The Astrophysical Journal, 996:97 (17pp), 2026 January 1 https://doi.org/10.3847/1538-4357/ae1446

Scalable Stellar Parameter Inference Using Python-based LASP: From CPU Optimization to GPU Acceleration

Jun-Chao Liang[1,2], Yin-Bi Li[1], A-Li Luo[1,2], Fang Zuo[1], Bing Du[1], Shuo Li[1,2], Xiao-Xiao Ma[1,2], Shu-Guo Ma[1], Hai-Ling Lu[1,2], Ke-Fei Wu[1], Zhi-Hua Zhong[3], Wen Hou[1], Xiao Kong[1], Shuo Ye[1,2], Li-Li Wang[4], and Hugh R. A. Jones[5]

[1] CAS Key Laboratory of Optical Astronomy, National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100101, People's Republic of China; ybli@bao.ac.cn, lal@nao.cas.cn
[2] School of Astronomy and Space Science, University of Chinese Academy of Sciences, Beijing 100049, People's Republic of China
[3] Center for Spatial Information Science, The University of Tokyo, Tokyo 153-8505, Japan
[4] School of Computer and Information, Dezhou University, Dezhou 253023, People's Republic of China
[5] School of Physics, Astronomy and Mathematics, University of Hertfordshire, UK
Received 2025 May 31; revised 2025 September 12; accepted 2025 September 26; published 2025 December 31

Abstract

To enhance the efficiency, scalability, and cross-survey applicability of stellar parameter inference in large spectroscopic datasets, we present a modular, parallelized Python framework with automated error estimation, built on the LAMOST Atmospheric Parameter Pipeline (LASP) originally implemented in IDL. Rather than a direct code translation, this framework refactors LASP with two complementary modules: LASP-CurveFit, a new implementation of the LASP fitting procedure that runs on a CPU, preserving legacy logic while improving data I/O and multithreaded execution efficiency; and LASP-Adam-GPU, a GPU-accelerated method that introduces grouped optimization by constructing a joint residual function over multiple observed and model spectra, enabling high-throughput parameter inference across tens of millions of spectra. Applied to 10 million LAMOST spectra, the framework reduces runtime from 84 to 48 hr on the same CPU platform and to 7 hr on an NVIDIA A100 GPU, while producing results consistent with those from the original pipeline. The inferred errors agree well with the parameter variations from repeat observations of the same target (excluding radial velocities), while the official empirical errors used in LASP are more conservative. When applied to DESI DR1, our effective temperatures and surface gravities agree better with APOGEE than those from the DESI pipeline, particularly for cool giants, while the latter performs slightly better in radial velocity and metallicity. These results suggest that the framework delivers reliable accuracy, efficiency, and transferability, offering a practical approach to parameter inference in large spectroscopic surveys. The code and DESI-based catalog are available via DOI: 10.12149/101679 and DOI: 10.12149/101675, respectively.

Unified Astronomy Thesaurus concepts: Astronomy data analysis (1858); Astronomy software (1855); Fundamental parameters of stars (555); GPU computing (1969); Radial velocity (1332); Surveys (1671)

1. Introduction

In recent years, the rapid development of large-scale spectroscopic surveys—such as the Radial Velocity Experiment (M. Steinmetz et al. 2006), the Large Sky Area Multi-object Fiber Spectroscopic Telescope (LAMOST; X.-Q. Cui et al. 2012; G. Zhao et al. 2012; A.-L. Luo et al. 2015), the Galactic Archaeology with HERMES (GALAH; G. M. De Silva et al. 2015), the Sloan Digital Sky Survey (SDSS; B. Yanny et al. 2009; S. R. Majewski et al. 2017; A. Almeida et al. 2023), the Dark Energy Spectroscopic Instrument (DESI; DESI Collaboration et al. 2025), the Gaia Radial Velocity Spectrometer (Gaia Collaboration et al. 2023; A. Recio-Blanco et al. 2023), the 4.2 m William Herschel Telescope Enhanced Area Velocity Explorer (S. Jin et al. 2024), the Multi-Object Optical and Near-infrared Spectrograph (M. Cirasuolo et al. 2020), the upcoming 4 m Multi-object Spectroscopic Telescope (R. S. de Jong et al. 2022), and the Chinese Space Station Telescope (CSST; Y. Gong et al. 2019; CSST Collaboration et al. 2025)—has provided, or is expected to provide, an unprecedented observational foundation for studies of stellar physics and Galactic structure.

These datasets enable the measurement of key stellar parameters, such as radial velocity (RV), effective temperature ($T_{\rm eff}$), surface gravity ($\log g$), metallicity ([Fe/H]), and dozens of elemental abundances. To efficiently and reliably infer these parameters, various automated pipelines have been developed, including the APOGEE Atmospheric Parameter and Chemical Abundance Pipeline (ASPCAP; A. E. García Pérez et al. 2016), the Cannon for GALAH DR1/DR2 (M. Ness et al. 2015; S. L. Martell et al. 2017; S. Buder et al. 2018), the Payne for GALAH DR4 (Y.-S. Ting et al. 2019; S. Buder et al 2025), the General Stellar Parametriser from spectroscopy in Gaia DR3 (A. Recio-Blanco et al. 2023), the RVS and SP pipelines in DESI (A. P. Cooper et al. 2023; S. E. Koposov et al. 2024, 2025), and the LAMOST Atmospheric Parameter Pipeline (LASP; A.-L. Luo et al. 2015). These pipelines have become essential tools for large-scale stellar parameter estimation, offering robust and automated solutions across diverse spectroscopic surveys.

However, with increasing sample sizes and expanding parameter dimensionality, existing pipelines still leave room for improvement in terms of computational efficiency and joint-modeling capabilities. Most do not systematically exploit GPU acceleration, making it challenging to process tens of millions of spectra efficiently. Moreover, except for the Payne (Y.-S. Ting et al. 2019) and its extension TransformerPayne (T. Różański et al. 2025), atmospheric parameters and

elemental abundances are often modeled independently, limiting the ability to capture interparameter correlations in high-dimensional spaces.

These limitations are particularly evident in the modular design of LASP. In the current LAMOST low-resolution survey, RV, $T_{\rm eff}$, $\log g$, and [Fe/H] for AFGK-type stars are inferred using LASP, which incorporates the University of Lyon Spectroscopic analysis Software (ULySS[6]), implemented in IDL and optimized via the MPFit algorithm (C. B. Markwardt 2009) (hereafter LASP-MPFit). In contrast, projected rotational velocity ($v \sin i$) and α-element abundance ([α/M]) are inferred using two separate Python-based modules developed by F. Zuo et al. (2024) and W. Hou (2025, personal communication), which rely on LASP-MPFit outputs as strong priors. This modular separation prevents fully coupled multiparameter inference, which not only propagates prior errors across sequential modules but also limits the framework's capacity to enforce joint constraints and capture parameter correlations, thereby potentially compromising both its accuracy and extensibility. Additionally, the IDL-based implementation is increasingly difficult to maintain and limits flexibility for adaptation to other surveys. Such architectural and methodological fragmentation is not unique to LASP; similar inconsistencies frequently arise when integrating stellar parameters derived from different survey pipelines. Differences in algorithmic design and modeling assumptions across surveys can introduce hard-to-quantify deviations in the resulting stellar parameters, complicating their cross-survey integration. Therefore, developing an efficient and readily implementable inference framework for high-dimensional joint optimization, with the flexibility to adapt to different surveys, will facilitate consistent analysis and integrated modeling of multisurvey data.

Motivated by these challenges, we develop a new Python-based architecture for LASP[7] (hereafter PyLASP) that supports efficient inference, multitask joint modeling, and cross-survey compatibility. The framework incorporates two optimization strategies: LASP-CurveFit, a CPU-based implementation of the Levenberg–Marquardt algorithm (K. W. Vugrin et al. 2007), and LASP-Adam-GPU, a GPU-accelerated method based on the Adam optimizer (D. P. Kingma & J. Ba 2014). LASP-CurveFit preserves the logic of LASP-MPFit while restructuring the codebase for improved I/O and multithreading performance using Python. LASP-Adam-GPU performs grouped optimization by minimizing a joint residual function over multiple spectra in each group, using the PyTorch library (A. Paszke et al. 2019) to support scalable and parallel inference for large datasets and high-dimensional parameter spaces. Thanks to the flexibility of Python and its extensive scientific ecosystem, this framework is easily extensible and transferable across surveys such as DESI and CSST. Given that LASP-MPFit forms the foundation for other parameter modules in LAMOST, its IDL-based implementation has certain limitations in computational efficiency and scalability. We therefore prioritize the reimplementation of this module, which serves as the baseline method for performance and consistency evaluation in subsequent experiments. Other modules, such as $v \sin i$ and [α/M], adopt different modeling strategies and are not yet included in the current framework. Their integration will be explored in future work as part of a unified framework for high-dimensional abundance inference.

This paper is organized as follows. In Section 2, we introduce the overall structure of PyLASP. Section 3 describes the datasets used to evaluate the performance of LASP-CurveFit and LASP-Adam-GPU. In Section 4, we present results on inference efficiency, robustness, error modeling, and generalization to DESI DR1. Section 5 provides an overall summary of this work.

2. Methodology

To support large-scale stellar parameter inference, we develop LASP-CurveFit and LASP-Adam-GPU based on LASP-MPFit. LASP-MPFit is designed to infer RV, $T_{\rm eff}$, $\log g$, and [Fe/H] for AFGK-type stars in the LAMOST survey (Y. Wu et al. 2011a, 2014; A.-L. Luo et al. 2015), using version 3.2 of the ELODIE library (Y. Wu et al. 2011b) and the ULySS package implemented in IDL (M. Koleva et al. 2009a, 2009b; Y. Wu et al. 2011b). LASP-CurveFit is designed for CPU environments, with a restructured parameter inference procedure that improves I/O efficiency and multithreading performance. In contrast, LASP-Adam-GPU introduces further optimization across multiple modules—such as data loading, spectral generation, and wavelength resampling—and adopts a grouped optimization strategy that minimizes the rms residual across multiple spectra, thereby enhancing inference throughput and enabling future extensions to multielement abundance inference.

2.1. LASP-CurveFit

LASP-CurveFit is primarily designed for efficient stellar parameter inference from individual spectra and is scalable to large spectroscopic datasets through parallelization with the `joblib` library (G. Varoquaux et al. 2024). Spectral data are read from FITS files using the `Astropy` library (The Astropy Collaboration et al. 2022) and organized into dictionary-based structures to facilitate rapid access during parameter inference.

The core components of LASP-CurveFit include modules for χ^2 computation, wavelength resampling, resolution degradation, and correction of shape differences between model and observed spectra. These modules retain logical consistency with LASP-MPFit. The main modifications are as follows: (1) Since LAMOST spectra are in relative flux, the observed spectra are median-normalized before fitting to standardize the flux scale and improve numerical stability, and to maintain consistency with LASP-Adam-GPU. This also facilitates potential future comparison of χ^2 values across spectra. (2) All computations are implemented using the `NumPy` library (C. R. Harris et al. 2020) and `SciPy` library (P. Virtanen et al. 2020). (3) The optimization step replaces the MPFit algorithm (C. B. Markwardt 2009) with `scipy.optimize.curve_fit`, which provides covariance-based error estimates for stellar parameters (K. W. Vugrin et al. 2007), thereby eliminating the need for the empirical error correction previously required in LASP-MPjFit.

2.2. LASP-Adam-GPU

LASP-Adam-GPU is designed for efficient and scalable stellar parameter inference from large spectroscopic datasets. The current implementation targets the inference of

[6] http://ulyss.univ-lyon1.fr/
[7] https://github.com/LiangJunC/PyLASP

atmospheric parameters ($T_{\rm eff}$, $\log g$, and [Fe/H]) and RV, leveraging GPU-based parallel optimization to enable high-throughput processing.

The framework consists of nine sequential stages: (1) definition of input spectral data formats; (2) generation of model spectra for arbitrary atmospheric parameters; (3)–(5) spectral processing to ensure consistency with observed spectra in wavelength, resolution, and continuum shape; (6) identification of bad pixels; (7) spectral comparison between processed models and observations; (8) iterative optimization of parameters, with repeated execution of steps (2)–(7) until convergence criteria are met; and (9) error estimation for the final parameters. The entire workflow is implemented with GPU parallelization to maximize computational efficiency. Details of each stage are described below.

