Journal of
open research software

Rabe, M 2020 Spectram: A MATLAB® and GNU Octave Toolbox for Transition Model Guided Deconvolution of Dynamic Spectroscopic Data. *Journal of Open Research Software,* 8: 13. DOI: https://doi.org/10.5334/jors.323

SOFTWARE METAPAPER

Spectram: A MATLAB® and GNU Octave Toolbox for Transition Model Guided Deconvolution of Dynamic Spectroscopic Data

Martin Rabe
Max-Planck-Institut für Eisenforschung GmbH, Düsseldorf, DE
m.rabe@mpie.de

Spectroscopic data, depending on an experimentally controllable variable, contains a wealth of information for researchers. However, complex spectra with overlapping peaks and multiple transitions complicate its straightforward interpretation and often the full contained information cannot be extracted. Here, the Spectram toolbox for MATLAB® and GNU Octave is described which was developed to analyse such data by a method based on singular value decomposition (SVD) and transition model coupled recombination. The method employs user-defined transition models, which depend on the control variable and are often known, or empirical descriptions of the transitions, which often can be guessed, to deconvolute such data. The outcome are the spectral components associated to the transitions and the model parameters. Both can be directly interpreted in terms of their physical meaning. Spectram can be applied to any desired spectroscopic technique and gives full freedom in the choice of the applied models, making it highly reusable.

Keywords: singular value decomposition; matrix least squares; optical spectroscopy; IR-spectroscopy; UV/Vis-spectroscopy; spectral deconvolution; multivariate data analysis
Funding statement: Funding by the European Union's Horizon 2020 research and innovation program under a Marie Skłodowska-Curie Grant (Agreement No. 705857) is acknowledged.

(1) Overview

Introduction

Spectroscopic methods are used in many fields of applied sciences to study dynamic processes. Such approaches yield multivariate spectroscopic data sets, i.e. intensity or absorbance data, depending on a photon energy equivalent such as frequency or wavelength on the one hand and a second control variable on the other hand. Here, control variables are experimental parameters that vary during the experiment in a controllable or measurable manner. These may be for instance temperature, time, incident angle or electrode potential.

The scientific questions underlying the experimental design can be as diverse as the possible techniques. In general, the control variable dependent development of a specific spectral response is probed. It depends on the concentration of a compound or may reflect its conformation, orientation or solvation. However, in complex samples a straightforward physical interpretation of such data is often hampered by its complexity which can be of different origins. In the rarest cases the measured spectral response consists of a single band depending uniquely on the control variable. More often, strongly overlapping spectral bands, with individual dependencies on the control variable, need to be disentangled. This can be caused by a complex sample structure for instance a mixture of similar components or a complex nature of the spectroscopic transition under study, such as transitions caused by chemical groups in different chemical environments or with different orientations.

As an example, a data set, as it might be observed in infrared (IR) spectroscopy, was generated (**Figure 1a**). Spectra are depicted for values of the control variable c gradually increasing in the range of 10, 15, ..., 80. In this example, c may be understood as a temperature and the studied process may be a temperature dependent structural transition. The data set was simulated assuming two separate two state chemical reactions (A $\rightarrow$ A' and B $\rightarrow$ B') with single absorbance peaks for each chemical species. This leads to broad spectra with up to 4 peaks overlapping. Upon variation of c a complex variation of the spectrum is obtained. The difference spectra illustrate that consideration of intensities at single wavenumbers would result in different c dependencies leading to erroneous conclusions on the underlying processes (**Figure 1a**, inset). Thus, to reveal the underlying transitions of such a data set, one must use the full spectroscopic data in the analysis.

One approach to the usage of the full data is fitting each spectrum with several single component peaks and determining their changes depending on *c*. Peak fitting software is readily available which provides convenient tools for such approaches, applicable for researchers of all experience levels. However, modeling such a system with independent peak fittings involves 4 highly overlapping components which leads to a high number of parameters per spectrum and non-unique solutions to the problem. The problem might get even more difficult, given that often the exact number of underlying peaks or their exact shape is not known with certainty.

However, often the underlying transition model, i.e. the *c* dependency of the spectral change for the studied process is better known than the actual spectra of the individual components. Examples for such transitions and their control variables are the concentration dependency of p*H* indicators given by the *Henderson Hasselbach* equation, the temperature dependency of equilibrium constants described by the *van't Hoff* equation or the temporal decay of species following a specific rate law (**Table 1**). When the the physical model is unknown, an empirically descriptive model can sometimes be devised to describe the principal trends in the data, for instance a linear or sigmoidal model that sufficiently describes many real physical processes.

