Glotaran

a tool for interactive global and target analysis of time-resolved spectroscopy and microscopy data



J. J. Snellenburg

Glotaran

a tool for interactive global and target analysis of time-resolved spectroscopy and microscopy data

Joris-Joost Snellenburg

Master's Thesis

for the completion of the custom master track Computational Biophysics,

a derivative of the master track Physical Sciences



Department of Physics and Astronomy

Faculty of Exact Sciences

VU University Amsterdam

De Boelelaan 1081HV

Amsterdam

THE NETHERLANDS

November 2010

This thesis was reviewed by:

1 Ivo H. M. van Stokkum

2 Erwin J. G. Peterman

Funding:

Part of the work carried out for this thesis was funded by the NWO Computational Science project 635.000.014.

Copyright:

© The Author 2010

Open Access
This thesis is distributed under the terms of the Creative Commons Attribution Noncommercial License which permits any noncommercial use, distribution, and reproduction in any medium, provided the original author(s) and source are credited.

Abstract

Biophysics is a branch of science that combines state-of-the-art measurement techniques from the fields of spectroscopy and microscopy with computational models to try to understand the mechanics underlying the behavior of complex biological systems. Because of the complexity of the physical process underlying these experiments, specialized software is required for the data-analysis of the experiments. Global and target analysis is a well-established technique to derive complex models from time-resolved spectroscopy and microscopy data. The open-source and cross-platform application Glotaran was developed to facilitate simplified global and target data-analysis of spectroscopic and microscopy data. Glotaran is shown to use global and target analysis techniques and combines this with interactive data-exploration and interactive visualization of data-analysis results. Some theoretical background on global and target analysis is contained within this work. The features of the application are documented and the usage of the application is demonstrated by means of a case-study on experimental data and a screencast with transcription. Finally, being an open-source project, the source of the application, as well as the necessary documentation is provided to further the development of the application by other scientists in the field. Some ideas for future work are given.

Keywords

Glotaran, Global and Target Analysis, spectroscopy data-analysis, microscopy data-analysis, Biophysics, data-analysis software, Problem Solving Environment (PSE), TIM, TIMP.

Preface

Glotaran is a software application that has been under development at the Vrije Universiteit Amsterdam, as part of a NWO funded research project. The goal of this project was to develop an open-source, cross-platform, interactive problem solving environment for hierarchical modeling of time-resolved spectra. The first step towards this goal was the development of the R-package TIMP by Dr. Katharine Mullen. R is a programming language and software environment for statistical computing and graphics and TIMP was designed to fully exploit the functionality available within this environment. The goal of TIMP was to realize an open-source implementation of the application TIM, an in-house global and target analysis software tool developed and maintained by Dr. Ivo van Stokkum, also the initiator of the project. The development of Glotaran was started to complement the functionality provided by TIMP and address its limitations, especially with respect to the graphical user interface, interactivity and ease-of-use. As such TIMP is still used as the primary computational back-end for Glotaran, but data-exploration, model development and analysis inspection occurs within Glotaran. The ultimate goal for Glotaran is to bring down the learning curve for global and target data-analysis of spectroscopic and microscopic data to a level where any biophysicists can feel comfortable exploring and fitting their own data.

The purpose of this thesis is to provide an introduction in the usage of Glotaran for global and target analysis of time-resolved spectroscopic data and the understanding thereof. In addition it serves as a starting point for future development of Glotaran (and the computational back-end TIMP). The following is a brief overview of the contents of the Chapters of this thesis:

- Chapter 1 serves as an introduction to the field of molecular biophysics and the aspects of global and target analysis therein, for which the software application Glotaran has been developed. In addition an overview of earlier work is given and the motivation for the development of Glotaran and its requirements are laid out.
- Chapter 2 provides the theoretical framework required to understand the process of global and target data analysis.
- Chapter 3 concerns the application itself. Discussed are: application architecture, design choices and the development process. In addition the various features of the application are documented and their usage is explained.
- Chapter 4 is a case study on the usage of Glotaran with real experimental data that has been analyzed with Glotaran. It serves as a starting guide for new users to Glotaran and the process of global and target analysis in general.
- Chapter 5 describes the conclusion of this project and discusses future work.

Contents

1 Introduction....9
1.1 Experiments in biophysics....9
1.1.1 The loop of hypothesis, experiment and data-analysis....9
1.1.2 Data-analysis in biophysics....10
1.2 Existing global data-analysis software....11
1.2.1 TIM....11
1.2.2 TIMP....12
1.2.3 Other existing software and algorithms....14
1.3 Requirements....14
1.4 Goal, purpose and aim....15
2 Theoretical background....16
2.1 Experimental process....16
2.1.1 Measurement types....16
2.1.2 Noise characteristics and resolution....17
2.1.2.1 Stochastic processes....17
2.1.2.2 Resolution....17
2.2 Mathematical Modeling....18
2.2.1 Model assumptions....18
2.2.1.1 Homogeneity....18
2.2.1.2 Separability....18
2.2.2 Model types....19
2.2.2.1 Kinetic models....20
2.2.3 Parameter estimation....20
2.3 Global analysis....21
2.3.1 First steps of global analysis....22
2.3.1.1 Singular value decomposition....22
2.3.1.2 Independent decays....24
2.3.1.3 Unbranched, unidirectional model....25
2.3.2 Target analysis....25
2.3.2.1 Compartmental models....25
2.4 Summary....28
3 Introducing Glotaran....30
3.1 The application....30
3.1.1 Introduction....30
3.1.2 Application architecture....30
3.1.2.1 NetBeans RCP....31
3.1.2.2 Glotaran – Core Modules....31
3.1.2.3 Glotaran – Supporting Modules....32
3.1.2.4 External libraries....32
3.2 Graphical User Interface....32
3.2.1 Main window....32
3.2.1.1 Project explorer....33
3.2.2 Data exploration windows....34
3.2.2.1 Time-resolved spectroscopy data....34
3.2.2.2 Time-resolved microscopy data....35
3.2.2.3 Other data types....36
3.2.3 Analysis scheme editor....36

3.2.3.1 Model specification window....37
3.2.3.2 Datasets specification window....38
3.2.3.3 Output specification window....38
3.2.3.4 Simulation specification window....38
3.2.4 Results presentation windows....39
3.2.4.1 Time-resolved spectroscopy analysis results....39
3.2.4.2 Time-resolved microscopy analysis results....41
3.3 Screencast....42
4 Case study....43
4.1 Introduction....43
4.2 Exploring the data....43
4.2.1 Data pre-processing....44
4.2.1.1 Window selection....44
4.3 Interactive modeling....44
4.3.1 Initial model....44
4.3.2 Extending the model....46
4.3.2.1 Adding dispersion....46
4.3.2.2 Coherent artifact....48
4.3.2.3 Weighing the data....49
4.3.2.4 A comparison....49
4.3.2.5 Target Analysis....50
4.3.2.6 Further analysis....51
5 Conclusions....52
5.1 Current usage....52
6 Future work....53
6.1 Graphical user interface....53
6.1.1 User assisted dispersion estimation....53
6.1.2 Visual compartmental modeling....53
6.1.3 Integration with iReport....53
6.2 Computational back-end....54
6.2.1 TIMP....54
6.2.2 Interchangeability with other computational back end....54
6.3 Application core....54
6.3.1 Integration with databases....54
6.3.2 Systems biology....55
7 Appendix A: The NetBeans platform architecture....56
8 Appendix B: Contributing to Glotaran....57
9 Appendix C: JFreeChart library....59
10 Appendix D: Glotaran screencast....61
Acknowledgments....66
Bibliography....67

1 Introduction

This chapter serves as an introduction to the concepts of global and target data-analysis in the context of biophysics. It motivates the development of Glotaran, the primary goal of this project. Section 1.1 functions as a brief introduction to the field of biophysics and science in general. Here the scientific process or loop of hypothesis, experiment and data-analysis is introduced. Also the nature of the data obtained in experiment in the field of biophysics is explained, and thus the need for advanced data-analysis software like Glotaran becomes clear. In section 1.2 the predecessors to Glotaran as well as other similar software are discussed in order to identify the general requirements for data-analysis software used in the field of biophysics. Section 1.3 then summarizes this list of requirements, and the outline for the implementation thereof in Glotaran is sketched. Finally section 1.4 discusses the goal and aims of this project and the purpose of this thesis.