1. *Storing batch data*. Spectroscopic data are read using the `Astropy` library and serialized into PyTorch tensor files (`.pt`), each containing 20,000 spectra. These files support efficient memory access during inference. In each optimization cycle, up to $N < 20{,}000$ spectra are loaded as a group and jointly processed on the GPU for parallel inference of $T_{\rm eff}$, $\log g$, [Fe/H], and RV.
2. *Generating N model spectra for each group*. For any set of N stellar atmospheric parameters $\theta_1, \theta_2, \ldots, \theta_N$ ($\theta_i = T_{\rm eff}$, $\log g$, [Fe/H]), we use PyTorch to generate a group of N model spectra:

$$F = \begin{pmatrix} F_1 \\ \vdots \\ F_N \end{pmatrix} = \begin{pmatrix} f_1(\theta_1, \theta_2, \ldots, \theta_{m_1}) \\ f_2(\theta_{m_1+1}, \ldots, \theta_{m_2}) \\ f_3(\theta_{m_2+1}, \ldots, \theta_{m_3}) \\ f_4(\theta_{m_3+1}, \ldots, \theta_{m_4}) \\ f_5(\theta_{m_4+1}, \ldots, \theta_N) \end{pmatrix} = \left(T_1, \ldots, T_q\right), \tag{1}$$

where F_i is the ith model spectrum; T_j denotes the jth column of F, corresponding to the flux values at the jth wavelength pixel across all N spectra; and q is the number of flux points in the model spectrum. The functions f_1 through f_5 are used to generate spectra over different $T_{\rm eff}$ regimes—$T_{\rm eff} \leqslant 4000$ K, $4000 < T_{\rm eff} \leqslant 4550$ K, $4550 < T_{\rm eff} \leqslant 7000$ K, $7000 < T_{\rm eff} \leqslant 9000$ K, and $T_{\rm eff} > 9000$ K, respectively—and satisfy

$$\begin{pmatrix} f_1(\theta_1, \theta_2, \ldots, \theta_{m_1}) \\ f_2(\theta_{m_1+1}, \ldots, \theta_{m_2}) \\ f_3(\theta_{m_2+1}, \ldots, \theta_{m_3}) \\ f_4(\theta_{m_3+1}, \ldots, \theta_{m_4}) \\ f_5(\theta_{m_4+1}, \ldots, \theta_N) \end{pmatrix} = \begin{pmatrix} f_1(\theta_1) \\ \vdots \\ f_1(\theta_{m_1}) \\ f_2(\theta_{m_1+1}) \\ \vdots \\ f_5(\theta_N) \end{pmatrix},$$

where $f_1(\theta_1), \ldots, f_5(\theta_N)$ are model spectra generated using the ELODIE spectral emulator in ULySS.[8]
3. *Resampling to the LAMOST linear wavelength grid*. We adopt a flux-conserving interpolation method (for a discussion of flux conservation, see A. C. Carnall 2017) to resample the model spectrum F (Equation (1)), originally defined on a linearly spaced wavelength grid λ_{11}, $\lambda_{12}, \ldots, \lambda_{1q}$ with a step size of $\Delta\lambda_1$, onto the LAMOST wavelength grid:

$$\lambda'_{21} = e^{\lambda_{21}}, \ldots, \lambda'_{2p} = e^{\lambda_{2p}},$$

where $\lambda_{2i} (i = 1, \ldots, p)$ denotes the logarithmically spaced wavelength sequence of LAMOST (base e), with a step size of $\Delta\lambda_2$, and p is the number of flux points in the observed spectrum. The entire resampling procedure consists of four steps:

a. *Constructing the wavelength integration nodes*. First, construct the wavelength integration nodes for the model spectrum F as

$$\lambda_{11} - \frac{\Delta\lambda_1}{2}, \lambda_{11} + \frac{\Delta\lambda_1}{2}, \ldots, \lambda_{11} + \frac{(2q-1)\Delta\lambda_1}{2},$$

and the corresponding wavelength integration nodes for LAMOST:

$$\lambda''_{21} = e^{\lambda_{21} - \frac{\Delta\lambda_2}{2}}, \ldots, \lambda''_{2(p+1)} = e^{\lambda_{21} + \frac{(2p-1)\Delta\lambda_2}{2}}.$$

b. *Computing the integrated flux of the model spectrum*. The model spectrum F is cumulatively integrated column-wise over its wavelength integration nodes to obtain

$$0, \quad T_1\Delta\lambda_1, \sum_{i=1}^{2} T_i\Delta\lambda_1, \ldots, \sum_{i=1}^{q} T_i\Delta\lambda_1. \tag{2}$$

c. *Computing the integrated flux at LAMOST integration nodes*. The integrated flux of F (Equation (2)) is linearly interpolated onto the LAMOST wavelength integration nodes to yield

$$\mathcal{F}_{\lambda''_{21}}, \mathcal{F}_{\lambda''_{22}}, \ldots, \mathcal{F}_{\lambda''_{2(p+1)}}. \tag{3}$$

d. *Computing the resampled flux on the LAMOST linear wavelength grid*. The fluxes at λ'_{21}, λ'_{22}, …, λ'_{2p} are obtained by differencing the integrated values from Equation (3). The N model spectra obtained through batch resampling are denoted as

$$F' = \begin{pmatrix} F_1' \\ F_2' \\ \vdots \\ F_N' \end{pmatrix} = \left(T_1', \quad T_2', \quad \ldots, \quad T_p'\right), \tag{4}$$

where

$$T_i' = \frac{\mathcal{F}_{\lambda''_{2(i+1)}} - \mathcal{F}_{\lambda''_{2i}}}{\lambda''_{2(i+1)} - \lambda''_{2i}},$$

and F_i' denotes the resampled model spectrum corresponding to the original F_i (Equation (1)).
4. *Convolving model spectra with a Gaussian broadening kernel to match the resolution of the observed spectra*. To match the resolution of the LAMOST spectra, we convolve the resampled model spectra F' (Equation (4)) using `torch.nn.functional.conv1d` in batches:

$$F'' = \begin{pmatrix} F_1'' \\ F_2'' \\ \vdots \\ F_N'' \end{pmatrix} = \begin{pmatrix} F_1' \otimes G(\mu_1, \sigma_1) \\ F_2' \otimes G(\mu_2, \sigma_2) \\ \vdots \\ F_N' \otimes G(\mu_N, \sigma_N) \end{pmatrix}, \tag{5}$$

where $G(\mu_i, \sigma_i)$ is a Gaussian kernel of shape $1 \times (2 \cdot \lceil |\mu_i| + 5\sigma_i \rceil + 1)$, and F_i'' denotes the spectrum

[8] http://ulyss.univ-lyon1.fr/uly_tgm_eval.html

obtained by convolving F'_i with this kernel. The kernel parameters μ_i and σ_i are specific to each spectrum and are introduced to accommodate varying resolution and RV offsets across the LAMOST spectra. In particular, μ_i is used to model the RV of the ith observed spectrum (R. P. Van Der Marel & M. Franx 1993; M. Koleva et al. 2009b; Y. Wu et al. 2011a, 2011b; P. Prugniel et al. 2011; M. Koleva & A. Vazdekis 2012; A.-L. Luo et al. 2015; M. Cappellari 2016; K. Sharma et al. 2016; N. Kumar et al. 2025), and σ_i encompasses both the instrumental broadening and the effects of rotation (A.-L. Luo et al. 2015). In the subsequent optimization process, all kernel parameters (a total of $2N$) are jointly inferred with the stellar atmospheric parameters (a total of $3N$).

5. *Correcting shape differences between model and LAMOST spectra*. To correct the shape differences between the model spectra F'' (Equation (5)) and the LAMOST spectra—which arise from flux calibration, Galactic extinction, and other effects that alter the spectral shape (A.-L. Luo et al. 2015)—we apply a multiplicative correction factor $P(x)b_i$ to each F''_i, yielding

$$\begin{pmatrix} \frac{L_1}{w_1} \\ \frac{L_2}{w_2} \\ \vdots \\ \frac{L_N}{w_N} \end{pmatrix} = \begin{pmatrix} (P(x)b_1)^T \odot F''_1 \\ (P(x)b_2)^T \odot F''_2 \\ \vdots \\ (P(x)b_N)^T \odot F''_N \end{pmatrix}, \tag{6}$$

where $\odot$ denotes the Hadamard product. L_i is the ith LAMOST spectrum (a $1 \times p$ row vector), and w_i is a weighting factor applied to L_i to reduce the flux differences among the N LAMOST spectra (see Section 3.3 for details). The matrix $P(x) \in \mathbb{R}^{p\times 51}$ contains the values of Legendre basis functions of degrees 0 through 50, evaluated at evenly spaced nodes,

$$x_j = -1 + \frac{2(j-1)}{p}, \quad j = 1,\ldots,p,$$

using `scipy.special.eval_legendre`. The coefficient vector $b_i \in \mathbb{R}^{51\times 1}$ is obtained by solving Equation (6) using weighted least squares (C. L. Lawson & R. J. Hanson 1995).

6. *Masking outlier pixels*. Based on the flux residuals between the observed and model spectra,

$$\epsilon_i = \frac{L_i}{w_i} - (P(x)b_i)^T \odot F''_i = \frac{L_i}{w_i} - F'''_i, \tag{7}$$

we use a masking vector A_i to mitigate the impact of anomalous pixels on parameter inference. Like LASP-MPFit, LASP-Adam-GPU supports two masking strategies: No Clean and Clean. The No Clean strategy masks only predefined problematic regions (e.g., the Na D absorption lines), while the Clean strategy incorporates the clipping algorithm[9] that iteratively identifies and masks outlier pixels through a three-step filtering process. To enable large-scale spectroscopic inference, we reimplement the Clean strategy from LASP-MPFit using PyTorch. While preserving the original masking logic, this implementation is restructured to support efficient spectrum-wise parallel execution across N spectra, making it suitable for GPU-accelerated workflows. It consists of the following three steps:

a. *Compute the primary outlier mask* B_i. For the jth pixel of the ith spectrum, if

$$|\epsilon_{i,j}| - f_{\rm shift} \cdot G_{i,j} > k \cdot \sigma_{\epsilon_i},$$

then $B_{i,j} = 0$; otherwise $B_{i,j} = 1$. Here, σ_{ϵ_i} is the standard deviation of ϵ_i computed from the unmasked pixels, and $G_{i,j} = \max(|F'''_{i,j} - F'''_{i,j-1}|, |F'''_{i,j} - F'''_{i,j+1}|)$ is the flux gradient at pixel j, used to assess whether $\epsilon_{i,j}$ can be attributed to minor wavelength shifts. The factor $f_{\rm shift}$ controls the comparison scale between the residual and the flux gradient. Following ULySS conventions, $f_{\rm shift}$ is set to 0.5 in the first iteration and reduced to 0.2 in subsequent iterations to implement a progressive filtering strategy. The threshold k starts from 3 and is relaxed to $k \in \{4, 5, 7\}$ if the outlier fraction exceeds 3%. This step targets sharp outliers that cannot be explained by the model, such as sky residuals, observational defects, or cosmic rays.

b. *Compute the adjacent outlier mask* C_i. For each pixel j already marked as an outlier in B_i, its neighboring pixels $j' \in \{j-1, j+1\}$ are examined. If

$$|\epsilon_{i,j'}| - f_{\rm shift} \cdot G_{i,j'} > 2 \cdot \sigma_{\epsilon_i},$$

then $C_{i,j'} = 0$; otherwise $C_{i,j'} = 1$. This step aims to eliminate secondary outliers that are adjacent to primary outliers but exhibit slightly smaller residuals, helping to suppress small-scale structures such as residual sky features.

c. *Compute the neighborhood outlier mask* D_i. For pixels j identified as an outlier in B_i or C_i, their neighbors $j' \in \{j-1, j+1\}$ are checked. If

$$\begin{cases} \epsilon_{i,j} \cdot \epsilon_{i,j'} > 0, \\ |\epsilon_{i,j'}| \leqslant |\epsilon_{i,j}|, \\ |\epsilon_{i,j'}| - f_{\rm shift} \cdot G_{i,j'} > \sigma'_{\epsilon_i}, \end{cases}$$

then $D_{i,j'} = 0$; otherwise $D_{i,j'} = 1$. Here, σ'_{ϵ_i} is the standard deviation of ϵ_i computed from the unmasked pixels. This step is repeated until the mask converges, i.e., $D_i = D_{i+1}$, with a maximum of 20 iterations. This step further eliminates boundary pixels near primary outliers, allowing effective removal of continuous spectral anomalies that cannot be fitted by the model.

After the above three steps, the final mask matrix for the N spectra is defined as

$$A = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_N \end{pmatrix} = \begin{pmatrix} B_1 \odot C_1 \odot D_1 \\ B_2 \odot C_2 \odot D_2 \\ \vdots \\ B_N \odot C_N \odot D_N \end{pmatrix}. \tag{8}$$

7. *Constructing the objective function*. To quantify the discrepancy between the N LAMOST spectra and the corresponding model spectra F'', we define the objective

[9] http://ulyss.univ-lyon1.fr/uly_fit.html

function as

$$\mathcal{L} = \sqrt{\frac{1}{pN}\sum_{i=1}^{N}(A_i \odot \epsilon_i)(A_i \odot \epsilon_i)^T}, \quad (9)$$

where ϵ_i and A_i are defined by Equations (7) and (8), respectively, representing the flux residuals and the pixel masking vector for the ith spectrum. To mitigate potential numerical instabilities introduced by wavelength resampling and multiplicative correction near the spectrum edges, edge pixels are excluded from the objective function by default.