An analysis approach that allows to employ physical or empirical transition models and uses all available spectral information has been developed by Shrager and Hendler [1, 2, 3]. It is used to deconvolute the data set by means of linear algebraic methods. In particular, the data is first decomposed by singular value decomposition (SVD) and subsequently recombined using a distinct transition model and a matrix least squares (MLS) minimization. Thus, the SVD-based MLS deconvolutes the spectral components as determined by the transition model, which is chosen by the investigator. This approach allows:

1. Determination of the number of independent transitions within the spectral data set
2. Determination of the transitions (f_1…f_3 in **Figure 1b**) and their model parameters (p_{i_j})
3. Determination of the base spectral components (d_1…d_3 in **Figure 1b**)

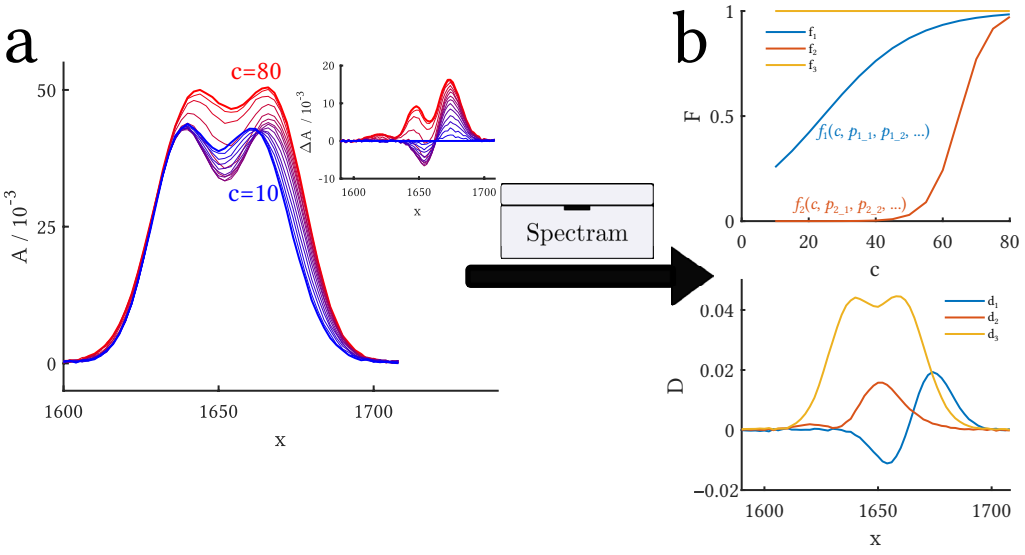

Figure 1: Input and output for SVD-based MLS by means of Spectram toolbox. **(a)** Spectral data set **A**(*x, c*) generated from two independent chemical transitions with the control variable *c*. In practice *x* can be any energy equivalent common in spectroscopy (wavenumber, frequency, wavelength). The inset shows the change in spectral intensities. Random error was added to the generated data. **(b)** Application of Spectram box results in two matrices **F** and **D** which describe the original data by $\mathbf{A} = \mathbf{D}\mathbf{F}^T$. **F** contains in its columns the transitions in *c* determined by the model functions (f_1 and f_2). The obtained parameters p_{i_j} may be physical quantities, when physical models are chosen over pure empirical descriptions. **D** contains in its columns the individual difference spectra $\mathbf{D}_i$ for each chemical compound.

Table 1: Examples for applications of control variable *c* dependent transitions that may be studied by SVD-based MLS using Spectram. *T*: temperature, *t*: time.

Control variable	Transition model function	Quantifiable model parameters
p*H*	*Henderson Hasselbach* equation	acid dissociation constant pK_a
T	*van't Hoff* equation	standard enthalpy change $\Delta H°$
t	rate law, qualitative description by exponential decay	rate constants *k*, half life $t_{½}$
any *c*	qualitative description for instance by sigmoidal	position of transition in *c*

Next to the spectral input data set, ordered in the matrix **A**(*x, c*), the researcher has to input the model functions f_1 and f_2 and a rank *r*. The spectral data **A**(*x, c*) is decomposed into two data sets **F** and **D** (**Figure 1b**). **F** contains the *c* dependent transitions and **D** contains the associated spectral components. In the example the last transition is kept constant with $f_3 = 1$. In this way the spectral component d_3 can be interpreted as a base spectrum and d_1 and d_2 as difference spectra. Thus, d_1 and d_2 indicate the change of the base spectrum d_3 upon variation in *c*. The gained information in this example is substantial: two independent, in *c* well separated transitions are revealed. These are associated to significant spectral changes which allow for an interpretation in terms of the underlying molecular process.

