration hashes and software version (v0.4.1), aiding peer reproduction. We provide example configurations used for this thesis alongside raw metric exports.

3 Configuration Space of DVA Systems

The design of Dynamic Vibration Absorbers (DVAs) involves selecting appropriate combinations of mechanical components—masses, springs, dampers, and inerters—to attach to a primary system for vibration mitigation. Each component contributes to the system's dynamic behavior, and their combinations result in a multitude of possible configurations.

3.1 Definition of Components and Parameters

Let:

- $\mathcal{M} = \{m_i \mid m_i \in [m_i^{\min}, m_i^{\max}],\ i = 1, \ldots, n_m\}$: Set of mass elements with viable ranges.

- $\mathcal{K} = \{k_j \mid k_j \in [k_j^{\min}, k_j^{\max}],\ j = 1, \ldots, n_k\}$: Set of spring elements with viable stiffness ranges.
- $\mathcal{C} = \{c_l \mid c_l \in [c_l^{\min}, c_l^{\max}],\ l = 1, \ldots, n_c\}$: Set of damping elements with viable damping coefficient ranges.
- $\mathcal{B} = \{b_p \mid b_p \in [b_p^{\min}, b_p^{\max}],\ p = 1, \ldots, n_b\}$: Set of inerter elements with viable inertance ranges.

Each parameter is bounded within a feasible design range, reflecting practical engineering constraints such as material properties, geometric limitations, and manufacturing capabilities.

3.2 Total Number of Configurations

$$N_{\text{total}} = (n_m + n_k + n_c + n_b) \tag{1}$$

$$N_{\text{config}} = \frac{N_{\text{total}} \times (N_{\text{total}} + 1)}{2} \tag{2}$$

This growth underscores the impracticality of exhaustively evaluating every possible configuration due to computational limitations.

4 Parameter Space and Design Variables

For a given configuration s, the parameter vector $\boldsymbol{\theta}_s$ comprises the design variables associated with the included components:

$$\boldsymbol{\theta}_s = [\theta_1, \theta_2, \ldots, \theta_{n_s}]^\top \tag{3}$$

where n_s is the number of parameters in configuration s, and each θ_i corresponds to a component parameter (e.g., mass, stiffness, damping coefficient, or inertance) within its viable range:

$$\theta_i \in [\theta_i^{\min}, \theta_i^{\max}] \tag{4}$$

The feasible parameter space for configuration s is thus defined as:

$$\Theta_s = \prod_{i=1}^{n_s} [\theta_i^{\min}, \theta_i^{\max}] \tag{5}$$

5 Problem Statement

5.1 Mechanical System Definition

5.1.1 Fully Coupled 2DOF - 3DOF System Setup

System Overview and Configuration In order to comprehend the fundamental concepts and operational principles of the advanced genetic algorithm methodology introduced in this work, a comprehensive, step-by-step analysis of the mechanical system is essential. The system under investigation is a sophisticated fully coupled 2DOF - 3DOF system, where "fully coupled" signifies that all system components are interconnected through masses, springs, dampers, and inerters, creating a complete vibrational system with comprehensive dynamic interactions.

The primary structure consists of a 2DOF main system with two primary masses representing different structural elements, augmented by three strategically positioned 1DOF Dynamic Vibration Absorbers (DVAs). This configuration allows for multi-modal vibration control targeting multiple resonant frequencies simultaneously. The system architecture includes:

- **Primary structural elements**: Two primary masses (M_1, M_2) representing the main structural components with their inertial properties
- **Dynamic Vibration Absorbers**: Three DVA masses (μ_1, μ_2, μ_3) with independent dynamic characteristics
- **Base excitations**: Lower and upper base motions providing external vibrational inputs
- **External forcing**: Direct force inputs applied to the primary masses
- **Complete coupling**: All components interconnected through mass, stiffness, damping, and inertial elements

The system's complexity arises from the extensive parameter space and coupling mechanisms. The complete system comprises 48 independent design parameters distributed across:

- 15 mass coupling parameters (β_1 through β_{15}) for inertial interconnections
- 15 stiffness parameters (λ_1 through λ_{15}) for elastic coupling
- 15 damping parameters (ν_1 through ν_{15}) for energy dissipation
- 3 DVA mass parameters (μ_1, μ_2, μ_3) for absorber sizing

This high-dimensional parameter space necessitates advanced optimization techniques capable of efficiently exploring the design domain while maintaining computational tractability.

Vibrational Modeling of the System and Assumptions The main 2DOF system is modeled using a comprehensive framework comprising masses, springs, dampers, and inerters, interconnected with both upper and lower bases. The modeling approach incorporates the following fundamental assumptions:

- **Linear elastic behavior**: All stiffness elements (K_1, K_2, K_3) exhibit linear force-displacement relationships within the operational range
- **Linear viscous damping**: All damping elements (C_1, C_2, C_3) follow linear velocity-dependent force relationships
- **Point mass representation**: All masses are treated as point masses with concentrated inertial properties
- **One-dimensional motion**: All system components move in a single translational direction
- **Time-invariant parameters**: All system parameters remain constant during operation
- **No geometric nonlinearities**: The system operates within the linear regime, excluding geometric stiffening effects
- **Perfect bonding**: All interconnections between components are assumed to be rigid and perfect

The base excitations are modeled with flexibility to represent various mechanical scenarios:

- **Foundation motion**: Representing actual physical ground motion or support excitation
- **Additional system modes**: Modeling extra degrees of freedom from more complex structures
- **Boundary condition variations**: Accommodating different support and mounting conditions

Each DVA is designed as an independent 1DOF system with its own mass, stiffness, damping, and inertial elements. The coupling between the primary system and DVAs, as well as between DVAs themselves, is achieved through comprehensive interconnection elements ensuring complete dynamic coupling.

Derivation of Governing Equations The governing equations for the fully coupled 2DOF-3DOF system are derived using Newton's method, applying Newton's second law to each degree of freedom while accounting for all interconnecting forces. The system's generalized coordinates are defined as:

$$\mathbf{q} = \begin{bmatrix} U_1(t) \\ U_2(t) \\ u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix}; \quad \dot{\mathbf{q}} = \begin{bmatrix} \dot{U}_1(t) \\ \dot{U}_2(t) \\ \dot{u}_1(t) \\ \dot{u}_2(t) \\ \dot{u}_3(t) \end{bmatrix}; \quad \ddot{\mathbf{q}} = \begin{bmatrix} \ddot{U}_1(t) \\ \ddot{U}_2(t) \\ \ddot{u}_1(t) \\ \ddot{u}_2(t) \\ \ddot{u}_3(t) \end{bmatrix} \tag{6}$$

where:

- $U_1(t)$, $U_2(t)$: Displacements of the primary masses at time t
- $u_1(t)$, $u_2(t)$, $u_3(t)$: Displacements of the DVA masses at time t

The equations of motion are expressed in matrix form as:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{F}(t) \tag{7}$$

where:

- $\mathbf{q}$: Generalized displacement vector, capturing displacements of the primary and DVA masses.
- $\mathbf{M}$: Mass matrix.
- $\mathbf{C}$: Damping matrix.
- $\mathbf{K}$: Stiffness matrix.
- $\mathbf{F}(t)$: External force vector, which includes both external loads and base motion effects.

The generalized coordinate vector is defined as:

$$\mathbf{q} = \begin{bmatrix} U_1 \\ U_2 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \tag{8}$$

Mass Matrix

$$[M] = \begin{bmatrix} M_1 + b_1 \\ +b_2 + b_3 & 0 & -b_1 & -b_2 & -b_3 \\ 0 & M_2 + b_4 \\ & +b_5 + b_6 & -b_4 & -b_5 & -b_6 \\ -b_1 & -b_4 & m_1 + b_1 \\ & & +b_4 + b_7 \\ & & +b_8 + b_9 \\ & & +b_{10} & -b_9 & -b_{10} \\ -b_2 & -b_5 & -b_9 & m_2 + b_2 \\ & & & +b_5 + b_9 \\ & & & +b_{11} + b_{12} & -b_{15} \\ -b_3 & -b_6 & -b_{10} & -b_{15} & m_3 + b_3 \\ & & & & +b_6 + b_{10} \\ & & & & +b_{13} + b_{14} \\ & & & & +b_{15} \end{bmatrix} \tag{9}$$

Damping Matrix

$$[C] = \begin{bmatrix} C_1 + C_2 \\ +C_3 & -C_3 & -c_1 & -c_2 & -c_3 \\ -C_3 & C_3 + C_4 \\ & +C_5 + c_4 \\ & +c_5 + c_6 & -c_4 & -c_5 & -c_6 \\ -c_1 & -c_2 & c_1 + c_4 \\ & & +c_7 + c_8 \\ & & +c_9 + c_{10} & -c_9 & -c_{10} \\ -c_2 & -c_5 & -c_9 & c_2 + c_5 \\ & & & +c_9 + c_{11} \\ & & & +c_{12} + c_{15} & -c_{15} \\ -c_3 & -c_6 & -c_{10} & -c_{15} & c_3 + c_6 \\ & & & & +c_{10} + c_{13} \\ & & & & +c_{14} + c_{15} \end{bmatrix} \tag{10}$$

