Modeling of Telluric Lines in WINERED Spectra Using PCA

Noriyuki Matsunaga
2025 September 4

Abstract

Ground-based spectra in the near-infrared are always imprinted with telluric absorption by Earth's atmosphere. Direct modeling is challenging due to the complexity and variability of atmospheric conditions. To model this absorption in observational spectra empirically, the PCA approach is useful and commonly used now. By observing a sample of hot, feature–poor standard stars under various airmass and weather conditions, we can describe these telluric lines with a small set of principal components (PCs).
Here we provide the scripts and a data set for constructing telluric absorption models of WINERED spectra. The accompanying Python toolkit
* builds order-by-order PCA bases (`get_eigens.py`);
* marks wavelength intervals where the average telluric spectrum is strongly absorbed (`make_tel_abs.py` / `make_tel_abs.sh`);
* checks that the basis reproduces every reference spectrum (`check_model.py`);
* fits science spectra (of any kind of objects unless the blends of intrinsic lines are too severe) with a linear combination of the PCs (`fit_model.py`)
One can mask the wavelength ranges with significant absorption of a target itself (not telluric), for which model spectra for stars with some spectral types and a utility script (`mask_from_obj.py`) are also provided.
With six PCs, the telluric profiles are reproduced with root-mean-square (RMS) residuals typically below 1% over 11 WINERED WIDE-mode orders (43–48, 52, 55–58). We ignore the orders 53 and 54, in which the telluric absorption is almost negligible. This toolkit facilitates robust telluric correction for WINERED WIDE-mode spectra and is adaptable to other instruments.

Analysis flow and helper scripts

Step 0: Convert all FITS spectra to 2-column text

```
python3 fits_to_txt.py          # prepare the dataset in each expected directory
```

Step 1. PCA basis (once)

```
python3 get_eigens.py           # creates ave_<order>.txt
```

Step 2. Telluric masks (once, all orders)

```
bash make_tel_abs.sh          # calls make_tel_abs.py m44 0.97 ...
```

Step 3. Quick self-check (optional, per order)

```
bash check_model.sh           # calls check_model.py m44 …
```

Step 4. Fit science spectra

```
bash test_fit_model.sh          # calls fit_model.py
```

Data preparation

We utilized 40 spectra of telluric standard stars whose intrinsic stellar lines had already been removed. These spectra were obtained on different dates and under varying airmass conditions. The main variation among them arises from differences in the strength of telluric absorption lines due to factors like airmass and humidity. We assume that the absorption line profiles at each wavelength do not vary significantly, although even moderate changes in line shape should still be manageable for modeling. Additionally, about four of the spectra were taken with a 200 μm slit instead of a 100 μm one, but we used them without distinguishing between the two. These spectra were also well reproduced using the derived models, suggesting that variations in telluric line widths can be effectively modeled.

Considering that the atmospheric absorption differs significantly between observations at Koyama Observatory in Japan and Chilean observatories (NTT, LCO), we restricted our analysis to WIDE-mode spectra obtained in Chile after 2017. The list of selected data is compiled in a file named `tel_list`. For each standard star, a FITS file with intrinsic stellar lines removed is located in a subdirectory under telluric/FITS. Since the subdirectory names vary depending on the observing run or analyst, we used the script `fits_to_txt.py` to locate the relevant FITS files, convert them to text format, and organize them into a txt subdirectory for each star.

Note that the line removal and analysis were performed using vacuum wavelengths (VAC), and all FITS and text files discussed here are in vacuum wavelengths. We analyze 11 spectral orders: 43–48, 52, and 55–58 (and maybe some other orders). The orders 53 and 54 are not included because they show negligible telluric absorption. Each order is treated independently, and the wavelength ranges slightly exceed the free spectral range (FSR). These ranges are defined in the list `order_wranges_with_buffer` in utils.py and used across scripts.