In our own research we have found SVD-based MLS very useful for the interpretation of large data sets from temperature dependent transmission IR spectroscopy [4], electrochemical in-situ attenuated total reflection (ATR)-IR spectroscopy [5], or time dependent photo luminescence spectroscopy [6]. In the course of these projects the SVD deconvolution was implemented in several scripts using MATLAB®. In the latest version the SVD-based MLS approach described by Shrager [2] was fully implemented. Here, only a shortened description to the application and the outcome of the method shall be given. For a complete and comprehensive discussion, also on the mathematical background and its implementation, the reader is referred to the 'beginners guide' by Hendler and Shrager [3].

Another set of related and well established techniques are multivariate curve resolution (MCR) or self-modeling curve resolutions (SMCR) [7, 8]. MCR techniques apply a set of constraints to the transitions and the spectral components during an alternating least squares minimisation. For these techniques software packages exist, for instance for MATLAB® (http://www.mcrals.info/).

Despite the fact that there is a broad application range of the SVD-based MLS method we are not aware of a published generalized implementation in MATLAB® or GNU Octave, that allows an application to a broad variety of experimental techniques. Thus, we restructured the scripts from the research projects to build up the Spectram toolbox. This toolbox is designed to allow a straightforward application of SVD-based MLS to any spectroscopic technique, using any desired control variable, with unlimited individually definable transitions models. These features enable a high reusability and extendibility of Spectram. The toolbox and especially its simple example scripts are released to serve as a jump start for researchers that can benefit from this method but are (still) unfamiliar with coding platforms like MATLAB® or hesitate, due to the work needed for the implementation of this approach. Furthermore, the toolbox may perfectly serve as a base for a user friendly GUI based program for specific applications or spectroscopic techniques.

Implementation and architecture

The Spectram box provides functions to easily follow the workflow described by Hendler and Shrager [3]. The main steps can be executed by applying the supporting functions (**Table 2**). Steps II – V may be repeated until satisfying results are obtained. An example script showing the process and its implementation step by step is included in the release (*typical_example.m*). A detailed documentation is provided to explain the application of the functions in detail.

For the data preparation (step I) standard MATLAB® and GNU Octave commands can be used. Three data inputs must be defined: an array *A* containing in its columns the spectral (intensity or absorbance) data for each value of the control variable, the row vector *c* containing the values of the control variable, and a column vector *x* containing the values of the energy equivalent. For the input *all*(*size*(*A*) == [*length*(*x*), *length*(*c*)]) must apply.

The toolbox performs the SVD by the MATLAB® function *svd*(*A,* 0), which returns three arrays *U*, *S*, and *V* which in linear algebra terms are matrices for which

$$\mathbf{A} = \mathbf{USV} \tag{1}$$

applies.

The rank, determined in step II, defines the number of signal containing components in **U**, **S**, and **V** that are used in the following matrix least squares process. Also, the rank is a lower limit for the number of transitions *n* in the model. An efficient way to determine the rank is the visual inspection of the SVD results (for a detailed description see [3]). For this step, a small app (*RankFinder.m*) is provided for convenience. Other ways to define the rank may be applied, as for instance use of the MATLAB® function *rank* (*A, ...*) or determination of the autocorrelation coefficients.

The key feature, leading to the high versatility of Spectram, is the ability to use any desired number of individual functions for transition models in step III. This

Table 2: Process steps for the SVD-based MLS decomposition and the supporting functions and programs provided by the Spectram box.

	Process Step	Spectram box function or command
I	Prepare data	
II	SVD and rank determination	*RankFinder(...)*
III	Construct transition model	*simple_model(...), model_fun, vecpar(...)*
IV	MLS recombination	*recombfit(...)*
V	Assess results	*eval_model(...), matres(...), plotmatres(...)*
VI	Repeat from II (optional)	

is realized by employing the MATLAB® and GNU Octave anonymous function formalism for the definition of the transition model functions. In Spectram a model consists of an $(n + 1)$-by-1 cell array containing n transition model functions specified as anonymous functions of the form *@(c, p1, p2, ..., pj)myfun(c, p1, p2, ..., pj)*. As shown in the example, the $(n + 1)^{th}$ cell may contain an anonymous function returning constants as for instance: *@(c) ones(size(c))*. Models can be constructed either by using the helper function *simple_model(...)* or manually, by defining the cell array. Also, a function library is provided by the *model_fun* class, containing some (empirical) standard model functions like constants, sigmoidals or exponentials.