8. *Minimizing the objective function*. We adopt the Adam optimizer (D. P. Kingma & J. Ba 2014) to minimize the objective function defined in Equation (9), using the initial parameter values provided by the correlation function interpolation (CFI) method (B. Du et al. 2012), which is consistent with the initialization strategy used in LASP-MPFit. To avoid distortion of the optimization trajectory caused by differences in parameter scales, we follow the variable rescaling strategy proposed by J. Nocedal & S. J. Wright (2006) and normalize all free parameters to the range $[-1, 1]$. The learning rate is set to 0.1, with all other Adam parameters left at their default values. The optimization is considered converged when any of the following three criteria is satisfied: (1) a maximum of 5000 iterations is reached; (2) the change in loss over 50 consecutive steps is less than 10^{-5}; or (3) the loss increases monotonically over 50 consecutive steps. This configuration of the learning rate and convergence threshold is designed to balance convergence stability and computational efficiency, and is examined in detail in Section 4.2.2. Upon convergence, the resulting $5N$ parameters are taken as the minimizer of Equation (9). To ensure consistent mask handling across N spectra, the Clean strategy updates A_i (Equation (8)) every 30 steps based on residuals, allowing up to 11 mask updates during the optimization.
9. *Estimating parameter errors*. Once the minimizer of Equation (9) is obtained, we estimate the parameter errors by computing the Jacobian matrix J_i of the residual vector ϵ_i (Equation (7)) at the solution point, using central finite differences. The covariance matrix Σ_i is then approximated via linear error propagation (K. W. Vugrin et al. 2007; M. Drosg 2009) as $\Sigma_i = \frac{\epsilon_i \epsilon_i^T}{p-5}(J_i^T J_i)^{-1}$, where $p-5$ is the number of degrees of freedom. Parameter errors are given by the square roots of the diagonal elements of Σ_i, and computed in parallel for all N spectra using PyTorch.

Beyond the current implementation, the modular architecture of LASP-Adam-GPU enables the possibility of future extension to the joint modeling of multidimensional elemental abundances. Such extensions could follow two main paths: (1) integrating an abundance module into the optimization loop via a multitask scheme to simultaneously infer multiple parameters—a strategy that has been widely adopted in deep learning tasks such as computer vision, natural language processing, and speech recognition (R. Collobert & J. Weston 2008; J.-T. Huang et al. 2013; I. Kokkinos 2017; R. Cipolla et al. 2018)—or (2) incorporating a high-dimensional spectral emulator, inspired by the Payne (Y.-S. Ting et al. 2019), trained on theoretical spectra to model all target parameters in a unified manner. Once developed, the new modules can be directly applied to different spectroscopic surveys, with resolution matching and related adjustments handled dynamically during inference—without retraining.

3. Data

LASP-MPFit performs parameter inference for low-resolution AFGK-type stellar spectra from LAMOST using a spectral emulator constructed from the ELODIE spectral library (A.-L. Luo et al. 2015). To evaluate the consistency between the Python and IDL versions of LASP, we use a crossmatched dataset of common stars observed by both LAMOST DR10 and APOGEE DR16 as the test sample.

3.1. ELODIE Library

The ELODIE library (J. Moultaka et al. 2004; P. Prugniel & C. Soubiran 2004; P. Prugniel et al. 2007) is based on echelle spectra taken with the eponymous spectrograph attached to the 1.93 m telescope of Observatoire de Haute-Provence. It contains 1962 spectra of 1388 stars, covering the wavelength range 3900–6800 Å at a resolving power of $R \approx 42{,}000$. Version 3.2 of the library spans a wide range of atmospheric parameters: $T_{\rm eff}$ from 3100 K to 59,000 K, $\log g$ from 0.0 to 5.0 dex, and [Fe/H] from −2.8 to +1.0 dex (Y. Wu et al. 2011a; A.-L. Luo et al. 2015).

To improve interpolation performance near the edges and sparsely populated regions of parameter space, ELODIE incorporates a set of "semiempirical" spectra constructed by combining observed and synthetic spectra (Y. Wu et al. 2011b). This enhances the emulator's extrapolation capability. The spectral emulator used in LASP is built on the ELODIE spectra degraded to a resolution of $R \approx 10{,}000$ (Y. Wu et al. 2011a), and uses separate high-order polynomial regression models for three stellar temperature classes, following the ULySS convention (M. Koleva et al. 2009b): hot ($T_{\rm eff} \geqslant 7000$ K), warm ($4000 < T_{\rm eff} \leqslant 9000$ K), and cold ($T_{\rm eff} \leqslant 4550$ K). Each model maps three atmospheric parameters to 14,501 flux points. The hot, warm, and cold models contain 19, 26, and 25 nonzero polynomial coefficients, respectively.

3.2. Survey Data and Reference Labels

3.2.1. The Low-resolution Spectra of LAMOST DR10

LAMOST is a Schmidt telescope located at the Xinglong Observatory northeast of Beijing, China. It is capable of simultaneously obtaining 4000 low-resolution ($R \sim 1800$) spectra in a single exposure, covering the wavelength range of 3700–9000 Å, with blue and red arms that overlap in the 5700–5900 Å region (X.-Q. Cui et al. 2012; G. Zhao et al. 2012; A.-L. Luo et al. 2015).

In the 10th data release (DR10 v1.0),[10] a total of 11,817,430 low-resolution spectra were released, of which 11,473,644 were classified as stellar spectra. The remainder include 263,444 galaxy spectra and 80,342 quasar spectra. LASP-MPFit has been applied to 7,478,650 AFGK-type stellar spectra in DR10 v1.0, yielding inferred values of $T_{\rm eff}$, $\log g$, [Fe/H], and RV.

[10] https://www.lamost.org/dr10/v1.0/

3.2.2. APOGEE Labels

APOGEE is one of the programs in SDSS-III (S. R. Majewski et al. 2017) and SDSS-IV (M. R. Blanton et al. 2017). It employs a high-resolution ($R \sim 22{,}500$) near-infrared spectrograph covering the wavelength range 1.51–1.70 μm, targeting primarily red giant stars in the Milky Way, as well as stars in the Large Magellanic Cloud, nearby dwarf galaxies, and a substantial number of cool dwarfs (FGKM types) (V. V. Smith et al. 2021).

In its 16th data release (DR16), APOGEE provided RV, atmospheric parameters, and up to 26 elemental abundances for approximately 430,000 stars (H. Jönsson et al. 2020). RV is determined via cross correlation, and ASPCAP derives stellar parameters with FERRE using precomputed, MARCS-based synthetic spectral grids that are compressed with principal component analysis (PCA); interpolation is performed in PCA coefficient space, with radial basis functions bridging gaps in these grids. The pipeline first determines the global parameters—$T_{\rm eff}$, $\log g$, overall metallicity [M/H], [α/M], [C/M], and [N/M]—together with microturbulent velocity and $v \sin i$ for dwarfs and macroturbulent velocity for giants; then, holding these fixed, it infers individual elemental abundances from element-specific spectral windows. Additionally, APOGEE employs independent external methods to calibrate the spectroscopic measurements: $T_{\rm eff}$ is calibrated using the infrared flux method; $\log g$ for giants is calibrated using asteroseismic data from Kepler field stars, while dwarf stars use asteroseismic values for warmer stars and isochrone-derived calibrations for cooler stars; and elemental abundances are calibrated through zero-point shifts to ensure solar neighborhood stars with solar [M/H] have mean [X/M] = 0. Owing to its high precision and stability, the APOGEE catalog has been widely used as a training reference for label transfer methods (J. Li et al. 2021; J. Liang et al. 2022; C. Wang et al. 2022). In this study, we adopt $T_{\rm eff}$, $\log g$, [Fe/H], and RV from APOGEE DR16 as external references to evaluate the consistency between the Python and IDL versions of LASP.

3.3. Data Preprocessing

We crossmatch the LAMOST DR10 AFGK-type stellar parameter catalog with APOGEE DR16, and select LAMOST spectra that have both APOGEE labels and CFI initial values, yielding a sample of 177,848 spectra. The restriction to spectra with CFI initialization is intended to systematically assess the sensitivity of PyLASP to different starting conditions (see Section 4.2.2). Before applying LASP-Adam-GPU for parameter inference, the following preprocessing steps are performed:

1. *Wavelength system conversion*. To ensure consistency in the wavelength reference system between LAMOST spectra and the ELODIE spectral library, we convert the LAMOST vacuum wavelengths to air wavelengths prior to parameter inference. Since LASP-MPFit performs spectral fitting in the wavelength range 4200–5700 Å during the first stage,[11] the conversion is applied only within this range. Each spectrum is truncated to its 1327 flux points within this range (the maximum number of pixels), and saved together with the corresponding wavelengths, the CFI initial values, the wavelength ranges before and after resampling (see step 3 in Section 2.2), and the polynomial coefficients used for generating the model spectra, in a single `.pt` file.
2. *Setting spectral weighting factors*. The fluxes of different LAMOST spectra exhibit differences in scale, which may affect the parameter-inference accuracy of LASP-Adam-GPU. To alleviate this, we assign each spectrum an empirical weight factor w_i (used in Equation (7)), defined as the median (0.5 quantile) of the flux values. This factor depends only on survey data quality, is fixed during optimization, and is not treated as a free parameter. We find that this setting performs well on LAMOST data in this work because it reduces intergroup numerical differences in LASP-Adam-GPU and decreases cases where unstable weight factors cause normalized flux values to become very large or near-zero. In practice, once the weight factor is well behaved (i.e., not near-zero or excessively large), the specific choice between 0.5 and 0.75 quantiles plays a secondary role for the inferred parameters; we therefore treat q as a fixed value rather than a tunable hyperparameter. However, when applying LASP-Adam-GPU to DESI spectra, the 0.5 quantile can cause the normalized flux of some spectra to become very large, whereas using the 0.75 quantile yields better results. Therefore, we recommend testing different quantiles (e.g., 0.5, 0.75) when adapting the method to other spectroscopic surveys.

4. Experiments

This section first introduces the efficiency-related parameters[12] in the Python and IDL versions of LASP, and evaluates the inference efficiency of each version using 10-million-scale spectroscopic datasets. After identifying the optimal efficiency configurations, we perform parameter inference on 177,848 LAMOST stellar spectra and compare the results from LASP-CurveFit and LASP-Adam-GPU with those from LASP-MPFit, focusing on the consistency of $T_{\rm eff}$, $\log g$, [Fe/H], and RV, as well as on the sources of discrepancies. We further assess the robustness of the Python implementation by testing different initial parameter settings. In addition, we compare the model-propagated errors from LASP-Adam-GPU with the empirical errors adopted in the official LASP-MPFit, which are estimated based on repeat observations and reduced χ^2, to quantify the differences between the two error models. Finally, we apply LASP-CurveFit and LASP-Adam-GPU to a DESI DR1 dataset to further evaluate their cross-survey applicability and performance.

4.1. Setting Efficiency Parameters and Evaluating Inference Efficiency

To determine the optimal efficiency settings for large-scale processing, we randomly select 10,000 spectra from the 177,848 LAMOST stellar spectra, and test the efficiency parameters of LASP-MPFit, LASP-CurveFit, LASP-Adam-CPU, and LASP-Adam-GPU across four computing platforms.

[11] LASP uses a two-stage inference (A.-L. Luo et al. 2015): $T_{\rm eff}$, $\log g$, [Fe/H], and RV are first inferred from the original spectrum, then reinferred after continuum correction. The final values are adopted if both inferences agree within a set threshold.

[12] There are two efficiency parameters: the number of parallel processes (`n_jobs`) used in LASP-MPFit and LASP-CurveFit, and the number of spectra processed per group (N) in LASP-Adam-CPU and LASP-Adam-GPU, as defined in Equation (9).

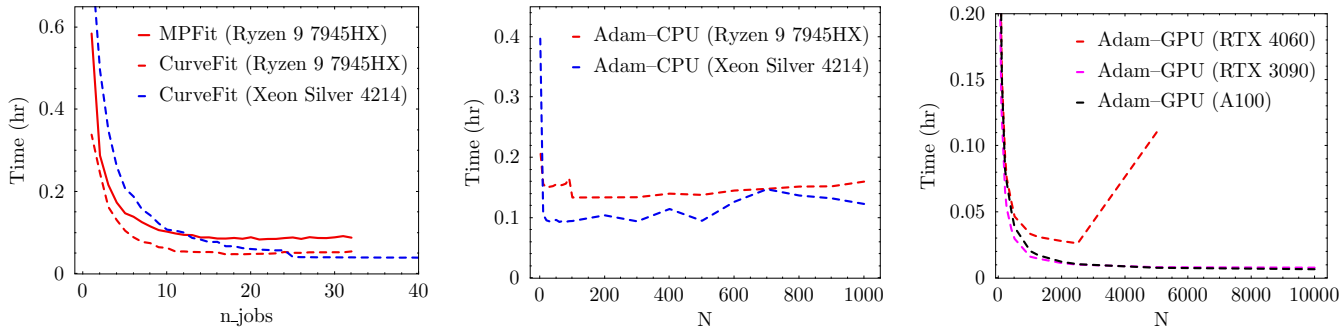

Figure 1. Comparison of the computational efficiency of different LASP versions across hardware platforms. The left panel shows the runtime of LASP-MPFit and LASP-CurveFit for processing 10,000 spectra under various efficiency parameter settings. The middle and right panels present the runtime of LASP-Adam on CPU and GPU platforms, respectively. The solid line indicates the IDL-based implementation, while dashed lines correspond to the Python-based version; line colors distinguish different hardware platforms. RTX 4060 refers to a mobile GPU integrated into a laptop with a Ryzen 9 7945HX processor. For clarity, the "LASP" prefix has been omitted from all legend labels in the figure.