Stiffness Matrix

$$[K] = \begin{bmatrix} K_1 + K_2 \\ +K_3 & -K_3 & -k_1 & -k_2 & -k_3 \\ -K_3 & K_3 + K_4 \\ & +K_5 + k_4 \\ & +k_5 + k_6 & -k_4 & -k_5 & -k_6 \\ -k_1 & -k_2 & k_1 + k_4 \\ & & +k_7 + k_8 \\ & & +k_9 + k_{10} & -k_9 & -k_{10} \\ -k_2 & -k_5 & -k_9 & k_2 + k_5 \\ & & & +k_9 + k_{11} \\ & & & +k_{12} + k_{15} & -k_{15} \\ -k_3 & -k_6 & -k_{10} & -k_{15} & k_3 + k_6 \\ & & & & +k_{10} + k_{13} \\ & & & & +k_{14} + k_{15} \end{bmatrix} \tag{11}$$

Force Vector

$$[F] = \begin{bmatrix} F_1(t) + C_1\dot{U}_{low} + C_2\dot{U}_{upp} + K_1U_{low} + K_2U_{upp} \\ F_2(t) + C_4\dot{U}_{low} + C_5\dot{U}_{upp} + K_4U_{low} + K_5U_{upp} \\ \beta_7\ddot{U}_{low} + \beta_8\ddot{U}_{upp} + 2\zeta_{dc}\omega_{dc}(\nu_7\dot{U}_{low} + \nu_8\dot{U}_{upp}) + \omega_{dc}^2(\lambda_7U_{low} + \lambda_8U_{upp}) \\ \beta_{11}\ddot{U}_{low} + \beta_{12}\ddot{U}_{upp} + 2\zeta_{dc}\omega_{dc}(\nu_{11}\dot{U}_{low} + \nu_{12}\dot{U}_{upp}) + \omega_{dc}^2(\lambda_{11}U_{low} + \lambda_{12}U_{upp}) \\ \beta_{13}\ddot{U}_{low} + \beta_{14}\ddot{U}_{upp} + 2\zeta_{dc}\omega_{dc}(\nu_{13}\dot{U}_{low} + \nu_{14}\dot{U}_{upp}) + \omega_{dc}^2(\lambda_{13}U_{low} + \lambda_{14}U_{upp}) \end{bmatrix} \tag{12}$$

5.1.2 Dimensionless Form

To simplify analysis, the system is normalized using dimensionless parameters listed in Table 1.

Table 1: Dimensionless Parameters for System Normalization

Parameter Group	Parameter	Definition
Mass Ratios	Γ	$\Gamma = \frac{M_2}{M_1}$
	μ_i	$\mu_i = \frac{m_i}{M_1}$
Inertial Coupling Ratios	β_i	$\beta_i = \frac{b_i}{M_1}$
Damping Ratios	$\mathcal{N}_i$	$\mathcal{N}_i = \frac{C_i}{C_1}$
	ν_i	$\nu_i = \frac{c_i}{C_1}$
Stiffness Ratios	Λ_i	$\Lambda_i = \frac{K_i}{K_1}$
	λ_i	$\lambda_i = \frac{k_i}{K_1}$
Decoupled Primary System	ω_{dc}	$\omega_{dc} = \sqrt{\frac{K_1}{M_1}}$
	ζ_{dc}	$\zeta_{dc} = \frac{C_1}{2M_1\omega_{dc}}$

Using these parameters, the dimensionless equations of motion are expressed as:

$$\bar{\mathbf{M}}\ddot{\mathbf{q}} + 2\zeta_{dc}\omega_{dc}\bar{\mathbf{C}}\dot{\mathbf{q}} + \omega_{dc}^2\bar{\mathbf{K}}\mathbf{q} = \bar{\mathbf{F}}(t) \tag{13}$$

The dimensionless mass, damping, and stiffness matrices, along with the force vector, are defined in Equations (14) to (17).

Dimensionless Mass Matrix

$$[\bar{M}] = \begin{bmatrix} 1+\beta_1 \\ +\beta_2+\beta_3 & 0 & -\beta_1 & -\beta_2 & -\beta_3 \\ 0 & \Gamma+\beta_4 \\ & +\beta_5+\beta_6 & -\beta_4 & -\beta_5 & -\beta_6 \\ & & \mu_1+\beta_1 \\ & & +\beta_4+\beta_7 \\ & & +\beta_8+\beta_9 \\ -\beta_1 & -\beta_4 & +\beta_{10} & -\beta_9 & -\beta_{10} \\ & & & \mu_2+\beta_2 \\ & & & +\beta_5+\beta_9 \\ -\beta_2 & -\beta_5 & -\beta_9 & +\beta_{11}+\beta_{12} & -\beta_{15} \\ & & & & \mu_3+\beta_3 \\ & & & & +\beta_6+\beta_{10} \\ & & & & +\beta_{13}+\beta_{14} \\ -\beta_3 & -\beta_6 & -\beta_{10} & -\beta_{15} & +\beta_{15} \end{bmatrix} \tag{14}$$

Dimensionless Damping Matrix

$$[\bar{C}] = \begin{bmatrix} 1+\mathcal{N}_2 \\ +\mathcal{N}_3+\nu_1 \\ +\nu_2+\nu_3 & -\mathcal{N}_3 & -\nu_1 & -\nu_2 & -\nu_3 \\ & \mathcal{N}_3+\mathcal{N}_4 \\ & +\mathcal{N}_5+\nu_4 \\ -\mathcal{N}_3 & +\nu_5+\nu_6 & -\nu_4 & -\nu_5 & -\nu_6 \\ & & \nu_1+\nu_4 \\ & & +\nu_7+\nu_8 \\ -\nu_1 & -\nu_2 & +\nu_9+\nu_{10} & -\nu_9 & -\nu_{10} \\ & & & \nu_2+\nu_5 \\ & & & +\nu_9+\nu_{11} \\ -\nu_2 & -\nu_5 & -\nu_9 & +\nu_{12}+\nu_{15} & -\nu_{15} \\ & & & & \nu_3+\nu_6 \\ & & & & +\nu_{10}+\nu_{13} \\ -\nu_3 & -\nu_6 & -\nu_{10} & -\nu_{15} & +\nu_{14}+\nu_{15} \end{bmatrix} \tag{15}$$

Dimensionless Stiffness Matrix

$$[\bar{K}] = \begin{bmatrix} 1+\Lambda_2 \\ +\Lambda_3+\lambda_1 \\ +\lambda_2+\lambda_3 & -\Lambda_3 & -\lambda_1 & -\lambda_2 & -\lambda_3 \\ & \Lambda_3+\Lambda_4 \\ & +\Lambda_5+\lambda_4 \\ -\Lambda_3 & +\lambda_5+\lambda_6 & -\lambda_4 & -\lambda_5 & -\lambda_6 \\ & & \lambda_1+\lambda_4 \\ & & +\lambda_7+\lambda_8 \\ -\lambda_1 & -\lambda_2 & +\lambda_9+\lambda_{10} & -\lambda_9 & -\lambda_{10} \\ & & & \lambda_2+\lambda_5 \\ & & & +\lambda_9+\lambda_{11} \\ -\lambda_2 & -\lambda_5 & -\lambda_9 & +\lambda_{12}+\lambda_{15} & -\lambda_{15} \\ & & & & \lambda_3+\lambda_6 \\ & & & & +\lambda_{10}+\lambda_{13} \\ -\lambda_3 & -\lambda_6 & -\lambda_{10} & -\lambda_{15} & +\lambda_{14}+\lambda_{15} \end{bmatrix} \tag{16}$$

Dimensionless Force Vector

$$[\bar{F}] = \begin{bmatrix} \frac{F_1(t)}{M_1} + 2\zeta_{dc}\omega_{dc}(\dot{U}_{low} + \mathcal{N}_2\dot{U}_{upp}) + \omega_{dc}^2(U_{low} + \Lambda_2 U_{upp}) \\ \frac{F_2(t)}{M_1} + 2\zeta_{dc}\omega_{dc}(\mathcal{N}_4\dot{U}_{low} + \mathcal{N}_5\dot{U}_{upp}) + \omega_{dc}^2(\Lambda_4 U_{low} + \Lambda_5 U_{upp}) \\ \beta_7\ddot{U}_{low} + \beta_8\ddot{U}_{upp} + 2\zeta_{dc}\omega_{dc}(\nu_7\dot{U}_{low} + \nu_8\dot{U}_{upp}) + \omega_{dc}^2(\lambda_7 U_{low} + \lambda_8 U_{upp}) \\ \beta_{11}\ddot{U}_{low} + \beta_{12}\ddot{U}_{upp} + 2\zeta_{dc}\omega_{dc}(\nu_{11}\dot{U}_{low} + \nu_{12}\dot{U}_{upp}) + \omega_{dc}^2(\lambda_{11} U_{low} + \lambda_{12} U_{upp}) \\ \beta_{13}\ddot{U}_{low} + \beta_{14}\ddot{U}_{upp} + 2\zeta_{dc}\omega_{dc}(\nu_{13}\dot{U}_{low} + \nu_{14}\dot{U}_{upp}) + \omega_{dc}^2(\lambda_{13} U_{low} + \lambda_{14} U_{upp}) \end{bmatrix} \tag{17}$$

Semi-Analytical Solutions for Harmonic Excitation The semi-analytical method assumes harmonic excitation and synchronized motion. The system response is expressed as:

$$\begin{bmatrix} U_1(t) \\ U_2(t) \\ u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} e^{j\omega t} \tag{18}$$

where A_1, A_2 are the vibration amplitudes of the primary masses and a_1, a_2, a_3 are the amplitudes of the DVA masses.

The harmonic excitations are defined as:

$$\begin{aligned} F_1(t) &= F_1 e^{j\omega t} \\ F_2(t) &= F_2 e^{j\omega t} \\ U_{Low}(t) &= A_{Low} e^{j\omega t} \\ U_{Up}(t) &= A_{Up} e^{j\omega t} \end{aligned} \tag{19}$$

Substituting the harmonic solutions into the equations of motion yields the frequency domain formulation:

$$\begin{bmatrix} A_1 \\ A_2 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \omega_{dc}^2 \left(-\Omega^2 \mathbf{M} + j2\zeta_{dc}\Omega \mathbf{C} + \mathbf{K}\right)^{-1} \mathbf{F} \tag{20}$$

where $\mathbf{F}$ is the complex amplitude vector of the forcing function.