1.1 Experiments in biophysics

Biophysics is a field in science that applies the principles of physics and chemistry and the methods of mathematical analysis and computer modeling to biological systems, with the ultimate goal of understanding at a fundamental level the structure, dynamics, interactions, and ultimately the function of biological systems. More specifically, biophysics seeks to explain biological function in terms of the physical properties of specific molecules and their interactions with their environment.

For the study of biological systems at the molecular level, a wide variety of experimental and simulation techniques are available. These techniques vary greatly in type (chemical, mechanical, electromagnetic, or other), complexity and applicability but have one common goal: the acquisition of as much (useful) data as possible.

However the usefulness of simulated or experimental data is often determined by the quality of the process of data-analysis that follows; there is only so much information that can be obtained from looking at the raw data itself. It is this process of data-analysis – and the tools required for it - that is the focus of this chapter.

1.1.1 The loop of hypothesis, experiment and data-analysis

In biophysics advanced optical spectroscopy and microscopy are commonly used techniques to obtain data in measurements on biological systems. By studying these measurements it is possible to obtain fundamental knowledge on the relation between the structure, function and dynamics of proteins and other biological molecules, both in single proteins and in supramolecular aggregates. This sounds relatively straight forward but it is actually a complicated multi-step process which is summarized in Figure 1.

The first step in any experiment (or simulation study) is to design the experiment (or

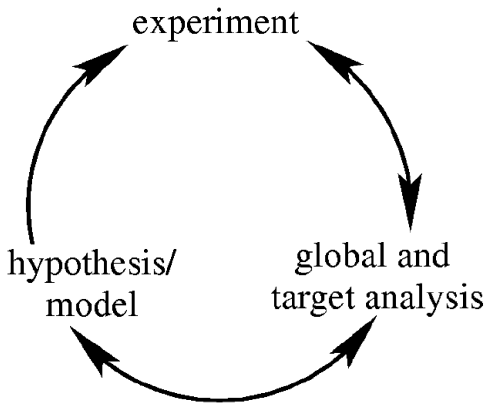


Figure 1: Relation between experiment, analysis and model

simulation) based on a hypothesis based on existing knowledge and assumptions about the system under study. The next step is to execute the experiment and acquire data.

The second step in the loop is to perform data-analysis making use of the technique of global and target analysis. Here global refers to a simultaneous analysis of all measurements, whereas target refers to the applicability of a particular model or scheme on the data. The importance of this particular technique and the demand on the analysis process will become clear throughout this thesis. The arrow in the figure leading back to experiment indicates that it is possible the analysis pinpoints some systematic experimental errors, and inspires improved experimentation.

Based on the results from the analysis, the final step is to accept or reject the initial hypothesis or define a new model that explains the data and conforms to knowledge of the physical system under investigation. Naturally any accepted, rejected or new hypothesis can be ground for a new experiment, thus closing the loop.

This loop of hypothesis, experiment and analysis is the road to discovery followed by any scientist looking for results, and every step in itself is a completed and often laborious affair. However, in this thesis the focus lies primarily on the step of data-analysis.

1.1.2 Data-analysis in biophysics

Science has come a long way since the first observation of fluorescence by Sir John Frederick William Herschel in 1845, who required little more than a bit of the widely encountered fluophore quinine dissolved in a glass of water and a curious and intelligent mind [1]. Nowadays in biophysics, data are often acquired using advanced techniques in optical spectroscopy and microscopy generating large amounts of complex data. The nature of these techniques and the samples to which they are applied to, means that the resulting data is often of such complexity that standard fitting procedures are not enough to resolve the information that lies encrypted within the measurements. Only by means of global and target analysis is it possible to

attempt to extract all this information from the analysis. And although software for standard fitting and data-analysis is plentiful, software for global and target analysis is scarce, especially the kind of software that can handle the acquired spectroscopy and microscopy data. The software that does exist is often limited in functionality, interactivity, usability, availability or a combination of these factors.

1.2 Existing global data-analysis software

In this section some of the tools used for global and target analysis are discussed, in particular the applications TIM and TIMP, both predecessors to Glotaran and also developed at the department of Computational Biophysics of the Vrije Universiteit Amsterdam. In addition a selection of other software applications and algorithms are discussed, provided some references were available. By carefully looking at other software, particularly their functionality and limitations, the requirements for Glotaran can be outlined.

1.2.1 TIM

TIM was developed as a prototype Problem Solving Environment (PSE) for problems in interactive modeling of multiway data [2]. Here multiway indicates that a property like emission or absorption is measured systematically as a function of independent experimental variables like time, detection or excitation wavelength, pH or temperature. The underlying concept can be illustrated by the flow chart depicted in Figure 2.

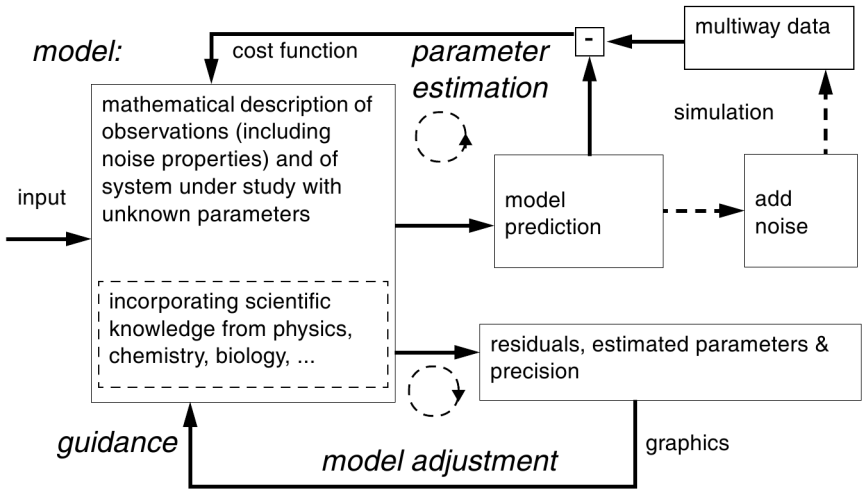


Figure 2 Flow chart of prototype PSE. Using the multiway data and the specified model the parameters are estimated in the upper loop.