Table 1
Configuration Ranges of Efficiency-related Parameters for LASP Modules on Different Devices

Device[a]	LASP-MPFit	LASP-CurveFit	LASP-Adam-CPU	LASP-Adam-GPU
	n_jobs[b]		N[c]	
Ryzen 9 7945HX	[1, 32]	[1, 32]	$\{d \mid d = 1 \vee (10 \mid d, d \leqslant 100) \vee (100 \mid d, d \leqslant 1000)\}$	$\{d \mid d = 1 \vee (d \mid 5000, 10 \mid d)\}$
Xeon Silver 4214	...[d]	[1, 48]	$\{d \mid d = 1 \vee (10 \mid d, d \leqslant 100) \vee (100 \mid d, d \leqslant 1000)\}$	...
RTX 3090	...	...	...	$\{d \mid d = 1 \vee (d \mid 10{,}000, 10 \mid d)\}$
A100	...	...	...	$\{d \mid d = 1 \vee (d \mid 10{,}000, 10 \mid d)\}$

Notes.
[a] Device: Ryzen 9 7945HX with RTX 4060 is a 16-core/32-thread AMD laptop processor; Xeon Silver 4214 is a 24-core/48-thread Intel server processor; and RTX 3090 and A100 are NVIDIA GPUs for desktop and data center applications, respectively.
[b] `n_jobs` represents the number of CPU multiprocessing processes.
[c] N represents the number of spectra in Equation (9).
[d] "..." indicates configurations not tested: LASP-MPFit requires licensed IDL software installed only on specific platforms, LASP-Adam-GPU was benchmarked exclusively on dedicated GPU accelerators (RTX 4060, RTX 3090, and A100), and CPU-optimized modules were evaluated on Xeon blade servers lacking high-performance GPUs.

The tested parameters include the number of parallel processes (`n_jobs`) for CPU-based methods and the number of spectra simultaneously processed by the objective function (N) for Adam-based methods (see Table 1). All tests are conducted under the No Clean strategy.

As shown in Figure 1, the inference time for LASP-MPFit and LASP-CurveFit decreases significantly with increasing `n_jobs`, stabilizing once `n_jobs` approaches half of the physical core count. On the same hardware, LASP-CurveFit achieves approximately 1.7 times the efficiency of LASP-MPFit even at `n_jobs=1`, and consistently outperforms it under all levels of parallelism, demonstrating the benefits of reconstruction-based optimization. For LASP-Adam, inference speed improves rapidly with increasing N. LASP-Adam-GPU reaches performance saturation around `n_jobs`≈2000, and on an RTX 4060 achieves a speedup of 191 times at `N=2500` compared to `N=1`, highlighting its strong parallel processing capability. LASP-Adam-CPU also benefits from increased N, performing optimally in the range $N \in [100, 700]$ on a Ryzen 9 CPU and $N \in [20, 500]$ on a Xeon platform, but remains significantly slower than the GPU version. Based on the shortest inference time across all configurations, we identify the optimal efficiency settings and provide recommended ranges for future use (see Table 2).

Using these optimal settings, we extrapolate the total time required to process 10 million spectra. On a laptop with a Ryzen 9 7945HX CPU and an RTX 4060 GPU, LASP-Adam-GPU completes the task in approximately 26 hr, outperforming LASP-CurveFit (48 hr) and LASP-MPFit (84 hr). LASP-Adam-CPU is the slowest, requiring 134 hr and is therefore not suitable for large-scale inference tasks. On high-performance platforms, LASP-Adam-GPU achieves excellent scalability: it completes 10 million spectra in about 8 hr on an RTX 3090 and 7 hr on an NVIDIA A100. Under the Clean strategy, inference time increases slightly due to the additional iterations required to dynamically mask outliers. For example, LASP-CurveFit (Ryzen 9) requires 63 hr, while LASP-Adam-GPU takes 76, 23, and 19 hr on RTX 4060, RTX 3090, and A100, respectively.

In summary, LASP-Adam-GPU is highly suitable for large-scale stellar parameter inference. LASP-CurveFit offers an efficient CPU-based alternative, while LASP-Adam-CPU is not recommended due to its limited performance scalability.

4.2. Evaluating Parameter Consistency and Robustness

4.2.1. Comparison with the Baseline Method

We evaluate the consistency between the parameters inferred by PyLASP and the baseline method LASP-MPFit, focusing on the performance of LASP-CurveFit and LASP-Adam-GPU. To ensure a valid comparison, we exclude the

Table 2
Recommendation and Optimal Efficiency Parameters for LASP Modules across Devices

Device	n_jobs				N			
	LASP-MPFit		LASP-CurveFit		LASP-Adam-CPU		LASP-Adam-GPU	
	RecRange[a]	OptVal[b]	RecRange[a]	OptVal[b]	RecRange[a]	OptVal[b]	RecRange[a]	OptVal[b]
Ryzen 9 7945HX	[10, 32]	21	[10, 32]	17	[100, 700]	100	[1000, 2500]	2500
Xeon Silver 4214	...	...	[24, 48]	47	[20, 500]	60	...	...
RTX 3090	...	...	...	...	...	...	[2000, 10,000]	10,000
A100	...	...	...	...	...	...	[2000, 10,000]	10,000

Notes.
[a] RecRange (recommended range): the recommended range of the efficiency parameters; "..." indicates that the configuration is not applicable.
[b] OptVal (optimal value): the optimal value of the efficiency parameters determined in this study.

Table 3
Sample Count Statistics of Common Targets between the Python and IDL Versions of LASP

Stellar Parameters	No Clean (versus LASP-MPFit)				Clean (versus LASP-MPFit)			
	LASP-CurveFit		LASP-Adam-GPU		LASP-CurveFit		LASP-Adam-GPU	
	Matched[a]	Outlier[b]	Matched[a]	Outlier[b]	Matched[a]	Outlier[b]	Matched[a]	Outlier[b]
RV	171,713	32	172,246	78	171,618	234	172,238	60
$T_{\rm eff}$	171,713	63	172,246	205	171,618	27	172,238	113
$\log g$	171,713	88	172,246	294	171,618	417	172,238	601
[Fe/H]	171,713	73	172,246	277	171,618	228	172,238	465

Notes.
[a] Number of matched samples for which both PyLASP and LASP-MPFit successfully infer the parameter, before outlier removal.
[b] Number of outliers, defined as matched samples where the parameter difference between PyLASP and LASP-MPFit exceeds 5 standard deviations.

following cases in which parameter inference fails: (1) those with negative multiplicative shape correction factors (see Step 5 in Section 2.2),[13] (2) those where the optimization does not converge within the maximum number of iterations, and (3) those for which the CFI-provided initial values fall outside the parameter space allowed by the ELODIE library. The number of valid samples after filtering is listed in Table 3.

As shown in Figure 2, the parameter offsets between the Python and IDL versions of LASP are close to zero across all four parameters: RV, $T_{\rm eff}$, $\log g$, and [Fe/H]. Under both the No Clean and Clean strategies, the standard deviations of the parameter differences between LASP-CurveFit and LASP-MPFit are $0.04/0.04\,{\rm km\,s^{-1}}$, $4/5\,{\rm K}$, $0.005/0.005\,{\rm dex}$, and $0.003/0.003\,{\rm dex}$, respectively. In comparison, LASP-Adam-GPU exhibits slightly larger discrepancies: $0.37/1.12\,{\rm km\,s^{-1}}$, $10/24\,{\rm K}$, $0.01/0.03\,{\rm dex}$, and $0.006/0.02\,{\rm dex}$. These differences primarily stem from two sources. First, the current version of LASP-Adam-GPU prioritizes computational efficiency and scalability, and does not yet implement the same failure detection mechanisms used in LASP-CurveFit and LASP-MPFit (e.g., masking spectra with anomalous multiplicative corrections). Second, under the Clean strategy, LASP-Adam-GPU employs a uniform early-stopping criterion for all spectra in a group, rather than dynamically determining convergence on a per-spectrum basis. Future versions will improve accuracy by refining failure detection and control strategies.

To determine which method yields more reliable parameter estimates in cases of significant disagreement, we identify spectra where any of the four parameters differs by more than 5σ between the Python and IDL versions of LASP. This subset, accounting for no more than 0.35% of the total sample (see Table 3), is compared against APOGEE reference labels (Figure 3) to evaluate which version is closer to the external standard. Most of these outliers are associated with low-quality LAMOST spectra (Figure 4). For $T_{\rm eff}$ under the No Clean strategy, LASP-CurveFit exhibits a deviation distribution more consistent with APOGEE than LASP-MPFit: the proportion of samples with $\Delta(T_{\rm eff}) \in [-500, 500]\,{\rm K}$ is approximately 18% higher for LASP-CurveFit, while that with $|\Delta(T_{\rm eff})| > 1000\,{\rm K}$ is about 32% lower, indicating enhanced robustness to outliers. The underlying cause of this phenomenon remains unclear, but it may be related to the scale-adjusting effect of median-based normalization, differences in optimizer convergence behavior, or the numerical stability of operations such as matrix inversion during parameter inference. Under the Clean strategy, the deviation distributions of LASP-CurveFit and LASP-MPFit relative to APOGEE become comparable, suggesting that Clean effectively improves robustness. By contrast, LASP-Adam-GPU shows slightly worse agreement with APOGEE in the $\Delta(T_{\rm eff}) = 1000$–$3000\,{\rm K}$ range. This is likely due to the absence of failure detection mechanisms and the use of a fixed early-stopping criterion for all spectra within a group, which may limit optimization depth for complex cases.

In summary, PyLASP achieves parameter consistency with LASP-MPFit for approximately 99.65% of the sample. LASP-CurveFit exhibits higher parameter accuracy for low-quality spectra, whereas LASP-Adam-GPU, by design, trades off some accuracy for computational efficiency. Given that the Clean strategy has minimal impact on the parameter consistency between LASP-CurveFit and LASP-MPFit, we speculate that introducing a dynamic per-spectrum Clean early-

[13] LASP-CurveFit treats negative multiplicative correction factors as failed fits by default. Note that both LASP-MPFit and LASP-CurveFit can dynamically adjust the order of the Legendre polynomial to reduce such failures.