Construction of basis spectra using PCA

The PCA basis spectra for modeling telluric absorption are constructed using the script `get_eigens.py`, which implements the following procedure:

We first compile the observed spectra of telluric standard stars from different nights and airmass conditions. For each spectral order, the corresponding wavelength range is defined (with buffer) in `utils.order_wranges_with_buffer`, and a fixed wavelength grid is generated using `utils.make_order_waves`. Spectra are interpolated onto this grid using cubic splines. Any pixels outside the observed range are padded with a continuum value of 1.0.

To ensure alignment between spectra, the second and subsequent spectra are shifted in wavelength and flux to minimize residuals relative to the running mean of previously processed spectra. Each interpolated spectrum is then standardized (i.e., mean-subtracted and divided by its standard deviation), and these standardized spectra are stacked into a matrix **X** of shape *(m × n)*, where *m* is the number of usable spectra and *n* is the number of wavelength pixels.

A covariance matrix of shape *(m × m)* is computed from **X**, and its eigenvalues and eigenvectors are obtained via symmetric eigen-decomposition (`numpy.linalg.eigh`). The first *p* eigenvectors (with the largest eigenvalues) define the principal components. We project **X** onto these eigenvectors to construct the basis spectra **A** (shape: *n* × *p*), where each column of **A** represents one basis spectrum.

The sign of each basis spectrum is adjusted so that absorption features consistently point downward. The resulting basis spectra are saved to `base_[order].txt`, and the corresponding wavelength grid is saved as `waves_[order].txt`.

To define a representative average spectrum, we take the mean of the original (non-standardized) spectra and fit it as a linear combination of the PCA basis vectors using linear least-squares. The resulting model is saved as `ave_[order].txt`. Once we get these average telluric spectra, `make_tel_abs.sh` (calling `make_tel_abs.py`) creates the wavelength regions that contain significant telluric absorption.

For diagnostic purposes, the script also produces two plots per spectral order:
(1) A plot of the eigenvalue spectrum to assess the variance captured by each component, and
(2) A plot of the scaled and normalized PCA basis vectors overlaid with the average and composed model spectra. The basis spectra are weighted by their corresponding eigenvalues and scaled for visibility.

This algorithm follows the PCA decomposition technique presented in Singh et al. (1998, MNRAS, 295, 312), adapted to the specific task of modeling telluric absorption features across a variety of observational conditions.

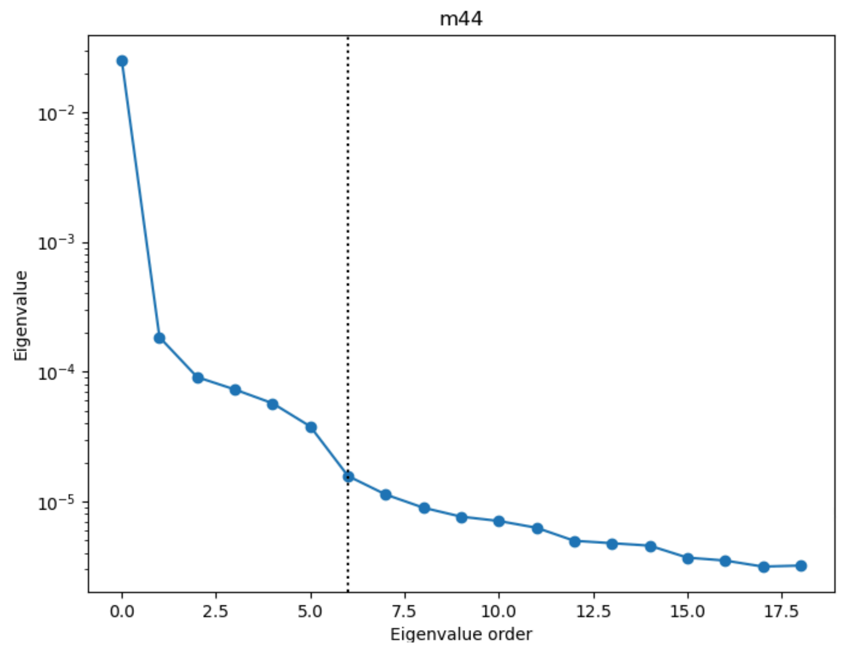


Figure 1 Eigen-value plot (log-scale) showing that the first six PCs capture >99 % of the variance.