At the highest level of operation two loops can be distinguished. In the top loop it is depicted how a model can be used to describe multiway data. This is known as

parameter estimation, or model fitting, or regression. In the bottom loop it is indicated that when there are many candidate models available, they should be tested, and using scientific knowledge a choice should be made. It is this model adjustment loop for which the PSE was developed. In the development phase of TIM the requirements for a good PSE were identified as follows:

- a language natural for the problem class should be used to specify the building blocks of the model,
- appropriate graphical output summarizing the results of the analysis should be produced in order to facilitate interactive data analysis by the human in the loop,
- the analysis processes should include a step where multiple models can be tested and discarded or accepted quickly based on the estimated parameters and residuals for each model,
- for the model specification a graphical user interface (GUI) should be developed, which supports collaborative research and enables distributed interactive modeling, where an expert in modeling and parameter estimation can analyze the data and an experimental scientist can contribute to the interactive modeling by discarding unrealistic models and suggesting model improvements.

The development on the PSE TIM started in 1990 and has continued to evolve ever since. In its current state most of the initially posed requirements have been met, save for the interactive GUI environment. The results from global and target data-analysis using TIM have found their way into many publications in peer-reviewed journals such as Nature, PNAS and BBA [3-5] emphasizing the importance of detailed global and target analysis of spectroscopic data.

TIM is written in the programming languages Fortran and C and makes use of the IMSL Numerical Libraries [6] for fast numerical computation, and the in-house developed USEPLA library for plotting. It currently runs under the Solaris operating system. A Linux port is in progress. The TIM kernel has already been ported to Linux. However the graphical backend still needs to be ported. TIM is not open source, nor publicly available, however access to the software is available through collaboration with its lead developer, Dr. I. H. M. van Stokkum[1].

1.2.2 TIMP

In an attempt to create an open-source, platform-independent (or cross-platform) software package that would implement most of the functionality available in TIM the development on TIMP started in 2003 as part of a NWO[2] funded project [7]

1 Dr. I.H.M. Van Stokkum, Department of Physics and Astronomy - Faculty of Sciences - Vrije Universiteit Amsterdam. **Address**: De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands. **Telephone**:+31(0)205987868. **Fax**:+31(0)205987999. **E-mail**: ivo@few.vu.nl

2 NWO: The Netherlands Organisation for Scientific Research that funds thousands of top researchers at universities and institutes and steers the course of Dutch science by means of

[7]. TIMP was designed to be a PSE for the fitting of separable nonlinear models under least-squares criteria to experimental data and has been implemented in the form of a R-package [8]. R is a language and environment for statistical computing and described in the R-manual [9]. TIMP itself is well documented in the dissertation of Dr. Katherin Mullen who created the software [10]. The implementation in the form of an R-package offers several advantages:

- R is open-source; the inner workings of R and its mathematical subroutines can easily be verified and validated.
- The R project provides a package system that allows individual packages like TIMP to be easily distributed, facilitating collaboration and the reproduction of research results.
- R has established itself as the lingua franca of statistical computing, and as such a wide variety of useful mathematical subroutines have been implemented.
- R supports object-oriented programming, allowing TIMP to be designed with extension to new types of data in mind. The object-oriented structure allows much of the core functionality of the package to be re-used when writing extensions to handle new types of data and models.

TIMP was designed to aid the process of model specification, optimization and validation; several case studies in [10] illustrate the usage of TIMP for these purposes. It should also be noted that TIMP has successfully been used in the data-analysis of several recently published papers [11-13] showcasing its capabilities and maturity.

The majority of the TIMP package is written in the programming language R and parts of it are written in Fortran and C. Every aspect of TIMP is open source, from the R environment under which it is running, to the actual code itself. Just as TIM has a language natural for the problem class to specify the building blocks of the model, so does TIMP. Extensive documentation on the model specification and plotting of results as well as sample scripts are available from the official project website [14]
[14]. For the production of graphics it uses the capabilities that come with the R environment. R is quite flexible in terms of the possibility to output a wide variety of plots, it is however lacking in the aspect of interactivity. Plots generated are typically static; zooming in to a higher level of detail is not possible and feedback based on interactivity with the plot is not possible. It is for this reason that the development of a graphical user interface for TIMP was started, keeping the powerful computational back end of TIMP and R but replacing the graphical front end with an implementation in Java [15]. This eventually became Glotaran, is discussed in Chapter 3.

1.2.3 Other existing software and algorithms

There is other data-analysis software available – some of it at least remotely similar in functionality to TIM or TIMP. A list of software and mathematical subroutines is

subsidies and research programmes.

given in Table 1. Although this table provides a fairly comprehensive list of data-analysis software for time-resolved spectroscopic data in existence, it should by no means be considered complete. Various individuals, companies, universities or other institutes have their own in-house developed analysis software but have not publicly released it [Holzwarth (Mülheim) , Boens (Leuven), Géza Groma (Biological Research Center (BRC) of the Hungarian Academy of Sciences)].

Name	**Developer / Company**
Prok-K / Prok-K II [16]	Applied Photophysics
FFS Data Processor; TRFD Fitting Software; TRFA Data Processor; BIFL Data Analyser [17]	Scientific Software Technologies Center, Department of Systems Analysis, Belarusian State University
DAS6, Decay Analysis Software [18]	Horiba Scientific
Asufit [19]	Evaldas Katilius, Arizona State University
The VARiable PROjection (VARPRO) method [20], source [21]	Golub & LeVeque

Table 1: Overview of data analysis tools in spectroscopic data analysis

It should be noted that none of the software packages listed in Table 1 meet all of the requirements as set out in the introduction of this chapter: open-source, platform-independent, capable of global and target analysis of spectroscopic data and support for interactive model exploration and visualization of analysis results. Glotaran was designed to fill exactly this void.

1.3 Requirements

The basic requirements for Glotaran are determined by the functionality and limitation of its predecessors. Ideally Glotaran would implement all of the functionality of its predecessors while addressing all of their limitation and extend the existing functionality with new and useful features. Practically the functionality is partially limited by the computational back-end that is used (in this case TIMP); therefore as new functionality is implemented in Glotaran the same functionality also needs to be implemented in the computational back-end. Some additional requirements were formulated based on practical requirements regarding collaborative development and feedback on initials version of the program. Summarized below are the initial requirements for Glotaran.

- The application must be open-source and platform-independent.
- The application must facilitate interactive data-exploration
- A simplified model specification is needed; a model specification that is intuitive to the average users and which uses a graphical user interface rather than a text based (syntax sensitive) model.

- The application must facilitate interactive results inspection (i.e.: zooming in on results, comparing data with model-based fits, comparing results from different models, etc.)
- The application should implement (most of) the functionality already provided by TIM and TIMP. Partially this is done by using TIMP as a computational engine for the calculations required for analysis.
- The application should be built on top of a Rich Client Platform to minimize the time spent on programming the basics (like the window environment, file input/output, editor capabilities, etc) and spend as much time as possible on the implementation of features. This requirement also adds much desired modularity which allows for easy extensibility by the development of new modules – without needing to rewrite large parts of the application.
- Collaborative development should be made as painless as possible by using a cross-platform IDE with support for version-controlled source code repositories.

1.4 Goal, purpose and aim

The goal of the project has been the development of an open-source, platform-independent global and target analysis tool for the analysis of spectroscopy and microscopy data in experiments in biophysics and other related disciplines. The purpose of this master thesis is the documentation of this process and the demonstration of the usage of the resulting software application Glotaran.
The aim of this thesis is to:

- Introduce the concepts of global and target analysis in the context of biophysics.
- Provide some mathematical background on model description, parameter estimation and global and target analysis.
- Document the features of Glotaran providing a basic 'user manual' for starting users of the application.
- Demonstrate the process of data-analysis within Glotaran by means of a case-study with real experimental data and a screencast documenting part of this process.
- Provide documentation for furthering the development of Glotaran by other researchers in the field.