The number of adjustable parameters for each model depends on n and the number of parameters in each individual transition model function. For all parameters start values must be defined, while lower and upper bounds can be defined optionally before starting the MLS routine. To aid in the correct construction of the vectors containing the start parameters and bounds, the parameter names lists are used. These are assignments of the model parameter names to the position of its value in the parameter vectors. A parameter names list is returned as 2nd output from *simple_model(...)* or can be constructed directly from the model using the function *vecpar(model)*. The parameter names lists are cell arrays of the form: {*"p1_1"; "p2_1"; ...; "pj_n"*}. Here, the latest numeral represents the position of the associated transition function in the model cell array. Note that the *vecpar(...)* function internally constructs the parameter names list from a parameter map returned by the function *mappar(...)*. The standard user will most probably not have to deal with parameter maps, still details about this concept are found in the documentation of the *mappar(...)* function.

The central computing step IV, the recombination by MLS is done by using the function *recombfit(...)*. It requires the data, the rank and the model as input. Internally, the function employs the function *lsqnonlin(...)* and thus also requires the same additional input parameters as well as it returns the same (optional) outputs. This includes parameters allowing an assessment of the fit quality such as the residuals. The *recombfit(...)* function implements the SVD-based MLS approach [2] and returns the parameters from the minimization procedure. For evaluation of the results (step V) the function values of the resulting model can be calculated by using the *eval_model(...)* function. This function returns an array F which contains in its columns the values for the individual transition model functions. In matrix notation

$$\mathbf{A} = \mathbf{D}\mathbf{F}^T \tag{2}$$

applies. To additionally evaluate $\mathbf{D}$ which contains the spectral information connected to the transitions in $\mathbf{F}$ the *matres(...)* function can be employed. Several methods for evaluating the results and the fit quality have been discussed elsewhere [3]. Some useful plots for evaluation, especially the comparison of the input $\mathbf{A}$ with the recombination $\mathbf{A}'$ as well as the comparison of $\mathbf{V}$ with the matrix $\mathbf{V}'$ constructed from the fit results may be directly created using *plotmatres(...)*. In most cases when approaching a new analytical problem, the first attempt will not lead to satisfying results and several attempts with different inputs will be needed. Usually, the researcher may try different models, numbers of transition model functions, ranks, starting parameters or bounds. Using the toolbox in a script or even as a base for a GUI substantially simplifies all these adaptions, especially the variations of the model.

It should be noted that apart from the formal function of the software (see also section: Quality control) the method comprises intrinsic pitfalls that may lead to meaningless or physical irrelevant results. Thus, a critical result evaluation must be part of the routine. Factors of major influence for the result quality are:

- Choice of appropriate number of transitions
- Choice of appropriate transition models
- Finding the minimum rank
- Using proper starting parameters and physically meaningful bounds

Users that are unfamiliar with the method are advised to use the provided example script to test the influence of improper choices of these parameters to gain experience with the observable results.

Furthermore, the data set under study may impose limitations to the usefulness of the method. For instance, noisy data with very closely spaced transitions or spectral components may not give satisfying results. Also, peak shifts cannot be deconvoluted completely by means of SVD, although in our experience small shifts may still be qualitatively separated when they are significantly well isolated in the control variable space.

Quality control

The core functionality, i.e. the MLS recombination implemented in *recombfit(...)*, has been tested thoroughly within MATLAB® in the application in several research projects over the past years since 2014. A simple test is the deconvolution of generated data sets. For instance, the provided example data set was generated, employing Gaussian peak shapes for two chemical transitions, the *Lambert-Beer* law and transitions model functions, arbitrarily chosen as *arctan* functions giving sigmoidal shapes. Furthermore, random error was added to the spectral data and the transition data. Deconvolution of this data set by means of different sigmoidal model functions, as shown in the example script, yields solutions in F that are qualitatively similar to the input functions and components in D similar to the difference spectra of the input data. The code was adapted partly to be compatible to GNU Octave and tested by means of the provided example script. Minor incompatibility problems with GNU Octave are still possible.