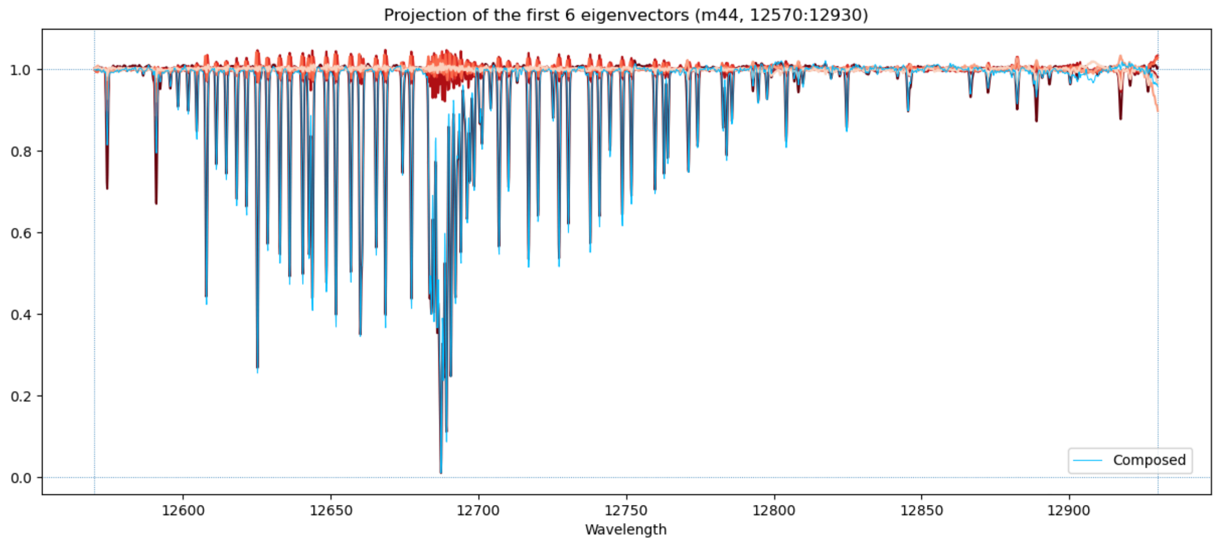


Figure 2 Basis spectra overlaid with average spectrum (black) and PCA reconstruction (cyan).

Modeling telluric absorption in standard star spectra

After the PCA basis vectors are in place, `check_model.py` validates how well they reproduce the same 40 reference spectra that were used to build the basis set:

1. **Input & setup**

   - The user supplies a single spectral order on the command line (e.g. `m44`).
   - For that order the script reads
      1. the wavelength grid `waves_[order].txt`,
      2. the fitted average spectrum `ave_[order].txt`, and
      3. the matrix of PCA basis vectors `base_[order].txt`.
   - A constant (all–ones) column is appended to the basis matrix so that a uniform continuum offset can be fitted simultaneously.

2. **Preparing each observed spectrum**
   For every telluric-standard star in `tel_list`:
   - The original spectrum is read and **aligned** to the PCA grid by a two-parameter correction returned by `utils.calc_adjust`:
      1. a wavelength shift (*xadjust*, in Å) and
      2. a small vertical offset (*yadjust*, in flux units) that minimises the difference from the average spectrum.
   - A cubic-spline interpolator transfers the observed fluxes onto the common wavelength grid. Pixels lying outside the star's actual coverage are ignored.

3. **Linear-combination fit**
   - Only grid points that fall within the star's observed range (`utils.check_pixels_use`) are used.
   - For these points, the code selects the corresponding rows of the augmented basis matrix and subtracts the mean level from the observed fluxes to center them around zero.
   - It then performs a linear least-squares fit (using numpy.linalg.lstsq) to determine the best combination of basis vectors that reproduces the observed flux pattern.
   - Once the optimal set of coefficients is obtained, the model spectrum is reconstructed by applying this linear combination of basis vectors and adding back the mean flux level that was initially removed.