2 Theoretical background

Time-resolved spectroscopy or microscopy is the study of dynamic processes in (bio) molecular systems by means of spectroscopic or microscopic techniques. Generally it involves the use of some source of illumination to change the properties of the system and then observing the effect of these changes by detecting fluorescence or absorption over time as a function of one or more variables. The result is some form of multiway data; typically 2-way data with intensity measured as a function of the independent experimental spectral variable wavelength λ and the independent experimental variable time t after excitation. Other experimental variables are temperature, polarization dependence (anisotropy), acidity (pH or pD), excitation wavelength or quencher concentration. In the case of microscopy techniques the spatial coordinate is typically also an experimental variable. The example of 2-way data with time and wavelength as the independent variables is used throughout this chapter.

In this chapter the theoretical framework for modeling these kinds of data is established. In section 2.1 the experimental process and the complications that arise in the measurement process and resulting data-types are explained. In section 2.2 the mathematical model to analyze this type of data is introduced. Finally in section 2.3 some aspects of global and target analysis are discussed.

2.1 Experimental process

To study dynamic processes in (bio)molecular systems various spectroscopic methods are used to collect information about the structure of the molecules and their concentrations. The measurement of the concentrations of different components as a function of time is necessary to study the dynamics of chemical reactions and of structural changes. However, with a mixture of n_{comp} components whose spectra overlap, concentrations can only be measured indirectly by observing the total spectrum at several time instants, for example using time-resolved absorption or fluorescence spectroscopy. The various types of measurement processes as well as the characteristics of the resulting data, including resolution and noise characteristics are discussed here.

2.1.1 Measurement types

In time-resolved spectroscopy, fluorescence or absorption is measured as a function of time at many wavelengths constituting a time-resolved spectrum. There are a few ways in which such a spectrum can be acquired:

- Measurements can be done simultaneously at a great number of wavelengths and at a certain time delay with respect to the exciting pulse. This is called a time-gated spectrum. A collection of such time-gated spectra at different time delays constitutes a time-resolved spectrum.
- Alternatively, at a particular wavelength a decay trace is measured as a function of time with respect to the exciting pulse. Again a collection of such

decay traces measured at different wavelengths constitutes a time-resolved spectrum.

- Detection of decay traces at a great number of wavelengths simultaneously, providing high resolution in both dimensions, is possible with a (synchroscan) streak camera in combination with a spectrograph [22], [23].

These measurement processes are used in combination with techniques like Time-resolved fluorescence spectroscopy [24], Transient-absorption spectroscopy [25] and Time-resolved infrared spectroscopy [3].

Another type of commonly encountered type of data is time-resolved microscopy data, which has a different underlying measurement process:

- A decay trace is measured at a certain wavelength after excitation but at a multitude of spatial coordinates at once.

This measurement process is used for instance in Fluorescence Lifetime Imaging Microscopy (FLIM) [26].

2.1.2 Noise characteristics and resolution

The relevance of differentiating between different types of measurements processes lies in the characteristics of the data. Different types of measurement may require different kinds of prepossessing (e.q. background subtraction) and can differ in their noise statistics. The resolution of the measurements are determined by a number of instrumental characteristics and by the stochastic processes underlying the measurements.

2.1.2.1 Stochastic processes

The underlying stochastic processes are determined by photon properties, especially in fluorescence measurements, but also ambient influences and instrumental properties. For instance, single photon timing fluorescence measurements constitutes a counting process which is Poisson distributed [27] where the variance is equal to the mean and all observations are independent. In contrast, the stochastic properties of the other types of measurements are much more uncertain. Typically the observations are assumed to be independent and contain additive normally distributed noise [4]. This means that the noise in the data is assumed to be normally distributed with a constant variance. This works for most cases and as such will be the assumption throughout this thesis.

2.1.2.2 Resolution

In time-resolved measurements, the time resolution is limited by both the width – in time - of the exciting laser pulse and the width of the detector response. The convolution of the shape of the exciting laser pulse and the detector response is termed the Instrument Response Function (IRF). This is an important aspect of modeling when analyzing ultrafast measurements (typically faster than a nanosecond) and should always be accounted for in these cases. For measurements performed on a much longer timescale (microseconds or even seconds) a nanosecond or picosecond

IRF can typically be assumed to be a delta function and does not require extra modeling.

Spectral resolution is determined by the characteristic of the spectrograph used; also in other types of measurements (spatially-resolved, pH-resolved, …) the resolution is determined by the detector or spectrometer used.

2.2 Mathematical Modeling

Time resolved spectroscopy or microscopy deals with experimental processes that extract information on the dynamics of a system, for instance a molecule, a protein or even a membrane complex. Different experimental processes result in different kinds of information but essential in time-resolved spectroscopy or microscopy is monitoring the relative concentrations of components that together constitute the measured time-resolved spectra. The study of these concentrations over time is necessary to elucidate the dynamics of chemical reactions or conformational changes that affect the working of biological systems. However, with a mixture of n_{comp} components whose spectra overlap concentrations can only be measured indirectly by observing the total spectrum at several time instants using time-resolved absorption or fluorescence spectroscopy.

2.2.1 Model assumptions

Given a set of experimental (or simulated) data, the following step is to describe the data in terms of a mathematical model. However before such a model can be defined, some assumption have to be made.

2.2.1.1 Homogeneity

A common assumption is that the system under study is homogenous, which means that a discrete set of parameters can be used to describe the data, rather than being inhomogeneous, which requires distributed parameters. This simplifies the analysis process and speeds up the computation. Glotaran currently only supports the fitting of a discrete parameter set.

2.2.1.2 Separability

The next assumption is separability. It is assumed that the spectroscopic properties of a mixture of components are a superposition of the spectroscopic properties of the individual components, weighted by their respective concentrations. [4], [28] In absorption spectroscopy this linear relationship between the absorbance (optical density (OD)) and the concentration of an absorber relates to Beer-Lambert's law. This means that the (noise-free) measured time-resolved spectrum ψ can be represented as a superposition of the contribution of n_{comp} different components:

$$\psi(t,\lambda)=\sum_{l=1}^{n_{comp}} c_l(t)\,\epsilon_l(\lambda) \tag{1}$$

Where $c_l(t)$ and $\varepsilon_l(\lambda)$ denote, respectively, the concentration and spectrum of component *l*. Or, in matrix notation:

$$\Psi = C\,E^T \tag{2}$$

Here Ψ represents the *m x n* data matrix denoting the time resolved spectrum, measured at n time instants and m wavelengths. Each column of the *m x* n_{comp} matrix **C** represents a concentration profile c_l of component *l* as function of the independent variable *t*. Likewise, each column of the *n x* n_{comp} matrix **E** represent a spectrum ε_l of component *l* as function of a second independent variable like λ. The number of components that can effectively be studied this is typically between one and ten, whereas the number of different wavelengths and the number of different time instants can vary from n_{comp} to thousands. It should be noted here that the quantity that can be estimated is the product $c_l\varepsilon_l$ which in itself is insufficient for the determination of the absolute values of c_l and ε_l. This aspect of modeling will be further discussed in §2.2.3.