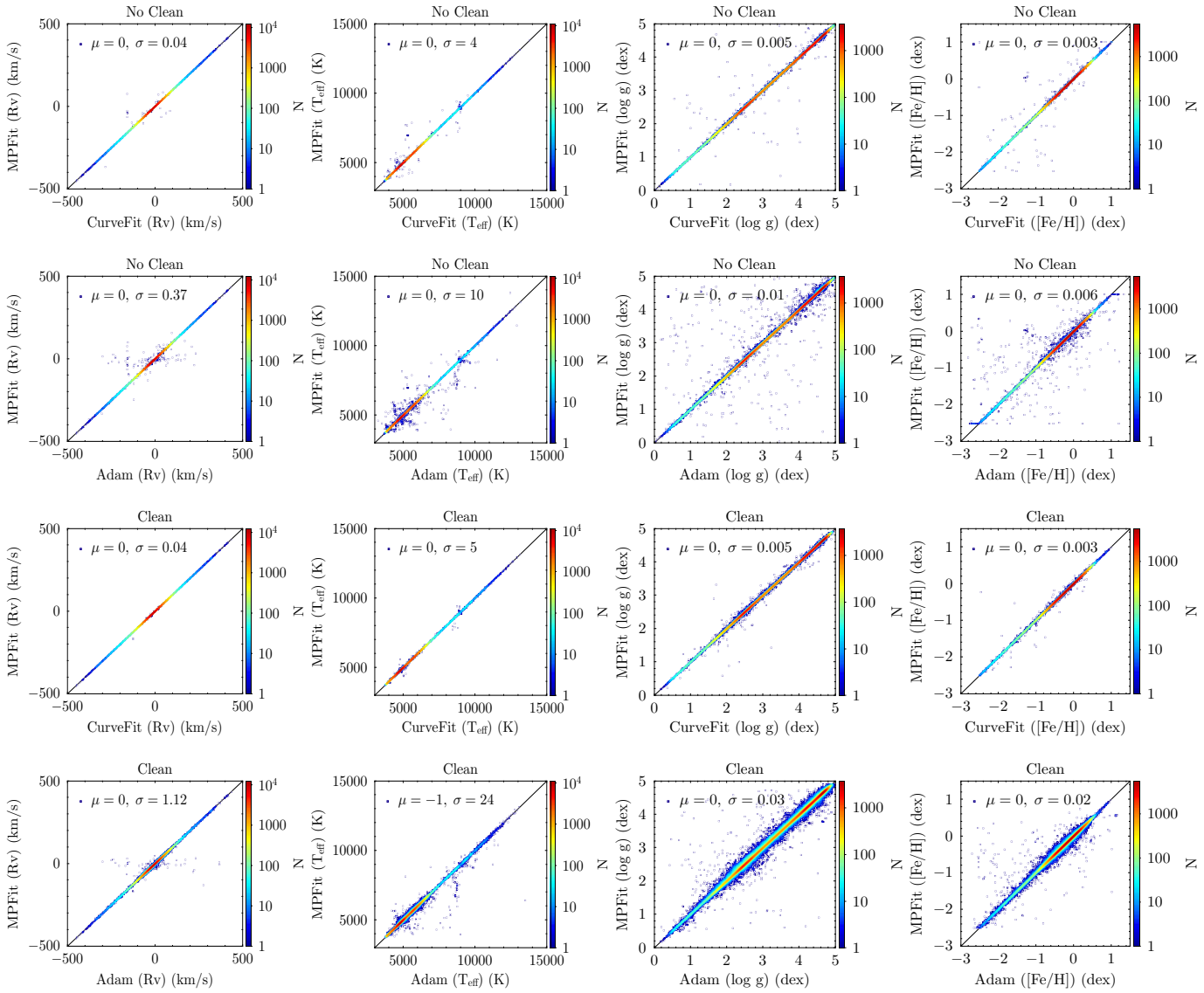

Figure 2. Comparison of RV (column 1), $T_{\rm eff}$ (column 2), $\log g$ (column 3), and [Fe/H] (column 4), inferred using LASP-CurveFit, LASP-Adam-GPU, and LASP-MPFit under both the No Clean and Clean strategies. From top to bottom, the four rows correspond to LASP-CurveFit versus LASP-MPFit (No Clean), LASP-Adam-GPU versus LASP-MPFit (No Clean), LASP-CurveFit versus LASP-MPFit (Clean), and LASP-Adam-GPU versus LASP-MPFit (Clean). The mean (μ) and standard deviation (σ) of the differences are calculated using `astropy.stats.sigma_clipped_stats` with `maxiters=1`, excluding values beyond 5σ. The "LASP" prefix has been omitted in all figure labels for clarity.

stopping strategy within LASP-Adam-GPU could substantially reduce the dispersion in parameter differences relative to LASP-MPFit, potentially restoring the consistency level observed under the No Clean condition.

4.2.2. Sensitivity to Initial Values

To evaluate the sensitivity of PyLASP implementations to initialization, we compare two approaches to 177,848 spectra: (1) a fixed initialization of ($T_{\rm eff}$, $\log g$, [Fe/H]) = (5000 K, 3 dex, −0.5 dex), simulating a worst-case scenario with no prior information, and (2) the use of CFI-derived initial values. As shown in Figure 5, LASP-CurveFit is insensitive to the choice of initial values: Under the No Clean strategy, only 0.09%, 0.11%, 0.10%, and 0.11% of samples show differences exceeding 1 km s^{-1} in RV, 100 K in $T_{\rm eff}$, 0.1 dex in $\log g$, and 0.1 dex in [Fe/H], respectively. The corresponding fractions under the Clean strategy are 0.13%, 0.10%, 0.17%, and 0.08%. Most of these discrepancies are associated with low-quality spectra with signal-to-noise ratios (S/N) below 20. In contrast, LASP-Adam-GPU is more sensitive to initialization for the three atmospheric parameters at $T_{\rm eff} > 8000$ K: under the No Clean strategy, 76%, 80%, and 78% of samples in this temperature range exhibit differences exceeding 100 K in $T_{\rm eff}$, 0.1 dex in $\log g$, and 0.1 dex in [Fe/H], respectively; under the Clean strategy, the corresponding values decrease to 65%, 68%, and 66%. These deviations appear to be independent of S/N. At $T_{\rm eff} \leqslant 8000$ K, LASP-Adam-GPU shows robustness comparable to that of LASP-CurveFit, with low sensitivity to initialization under both masking strategies.

To investigate the causes of this behavior, we analyze the problem from two perspectives:

1. *Impact of optimizer learning rate and convergence threshold.* To evaluate the impact of optimizer configurations on the inferred parameters under different

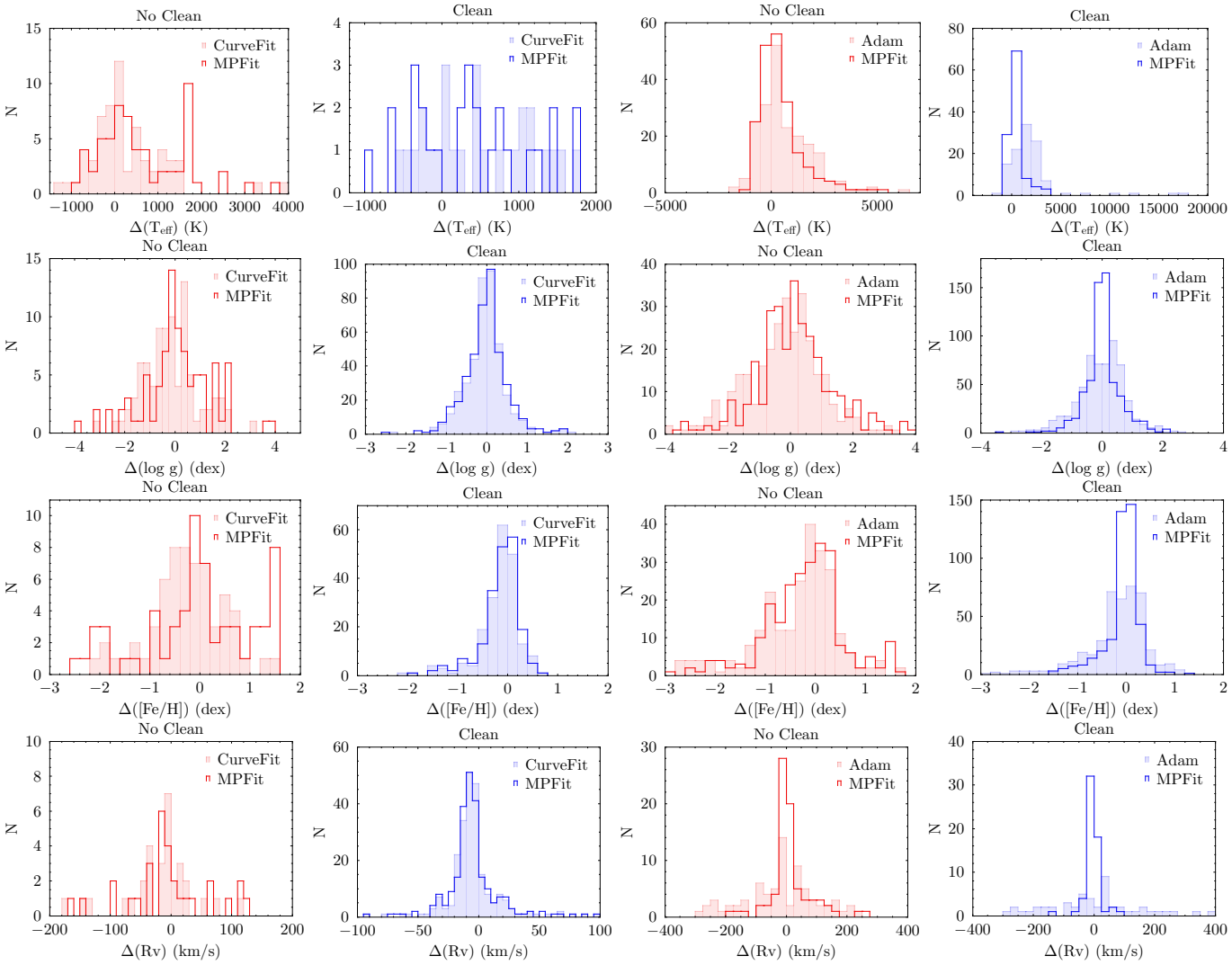

Figure 3. Residual distributions between LASP-derived parameters and APOGEE DR16 labels for outlier cases. Only parameters with internal discrepancies exceeding 5σ between the Python and IDL versions are included. Each row corresponds to a stellar parameter ($T_{\rm eff}$, $\log g$, [Fe/H], and RV, from top to bottom), and each column represents a different LASP implementation and pixel masking method: LASP-CurveFit with No Clean, LASP-CurveFit with Clean, LASP-Adam-GPU with No Clean, and LASP-Adam-GPU with Clean (from left to right). In each panel, the residuals between APOGEE labels and the parameters derived by both LASP-MPFit and PyLASP are shown, allowing direct comparison of their consistency with APOGEE. For clarity, the "LASP" prefix is omitted from all legend labels.

initialization strategies, we randomly select 10,000 spectra and evaluate the consistency of LASP-Adam-GPU results against those from LASP-CurveFit (using CFI initialization), under a range of learning rates (1, 0.1, 0.01, 10^{-3}, and 10^{-4}) and two convergence criteria (requiring the loss to vary by less than 10^{-7} or 10^{-5} over 50 consecutive iterations). As shown in Figure 6, second and final panels (corresponding to fixed initialization), the consistency of $T_{\rm eff}$ between LASP-Adam-GPU and LASP-CurveFit improves with stricter convergence thresholds (e.g., 10^{-7}) at a fixed learning rate, and with smaller learning rates when the convergence threshold is held constant (for learning rates below 0.1). However, these improvements come with a substantial computational cost: for instance, with a learning rate of 10^{-4} and a threshold of 10^{-7}, the runtime is approximately 14 times longer than that for a learning rate of 0.1; even at a 10^{-5} threshold, the factor remains as high as 4. Inappropriate learning rate settings can lead to specific clustering patterns in the inferred $\log T_{\rm eff}$ with LASP-Adam-GPU: when the learning rate is too small, potentially incomplete convergence within the 5000-iteration limit set in Section 2.2 causes values to stagnate near the initialization point (e.g., $\log T_{\rm eff} \approx 3.7$); conversely, excessively large learning rates may cause the optimizer to overshoot the optimal region, with updates stepping beyond reasonable ranges and resulting in clustering near $\log T_{\rm eff} \approx 4.5$. At a learning rate of 0.1, a small accumulation near $\log T_{\rm eff} \approx 3.95$ is also observed. Similar trends are observed under CFI initialization (first and third panels): stricter convergence improves consistency, while clustering due to inappropriate learning rate settings (except 0.1) persists. Therefore, we infer that the clustering at $\log T_{\rm eff} \approx 3.95$ likely reflects inappropriate initialization rather than the learning rate or convergence threshold, whereas clustering at other locations is attributable to learning rate settings. Balancing computational efficiency and parameter consistency, the

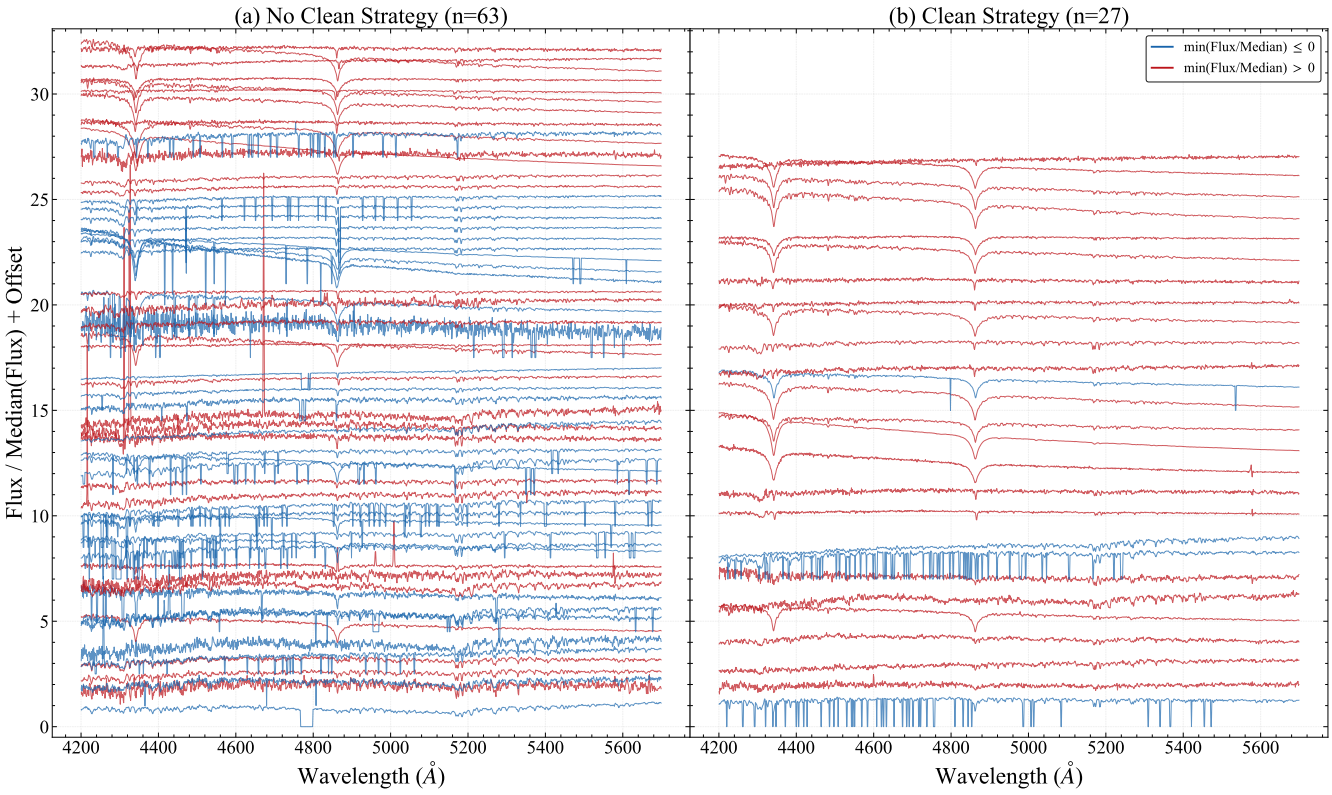

Figure 4. Spectral flux of stars with $T_{\rm eff}$ differences exceeding 5σ between LASP-CurveFit and LASP-MPFit. The left panel (a) shows the spectra under the No Clean strategy ($n = 63$), while the right panel (b) shows the spectra under the Clean strategy ($n = 27$). Blue lines represent spectra with negative flux values, while red lines represent spectra with no negative flux values.

configuration with a learning rate of 0.1 and a convergence threshold of 10^{-5} yields the best overall performance: in the first two panels, where LASP-Adam-GPU is initialized with CFI and fixed values, respectively, the fraction of spectra for which the $T_{\rm eff}$ difference between LASP-Adam-GPU and LASP-CurveFit exceeds 100 K is 0.1% and 1.9%—the lowest among all learning rate configurations at this convergence threshold. We therefore recommend this configuration as the default for LASP-Adam-GPU.