4. **Diagnostics**

   - The script calculates the standard deviation of the difference between the observed and model spectra for each star. This value, along with the number of fitted pixels and the applied wavelength shift, is printed as a diagnostic. In most cases, the standard deviation is less than 1%, indicating that the six principal components are sufficient to reproduce nearly all telluric absorption features.
   - For visual inspection the script writes one PNG per star (saved under `check_model/[order]/`) containing
      1. the observed flux (red) and the fitted model (blue), and
      2. the residual curve below it, with σ annotated.
         Vertical dotted lines mark the nominal order limits; the continuum level is shown as a horizontal dotted line.

This end-to-end check demonstrates that, even when a constant continuum term is fitted simultaneously, a linear combination of the six PCA basis vectors can reproduce the telluric absorption in every reference spectrum to better than ~1 % rms, validating the basis for subsequent application to science targets.

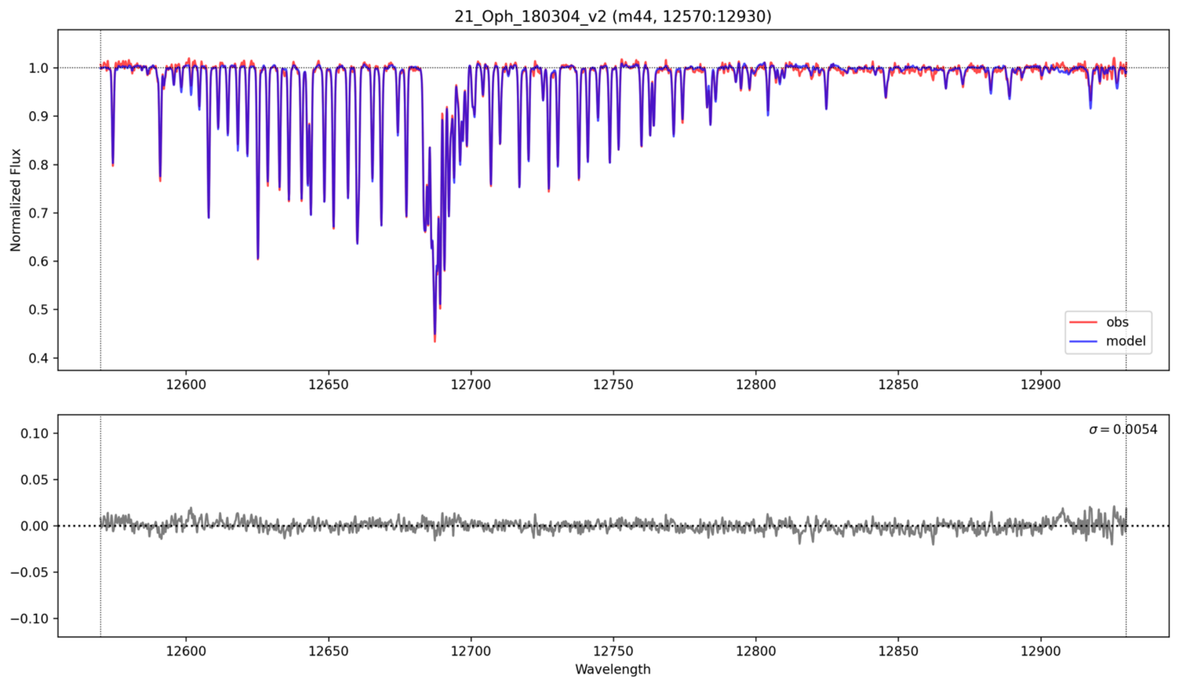


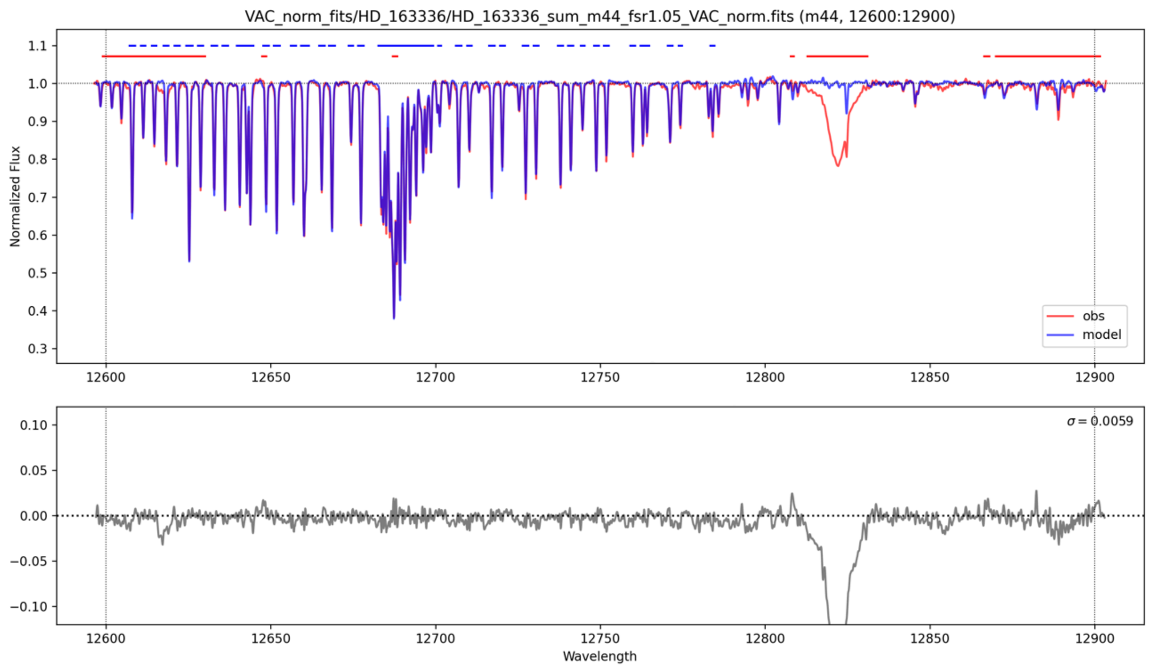
Figure 3 Per-star diagnostic: (top panel) observed (red) vs. model (blue); (bottom panel) residuals with σ annotation.

Modeling telluric absorption in new observed spectra

We use the script `fit_model.py` to apply the PCA-based model to new observed spectra. It fits a linear combination of the six basis spectra to reproduce the telluric absorption features. `fit_model.py` supports both text and FITS formats as input/output.