5.2 Optimization Problem Formulation

Building upon the complex 2DOF-3DOF mechanical system described in the previous section, the optimization problem is formulated to find optimal Dynamic Vibration Absorber (DVA) parameters that minimize deviations from desired system performance characteristics. The system comprises 48 independent design parameters distributed across 15 mass coupling coefficients (β_1 through β_{15}), 15 stiffness parameters (λ_1 through λ_{15}), 15 damping parameters (ν_1 through ν_{15}), and 3 DVA mass parameters (μ_1, μ_2, μ_3).

The optimization problem can be mathematically stated as:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) = f_{primary}(\mathbf{x}) + f_{sparsity}(\mathbf{x}) + f_{error}(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x}_L \leq \mathbf{x} \leq \mathbf{x}_U \end{aligned} \tag{21}$$

where:

- $\mathbf{x} \in \mathbb{R}^{48}$ is the design parameter vector
- $\mathbf{x}_L, \mathbf{x}_U \in \mathbb{R}^{48}$ are the lower and upper parameter bounds
- $f_{primary}(\mathbf{x})$, $f_{sparsity}(\mathbf{x})$, $f_{error}(\mathbf{x})$ are the objective function components

5.2.1 Objective Function Definition

The overall objective function is a weighted sum of three distinct components, each addressing different aspects of the optimization problem:

$$f(\mathbf{x}) = f_{primary}(\mathbf{x}) + f_{sparsity}(\mathbf{x}) + f_{error}(\mathbf{x}) \tag{22}$$

where each component is precisely defined and serves a specific purpose in the optimization process.

Primary Objective Function ($f_{primary}(\mathbf{x})$): The primary objective measures the deviation of the system's singular response from the optimal target value of 1.0:

$$f_{primary}(\mathbf{x}) = |C_s(\mathbf{x}) - 1.0| \tag{23}$$

where the singular criteria $C_s(\mathbf{x})$ is computed as:

$$C_s(\mathbf{x}) = \sum_{i=1}^{5} CM_i(\mathbf{x}) \tag{24}$$

and $CM_i(\mathbf{x})$ is the composite measure for mass i:

$$CM_i(\mathbf{x}) = \sum_{j} w_{ij} \cdot \frac{a_{ij}(\mathbf{x})}{t_{ij}} \tag{25}$$

The composite measure integrates multiple performance criteria for each mass, where:

- w_{ij}: Weight coefficient for criterion j of mass i
- $a_{ij}(\mathbf{x})$: Actual performance value for criterion j of mass i
- t_{ij}: Target performance value for criterion j of mass i

The performance criteria include peak positions, peak values, bandwidths, slopes, and area under the curve, as extracted from the FRF analysis. The weights allow prioritization of different performance aspects, with typical values ranging from 0.05 to 1.0 depending on the importance of each criterion.

Sparsity Penalty Function ($f_{sparsity}(\mathbf{x})$): The sparsity penalty encourages solutions with smaller parameter magnitudes:

$$f_{sparsity}(\mathbf{x}) = \alpha \sum_{k=1}^{48} |x_k| \tag{26}$$

where:

- α is the sparsity weight coefficient
- x_k represents the k-th design parameter
- The L1 regularization promotes sparse solutions by penalizing non-zero parameter values

This function serves multiple purposes:

1. **Regularization**: Prevents overfitting to specific frequency ranges by discouraging overly complex parameter combinations
2. **Practical Implementation**: Encourages simpler DVA configurations that are easier to manufacture and maintain
3. **Robustness Enhancement**: Reduces sensitivity to parameter variations in real-world applications

Percentage Error Component ($f_{error}(\mathbf{x})$): The percentage error component captures detailed performance deviations:

$$f_{error}(\mathbf{x}) = \frac{1}{\gamma} \sum_{i} \sum_{j} |PD_{ij}(\mathbf{x})| \tag{27}$$

$$PD_{ij}(\mathbf{x}) = \left(\frac{a_{ij}(\mathbf{x}) - t_{ij}}{|t_{ij}|} \right) \times 100\% \tag{28}$$

where:

- γ is the scaling factor (default: 1000)
- $PD_{ij}(\mathbf{x})$ is the percentage difference for criterion j of mass i
- The absolute value prevents cancellation between positive and negative errors

This component ensures comprehensive evaluation by:

- Capturing all performance criteria deviations in the objective function
- Enabling consideration of both primary response characteristics and detailed performance metrics
- Allowing different criteria to have varying importance through target value specification
- Providing a normalized error measure that can be compared across different criteria types
- Balancing the contribution of detailed metrics with the primary objective through the scaling factor γ

The scaling factor γ plays a crucial role in balancing the contribution of the percentage error component with the other objectives. A larger γ reduces the influence of detailed percentage errors, while a smaller γ increases their importance in the overall optimization.

5.2.2 Performance Criteria Hierarchy

The multi-objective framework evaluates performance across several hierarchical categories:

1. **Modal Performance Criteria:**
    - Peak positions ($\omega_{peak,i}$): Target resonant frequencies for each mass
    - Peak values ($A_{peak,i}$): Target response amplitudes at resonant frequencies
    - Critical for modal alignment and vibration control effectiveness
2. **Inter-Modal Criteria:**
    - Bandwidths ($\Delta\omega_{i,j}$): Target frequency ranges between resonances
    - Slopes ($s_{i,j}$): Rate of change between peaks
    - Affects system stability and control bandwidth
3. **Global Response Criteria:**
    - Area under curve: Total response energy across frequency range

- Maximum slope: Maximum rate of amplitude change
- Provides overall system response characteristics

Weight Assignment Strategy: The assignment of weights to each objective function is a critical aspect of the optimization process, as it directly influences the balance between different performance criteria in the total fitness function. In this methodology, the program calculates $C_s - 1$ as one of the objective functions, and it is important to ensure that the contribution of each objective is properly normalized.

To achieve a meaningful and interpretable fitness value, it is recommended that the sum of all weights assigned to the objectives does not exceed 1, and ideally, all weights should sum exactly to 1. This normalization ensures that each objective's influence is proportional and that the overall fitness function is calculated correctly, preventing any single criterion from dominating the optimization process due to disproportionate weighting.

The general structure for assigning weights is as follows:

$$w_{ij} = w_i \cdot w_{base,j} \tag{29}$$

where:

- w_i: Weight associated with a specific mass or subsystem (often set to 1.0 for uniform treatment, but can be adjusted for prioritization)
- $w_{base,j}$: Base weight assigned to each criterion type j (e.g., peak position, peak value, bandwidth, slope, area, etc.)

The sum of all w_{ij} across all masses and criteria should satisfy:

$$\sum_{i,j} w_{ij} \leq 1 \tag{30}$$

and, for best normalization, it is preferable to enforce

$$\sum_{i,j} w_{ij} = 1 \tag{31}$$

This approach ensures that the fitness function remains consistent and interpretable, especially when combining $C_s - 1$ with other objectives. However, since the program is open-source, users have the flexibility to modify the weighting scheme as needed for their specific application or research focus. Adjusting the weights allows for custom prioritization of objectives, but care should be taken to maintain the normalization condition for optimal performance and comparability of results.

Objective Function Interactions:

In multi-objective optimization, especially in the context of genetic algorithms for engineering design, the total objective function $f_{total}(\mathbf{x})$ is typically composed of several distinct terms, each representing a different aspect of system performance or a different design goal. These terms may include, for example, a primary performance objective (such as minimizing vibration amplitude), a sparsity-promoting penalty (to encourage simpler or more efficient designs), and an error or constraint penalty (to penalize infeasible or undesirable solutions).

5.2.3 Fitness Evaluation through FRF Analysis

FRF Mathematical Foundation: The FRF analysis is based on the frequency domain representation of the system dynamics:

$$\mathbf{X}(\omega) = \mathbf{G}(\omega)\mathbf{F}(\omega) \tag{32}$$

where:

- $\mathbf{X}(\omega) \in \mathbb{C}^5$: Displacementresponse vector (5 DOF)
- $\mathbf{G}(\omega) \in \mathbb{C}^{5\times5}$: Frequency response function matrix
- $\mathbf{F}(\omega) \in \mathbb{C}^5$: Forcing function vector including external and base excitations

The frequency response function matrix is computed as:

$$\mathbf{G}(\omega) = \left[-\omega^2\mathbf{M} + j\omega\mathbf{C} + \mathbf{K}\right]^{-1} \tag{33}$$

where $\mathbf{M}$, $\mathbf{C}$, $\mathbf{K}$ are the system matrices defined in the mechanical system formulation.