2. *Connection between initialization bias and emulator structure*. To assess whether the stability of LASP-Adam-GPU is affected by the piecewise structure of the spectral emulator (Step 2 in Section 2.2), we test an alternative fixed initialization of ($T_{\rm eff}$, $\log g$, [Fe/H]) = (7500 K, 3 dex, −0.5 dex). As shown in Figure 7, similar to the behavior observed in Figure 6, inappropriate learning rates cause $\log T_{\rm eff}$ values inferred by LASP-Adam-GPU to cluster near the initialization or around 3.7 and 4.5, indicating that such clustering arises primarily from the learning rate rather than from the initialization strategy. Importantly, under the alternative initialization with a learning rate of 0.1, the previously observed clustering near the emulator's piecewise boundary at $\log T_{\rm eff} \approx 3.95$ (in the second and final panels of Figure 6) is notably suppressed in Figure 7. This result indicates that LASP-Adam-GPU exhibits directional sensitivity to the initial $T_{\rm eff}$: optimization from lower to higher temperatures tends to be less stable, whereas the reverse direction is relatively robust. This may reflect the local gradient discontinuities along the $T_{\rm eff}$ axis, introduced by the emulator's piecewise design, which can interfere with the optimizer's update trajectory.

In summary, LASP-CurveFit and LASP-Adam-GPU exhibit distinct behaviors with respect to initialization sensitivity. LASP-CurveFit yields stable results across the entire parameter space and is therefore suitable for scientific applications that demand high robustness. For LASP-Adam-GPU, CFI-based initialization is preferred when available; if such initialization is not accessible, we recommend adopting (7500 K, 3 dex, −0.5 dex) as the default initial guess to improve inference accuracy in high-temperature regions.

4.3. Error Analysis

To evaluate the error estimation capability of PyLASP in stellar parameter inference, we examine the consistency between model errors estimated by the first-stage LASP-Adam-GPU and the corresponding repeat-observation errors of the same targets, and compare it with that between the official two-stage empirical errors from LAMOST and their respective repeat-observation errors. The model errors are derived from error propagation theory, which estimates how spectral noise propagates from pixel space to parameter space. In contrast, repeat-observation errors are computed from the dispersion among multiple independent observations of the same target, under the assumption of stable stellar properties and similar observing conditions. Since both types of errors reflect the influence of spectral noise, the repeat-observation errors serve as an empirical reference for evaluating the reliability of model errors. If the error model is reasonable, the two should be

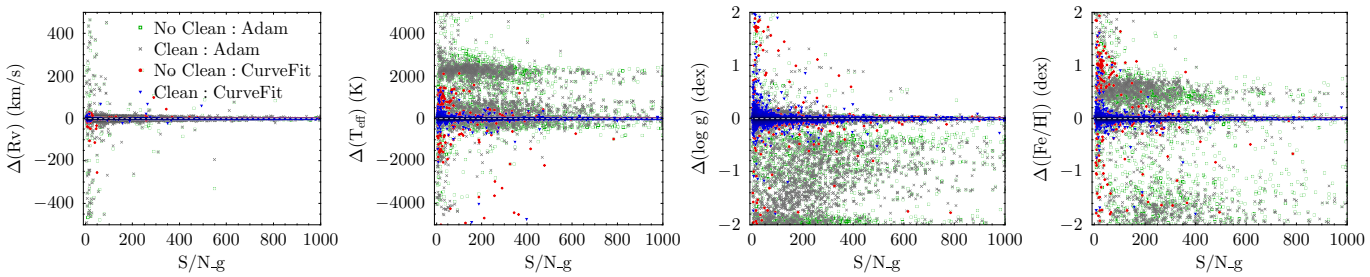

Figure 5. Stability test of initial values in PyLASP. Δ denotes the difference between stellar parameters inferred using fixed initial values ($T_{\rm eff}$, $\log g$, [Fe/H]) = (5000 K, 3 dex, -0.5 dex) and those inferred using CFI-derived initial values. For clarity, the legend is shown only in the first panel, and the "LASP" prefix is omitted from all legend labels.

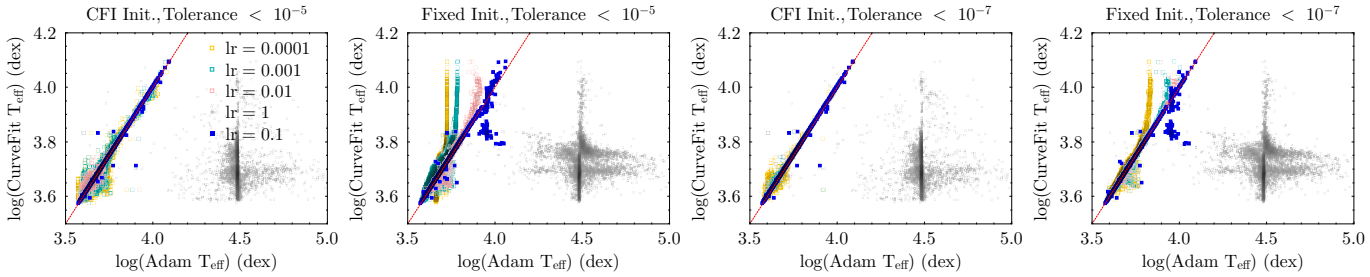

Figure 6. Comparison of $T_{\rm eff}$ between LASP-Adam-GPU and LASP-CurveFit under varying initialization schemes, based on 10,000 randomly selected spectra. LASP-Adam-GPU uses either CFI-derived values (first and third panels) or fixed initial values (5000 K, 3 dex, -0.5 dex) (second and fourth panels), while LASP-CurveFit consistently uses CFI-derived values. The first two panels use a convergence threshold of 10^{-5}, and the last two use 10^{-7}. Colors of the scatter points indicate different learning rates (lr). The x-axis shows LASP-Adam-GPU results, and the y-axis shows LASP-CurveFit results. For clarity, the legend is shown only in the first panel, and the "LASP" prefix is omitted in all figure labels.

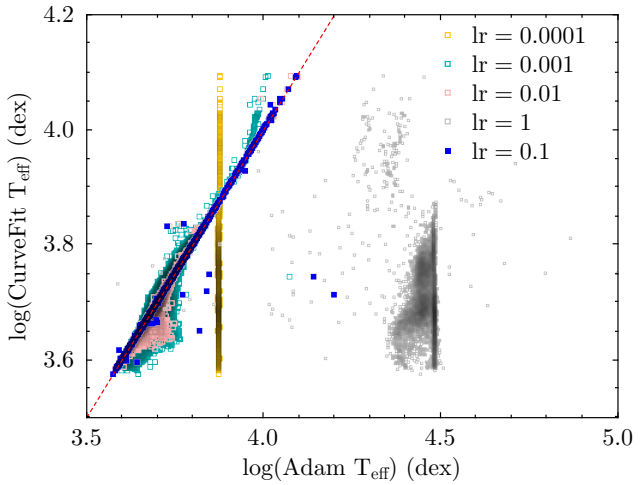

Figure 7. Comparison of $T_{\rm eff}$ between LASP-Adam-GPU and LASP-CurveFit under different initialization schemes, based on 10,000 randomly selected spectra. LASP-Adam-GPU uses fixed initial values of (7500 K, 3 dex, -0.5 dex) and applies a convergence threshold of 10^{-5}, while LASP-CurveFit uses CFI-derived initial values. Scatter point colors indicate different learning rates. The x-axis shows LASP-Adam-GPU results, and the y-axis shows LASP-CurveFit results. The "LASP" prefix is omitted in all figure labels for clarity.

statistically consistent and exhibit similar trends with respect to S/N. For this analysis, we select 25,878 spectra from a total of 177,848 that have at least two repeat observations, S/N in the g band (S/N_g) greater than zero, and less than 10% variation in S/N_g across epochs, to ensure robust error assessment.

As shown in Figure 8, the model-based errors from LASP-Adam-GPU for $T_{\rm eff}$, $\log g$, and [Fe/H] are statistically consistent with the corresponding repeat-observation errors, suggesting that the model provides reasonable error estimates. At S/N_g < 20, these model errors are generally larger than the official LAMOST empirical errors, while at S/N_g > 20, they become smaller. This indicates that LASP-Adam-GPU provides an S/N-sensitive error response and better captures noise-driven variability than the official empirical formula. In contrast, the behavior of RV errors deviates notably. LASP-Adam-GPU consistently underestimates RV errors relative to repeat observations, which in turn lie below the official empirical values. A similar phenomenon is observed in the DESI RVS catalog. We speculate that this discrepancy mainly arises from the fact that the model does not include systematic effects such as wavelength calibration residuals and instrumental response variations in the error propagation process, and these factors have a more significant impact on RV measurements. Since repeat observations can partially reflect such systematic errors, while the model errors are solely derived from the propagation of spectral noise, the estimated results from the two approaches exhibit a systematic deviation of approximately 1.5 km s^{-1} in the RV dimension. These results suggest that, with the exception of RV, the model errors estimated by LASP-Adam-GPU are generally consistent with the repeat-observation errors, and are more reasonable in the low-S/N regime.

4.4. Application to DESI

DESI is a state-of-the-art, highly multiplexed spectroscopic facility mounted on the Mayall 4 m telescope at the Kitt Peak National Observatory (J. H. Silber et al. 2023; T. N. Miller et al. 2024; C. Poppett et al. 2024). In addition to its cosmological program, DESI conducts the Milky Way Survey (MWS) during bright-time conditions to obtain extensive spectroscopy of Galactic stellar populations. These stellar

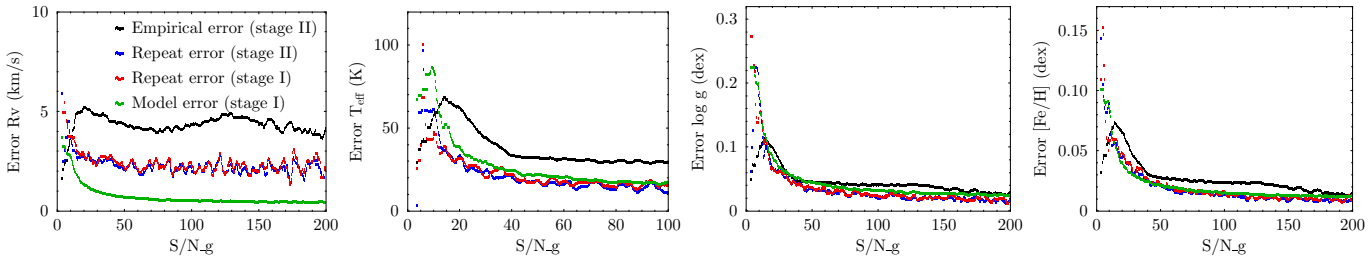

Figure 8. Comparison of the 0.5 quantile (median) errors for LASP-Adam-GPU (stage I), repeat observation (stage I), LAMOST official empirical formula (stage II), and LAMOST official repeat observation (stage II) as a function of `S/N_g`. The errors are smoothed using an Epanechnikov kernel (smoothing factor is set to 2). For clarity, the legend is shown only in the first panel.

spectra span 3600–9800 Å, with spectral resolution increasing from $R \sim 2000$ at 3600 Å to $R \sim 5000$ at 9800 Å. For the MWS program, these spectra are analyzed by two independent pipelines for stellar parameter inference (S. E. Koposov et al. 2025): the RVS pipeline uses RVSpecFit with neural network emulators trained on PHOENIX spectra to infer RV, $T_{\rm eff}$, $\log g$, [Fe/H], and [α/Fe], while the SP pipeline employs FERRE based on Kurucz models to infer $T_{\rm eff}$, $\log g$, [Fe/H], and the abundances of 10 individual elements.