The goal now is the recovery of the entries of **C** or **E** in terms of physically significant parameters (e.g. describing the decay rate of a component, or the location of the maximum of the spectrum of a component). In order to achieve this, first a model-based description for either **C** (kinetic model), **E** (spectral model) or both (spectrotemporal model) is needed, from which the relevant parameters can be estimated. These model-based descriptions are discussed in the following section.

2.2.2 Model types

As the measured time-resolved spectra is considered to be a superposition of the contribution from n_{comp} different components, it should be – at least theoretically - possible to model the spectroscopic and kinetic properties describing these components. However, finding the right model can be difficult; in practice various problems can arise. First, the number of components in the system may be unknown. Second, in general neither the concentration profile $c_l(t)$ or the spectra $\varepsilon_l(\lambda)$ are known and because only the product $c_l\varepsilon_l$ can be estimated, this poses a problem. Third, experimental properties like noise and finite (temporal, spectral, spatial, …) resolution limit the accuracy of any model fitted to the data, and may even obscure components that could otherwise be modeled.
However, often the experimentalist has some a priori knowledge that can aid the modeling process. For instance, from literature or earlier experiments he or she might know which shapes of concentration profiles or spectra are realistic and can impose so called constraints on the model.
In global and target analysis three different types of models exists:

- A kinetic model; in this case a parametrized model for the concentrations of the components ($c_l(t)$) is used.

- A spectral model; in this case a parametrized model for the spectra of the components ($\varepsilon_l(\lambda)$) is used. A spectral model is more phenomenological than a kinetic model and generally requires more parameters than kinetic models.
- A spectrotemporal model; in this case knowledge about kinetic and spectral properties of the components is combined.

In the initial release of Glotaran (version 1.0.0) as described in this thesis only the framework for kinetic modeling is in place although the application can be extended to support spectral and spectrotemporal models as well. In this chapter the focus lies on the description of the kinetic model.

2.2.2.1 Kinetic models

A kinetic model describes the concentrations of components in a measurement over time; i.e. it directly models the dynamics of the sample. Typically, measurements are described by first order kinetics, meaning that the concentrations are described by linear first order differential equations [4]. The solution for a system of linear differential equations is given by a sum of exponentially decaying functions convolved with the function describing the initial condition, in experimental physics termed the Instrument Response Function. The basic form of a kinetic model describing the measured data Ψ as a function of time and the independent spectral variable λ can then be represented as

$$\psi(t,\lambda)=\sum_{l=1}^{n_{comp}}(\exp(-k_l t)\circ i(t))\epsilon_l(\lambda) \tag{3}$$

Where k_l is the rate of exponential decay of component l and the symbol '$\circ$' indicates the convolution with the IRF $i(t)$. The amplitudes of the concentration profiles are described by the spectra $\varepsilon_l(\lambda)$. In matrix notation, equation 3 can be formulated as follows:

$$\Psi = C(\theta_l)\,E^T \tag{4}$$

Here $\boldsymbol{C}(\theta_l)$ is the matrix of concentration profiles as determined by the vector of non-linear parameters θ_l. $\mathbf{E}$ is the matrix of spectral amplitudes for which the entries are estimated as so called conditionally linear parameters. By treating the entries of $\mathbf{E}$ as conditionally linear parameters it is possible to recover the spectral shapes without actually specifying the parameters for the underlying spectral model [29]. The problem of parameter estimation is central to the process of global analysis of time-resolved data and is discussed in more detail in the following section.

2.2.3 Parameter estimation

Given a parametric model-based description of the data as given by equation 4 the next problem is estimating these parameters in terms of physically significant values. There are two kinds of parameters that are estimated: the intrinsically non-linear parameters (the parameter vector θ_l) and the conditionally linear parameters ($\mathbf{E}^T$) in (4). The non-linear parameters are those that directly make up the model specification. For these parameters initial starting values need to be provided, as they cannot be

calculated in any way. The conditionally linear parameters *can* be solved for, conditionally on the unknown non-linear parameters. In the case of a kinetic model this means optimizing for the vector of parameters that determine the lifetime of each component, while at the same time finding the optimum for the amplitudes of the spectra that together minimize the residuals. Mathematically this can be represented as

$$min\left\|(\psi - \sum_{l=1}^{n_{comp}} (c(\theta_l))\epsilon_l)\right\|^2 \tag{5}$$

Where ψ is the m x n data matrix, c is a function of the nonlinear parameter vector θ_l that outputs a vector of length m, ε_l is a vector of length n with the conditionally linear parameters that are estimated from the parameter vector θ_l, and finally ||.|| is the matrix 2-norm. In matrix notation this can be formulated as

$$min\|\Psi - C(\theta_l)\,E^T\| \tag{6}$$

Here Ψ is the m x n data matrix, C is the matrix of concentrations that depend on the estimated non-linear parameter vector θ_l and E^T is the matrix composed of the vectors of conditionally linear parameters.

The problem formulated in equations 5 and 6 is solved by the technique called variable projection. Variable projection is a technique introduced by Golub and Pereyra, for reducing the separable estimation problem to one of minimizing a sum of squares in the nonlinear parameters only [30]. In the computational back-end for Glotaran, TIMP, a partitioned version of the variable projection algorithm has been implemented that allows estimating the free parameters of the model for matrix data without requiring large memory resources [31].

2.3 Global analysis

It has become clear that given a set of experimental data (§2.1), with its origin in some time-resolved experimental process (§2.1.1), assuming that the system under study is homogenous (§2.2.1.1) and separable (§2.2.1.2) and given a proper parametrization of the data in the form of a parametric model (§2.2.2) with intrinsically non-linear and conditionally linear parameters (§2.2.3), it is possible to apply the technique of variable projection in order to estimate the values of the parameters underlying the measurements in terms of physically significant parameters that describe the (biological) system under study. The use of a unified separable nonlinear model to describe all measurements collected over multiple independent variables, and possibly over the course of many experiments, is what is referred to as global analysis [8]. This section explains the process of globally fitting data with a few simple models. In addition a more complex class of global analysis model is discussed called target analysis, in which compartmental models are used.

2.3.1 First steps of global analysis

As has become clear from §2.2 the first part in global analysis using kinetic models is to fit the data with a sufficient number of exponential decays and their amplitudes (i.e. the lifetimes of the components). The number of components can be estimated based on a priori knowledge but more often than not is the result of evaluating the results of a singular value decomposition on the data matrix as explained in §2.3.1.1. The starting values for the amplitudes of the components are often based on a rough but educated guess based on the understanding of the timescales of the processes under study.

2.3.1.1 Singular value decomposition

The singular value decomposition (SVD) is an important mathematical tool in global analysis. In Glotaran it is used as a data-inspection tool, to estimate the number of linear independent components in the data and to check for potential outliers or spot particularly noisy regions in the data. In addition it is used as a results-inspection method, to evaluate the quality of the fit and determine if perhaps additional components need to be modeled. Here this process is explained with the aid of a practical example.

The SVD theorem states that any *m x n* matrix has a singular value decomposition of the form:

$$\Psi_{m\times n} = U_{m\times m} S_{m\times n} W^{T}_{n\times n} \tag{7}$$

Where U and W are orthogonal matrices, containing respectively the left and the right singular vectors and S is a matrix which is zero expect for its diagonal, which holds the singular values or eigenvalues. The singular values give some indication of the number of components in the data. With n_{comp} and noise-free data there are exactly n_{comp} significant singular values (different from 0): $s_1 \geq s_2 \geq s_{ncomp} > s_{ncomp+1} = \ldots = 0$. In the case data with noise where the singular values are typically not equal to zero the significant singular values are those significantly different from the rest. This is best visualized in the form of a screeplot where the singular values are plotted on a logarithmic range axis, see Figure 3.