Performance Metric Calculations: Each mass response undergoes comprehensive analysis:

1. **Peak Frequency (X-Value) Analysis:**

$$\omega_{peak,i} = \omega_{j^*} \quad \text{where} \quad j^* = \arg\max_j |X_i(\omega_j)|, \quad \forall i = 1, \ldots, 5 \tag{34}$$

2. **Peak Amplitude (Y-Value) Analysis:**

$$A_{peak,i} = |X_i(\omega_{peak,i})| \quad \forall i = 1, \ldots, 5 \tag{35}$$

3. **Bandwidth Analysis:**

$$\Delta\omega_{i,j} = |\omega_{peak,j} - \omega_{peak,i}| \quad \forall i < j \tag{36}$$

4. **Slope Analysis:**

$$s_{i,j} = \frac{A_{peak,j} - A_{peak,i}}{\omega_{peak,j} - \omega_{peak,i}} \quad \forall i < j \tag{37}$$

5. **Area Analysis:**

$$Area_i = \int_{\omega_{start}}^{\omega_{end}} |X_i(\omega)| d\omega \quad \forall i = 1, \ldots, 5 \tag{38}$$

The area integral is computed using Simpson's rule for numerical accuracy:

$$Area_i \approx \frac{h}{3} \left[|X_i(\omega_0)| + 4 \sum_{k=1}^{n/2} |X_i(\omega_{2k-1})| + 2 \sum_{k=1}^{n/2-1} |X_i(\omega_{2k})| + |X_i(\omega_n)| \right] \tag{39}$$

where h is the frequency step size and n is the number of frequency points.

6 Traditional Genetic Algorithm Methodology

6.1 Algorithmic Framework

The baseline GA follows the classic evolve–evaluate–select loop with real-valued representations tailored to the DVA design vector $\mathbf{x}$ and FRF-based fitness in Eq. (22). Individuals are length-P floating vectors, bounds are enforced by projection, and fixed parameters are respected.

Representation and constraints

- **Genome**: $\mathbf{x} = [x_1, \ldots, x_P]^\top$ with $x_i \in [\theta_i^{\min}, \theta_i^{\max}]$; fixed parameters are pinned to their fixed values.
- **Projection to bounds**: after any variation, each gene is clamped

$$x_i \leftarrow \min\{\, \theta_i^{\max},\ \max\{\theta_i^{\min},\ x_i\}\,\}, \quad i = 1, \ldots, P. \tag{40}$$

Fitness evaluation Each individual is evaluated by the FRF pipeline (Sec. **??**), yielding the composite singular criterion C_s in Eq. (24) and the detailed percentage differences in Eq. (27). The total fitness is Eq. (22):

$$f(\mathbf{x}) = \big|C_s(\mathbf{x}) - 1.0\big| \ + \ \alpha \sum_{k=1}^{P} |x_k| \ + \ \frac{1}{\gamma} \sum_{i,j} \big|PD_{ij}(\mathbf{x})\big|. \tag{41}$$

Selection, crossover, mutation, replacement

- **Selection**: tournament selection with size 3 (balance pressure and diversity).

- **Crossover**: *Blend* crossover (BLX-α) with $\alpha = 0.5$ acts gene-wise on pairs.
- **Mutation**: per-gene perturbation with probability indpb $= 0.1$, using a uniform perturbation bounded to 10% of each gene span; then projection (Eq. 40).
- **Replacement**: generational replacement of the full population with the offspring.

Termination The loop terminates when either (i) the best fitness reaches the tolerance $f_{\min} \leq \texttt{ga\_tol}$, or (ii) the generation budget $G_{\max}$ is exhausted.

6.2 Baseline GA for DVA Design

Algorithm 1 Baseline Genetic Algorithm for DVA parameter optimization

Require: Bounds $[\theta^{\min}, \theta^{\max}]$, fixed mask/values, population size N, generations $G_{\max}$, cxpb, mutpb, indpb, tolerance τ

Initialize population $\mathcal{P}_0$ of N individuals uniformly in bounds; set fixed genes.
for $g = 1$ to $G_{\max}$ **do**
  Evaluate each $\mathbf{x} \in \mathcal{P}_{g-1}$ via FRF to compute $f(\mathbf{x})$.
  if $\min_{\mathbf{x} \in \mathcal{P}_{g-1}} f(\mathbf{x}) \leq \tau$ **then**
    break
  end if
  $\mathcal{M} \leftarrow$ tournament-select N parents from $\mathcal{P}_{g-1}$ (size 3).
  Form offspring $\mathcal{O}$ by pairing parents and applying BLX-0.5 with prob. cxpb.
  for each offspring $\mathbf{x} \in \mathcal{O}$ **do**
    if rand $<$ mutpb **then**
      mutate each gene with prob. indpb by a uniform perturbation in $\pm 0.1\,(\theta_i^{\max} - \theta_i^{\min})$; enforce fixed genes and project with Eq. (40)
    end if
  end for
  Evaluate invalid individuals in $\mathcal{O}$; set $\mathcal{P}_g \leftarrow \mathcal{O}$.
end for
return best individual and fitness.

6.3 Complexity and limitations

Let C_{FRF} be the average cost of one FRF evaluation and N the population size. The time per generation is $\Theta(N\, C_{\mathrm{FRF}}) + \mathcal{O}(N)$; crossover/mutation are linear. Classical drawbacks are: (i) premature convergence under fixed operators, (ii) sensitivity to hyperparameters, (iii) costly fitness evaluations, and (iv) random seeding that may under-sample high-quality regions. These motivate the advanced controllers and screening introduced next.

7 Advanced Genetic Algorithm Methodology

7.1 Overview of Advanced Features

DeVana extends the baseline GA with controllers and samplers that adapt online: (i) **adaptive operator control** via an ML bandit or a lightweight RL agent that tunes crossover cxpb, mutation mutpb, and population size N generation-by-generation; (ii) **advanced seeding** with Sobol/LHS low-discrepancy designs and a **NeuralSeeder** that learns promising regions using UCB or EI; and (iii) **surrogate-assisted screening** that pre-filters offspring using a KNN surrogate with novelty-based exploration. All mechanisms respect fixed parameters and bounds.

7.2 Machine Learning Bandit Controller for Parameter Adaptation

The controller defines a discrete action space $\mathcal{A} = \{(\delta_{\text{cx}}, \delta_{mu}, \pi)\}$ with relative changes to cxpb and mutpb and a multiplier π for population size. At generation t, it selects

$$a_t \;=\; \arg\max_{a\in\mathcal{A}} \Big[\hat{R}_t(a) \;+\; c\sqrt{\tfrac{\ln t}{N_t(a)}}\Big], \tag{42}$$

where $\hat{R}_t(a)$ is the average reward observed for action a, $N_t(a)$ its count, and $c > 0$ controls exploration. The chosen action updates

$$\text{cxpb}_{t+1} = \text{clip}(\text{cxpb}_t\,(1+\delta_{\text{cx}}),\ [\text{cxpb}_{\min}, \text{cxpb}_{\max}]), \tag{43}$$

$$\text{mutpb}_{t+1} = \text{clip}(\text{mutpb}_t\,(1+\delta_{\mu}),\ [\text{mutpb}_{\min}, \text{mutpb}_{\max}]), \tag{44}$$

$$N_{t+1} = \text{round}(\text{clip}(\pi\, N_t,\ [N_{\min}, N_{\max}])). \tag{45}$$

Reward shaping Consistent with the implementation, the reward balances improvement, speed, diversity, and effort:

$$R_t \;=\; \frac{\max\{0,\ f^\star_{t-1} - f^\star_t\}}{\max\{\epsilon,\ T_t\}}\,\frac{1}{\max\{1,\ E_t\}} \;-\; w_{\text{div}}\,\big|\text{CV}_t - \text{CV}^{\text{target}}\big|, \tag{46}$$

where $f^\star_t$ is the best fitness in generation t, T_t is generation time, E_t the number of evaluations, and $\text{CV}_t = \frac{\sigma_t}{|\mu_t|+\epsilon}$ is the coefficient of variation of fitness. A blended estimator $\tilde{R}_t = w_{\text{hist}}\,\hat{R}_{t-1}(a) + w_{\text{cur}}\,R_t$ stabilizes updates.

Algorithm 2 ML-Bandit controller (per generation)

1: Compute statistics $\mu_t, \sigma_t, f^\star_t, T_t, E_t$ from the finished generation.
2: For each $a \in \mathcal{A}$, compute UCB score via Eq. (42); select a_t.
3: Update $(\text{cxpb}, \text{mutpb}, N)$ by the selected action with clipping.
4: Observe R_t via Eq. (46), update counts and averages for a_t.

7.3 Reinforcement Learning-Based Control of GA Parameters

A compact Q-learning agent with an ϵ-greedy policy selects actions $a \in \mathcal{A}$ conditioned on a coarse state s (e.g., improvement/no-improvement). The update is

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(R + \gamma \max_{a'} Q(s',a') - Q(s,a)\right), \tag{47}$$

with learning rate α, discount γ, and exploration ϵ decayed each generation. Actions map to the same $(\delta_{\text{cx}}, \delta_{\mu}, \pi)$ triplets and are clipped to guardrails as above.

Algorithm 3 RL controller (per generation)

1: Observe state s_t (e.g., $\mathbb{1}[f_t^\star < f_{t-1}^\star]$).
2: With prob. ϵ select random a_t, else $\arg\max_a Q(s_t, a)$.
3: Apply a_t to update $(\text{cxpb}, \text{mutpb}, N)$ with clipping.
4: Receive reward R_t per Eq. (46), observe s_{t+1}; update Q.
5: Decay $\epsilon \leftarrow \epsilon \cdot \epsilon_{\text{decay}}$.

7.4 Sobol and Latin Hypercube Seeding

For $d = P$ free parameters with bounds ℓ_i, u_i, a QMC engine generates m points $\mathbf{z}_k \in [0,1]^d$ using either Sobol or LHS, then scales

$$x_{k,i} \;=\; \ell_i + z_{k,i}\,(u_i - \ell_i), \quad i = 1, \ldots, d,\; k = 1, \ldots, m, \tag{48}$$

and overwrites fixed coordinates. QMC seeding improves space-filling vs. pure random initialization.