To assess the applicability of PyLASP to other spectroscopic surveys, we apply it to the MWS in DESI DR1, comprising 6,369,991[14] spectra used to infer RV, $T_{\rm eff}$, $\log g$, and [Fe/H]. The PyLASP inference is performed over the wavelength range 4200–5700 Å, sampled in base-10 logarithmic wavelength space with a step size of 0.0001. The initial values for atmospheric parameter optimization are set to (7500 K, 3 dex, −0.5 dex), and LASP-Adam-GPU is configured with a default learning rate of 0.1 and a convergence threshold of 10^{-5}. Under the No Clean strategy, LASP-CurveFit and LASP-Adam-GPU complete the inference in 35 and 8 hr, respectively; with the Clean strategy, the runtimes increase to 43 and 14 hr. The APOGEE stellar parameter catalog is adopted as the reference benchmark,[15] and the parameters inferred by PyLASP and by DESI's two official pipelines (RVS and SP) are independently compared against it.

As shown in Figure 9, the Clean strategy in PyLASP improves the consistency of $T_{\rm eff}$, $\log g$, and [Fe/H] over the No Clean strategy, indicating effective removal of bad pixels in DESI spectra. Compared to the RVS and SP pipelines, PyLASP shows better agreement with APOGEE for $T_{\rm eff}$ and $\log g$, while the DESI pipelines yield better consistency in RV and [Fe/H]. Notably, PyLASP exhibits a systematic underestimation of [Fe/H] in the low-S/N regime (`S/N_B` < 50). This bias may arise from the limited wavelength coverage of the ELODIE spectra, which does not extend to the blue end of DESI spectra where many metal lines are located. In contrast, the broader wavelength range used in DESI's RVS and SP pipelines may provide greater robustness in this region. We further note that, beyond the systematic [Fe/H] bias, other outliers relative to APOGEE are mainly concentrated at low `S/N_B` for LASP-CurveFit, whereas a smaller subset of outliers is also exhibited at high `S/N_B` for LASP-Adam-GPU. In these high `S/N_B` cases, roughly one-third of the affected spectra contain negative or zero flux, while another portion exhibits pronounced absorption or emission features. These parameter anomalies in LASP-Adam-GPU results at high `S/N_B`, relative to APOGEE, are mainly attributable to two factors. First, LASP-Adam-GPU, unlike LASP-CurveFit, lacks explicit failure detection mechanisms for parameter inference —for instance, it still treats results as valid even when the multiplicative correction factors $P(x)b_i$ are nonpositive, indicating that anomalous spectral features in the observed spectra cause an abnormal pseudocontinuum, which destabilizes the objective function (Equation (9)) and results in unreliable parameter inferences. Second, LASP-Adam-GPU applies a fixed, intergroup Clean strategy rather than adapting the iterative process to individual spectra. As a result, anomalous spectral features within spectra of the same group may not be fully masked, potentially leading to unreliable parameter inferences. These limitations will be addressed in future versions of PyLASP. Additionally, implementation details such as the order of Legendre polynomials and the resampling step size used in PyLASP may also contribute to differences in RV. A more detailed analysis will be provided in S. Li et al. (2025, in preparation).

4.5. Future Improvements

To further improve the efficiency, robustness, and reliability of parameter inference in PyLASP, we propose the following three directions for future development:

1. *Enhancing curvature awareness in the optimizer*. Compared to LASP-CurveFit, LASP-Adam-GPU generally requires more iterations to converge, primarily because the Adam optimizer relies solely on first-order gradients and lacks sensitivity to the curvature of the objective function. In contrast, LASP-CurveFit captures local curvature through an approximate Hessian, enabling faster convergence with fewer iterations. To reduce the number of iterations and improve optimization efficiency, we plan to incorporate curvature information into LASP-Adam-GPU using second-order central differences to approximate the Hessian. This approach can be designed to take advantage of parallel computation, balancing speed and accuracy. In addition, symbolic differentiation or learned derivative models may be employed to further reduce the cost of gradient evaluation.
2. *Improving the adaptiveness of the Clean strategy*. Currently, LASP-Adam-GPU applies a fixed iteration schedule for outlier pixel rejection across all spectra in a group, without adjusting for individual spectral quality. This uniform scheme limits the ability to detect outliers in low-quality spectra and may contribute to

[14] The full MWS catalog contains 6,372,607 spectra; we exclude 2608 labeled as BAD and eight as NON based on the `OBJTYPE` field.

[15] APOGEE provides higher spectral resolution ($R \sim 22{,}500$ versus $R \sim 2000$–5000 for DESI), reduced interstellar extinction in the H band, and well-established external calibrations of stellar parameters.

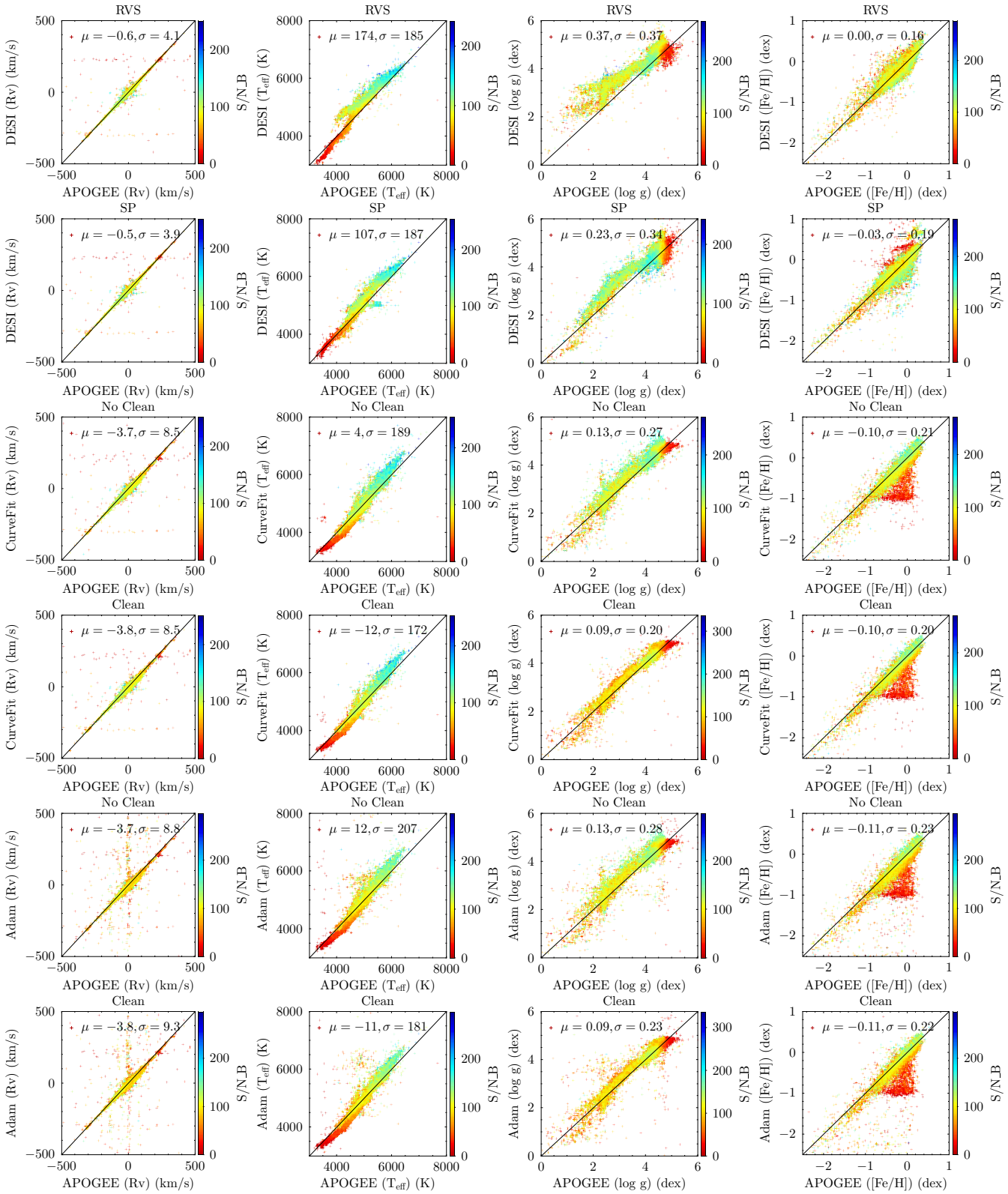

Figure 9. Comparison of inferred parameters with APOGEE DR16 labels. From top to bottom, the six rows correspond to comparisons between APOGEE DR16 and (1) the DESI RVS pipeline, (2) the DESI SP pipeline, (3) LASP-CurveFit with the No Clean strategy, (4) LASP-CurveFit with the Clean strategy, (5) LASP-Adam-GPU with the No Clean strategy, and (6) LASP-Adam-GPU with the Clean strategy. From left to right, the columns show RV, $T_{\rm eff}$, $\log g$, and [Fe/H]. The μ and σ of the differences are computed using `astropy.stats.sigma_clipped_stats`, excluding values beyond 5σ. For clarity, the "LASP" prefix is omitted in all figure labels.

discrepancies with LASP-MPFit under the Clean strategy. To enhance robustness, future versions will implement adaptive Clean control based on spectrum-specific properties, such as S/N or fitting residuals, allowing for dynamic adjustment within groups and improving consistency with the LASP-MPFit Clean implementation.

3. *Refining parameter validation for catalog outputs*. Due to limited spectral quality, certain targets may exhibit abnormal emission lines or unflagged bad pixels not recorded in the official FITS files, which can compromise parameter reliability. Although the Clean strategy helps mitigate the effects of such artifacts, significant discrepancies between results from the Clean and No Clean modes often indicate unreliable parameters. We therefore propose the use of robust statistical metrics (e.g., sigma clipping) to implement a parameter consistency check, identify and exclude unreliable results, and prioritize Clean-mode parameters in the final catalog. This screening mechanism will improve the scientific utility and reliability of the recommended catalog.

5. Conclusions

Based on the original IDL version of LASP, we have developed a Python-based implementation tailored for large-scale stellar spectral analysis. The new framework incorporates two optimization strategies—LASP-CurveFit and LASP-Adam-GPU—within a modular architecture, and we systematically evaluate its performance in terms of inference efficiency, parameter consistency, error modeling, and cross-survey applicability. The PyLASP code and DESI-based catalog are available via DOI: 10.12149/101679 and DOI: 10.12149/101675, respectively.[16] The main conclusions are as follows:

1. *Improved efficiency in parameter inference*. On the same notebook platform, the runtime efficiency of LASP-Adam-GPU and LASP-CurveFit reaches 3.23 and 1.75 times that of LASP-MPFit, respectively. In addition, LASP-Adam-GPU enables fast parameter inference for tens of millions of stellar spectra on high-performance GPUs, completing 10 million spectra in 8 and 7 hr on the RTX 3090 and A100, respectively. These results demonstrate the framework's strong parallel scalability and computational efficiency.
2. *Consistent parameter estimates with the IDL version*. The Python and IDL versions yield highly consistent parameter estimates, with discrepancies observed in fewer than 0.35% of cases. Further analysis indicates that abnormal flux patterns are the primary cause of outliers (Figure 4); applying the Clean strategy significantly reduces such inconsistencies. Comparison with APOGEE labels shows that the Python version achieves better agreement under the No Clean strategy, while LASP-Adam-GPU performs slightly worse under the Clean mode, possibly due to the use of a fixed number of Clean iterations without spectral-quality-aware adaptation.
3. *Different sensitivity to initialization*. LASP-CurveFit is robust to initial values. In contrast, due to the piecewise structure of the spectral emulator in the $T_{\rm eff}$ dimension, LASP-Adam-GPU shows increased sensitivity to the initial temperature when $T_{\rm eff} > 8000$ K. To improve the inference accuracy of LASP-Adam-GPU in the high-temperature regime, we recommend using the initial values provided by the CFI when available. If CFI initialization is not accessible, we suggest adopting (7500 K, 3 dex, −0.5 dex) as the default initial guess.
4. *Model-based errors are consistent with repeat observations*. The atmospheric parameter errors estimated by PyLASP are highly consistent with the random errors calculated from multiple observations of the same target, and outperform the empirical correction scheme based on fitted functions. The tendency of RV errors to be smaller than the corresponding repeat-observation errors is also observed in the DESI data, indicating this bias may be widespread and deserves further investigation.
5. *Reliable performance across different surveys*. When applied to DESI DR1 data, PyLASP produces $T_{\rm eff}$ and $\log g$ values that agree well with APOGEE labels. For [Fe/H], performance is slightly worse than the DESI RVS and SP pipelines in low-S/N regions, likely due to the limited wavelength coverage of the ELODIE-based templates. RV precision is also slightly lower, possibly due to the impact of spectral sampling resolution.
6. *Recommended use cases and application conditions*. PyLASP, built on the ELODIE library, targets optical spectra with $R < 10{,}000$ and wavelength coverage 3900–6800 Å, provided the targets' atmospheric parameters lie within $T_{\rm eff} = 3100$–$59{,}000$ K, $\log g = 0$–5 dex, and $[{\rm Fe/H}] = -2.8$–1 dex. Within this domain, the framework adaptively matches observational resolution without retraining the spectral emulator. If the resolution, wavelength range, or parameter domain exceeds ELODIE's coverage, extend the framework by replacing the model-spectrum generation module and adjusting the wavelength sampling. For applications, LASP-CurveFit with the Clean strategy is preferred for small-scale to medium-scale datasets ($\leqslant$100,000) or low-quality spectra, owing to its robustness to initialization and effective handling of anomalous flux features; LASP-Adam-GPU is recommended for large-scale analyses ($>$100,000) or joint optimization in high-dimensional parameter spaces, being better suited to multivariate optimization when the number of free parameters ranges from a few dozen to tens of thousands, where CurveFit may become inefficient.