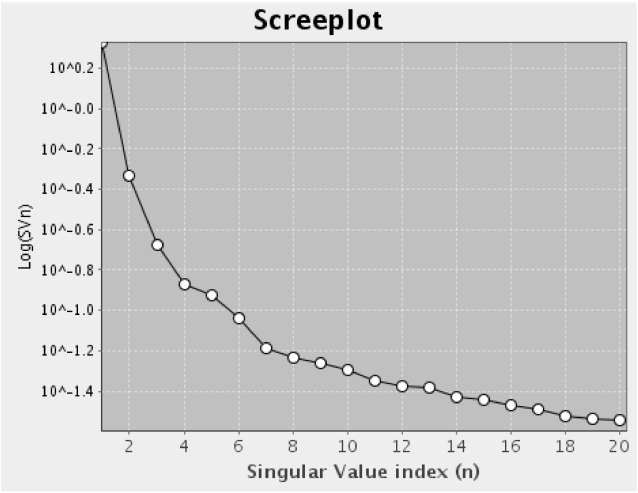


Figure 3: Screeplot of the first 20 singular values of the SVD of typical data with noise

As can be seen from this screeplot, in the case of noisy data it is not as clear cut how many components are underlying the data. In this case there seem to be at least 3 components and perhaps 4. It's often best practice to start with as little components as possible and then look at the SVD of the residual matrix for more significant values. However, it is essential to take into account the structure of the left and right singular vectors as well, as they better reflect the physical aspects of a particular component. In this case the matrix of left singular vectors represents the time-dependence of the particular components, see Figure 4.

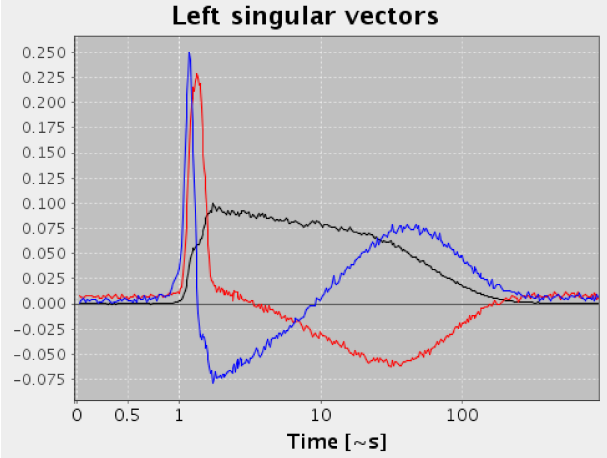


Figure 4: The first three left singular vectors representing the time dependence of components represented by the first three singular values

By visualizing only the first few components it is easier to qualitatively evaluate the individual contribution of the components to the dataset as a whole. Usually the number of visualized traces is incremented by one until the structure is no longer

clearly distinct from noise. Typically this value is plus or minus one around the value estimated from the screeplot.

Inspection of the right singular vectors can also be helpful, here they represent the spectral-dependence of the particular components, see Figure 5. They can be particularly helpful in determining the quality of the data on the edges of the spectra and might give an indication if weighting of the data or even a cut-off from a particular wavelength is required.

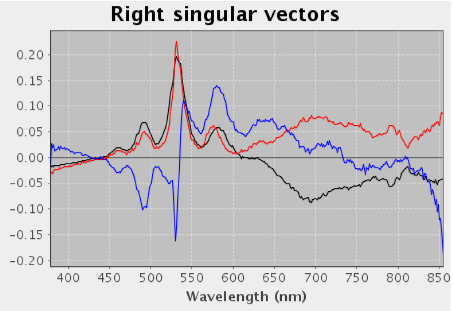


Figure 5: The first three right singular vectors representing the spectral dependence of the components represented by the first three singular values.

Apart from being a useful pre-analysis tool, the SVD can also be a powerful tool in evaluating the quality of the results. For instance, any structure in the first one or two residual vectors could suggest that the quality of the fit might be improved by adding another component to the analysis. The quality of the fit can by quantitatively represented by the sum of squares of errors. This value can directly be calculated from the residual matrix. The SVD of the residual matrix Z is

$$Z = \Psi - CE^T = U_Z S_Z W_Z^T \tag{8}$$

Now because U and W are unitary matrices the sum of squares of the residual trace (Z^TZ) is equal to the sum of squares of the singular values, which can be represented by the value Λ_Z as

$$\Lambda_Z = \sum_{i=1}^{min(n,m)} s_{Z,i}^2 \tag{9}$$

It is this number that can be used to numerically compare different models for the name dataset. The RMS value as reported by Glotaran is simply the sum of squares of the errors divided by the number of degrees of freedom.

2.3.1.2 Independent decays

As mentioned before, the first part in global analysis using a kinetic scheme is to fit the data with a sufficient number of exponential decays and their amplitudes. This is the simplest kinetic scheme where every component decays independently (or in

parallel) resulting in decay associated spectra, one for every component, the collection of which is referred to as the Decay Associated Spectra (DAS). In the case of difference absorption spectroscopy the amplitudes associated with the exponential decays are termed Decay Associated Difference Spectra (DADS).

2.3.1.3 Unbranched, unidirectional model

Apart from global analysis with independent decays, the simplest analysis uses a kinetic scheme where the components decay sequentially (most often with increasing lifetimes), this is termed the unbranched, unidirectional model ($1 \rightarrow 2 \rightarrow \ldots \rightarrow n_{comp}$). The individual components are referred to as compartments. In this scenario the first compartment gets populated by an input excitation pulse and subsequently forms the second component. The second component decays into the third compartment and so on, until finally the nth component decays to the ground state. The unbranched, unidirectional model is also termed a sequential model. The estimated spectra are called Evolution Associated Spectra (EAS) or Evolution Associated Difference Spectra (EADS) in the case of difference absorption spectroscopy; the spectra reflect the spectral evolution, e.g. the third EAS or EADS rises with the lifetime of the second component and decays with the third lifetime. In the sequential model any back-reactions are ignored on the assumption that the energy losses are large enough that the reverse reaction rates are negligible. The same way it is assumed that there are no losses in the chain $1 \rightarrow 2 \rightarrow \ldots \rightarrow n_{comp}$ to the ground state.

2.3.2 Target analysis

The combination of global analysis with testing a specific photophysical or photochemical model is called target analysis. The difference with normal global analysis is the interpretation given to the estimated spectra. The simplest target analysis still uses independent decays, or the unbranched, unidirectional model but the interpretation then is that there are physical processes underlying the data that can and should be modeled this way. For instance when the sequential model with increasing lifetimes represent the correct physicochemical picture, then the EAS correspond to the true spectra for each intermediate state or species. The EAS in this case are then referred to as the Species Associated Spectra (SAS).

In the process of global analysis of time-resolved data the simple model representing independent decays or the unbranched, unidirectional model will go a long way, especially when 'exploring' the data. Nevertheless, often at some point it becomes desirable to test a more complicated scheme that is a combination of the parallel and sequential model and involves branching, back-reactions or multiple compartment excitation. In this case a so called linear time-invariant compartmental model can be used, the next section describes the mathematics behind a compartmental model.