7.5 Neural Network-Based Population Seeding (NeuralSeeder)

The NeuralSeeder maintains a dataset $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}$ of evaluated points and trains a lightweight ensemble to model $y \approx f(\mathbf{x})$. It proposes candidates by optimizing an acquisition function such as UCB or EI over a candidate pool, with exploration fraction ϵ and UCB scale β adapted online:

$$\text{UCB}(\mathbf{x}) = \mu(\mathbf{x}) - \beta\,\sigma(\mathbf{x}) \quad \text{(minimization)}, \tag{49}$$

$$\text{EI}(\mathbf{x}) = \mathbb{E}\big[\max\{0, f^\star - Y(\mathbf{x})\}\big]. \tag{50}$$

A diversity constraint (minimum distance in normalized space) avoids mode collapse. When the controller resizes the population, additional individuals are drawn from NeuralSeeder once it has seen enough data; otherwise it falls back to the configured seeding method.

7.6 Surrogate Model-Assisted Fitness Evaluation

To reduce FRF calls, a KNN surrogate pre-screens offspring. Let $\tilde{\mathbf{x}}$ be a candidate and define normalized coordinates $z_i = (x_i - \ell_i)/(u_i - \ell_i)$. The KNN predictor is

$$\hat{f}(\tilde{\mathbf{x}}) = \frac{1}{k} \sum_{(\mathbf{x}^{(n)}, y^{(n)}) \in \mathcal{N}_k(\tilde{\mathbf{x}})} y^{(n)}, \quad \mathcal{N}_k(\tilde{\mathbf{x}}) = \arg\min_{\mathcal{S},\, |\mathcal{S}|=k} \sum_{(\mathbf{x},\cdot)\in\mathcal{S}} \|\mathbf{z} - \tilde{\mathbf{z}}\|_2. \tag{51}$$

A pool of size $M = \lceil \phi\, Q \rceil$ (with Q invalid offspring and pool factor $\phi \geq 1$) is generated by cloning and light genetic operations; the top q by $\hat{f}$ are *exploited* and an *explore* subset is added by maximizing novelty

$$\mathcal{N}(\tilde{\mathbf{x}}) = \min_{\mathbf{x}^{(n)} \in \mathcal{D}} \|\tilde{\mathbf{z}} - \mathbf{z}^{(n)}\|_2. \tag{52}$$

Only the chosen subset is evaluated by FRF; others are discarded, yielding substantial savings when FRF is expensive.

Algorithm 4 Surrogate screening per generation

Require: Invalid offspring set $\mathcal{U}$ of size Q, pool factor ϕ, explore fraction η, KNN k

1: Build pool $\mathcal{P}$ of size $M = \max\{Q, \lceil \phi Q \rceil\}$ by cloning $\mathcal{U}$ and applying light crossover/-mutation (with bounds and fixed genes).
2: Predict $\hat{f}$ for all $\mathbf{x} \in \mathcal{P}$; sort ascending.
3: Select $q = Q$ exploit candidates with smallest $\hat{f}$.
4: From the remainder, select ηq most novel by maximizing $\mathcal{N}(\cdot)$; add to chosen set.
5: Evaluate FRF only on the chosen set; fill remaining offspring slots by best evaluated if needed.

7.7 Legacy adaptive rates

As a fallback, a heuristic adapts $(\text{cxpb}, \text{mutpb})$ based on diversity $\sigma/|\mu|$ and stagnation counter: low diversity increases mutation and decreases crossover, high diversity does the opposite, with periodic adjustments.

8 Benchmarking and Robustness Analysis

DeVana instruments the optimization with detailed metrics to quantify efficiency and robustness across controllers and seeding strategies.

8.1 Recorded metrics

- **Per generation**: best/mean/std fitness, population size, $(\text{cxpb}, \text{mutpb})$, evaluations, time breakdown (selection/crossover/mutation/evaluation), total generation time.

- **System**: CPU%, per-core CPU, memory usage/details, I/O counters, network, thread count.
- **Controllers**: action histories and rewards for ML-bandit/RL; rate/adaptation history.

8.2 Key performance indicators (KPIs)

Let $f_g^\star$ be best fitness at generation g and G the total generations.

$$g_\tau = \min\{g : \ f_g^\star \le \tau\} \quad \text{(time-to-tolerance)}, \tag{53}$$

$$\text{AUC} = \sum_{g=1}^{G} f_g^\star \quad \text{(area under best-fitness curve; lower is better)}, \tag{54}$$

$$\bar{T} = \frac{1}{G}\sum_{g=1}^{G} T_g, \quad \bar{E} = \frac{1}{G}\sum_{g=1}^{G} E_g, \tag{55}$$

$$\Delta f_g = \max\{0, f_{g-1}^\star - f_g^\star\}, \quad \overline{\Delta f} = \frac{1}{G-1}\sum_{g=2}^{G} \Delta f_g. \tag{56}$$

Reliability is reported as the fraction of runs achieving $f_G^\star \le \tau$.

8.3 Benchmark design

We adopt a factorial study over controllers (Fixed/Adaptive/ML-Bandit/RL) and seeders (Random/Sobol/LHS/Neural), with and without surrogate screening. For each cell, run R replicates with different seeds; collect KPIs and resource metrics.

Ablation studies We perform ablations by enabling one advanced feature at a time: (i) adaptive controllers only; (ii) surrogate screening only; (iii) advanced seeding only; and (iv) combinations. This isolates the marginal impact of each component.

Sensitivity analyses We probe sensitivity to: (a) controller exploration constants, (b) surrogate pool factor ϕ and k, (c) NeuralSeeder exploration fraction and acquisition scale, and (d) objective weights. Outcomes are reported on g_τ, AUC, and final best fitness.

External validity We validate across multiple mechanical scenarios (mass ratios, damping regimes, and different target criteria) to ensure conclusions generalize beyond a single configuration.

8.4 Statistical analysis

Report medians and IQRs across replicates; compare cells using non-parametric tests (e.g., Wilcoxon) and bootstrap 95% CIs for median differences of AUC and final best fitness. When multiple factors are varied, use aligned ranks or a permutation-based two-way ANOVA analogue on AUC.

8.5 Reporting

Provide: (i) convergence traces with shaded IQR, (ii) boxplots of KPIs, (iii) Pareto plots of final fitness vs. wall-time/evaluations, (iv) controller rate/population trajectories, (v) resource timelines, and (vi) ablation/sensitivity summaries.

9 Case studies and validation

We demonstrate DeVana on representative scenarios: (i) baseline 2DOF–3DOF configuration with two primary targets; (ii) high-damping regime; (iii) multi-peak realignment with stringent slope uniformity; and (iv) sensitivity to inerter couplings. For each case we compare the algorithm suite, with and without advanced features, and report KPIs and FRF visualizations.

9.1 Experimental setup

We fix bounds, targets, and weight schema per Sec. **??**. Each optimizer uses identical stopping criteria and evaluation budgets. Controllers and surrogates follow their default guardrails unless stated.

9.2 Results overview

Across cases, adaptive GA controllers reduce time-to-tolerance and evaluations, surrogate screening yields the largest savings when FRF is most expensive, and NeuralSeeder improves early convergence and reliability. Results are detailed in Sec. 8.

10 Statistical synthesis of recommended DVA parameter ranges

10.1 Motivation and overview

Multiple independent GA runs provide a sample of optimized parameter vectors that encode how the system converges under different random seeds, seeding strategies, and

adaptive controllers. Rather than reporting a single point estimate, we construct statistically robust *recommended ranges* for every design parameter. These ranges capture where high-performing solutions concentrate, improve interpretability, and guide robust engineering choices.

This section details a step-by-step, statistically grounded procedure that aggregates results from many runs and produces parameter ranges using several complementary criteria. It also defines how to compare, combine, and visualize these ranges, and it gives principled rules for selecting the final recommended interval per parameter.

10.2 Data model and notation

Let there be R benchmark runs of the GA. Each run $r \in \{1, \dots, R\}$ returns:

- Best solution vector $\mathbf{x}^{(r)} = [x_1^{(r)}, \dots, x_P^{(r)}]^\top$ of dimension P (number of design parameters),
- Best fitness value $f^{(r)}$ (lower is better),
- Parameter name list $\{p_1, \dots, p_P\}$ consistent across runs.

For each parameter p_j we collect the empirical sample

$$\mathcal{X}_{p_j} = \{\, x_j^{(1)}, x_j^{(2)}, \dots, x_j^{(R)} \,\}, \qquad j = 1, \dots, P. \tag{57}$$

We also define an optional performance subset using the q-quantile of fitness:

$$\mathcal{I}_q = \{\, r : \; f^{(r)} \le Q_f(q) \,\}, \quad \mathcal{X}_{p_j}^q = \{\, x_j^{(r)} : \; r \in \mathcal{I}_q \,\}, \tag{58}$$

where $Q_f(q)$ is the q-quantile of $\{f^{(r)}\}_{r=1}^R$.

10.3 Preprocessing and quality control

Prior to range estimation we apply the following checks:

P1 Validation: discard runs with missing or incompatible dimension P, or non-finite entries.

P2 Alignment: ensure parameter names $\{p_j\}$ align across runs.

P3 Optional winsorization: for visual diagnostics only, we may clip the top/bottom 1% to prevent extreme values from dominating plots; the statistical criteria below are robust and do not require it.

10.4 Range estimation criteria

To provide statistically meaningful and robust recommended ranges for each design parameter p_j, we compute six complementary interval criteria. Each criterion yields an interval $[L_j^{(c)}, U_j^{(c)}]$ for parameter p_j, where c indexes the criterion. These criteria are implemented in the GUI (see `codes/gui/main_window/ga_mixin.py`, e.g., `_iqr_range`, `_p5_p95`, `_tukey_whisker`, `_shortest_interval`, `_top_quantile_p5_p95`, `_trimmed_mean_mad`) and operate directly on the empirical samples of each parameter, as produced by `GAWorker.finished`. Below, we describe each criterion in detail, including the meaning of all parameters and their statistical motivation.