It is not strictly necessary to remove intrinsic stellar lines beforehand. For example, the method works well with A0-type standard stars and Cepheids, which have relatively few strong absorption lines. However, if there are too many intrinsic lines, the modeling accuracy may decrease.

The script `fit_model.py` accepts various command-line options. Running it with the -h flag displays a brief help message. Options include excluding the spectral edges from the fit (`-t -e EDGE_FRAC`) and specifying wavelength ranges to exclude (`--reject1 REJECT1, --reject2 REJECT2`). The key difference is that `REJECT1` excludes all wavelengths in the specified range, even if they contain telluric lines, whereas REJECT2 allows telluric pixels within that range to be included in the fit. For spectra with many absorption lines of the target itself, masking such absorption from the fitting (with the --mask_obj option) would be beneficial (see more details below). The -t option can be useful because strange distortion sometimes appears towards the edge of a spectrum. With the recommended -x option, a small offset in wavelength is determined and adjusted. The script also supports iterative fitting with sigma clipping for pixels with large residuals. This clipping is treated as REJECT2, meaning telluric pixels in these ranges are still used for fitting if relevant. The telluric line wavelength regions are defined in the text files eigen_txt/tel_abs_[order].txt , which can be created with make_tel_abs.py. Using the -a option enables analysis on spectra in air wavelength scale instead of vacuum. Other options are found in the help message.