2.3.2.1 Compartmental models

A compartmental models is a complex kinetic schema involving branching, back-reactions or multiple compartment excitation; consequently it allows for much more

flexibility in modeling data. However, the danger lies exactly in this flexibility. One very important aspect of using compartmental models is the realization that different compartmental schemes can result in the same model output, rendering the system structurally (un)identifiable and considerable caution should be used in the interpretation of such models. Structural unidentifiability is caused by incomplete information on the system but may be resolved by combining measurements or making certain spectral assumptions. However, even a structurally identifiable model may be numerically unidentifiable [32].

In a compartmental model, transitions between compartments are described by microscopic rate constants, which constitutes the off-diagonal elements of the transfer matrix K. Each compartment is represented by a column in this K-matrix. The lifetimes of the components are the reciprocals of the eigenvalues of this matrix. The concentration of each compartment is described by a vector $c(t) = [c_1(t) \ldots c_{ncomp}(t)]^T$. Thus a linear compartmental model with n_{comp} compartments is described by a differential equation for these concentrations:

$$\frac{d}{dt}c(t) = K\,c(t) + j(t) \tag{10}$$

Where the input to the system is described by a vector $j(t) = i(t)\,[1\ x_2 \ldots x_l]^T$, where i(t) is described by the IRF and x_l represent possible extra input to compartment *l*. The solution for equation 10 is an exponential function convolved with the IRF, which can be used to calculate the concentration matrix.

What is important in the compartmental model is understanding how to specify the K-matrix, the scaling parameters and the input for each compartment. A compartmental model can quickly become quite complex and key is keeping overview. To this end it often helps to draw a diagram in which the dynamics are more easily represented, see Figure 6.

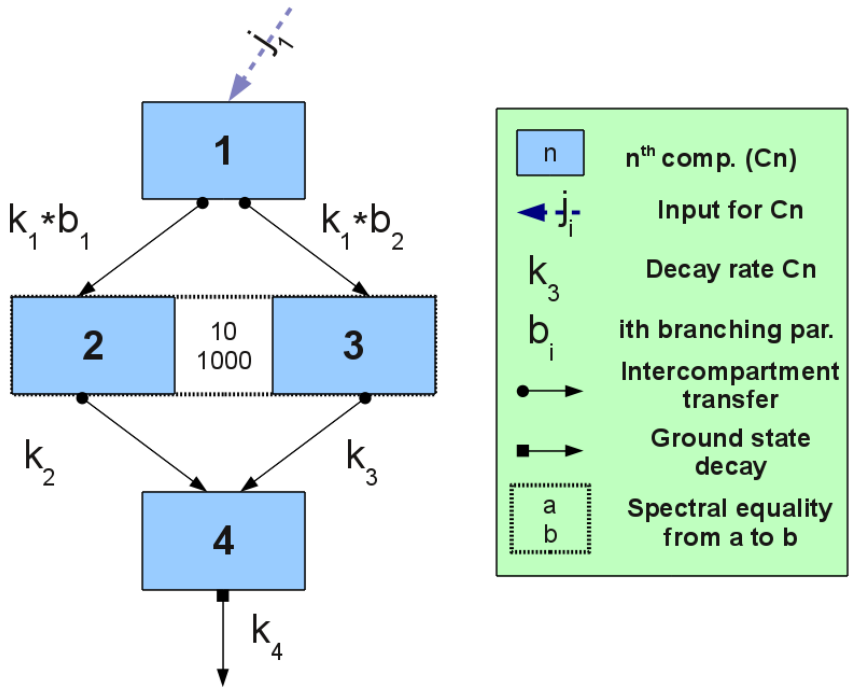


Figure 6: Example compartmental model

Ideally the way to specify a compartmental models is by drawing a diagram as above. In this compartmental model the first compartment is populated as a direct result excitation, as represented by the IRF multiplied with j_1. This compartment then evolves into the second and third compartment, with kinetic rate k_1 multiplied by branching ratios b_1 and b_2 respectively. The second and third component share the same spectrum from 10 nm to 1000 nm. Both compartment then evolve into the final compartment with rates k_2 and k_3. This compartment then finally decays to the ground state with rate k_4.

Although a diagram of a compartmental model as illustrated in Figure 6 is a very intuitive way of visualization, ultimately this needs to be translated into a language the computer understands. This is why, for the moment, a slightly more abstract representation is used as demonstrated by a screenshot of the K-matrix editor in Glotaran version 1.0 in Figure 7.

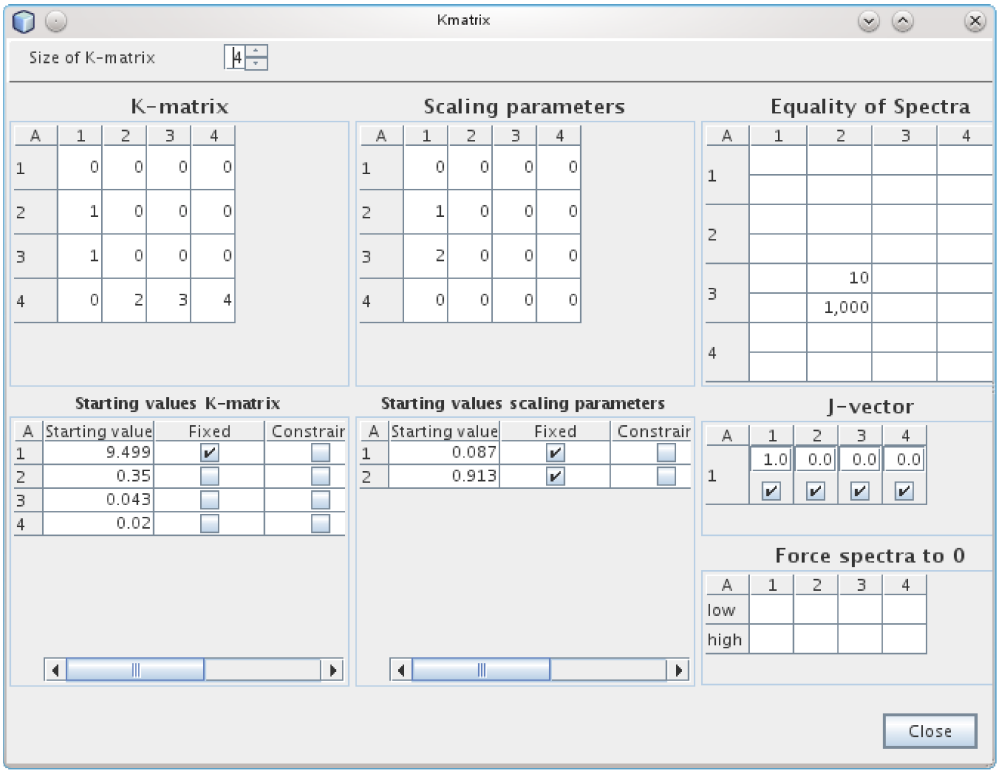


Figure 7: K-matrix editor in Glotaran version 1.0. Here a compartmental model with 4 compartments is modeled, the K-matrix is specified (top left) where the numbers represent the indices of the decay rates; the decay rates are specified (bottom left) as well as the branching parameters for the decay of compartment 1 into 2 and 3 (center top) with fixed starting values (bottom center). The input for the compartments in the form of the j-vector is specified (center right). In addition the spectra for compartment 2 and 3 are equated from (in this case) 10 to 1000 nm, i.e. over the full width of the spectrum.