In summary, PyLASP significantly improves the efficiency of large-scale spectral parameter inference and demonstrates reliable performance in terms of parameter accuracy, error modeling, and cross-survey applicability. These developments establish a robust computational foundation for extending LASP to high-dimensional label inference, chemical abundance modeling, and automated analysis pipelines in upcoming large-scale spectroscopic surveys.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (12273075, 12273078, and 12411530071) and the National Astronomical Observatories of the Chinese Academy of Sciences (No. E4ZR0516). We also acknowledge support from a Royal Society IEC\NSFC\233140 exchange grant.

Guoshoujing Telescope (the Large Sky Area Multi-object Fiber Spectroscopic Telescope, LAMOST) is a National Major Scientific Project built by the Chinese Academy of Sciences. Funding for the project has been provided by the National Development and Reform Commission. LAMOST is operated

[16] Each of these DOIs represents version 1.0 of the specified repository. DOI: 10.12149/101678 and DOI:10.12149/101674 will resolve to the latest version of each repository.

and managed by the National Astronomical Observatories, Chinese Academy of Sciences.

Funding for the Sloan Digital Sky Survey IV has been provided by the Alfred P. Sloan Foundation, the U.S. Department of Energy Office of Science, and the Participating Institutions.

SDSS-IV acknowledges support and resources from the Center for High Performance Computing at the University of Utah. The SDSS website is www.sdss4.org.

SDSS-IV is managed by the Astrophysical Research Consortium for the Participating Institutions of the SDSS Collaboration including the Brazilian Participation Group, the Carnegie Institution for Science, Carnegie Mellon University, Center for Astrophysics—Harvard & Smithsonian, the Chilean Participation Group, the French Participation Group, Instituto de Astrofísica de Canarias, the Johns Hopkins University, Kavli Institute for the Physics and Mathematics of the Universe (IPMU)/University of Tokyo, the Korean Participation Group, Lawrence Berkeley National Laboratory, Leibniz Institut für Astrophysik Potsdam (AIP), Max-Planck-Institut für Astronomie (MPIA Heidelberg), Max-Planck-Institut für Astrophysik (MPA Garching), Max-Planck-Institut für Extraterrestrische Physik (MPE), National Astronomical Observatories of China, New Mexico State University, New York University, University of Notre Dame, Observatário Nacional/MCTI, the Ohio State University, Pennsylvania State University, Shanghai Astronomical Observatory, the United Kingdom Participation Group, Universidad Nacional Autónoma de México, University of Arizona, University of Colorado Boulder, University of Oxford, University of Portsmouth, University of Utah, University of Virginia, University of Washington, University of Wisconsin, Vanderbilt University, and Yale University.

This research used data obtained with the Dark Energy Spectroscopic Instrument (DESI). DESI construction and operations are managed by the Lawrence Berkeley National Laboratory. This material is based on work supported by the U.S. Department of Energy, Office of Science, Office of High-energy Physics, under contract No. DE-AC02-05CH11231, and by the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility under the same contract. Additional support for DESI was provided by the U.S. National Science Foundation (NSF), Division of Astronomical Sciences, under contract No. AST-0950945 to the NSF's National Optical-infrared Astronomy Research Laboratory; the Science and Technology Facilities Council of the United Kingdom; the Gordon and Betty Moore Foundation; the Heising-Simons Foundation; the French Alternative Energies and Atomic Energy Commission (CEA); the National Council of Humanities, Science and Technology of Mexico (CONAHCYT); the Ministry of Science and Innovation of Spain (MICINN); and the DESI Member Institutions: www.desi.lbl.gov/collaborating-institutions/. The DESI collaboration is honored to be permitted to conduct scientific research on I'oligam Du'ag (Kitt Peak), a mountain with particular significance to the Tohono O'odham Nation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. National Science Foundation, the U.S. Department of Energy, or any of the listed funding agencies.

ORCID iDs

Jun-Chao Liang https://orcid.org/0009-0001-9085-8718
Yin-Bi Li https://orcid.org/0000-0001-7607-2666
A-Li Luo https://orcid.org/0000-0001-7865-2648
Fang Zuo https://orcid.org/0000-0002-9081-8951
Bing Du https://orcid.org/0000-0001-6820-6441
Shuo Li https://orcid.org/0000-0002-8913-3605
Xiao-Xiao Ma https://orcid.org/0000-0002-9279-2783
Shu-Guo Ma https://orcid.org/0000-0001-5066-5682
Hai-Ling Lu https://orcid.org/0000-0002-8252-8743
Ke-Fei Wu https://orcid.org/0009-0009-0513-7042
Zhi-Hua Zhong https://orcid.org/0009-0003-6636-8203
Wen Hou https://orcid.org/0000-0003-0716-1029
Xiao Kong https://orcid.org/0000-0001-8011-8401
Shuo Ye https://orcid.org/0009-0005-6200-2778
Li-Li Wang https://orcid.org/0000-0001-7783-9662
Hugh R. A. Jones https://orcid.org/0000-0003-0433-3665

References

Almeida, A., Anderson, S. F., Argudo-Fernández, M., et al. 2023, ApJS, 267, 44
The Astropy Collaboration, Price-Whelan, A. M., Lim, P. L., et al. 2022, ApJ, 935, 167
Blanton, M. R., Bershady, M. A., Abolfathi, B., et al. 2017, AJ, 154, 28
Buder, S., Asplund, M., Duong, L., et al. 2018, MNRAS, 478, 4513
Buder, S., Kos, J., Wang, X. E., et al. 2025, PASA, 42, e051
Cappellari, M. 2016, MNRAS, 466, 798
Carnall, A. C. 2017, arXiv:1705.05165
Cipolla, R., Gal, Y., & Kendall, A. 2018, in Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE), 7482
Cirasuolo, M., Fairley, A., Rees, P., et al. 2020, Msngr, 180, 10
Collobert, R., & Weston, J. 2008, in Proc. 25th Int. Conf. Machine Learning (ACM), 160
Cooper, A. P., Koposov, S. E., Allende Prieto, C., et al. 2023, ApJ, 947, 37
CSST Collaboration, Gong, Y., Miao, H., et al. 2025, arXiv:2507.04618
Cui, X.-Q., Zhao, Y.-H., Chu, Y.-Q., et al. 2012, RAA, 12, 1197
de Jong, R. S., Bellido-Tirado, O., Brynnel, J. G., et al. 2022, SPIE, 12184, 1218414
De Silva, G. M., Freeman, K. C., Bland-Hawthorn, J., et al. 2015, MNRAS, 449, 2604
DESI Collaboration, Abdul-Karim, M., Adame, A. G., et al. 2025, arXiv:2503.14745
Drosg, M. 2009, Dealing with Uncertainties: A Guide to Error Analysis (Springer)
Du, B., Luo, A., Zhang, J., Wu, Y., & Wang, F. 2012, SPIE, 8451, 845137
Gaia Collaboration, Vallenari, A., Brown, A. G. A., et al. 2023, A&A, 674, A1
García Pérez, A. E., Prieto, C. A., Holtzman, J. A., et al. 2016, AJ, 151, 144
Gong, Y., Liu, X., Cao, Y., et al. 2019, ApJ, 883, 203
Harris, C. R., Millman, K. J., van der Walt, S. J., et al. 2020, Natur, 585, 357
Huang, J.-T., Li, J., Yu, D., Deng, L., & Gong, Y. 2013, in Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (IEEE), 7304
Jin, S., Trager, S. C., Dalton, G. B., et al. 2024, MNRAS, 530, 2688
Jönsson, H., Holtzman, J. A., Prieto, C. A., et al. 2020, AJ, 160, 120
Kingma, D. P., & Ba, J. 2014, arXiv:1412.6980
Kokkinos, I. 2017, in Proc. IEEE Conf. Computer Vision and Pattern Recognition *(Honolulu, HI, USA, 21-26 July 2017)* (IEEE), 6129
Koleva, M., De Rijcke, S., Prugniel, P., Zeilinger, W. W., & Michielsen, D. 2009a, MNRAS, 396, 2133
Koleva, M., Prugniel, P., Bouchard, A., & Wu, Y. 2009b, A&A, 501, 1269
Koleva, M., & Vazdekis, A. 2012, A&A, 538, A143
Koposov, S. E., Allende Prieto, C., Cooper, A. P., et al. 2024, MNRAS, 533, 1012
Koposov, S. E., Li, T. S., Prieto, C. A., et al. 2025, arXiv:2505.14787
Kumar, N., Prugniel, P., & Singh, H. P. 2025, NewA, 119, 102416
Lawson, C. L., & Hanson, R. J. 1995, Solving Least Squares Problems (SIAM)
Li, J., Liu, C., Zhang, B., et al. 2021, ApJS, 253, 45
Liang, J., Bu, Y., Tan, K., et al. 2022, AJ, 163, 153
Luo, A.-L., Zhao, Y.-H., Zhao, G., et al. 2015, RAA, 15, 1095
Majewski, S. R., Schiavon, R. P., Frinchaboy, P. M., et al. 2017, AJ, 154, 94
Markwardt, C. B. 2009, ASPC, 411, 251

Martell, S. L., Sharma, S., Buder, S., et al. 2017, MNRAS, 465, 3203
Miller, T. N., Doel, P., Gutierrez, G., et al. 2024, AJ, 168, 95
Moultaka, J., Ilovaisky, S., Prugniel, P., & Soubiran, C. 2004, PASP, 116, 693
Ness, M., Hogg, D. W., Rix, H.-W., Ho, A. Y. Q., & Zasowski, G. 2015, ApJ, 808, 16
Nocedal, J., & Wright, S. J. 2006, Numerical Optimization (Springer)
Paszke, A., Gross, S., Massa, F., et al. 2019, arXiv:1912.01703
Poppett, C., Tyas, L., Aguilar, J., et al. 2024, AJ, 168, 245
Prugniel, P., & Soubiran, C. 2004, arXiv:astro-ph/0409214
Prugniel, P., Soubiran, C., Koleva, M., & Borgne, D. L. 2007, arXiv:astro-ph/0703658
Prugniel, P., Vauglin, I., & Koleva, M. 2011, A&A, 531, A165
Recio-Blanco, A., de Laverny, P., Palicio, P. A., et al. 2023, A&A, 674, A29
Różański, T., Ting, Y.-S., & Jabłońska, M. 2025, ApJ, 980, 66
Sharma, K., Prugniel, P., & Singh, H. P. 2016, A&A, 585, A64
Silber, J. H., Fagrelius, P., Fanning, K., et al. 2023, AJ, 165, 9
Smith, V. V., Bizyaev, D., Cunha, K., et al. 2021, AJ, 161, 254
Steinmetz, M., Zwitter, T., Siebert, A., et al. 2006, AJ, 132, 1645
Ting, Y.-S., Conroy, C., Rix, H. W., & Cargile, P. 2019, ApJ, 879, 69
Van Der Marel, R. P., & Franx, M. 1993, ApJ, 407, 525
Varoquaux, G., Grisel, O., Estève, L., et al., 2024 joblib/joblib v1.4.2, Zenodo, doi:10.5281/zenodo.14915602
Virtanen, P., Gommers, R., Oliphant, T. E., et al. 2020, NatMe, 17, 261
Vugrin, K. W., Swiler, L. P., Roberts, R. M., Stucky-Mack, N. J., & Sullivan, S. P. 2007, WRR, 43, W03423
Wang, C., Huang, Y., Yuan, H., et al. 2022, ApJS, 259, 51
Wu, Y., Du, B., Luo, A., Zhao, Y., & Yuan, H. 2014, IAUS, 306, 340
Wu, Y., Luo, A.-L., Li, H.-N., et al. 2011a, RAA, 11, 924
Wu, Y., Singh, H. P., Prugniel, P., Gupta, R., & Koleva, M. 2011b, A&A, 525, A71
Yanny, B., Rockosi, C., Newberg, H. J., et al. 2009, AJ, 137, 4377
Zhao, G., Zhao, Y.-H., Chu, Y.-Q., Jing, Y.-P., & Deng, L.-C. 2012, RAA, 12, 723
Zuo, F., Luo, A.-L., Du, B., et al. 2024, ApJS, 271, 4