10.4.1 Interquartile Range (IQR, Q1–Q3)

$$L_j^{\mathrm{IQR}} = Q_{x_j}(0.25), \qquad U_j^{\mathrm{IQR}} = Q_{x_j}(0.75). \tag{59}$$

Here, $Q_{x_j}(q)$ denotes the q-th quantile of the sample $\mathcal{X}_{p_j}$ for parameter p_j. The IQR interval captures the central 50% of the observed values for p_j, i.e., the range between the first quartile (25th percentile) and the third quartile (75th percentile). This method is highly robust to outliers and is especially effective when the distribution of p_j is unimodal or only mildly skewed. By focusing on the middle half of the data, it provides a compact summary of where most solutions concentrate, but it intentionally ignores the tails of the distribution.

10.4.2 P5–P95 Quantile Interval

$$L_j^{\mathrm{P5–P95}} = Q_{x_j}(0.05), \qquad U_j^{\mathrm{P5–P95}} = Q_{x_j}(0.95), \tag{60}$$

This interval is defined by the 5th and 95th percentiles of the sample for parameter p_j. It includes the central 90% of the data, thus providing a broader coverage than the IQR. The P5–P95 interval is moderately robust to outliers: it trims the most extreme 5% of values at each end, which is useful for distributions with moderately heavy tails. This criterion is often preferred when stakeholders desire a more generous envelope that still excludes the most extreme solutions.

10.4.3 Tukey Whiskers (Classical Outlier Trimming)

$$\left[L_j^{\mathrm{Tukey}}, U_j^{\mathrm{Tukey}}\right] = \left[\min\left\{x \in \mathcal{X}_{p_j} : x \geq Q_1 - 1.5\,\mathrm{IQR}\right\},\ \max\left\{x \in \mathcal{X}_{p_j} : x \leq Q_3 + 1.5\,\mathrm{IQR}\right\}\right] \tag{61}$$

This approach trims classical outliers—values that fall outside the Tukey fences—while preserving the full variability of the interior data. It is especially interpretable and useful

when a few extreme runs would otherwise distort the recommended range, but you still want to include as much of the "typical" data as possible.

10.4.4 Shortest α High-Density Interval (HDI)

The HDI criterion seeks the most compact interval containing a specified fraction α of the data. For parameter p_j, let $\alpha \in (0,1)$ (we use $\alpha = 0.68$ by default, analogous to a 1-sigma interval for a normal distribution). Sort the sample values $v_1 \leq \cdots \leq v_n$ and set $k = \lfloor \alpha n \rceil$. The HDI is the shortest interval among all contiguous windows of length k:

$$[L_j^{\mathrm{HDI}}, U_j^{\mathrm{HDI}}] = \underset{1 \leq s \leq n-k+1}{\arg\min} \, (v_{s+k-1} - v_s). \tag{62}$$

This method is non-parametric and adapts to the actual shape of the sample distribution, making it especially effective for skewed or multi-modal data. The HDI focuses on the densest region of solutions, which is often where the most promising or robust parameter values lie.

10.4.5 Top-q Performance P5–P95

This criterion restricts attention to the best-performing fraction of runs, as measured by the fitness value $f^{(r)}$ for each run r. Let $q \in (0,1)$ denote the top fraction to consider (default $q = 0.25$, i.e., the best 25% of runs). Define the index set $\mathcal{I}_q = \{r : \ f^{(r)} \leq Q_f(q)\}$, where $Q_f(q)$ is the q-th quantile of the fitness values. The subset of parameter samples from these top runs is $\mathcal{X}_{p_j}^q$. The interval is then:

$$L_j^{\mathrm{Top}} = Q_{x_j \in \mathcal{X}_{p_j}^q}(0.05), \qquad U_j^{\mathrm{Top}} = Q_{x_j \in \mathcal{X}_{p_j}^q}(0.95). \tag{63}$$

This approach highlights the range where high-performing solutions for p_j are concentrated, making it particularly useful when the goal is to recommend parameter values that are most likely to yield elite performance. Note that this criterion is sensitive to the choice of q and requires a sufficient number of top-performing samples for stability.

10.4.6 Trimmed Mean $\pm$ 1.5 MAD (TMAD, Robust)

This robust criterion combines the trimmed mean and the median absolute deviation (MAD) to define a stable interval, even for small sample sizes or heavy-tailed distributions. For parameter p_j, let $m = \mathrm{median}(\mathcal{X}_{p_j})$ and $\mathrm{MAD} = \mathrm{median}(|x - m|)$, which measures the typical deviation from the median. The robust scale estimate is $\sigma_{\mathrm{rob}} = 1.4826\,\mathrm{MAD}$ (the constant makes it consistent with the standard deviation for normal data). Optionally, a 10% symmetric trimming is applied before computing the mean $\tilde{\mu}$ (i.e., the mean of the

central 80% of values). The interval is:

$$L_j^{\mathrm{TMAD}} = \max\{\min \mathcal{X}_{p_j},\ \tilde{\mu} - 1.5\,\sigma_{\mathrm{rob}}\}, \quad U_j^{\mathrm{TMAD}} = \min\{\max \mathcal{X}_{p_j},\ \tilde{\mu} + 1.5\,\sigma_{\mathrm{rob}}\}. \tag{64}$$

This method yields compact, outlier-resistant intervals and is especially reliable when the number of runs R is small or the data are contaminated by outliers.

10.5 Data ingestion and aggregation (GAWorker → GUI)

Each optimization run is executed by `GAWorker.run()`, which, upon completion, emits a set of outputs via `GAWorker.finished`:

- `final_results`: The full results of the run, optionally including `singular_response` (detailed response data) and `benchmark_metrics` (summary statistics and metadata).
- `best_ind`: The best solution vector (i.e., the parameter values for the best individual found in the run).
- `parameter_names`: An ordered list of parameter names, ensuring consistent mapping across runs.
- `best_fitness`: The fitness value associated with the best solution.

The GUI mixin aggregates these outputs into a structured table, where each row corresponds to a run and columns include `run_id`, `fitness`, and one column for each parameter (named according to `parameter_names`). Additional metadata—such as the seeding method (`seeding_method`), controller type (e.g., fixed, adaptive, ML-bandit, RL), and surrogate model settings—are attached from `benchmark_metrics` to enable filtering and stratified analysis. Parameters that are fixed (i.e., have collapsed bounds and do not vary across runs) are automatically detected and treated as degenerate intervals (single values).

Special edge cases handled by the GUI include:

- **Fixed parameters:** If a parameter is fixed across all runs, all criteria return the degenerate interval $[\theta^{\min}, \theta^{\max}]$ for that parameter, reflecting its lack of variability.
- **Insufficient top-performing samples:** If the Top-q subset contains fewer than 5 samples, the Top-q P5–P95 interval is considered unstable and is omitted in favor of more robust criteria (HDI, IQR, TMAD).

10.6 Criterion robustness and guidance

Each interval criterion has distinct statistical properties and is suited to different data characteristics:

- **IQR:** Highly robust to outliers and provides a compact interval. It is a good default when the parameter distribution is unimodal or only mildly skewed. By design, it ignores the tails, focusing on the central bulk of solutions.
- **P5–P95:** Offers broader coverage than IQR, including 90% of the data. It is moderately robust and is appropriate when a more generous envelope is desired, while still trimming the most extreme values.
- **Tukey:** Implements classical outlier trimming using the 1.5 IQR rule. It is interpretable and effective when a few extreme runs would otherwise distort the range, but you want to retain as much of the "typical" data as possible.
- **$\mathbf{HDI}_{0.68}$:** Finds the shortest interval containing 68% of the data, regardless of the distribution's shape. It excels for skewed or multi-modal samples and is especially useful for identifying dense "sweet spots" in the parameter space.
- **Top-q P5–P95:** Focuses on the best-performing fraction of runs (e.g., top 25% by fitness). This criterion is ideal when the recommended range should reflect where elite solutions are concentrated, but it is sensitive to the choice of q and requires a sufficient number of top-performing samples.
- **TMAD:** Based on the median and MAD, with optional trimming, this is the most stable criterion for small sample sizes or when the data are heavy-tailed. It is conservative but highly reliable, making it a strong default in challenging scenarios.

Recommended usage: For most applications, begin with TMAD or IQR for stability and robustness. Add HDI to capture the densest region of solutions, especially if the distribution is skewed or multi-modal. Report P5–P95 for a broad, stakeholder-friendly envelope. Use Top-q to highlight elite performance corridors, and apply Tukey when explicit, interpretable outlier handling is required.

10.7 Algorithms

Below we provide efficient, implementation-ready pseudocode for two of the more complex criteria, matching the helper functions in `ga_mixin.py`. All variables are defined for clarity.