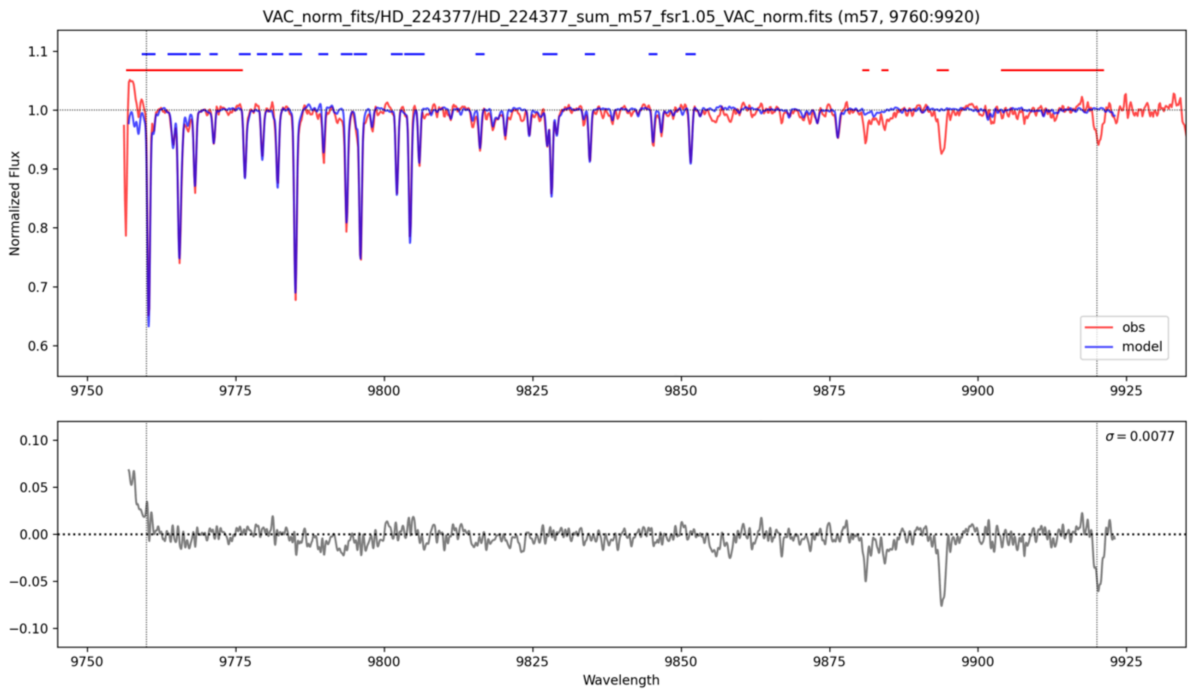


Figure 4 Fit summary: observed (red) vs. model (blue); blue bars = `tel_abs` regions, grey = σ-clipped `REJECT2`, red = user-defined `REJECT1`; residual panel below.

Masking the target's absorption from the telluric fitting

`mask_from_model.py` is a utility script used to identify and mark wavelength regions of significant object absorption in a given model spectrum. This is critical when modeling telluric absorption in observed spectra, since deep stellar or molecular lines intrinsic to the target star can otherwise bias the PCA-based telluric correction. By masking these regions, we ensure that only telluric features are used in the fit.
The script provides a function mask_from_model() for use in other scripts (e.g., fit_model.py), as well as command-line usage for standalone testing and visualization. When you fit the telluric model with fit_model.py, you can make this masking with the options `--mask_obj`, `--obj_rv_add`, and `--obj_thresh`.

What does `mask_from_model.py` do?

- Reads a model spectrum (wavelength + normalized flux)
- Shifts the model to account for a user-specified radial velocity
- Detects significant absorption features (where flux drops below a given threshold)
- Adds buffer pixels to broaden the mask region
- Merges overlapping or adjacent regions into contiguous wavelength ranges
- Outputs the masked wavelength ranges as a comma-separated list

- Optionally saves a plot showing the model and masked regions

Acknowledgements

I’m grateful to Hiroaki Sameshima for his suggestion to use the PCA approach for the WINERED telluric correction and following discussions. I also thank Daisuke Taniguchi for discussions on the algorithm and toolkit.