(2) Availability

Operating system

The Spectram toolbox requires MATLAB® 9.0 or higher or GNU Octave 4.4.1 or higher. Thus, it can run on operating

systems for which these are available (including Microsoft Windows 7/10, macOS and Linux distributions).

Programming language
MATLAB®, GNU Octave

Dependencies
MATLAB® (9.0): Optimization Toolbox (8.5)
GNU Octave (4.4.1): optim (1.6.0) package. Optim requires the packages struct (1.0.15), statistics (1.4.0) and io (2.4.12). The installation and the loading of these packages is described in the *readme.md* and the example script.

List of contributors
Martin Rabe

Software location
Code repository
Name: GitHub
Identifier: https://github.com/mrtnrb/Spectram/
Licence: MIT
Date published: 06/02/20

Archive
Name: Zenodo
Persistent identifier: DOI: 10.5281/zenodo.3800533
Licence: MIT
Publisher: Martin Rabe
Date published: 06/05/20
Version published: v1.0.0

Language
English

(3) Reuse potential
The toolbox provides a set of functions that enable the stepwise application of the analysis. There are no limitations on the number of transition model functions or limitations on the applicable control variable. Furthermore, individual model functions can be applied. Thus, the reuse potential is high, so that applications with any dynamic spectroscopic method are possible.

General support, specific feature requests or bug reports can be submitted by opening an issue in the GitHub repository or by email to the author. Users can fork the repository to develop applications or GUIs for specific methods or spectroscopies.

Acknowledgements
Andreas Erbe is acknowledged for general support and critical reading of the manuscript.

Competing Interests
The author has no competing interests to declare.

References
1. **Shrager, R I** and **Hendler, R W** 1982 Titration of individual components in a mixture with resolution of difference spectra, pKs, and redox transitions. *Anal. Chem.,* 54(7): 1147–1152. DOI: https://doi.org/10.1021/ac00244a031
2. **Shrager, R I** 1986 Chemical transitions measured by spectra and resolved using singular value decomposition. *Chemom. Intell. Lab. Syst.,* 1(1): 59–70. DOI: https://doi.org/10.1016/0169-7439(86)80026-0
3. **Hendler, R W** and **Shrager, R I** 1994 Deconvolutions based on singular value decomposition and the pseudoinverse: a guide for beginners. *J. Biochem. Biophys. Methods*, 28(1): 1–33. DOI: https://doi.org/10.1016/0165-022X(94)90061-2
4. **Rabe, M, Zope, H R** and **Kros, A** 2015 Interplay between lipid interaction and homo-coiling of membrane-tethered coiled-coil peptides. *Langmuir*, 31(36): 9953–9964. DOI: https://doi.org/10.1021/acs.langmuir.5b02094
5. **Niu, F, Rabe, M, Nayak, S** and **Erbe, A** 2018 Vibrational spectroscopic study of pH dependent solvation at a Ge(100)-water interface during an electrode potential triggered surface termination transition. *J. Chem. Phys.,* 148(22): 222824. DOI: https://doi.org/10.1063/1.5018796
6. **Tecklenburg, S** 2019 *Defect Formation and Evolution in Zinc Oxide: from Semiconductors to Corrosion.* Doctoral thesis, Department of Chemistry and Biochemistry, Ruhr-Universität Bochum, Bochum, Germany.
7. **Tauler, R, Izquierdo-Ridorsa, A** and **Casassas, E** 1993 Simultaneous analysis of several spectroscopic titrations with self-modelling curve resolution. *Chemom. Intell. Lab. Syst.,* 18(3): 293–300. DOI: https://doi.org/10.1016/0169-7439(93)85006-3
8. **Tauler, R, Smilde, A** and **Kowalski, B** 1995 Selectivity, local rank, three-way data analysis and ambiguity in multivariate curve resolution. *J. Chemom.,* 9(1): 31–58. DOI: https://doi.org/10.1002/cem.1180090105

How to cite this article: Rabe, M 2020 Spectram: A MATLAB® and GNU Octave Toolbox for Transition Model Guided Deconvolution of Dynamic Spectroscopic Data. *Journal of Open Research Software,* 8: 13. DOI: https://doi.org/10.5334/jors.323

Submitted: 07 February 2020 **Accepted:** 12 May 2020 **Published:** 09 June 2020

Copyright: © 2020 The Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International License (CC-BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited. See http://creativecommons.org/licenses/by/4.0/.

]u[*Journal of Open Research Software* is a peer-reviewed open access journal published by Ubiquity Press.

OPEN ACCESS