Algorithm 5 Shortest α-HDI for a parameter sample

Require: Sample $\mathcal{X} = \{x_1, \ldots, x_n\}$ for parameter p_j; desired coverage $\alpha \in (0, 1)$ (default 0.68)

$v \leftarrow \text{sort}(\mathcal{X})$ ascending // Sort the sample values
$k \leftarrow \max\{1, \lfloor \alpha n \rceil\}$ // Number of points in the interval
$w^\star \leftarrow +\infty$, $s^\star \leftarrow 1$ // Initialize best width and start index
for $s = 1$ to $n - k + 1$ **do**
  $w \leftarrow v_{s+k-1} - v_s$ // Width of current window
  if $w < w^\star$ **then**
    $w^\star \leftarrow w$, $s^\star \leftarrow s$
  end if
end for
return $[v_{s^\star},\ v_{s^\star+k-1}]$ // Shortest interval found

Algorithm 6 Top-q performance P5–P95

Require: Paired samples $\{(x^{(r)}, f^{(r)})\}_{r=1}^{R}$ for parameter p_j and fitness; top fraction $q \in (0, 1)$ (e.g., $q = 0.25$)

$\tau \leftarrow Q_{\{f^{(r)}\}}(q)$ // Fitness threshold for top q fraction
$\mathcal{J} \leftarrow \{r :\ f^{(r)} \le \tau\}$ // Indices of top-performing runs
$\mathcal{X}^q \leftarrow \{x^{(r)} :\ r \in \mathcal{J}\}$ // Parameter values from top runs
$L \leftarrow Q_{\mathcal{X}^q}(0.05)$, $U \leftarrow Q_{\mathcal{X}^q}(0.95)$ // 5th and 95th percentiles
return $[L, U]$

10.8 Aggregation, comparison, and decision rules

Once all per-criterion intervals $[L_j^{(c)}, U_j^{(c)}]$ are computed for each parameter p_j, we aggregate and compare them using the following metrics and rules:

- **Width and center:** For each criterion c, the width is $W_j^{(c)} = U_j^{(c)} - L_j^{(c)}$ and the center is $C_j^{(c)} = (U_j^{(c)} + L_j^{(c)})/2$. These summarize the size and location of each interval.

- **Union band:** The union interval $[L_j^{\cup}, U_j^{\cup}] = [\min_c L_j^{(c)},\ \max_c U_j^{(c)}]$ covers all values included by any criterion.

- **Intersection band:** The intersection $[L_j^{\cap}, U_j^{\cap}] = [\max_c L_j^{(c)},\ \min_c U_j^{(c)}]$ is the region where all intervals overlap (if non-empty).

- **Majority intersection:** To avoid bias from a single criterion, we can intersect only those intervals supported by at least M criteria (e.g., $M = 3$ out of 6).

- **Consensus score:** $S_j^{\text{cons}} = \dfrac{\max\{0,\ U_j^{\cap} - L_j^{\cap}\}}{\max\{\varepsilon,\ U_j^{\cup} - L_j^{\cup}\}}$, where $\varepsilon \ll 1$ prevents division by zero. This score quantifies the degree of agreement among criteria (1 = perfect consensus, 0 = no overlap).

- **Pairwise Intersection-over-Union (IoU):** For any two criteria c_1 and c_2, $\mathrm{IoU}_j^{(c_1,c_2)} = \dfrac{\max\{0,\, \min(U_j^{(c_1)}, U_j^{(c_2)}) - \max(L_j^{(c_1)}, L_j^{(c_2)})\}}{\max\{\varepsilon,\, \max(U_j^{(c_1)}, U_j^{(c_2)}) - \min(L_j^{(c_1)}, L_j^{(c_2)})\}}$ measures the overlap between intervals.

- **Normalized width:** $\tilde{W}_j^{(c)} = W_j^{(c)} / \max\{\varepsilon,\, U_j^{\cup} - L_j^{\cup}\}$ expresses each interval's width as a fraction of the union width, allowing direct comparison of compactness across criteria.

Recommended range selection. The following decision rules are used to select the final recommended interval for each parameter p_j:

R1 If the intersection band $[L_j^{\cap}, U_j^{\cap}]$ is non-empty and its width $W_j^{\cap}$ exceeds a minimal engineering tolerance, use this as the recommended range.

R2 If the intersection is empty or too narrow, use the *majority intersection* (intersection of at least M criteria). If this is also empty, fall back to the shortest width $W_j^{(c)}$ among the robust criteria (HDI, IQR, TMAD).

R3 For parameters that show strong sensitivity to performance (e.g., high correlation with fitness), prefer the *Top-q* or *HDI* interval, as these focus on elite or dense regions.

R4 Always enforce physical or engineering bounds: clamp the final recommended interval $[L_j, U_j]$ to the allowed range $[\theta_j^{\min}, \theta_j^{\max}]$ for parameter p_j.

10.9 Diagnostics and visualization

To support interpretation and transparency, we generate two main types of diagnostics for each parameter:

1. **Stacked interval bands:** For each parameter, we plot stacked, color-coded rectangles representing the intervals from all criteria. This visualization makes it easy to see where criteria agree or disagree, and to compare the widths and locations of the intervals.

2. **Width heatmap:** We construct a matrix where each row is a parameter and each column is a criterion, with entries showing the interval width $W_j^{(c)}$ (optionally normalized by the union width). This heatmap highlights which parameters are tightly constrained (stable) and which are more uncertain.

10.10 Statistical guidance on criterion choice

The choice of interval criterion should be guided by the characteristics of the data and the goals of the analysis:

- **Small sample size (R) or heavy outliers:** Prefer *TMAD* and *IQR*, as these are most robust and stable.
- **Skewed or multi-modal parameter distributions:** Use *HDI*, which adapts to the actual density and can capture multiple modes or asymmetry.
- **When the recommended range should reflect high performance:** Use *Top-q*, which focuses on the best-performing solutions.
- **When broader coverage is acceptable or desired:** Use *P5–P95*, which includes most of the data while trimming only the extremes.
- **When explicit, interpretable outlier handling is needed:** Use *Tukey*, which applies a classical, well-understood rule for outlier exclusion.

10.11 Complexity and implementation notes

The computational complexity of the statistical range synthesis procedure is dominated by the sorting operations required for quantile-based criteria. For each parameter, most criteria (such as IQR, P5–P95, Tukey, and HDI) require sorting the R values collected from independent optimization runs, resulting in a per-parameter complexity of $O(R \log R)$. Simple quantile calculations (e.g., computing the median or a single percentile) can be performed in $O(R)$ time using selection algorithms, but in practice, sorting is used for flexibility and simplicity, especially when multiple quantiles or more complex intervals (like HDI or Tukey) are needed.

Since the procedure is repeated for each of the P parameters, the total computational cost scales as $O(PR \log R)$. This linear scaling in the number of parameters P and near-linear scaling in the number of runs R makes the method practical and efficient even for high-dimensional problems or when aggregating results from hundreds of optimization runs.

In the software implementation, the GUI leverages efficient, vectorized operations provided by NumPy and Pandas. These libraries allow for rapid computation of quantiles, sorting, and interval construction across all parameters and runs. The mapping from the mathematical definitions of each criterion to code is direct: for example, the IQR is computed using `numpy.percentile` or `pandas.Series.quantile`, and the HDI is implemented via a sliding window over the sorted samples. This ensures that the statistical analysis remains both robust and performant, supporting interactive exploration and visualization in the GUI.

10.12 End-to-end procedure (from optimization to ranges and comparison)

The following algorithm outlines the complete workflow, starting from the outputs of repeated optimization runs and culminating in the generation, comparison, and export of recommended parameter ranges. Each step is designed to ensure traceability, robustness, and transparency in the statistical synthesis process.

Algorithm 7 Aggregate and Validate Optimization Run Data

Require: R independent optimization runs executed via `GAWorker.run()`; each run outputs a tuple `(final_results, best_ind, parameter_names, best_fitness)`; parameter bounds $[\theta_j^{\min}, \theta_j^{\max}]$ for all j; optional metadata (e.g., seeding method, controller type, surrogate usage)

1: **Data collection:** In the GUI, aggregate all runs into a structured table. Each row corresponds to a run and contains the run identifier (`run_id`), the final fitness value (`fitness`), and one column for each parameter in `parameter_names[j]`.

2: **Validation and alignment:** Ensure that parameter names and ordering are consistent across all runs. Remove any runs that are incomplete, invalid, or contain missing data to maintain data integrity.

3: **Sample construction:** For each parameter p_j, extract the set of observed values across all runs, forming the sample $\mathcal{X}_{p_j} = \{x_j^{(r)}\}_{r=1}^{R}$. Simultaneously, collect the corresponding fitness values $\{f^{(r)}\}_{r=1}^{R}$. Identify parameters that are fixed (i.e., have the same value in all runs).

4: **Top-q subset (optional):** If desired, define a subset of runs corresponding to the top-performing fraction q (e.g., $q = 0.25$). Compute the fitness threshold $\tau = Q_f(q)$, and let $\mathcal{I}_q = \{r : f^{(r)} \leq \tau\}$ be the indices of the top runs. For each parameter, extract the corresponding values $\mathcal{X}_{p_j}^{q}$.

Algorithm 8 Compute Statistical Intervals for Each Parameter

Require: For each parameter p_j, sample $\mathcal{X}_{p_j}$ (and optionally $\mathcal{X}_{p_j}^q$), parameter bounds $[\theta_j^{\min}, \theta_j^{\max}]$

1: **for** $j = 1$ to P **do**
2: **if** p_j is fixed **then**
3: For all criteria, set the interval to the full allowed range $[\theta_j^{\min}, \theta_j^{\max}]$ (since no variability is observed), and **continue** to the next parameter.
4: **end if**
5: **Interval computation:** For parameter p_j, compute the recommended intervals according to each statistical criterion using GUI helper functions:

- **IQR:** Interquartile range (25th to 75th percentile)
- **P5–P95:** 5th to 95th percentile interval
- **Tukey:** Tukey's outlier rule (using 1.5× IQR)
- **HDI$_{0.68}$:** Highest Density Interval containing 68% of the data
- **Top-q P5–P95:** 5th to 95th percentile within the top-q subset (if $|\mathcal{I}_q|$ is sufficiently large)
- **TMAD:** Trimmed Median Absolute Deviation interval

6: **end for**

Algorithm 9 Compute Interval Metrics and Compare Criteria

Require: For each parameter p_j, intervals $[L_j^{(c)}, U_j^{(c)}]$ for all criteria c

1: **for** $j = 1$ to P **do**
2: For each criterion c, calculate the interval width $W_j^{(c)} = U_j^{(c)} - L_j^{(c)}$ and center $C_j^{(c)} = (U_j^{(c)} + L_j^{(c)})/2$.
3: Compute the union interval $[L_j^{\cup}, U_j^{\cup}]$ (spanning all criteria) and the intersection interval $[L_j^{\cap}, U_j^{\cap}]$ (where all intervals overlap).
4: Quantify agreement and differences among criteria by computing:

- **Consensus score** S_j^{cons} (fraction of overlap among all intervals)
- **Pairwise Intersection-over-Union (IoU)** between all pairs of criteria
- **Normalized widths** $\tilde{W}_j^{(c)}$ (width of each interval relative to the union width)

5: **end for**

Algorithm 10 Select Recommended Parameter Ranges

Require: For each parameter p_j, all computed intervals, metrics, and parameter bounds $[\theta_j^{\min}, \theta_j^{\max}]$

1: **for** $j = 1$ to P **do**

2: Apply the decision rules (R1–R4) to select the final recommended interval $[L_j, U_j]$ for parameter p_j. This may involve using the intersection, majority intersection, or the most robust criterion, and always clamping to the physical bounds $[\theta_j^{\min}, \theta_j^{\max}]$.

3: **end for**

Algorithm 11 Visualize, Report, and Export Results

Require: For all parameters, per-criterion intervals, metrics, and final recommended ranges

1: Present the results in the GUI as follows:

- Per-criterion tables showing $[L_j^{(c)}, U_j^{(c)}, W_j^{(c)}, C_j^{(c)}]$ for all parameters
- Stacked-interval plots visualizing the intervals for each parameter and criterion
- Overlays comparing intervals across criteria, width heatmaps, and summary metrics

2: Export all computed intervals, union/intersection bands, comparison metrics, and final recommended ranges $[L_j, U_j]$ to CSV files. These are saved alongside the `benchmark_metrics` for full traceability and reproducibility of the analysis.

10.13 Mapping to software implementation

The GUI adds a *Parameter Ranges* benchmarking tab with one subtab per criterion and a *Compare Criteria* subtab. Each subtab presents: (i) a table of $[L_j^{(c)}, U_j^{(c)}, W_j^{(c)}, C_j^{(c)}]$ across all parameters; (ii) a horizontal stacked-rectangle visualization; and (iii) export options (CSV). The comparison subtab overlays stacked rectangles across selected criteria and provides a combined table to quantitatively compare ranges.

Engineering outcome. The final ranges provide defensible, transparent envelopes for each DVA parameter, reflecting both central tendency and high-performance concentration while remaining robust to outliers and distributional irregularities.

11 Benchmarking protocol: single-run and 100-run statistical study

11.1 Rationale and scope

This protocol evaluates DeVana under a practically motivated setting where the **primary objective is bandwidth avoidance on the host structure**. Because the engineering

concern is to keep the main structure quiet in service, we restrict performance accounting to the first two degrees of freedom (primary masses M_1 and M_2). The inter-modal band between two anchor resonances is treated as a protected corridor ("do-not-excite"), and all criteria for absorber-attached coordinates are set to zero weight. This concentrates the optimizer on what matters most: shaping the host dynamics.

11.2 Benchmarking system and justifications

We adopt a compact, dimensionless baseline aligned with the GUI inputs you supplied. Values are chosen for neutrality, clarity, and reproducibility:

- **Symmetric host**: Mass ratio $\Gamma = 1.0$. A symmetric 2DOF host is the canonical baseline and avoids trivial asymmetry effects.
- **Low primary damping**: $\zeta_{dc} = 0.01$. Light damping exposes sharp modal structure, making bandwidth shaping meaningful.
- **Reference frequency**: $\omega_{dc} = 100\,\mathrm{s}^{-1}$. This sets the non-dimensionalization scale used throughout the thesis.
- **Balanced base motions and forcing**: $A_{\mathrm{LOW}} = A_{\mathrm{UPP}} = 0.05$, $F_1 = F_2 = 100$. These neutral inputs do not privilege either primary mass and are consistent with your GUI configuration.
- **Primary stiffness and damping ratios**: From the GUI, $\Lambda_1 = 1.0$, $\Lambda_2 = 1.0$, and $\Lambda_3 = \Lambda_4 = \Lambda_5 = 0.5$; the primary damping ratios satisfy $\mathcal{N}_1 = \cdots = \mathcal{N}_5 = 0.75$. Ratios maintain dimensionless portability across scales.
- **Anchor peak targets and weights**: Peak positions at $100\,\mathrm{s}^{-1}$ and $300\,\mathrm{s}^{-1}$ with per-peak weights 0.25 (your GUI: Peak Position 2 and 3 set to 100 and 300; corresponding weights set to 0.25; others zero).

Unless otherwise specified, all absorber-related parameters (couplings $\beta_i, \lambda_i, \nu_i$ and mass ratios μ_i) are free within engineering bounds and are optimized; however, *the fitness in this benchmark depends only on the primary coordinates* U_1 and U_2.

11.3 Bandwidth-avoidance criterion and weights

Let $\Omega = \omega/\omega_{dc}$. With $\omega_{dc} = 100\,\mathrm{s}^{-1}$, the anchor resonances are $\Omega_1^\star = 1$ and $\Omega_2^\star = 3$. Define a protected inter-modal band

$$\mathcal{I}_{\mathrm{avoid}} = \big[\,\Omega_1^\star + \delta\Omega,\ \Omega_2^\star - \delta\Omega\,\big], \qquad \delta\Omega = 0.10.$$

We penalize the *normalized inter-modal area* of each primary response:

$$B = \Omega_2^\star - \Omega_1^\star - 2\,\delta\Omega, \tag{65}$$

$$A_i^{\text{avoid}} = \frac{1}{B}\int_{\Omega\in\mathcal{I}_{\text{avoid}}} \big|X_i(\Omega)\big|\,\mathrm{d}\Omega, \qquad i\in\{1,2\}. \tag{66}$$

To softly anchor the two resonances we include peak-position consistency for the primary masses

$$E_1^{\text{pos}} = \left|\tfrac{\Omega_{\text{peak},1}-\Omega_1^\star}{\Omega_1^\star}\right|, \qquad E_2^{\text{pos}} = \left|\tfrac{\Omega_{\text{peak},2}-\Omega_2^\star}{\Omega_2^\star}\right|. \tag{67}$$

The benchmark's singular criterion (cf. Eq. (24)) is specialized to

$$C_s^{\text{bench}} = w_{A,1}\,A_1^{\text{avoid}} + w_{A,2}\,A_2^{\text{avoid}} + w_{\text{pos},1}\,E_1^{\text{pos}} + w_{\text{pos},2}\,E_2^{\text{pos}}, \tag{68}$$

with a **bandwidth-avoidance focus** implemented by

$$w_{A,1} = w_{A,2} = 0.25, \quad w_{\text{pos},1} = w_{\text{pos},2} = 0.25, \quad \sum w = 1, \quad \text{all other GUI weights} = 0. \tag{69}$$

This adheres to the normalization guidance (Eq. (29)) and uses only the primary-mass terms. The total fitness retains Eq. (22): $|C_s^{\text{bench}} - 1|$ replaces the generic singular term and is combined with sparsity and percentage-error components.

11.4 Part I: Single run with all features enabled

We exercise the full integrated stack on the system above:

1. **GA core** with bound projection (Eq. (40)) and fixed-gene handling.
2. **ML-bandit controller** (Eqs. (42)–(46)) to adapt cxpb, mutpb, and population size within guardrails.
3. **Seeding**: Sobol/LHS QMC plus **NeuralSeeder** (Eq. 49) when sufficient data exist.
4. **Surrogate screening**: KNN with novelty to pre-filter offspring before FRF calls.
5. **Instrumentation**: full per-generation KPIs and artifacts; FRFs for U_1 and U_2 over a grid resolving the avoid band and both anchors (at least $\Delta\Omega \le 0.01$).

Budgets and controller guardrails follow release defaults. The random seed is recorded for exact reproducibility.

11.5 Part II: 100-run statistical range study

We repeat the full-feature setup of Sec. **??** for $R = 100$ independent seeds (seeds 1–100). For each run we persist `(final_results, best_ind, parameter_names, best_fitness)`

and all benchmarking metadata. The statistical pipeline of Sec. 10 is then applied with the following lens:

- **Performance lens**: evaluation is driven solely by the primary-mass criterion in Eq. (**??**); absorber mass coordinates are not weighted.
- **Intervals**: IQR, P5–P95, Tukey, $\mathrm{HDI}_{0.68}$, Top-q with $q = 0.25$, and TMAD; Top-q is formed using the total fitness of Eq. (22) under the bandwidth-avoidance singular term.
- **Diagnostics**: consensus/IoU metrics and stacked-interval plots exported alongside KPIs for transparency.

Interpretation prioritizes parameters that directly influence the primary masses (e.g., direct couplings and attachments) to maintain alignment with the engineering objective of bandwidth protection.

11.6 Success criteria and non-goals

This section is methodological: it specifies *what* is evaluated and *how*, not *how well*. Success is the ability to reproduce a single-run evaluation and obtain stable parameter intervals across 100 runs without changing defaults. Numerical outcomes are intentionally deferred to results sections.