2.4 Summary

Even the simplest of global analysis requires a basic understanding of the mathematical concepts behind it. And before that, a sound understanding of the experimental processes that result in the experimental data under study. In this chapter it was shown that, assuming homogeneity of the data, meaning that the data can be represented with a discrete set of parameters, and assuming separability, meaning that the data can be modeled as a superposition of the spectroscopic properties of the individual components, it is possible to define a mathematical model with which to fit the data. Typically in time-resolved spectroscopy or microscopy a kinetic model can be used for this. Kinetic models in their simplest form can be represented by a number of independently decaying components or compartments, or by an unbranched unidirectional scheme of compartments. These models are referred to as parallel and sequential models respectively. Application of such models in a least square fitting process by means of variable projection is referred to as global analysis and results in the estimated rate constants (the reciprocal of the lifetimes) per component and their

evolution associated spectra (EAS) and decay associated spectra (DAS). The DAS are interpreted as loss or gain of emission or absorption with a certain lifetime, whereas the EAS represent the spectral evolution, e.g. the third EAS rises with the second lifetime and decays with the third lifetime. When the parallel and sequential models are combined into a more complex so called compartmental the process of global analysis is referred to as target analysis, because now a specific target model is tested resulting in the species associated spectra (SAS) representing the true concentrations of the modeled compartments over time. What is left then is the actual physical interpretation of the estimated parameters.

3 Introducing Glotaran

"If I have seen further it is by standing on the shoulders of Giants."
- Sir Isaac Newton 1676

3.1 The application

In this chapter the software package Glotaran is introduced. Glotaran stands for **Glo**bal and **Tar**get **An**alysis and as such is sometimes also written as GloTarAn. Glotaran was born out of the need for an open-source, platform independent and extensible data-analysis tool with support for interactive data exploration, visual modeling and interactive visualization of analysis results. Development on Glotaran was started at the physics department of the VU University Amsterdam as a graphical user interface (GUI) for the R-package TIMP [8]. Glotaran builds on top of existing software by addressing limitations, providing new functionality and simplifying the process of model-based data-analysis for the user where possible.

3.1.1 Introduction

Glotaran was first conceived as a graphical user interface for TIMP, and therefore initially referred to as TIMPGUI. A working version of this program was first publicly presented at the useR! conference in Dortmund, Germany in 2008 as the future graphical front end for TIMP [15]. During the presentation a demonstration of a simple global analysis of streak camera data was given. Since then the application has matured and it is now possible to do global as well as target analysis and simulation of target models. In chapter 3.4 these features will be further discussed. In this chapter an overview of the application's architecture is discussed.

3.1.2 Application architecture

The Glotaran application is completely written in the programming language Java and is being developed on top of the NetBeans Rich Client Platform[3] (NetBeans RCP) using the NetBeans Integrated Development Environment[4] (NetBeans IDE). It relies on several external libraries for plotting, data-handling and communication with the R Computing Platform. Glotaran is completely open-source and the source code is published under the terms of the GNU General Public License. It is cross-platform and runs under any operating system that has the latest version of the Java Virtual Machine[5] installed. Figure 8 shows the application architecture diagram.

3 More information at http://platform.netbeans.org
4 More information at http://netbeans.org/features/index.html
5 More information at http://www.java.com/en/

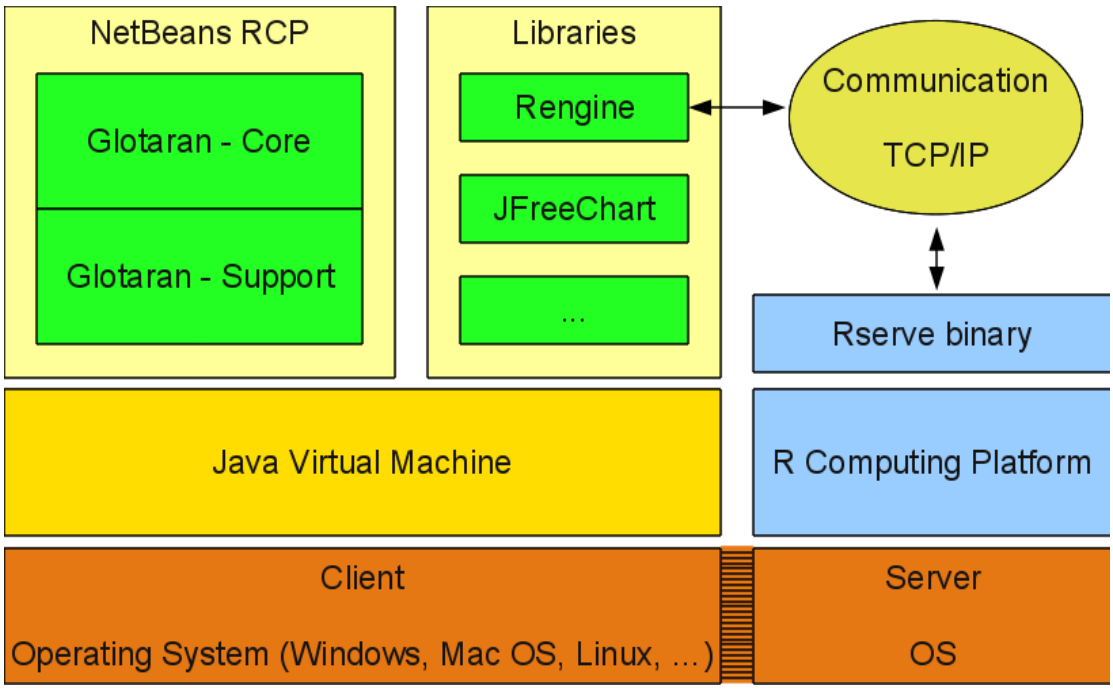


Figure 8: schematic of the interaction between Glotaran, R and Java.

The application diagram shows Glotaran running on top of the NetBeans RCP, which sits on top of the Java Virtual Machine (JVM) controlled by the client's OS. The Rserve library used facilitates the communication with the Rserve binary via the TCP/IP protocol [33]. The Rserve binary is launched by the R Computing Platform (R) controlled by the server's OS. The client and the server can be physically the same machine, allowing for the fastest communication between Java and R. However, the sever can also be a much faster computing server located somewhere in the network (or the Internet), allowing for much faster processing (at the price of slower communication). In addition the client would not need to have R installed. In the following section the various blocks in the application diagram are discussed further.

3.1.2.1 NetBeans RCP

Glotaran is not a traditional stand-alone Java application, it is actually a collection of modules that integrates into the NetBeans rich client platform (RCP). The NetBeans RCP is a reusable framework for simplifying the development of Java Swing desktop applications. The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application rather than spend their time programming the basic functionality that makes up a desktop application (i.e. interface management, windows management, storage management, etc). More details on the NetBeans platform can be found in Appendix A.

3.1.2.2 Glotaran – Core Modules

The core of Glotaran consists of the modules that provide the user interface components and handle user input. This includes the data visualization, the visual modeling, the result visualization and the analysis worker (the module that is

responsible for interpreting the user's input and delegate it to the available computational support module). The idea behind separating interpretation of the module and the actual computation is interchangeability. Currently all communication is done within the R-package TIMP but potentially the computation could be done by any software package as long as the interface with Glotaran is clearly defined in a computational support module. In this way it would for instance be possible at some point to perform the computations for analysis within a C++ application like TIM, or even within Glotaran itself.

3.1.2.3 Glotaran – Supporting Modules

The supporting modules for Glotaran provide a wide variety of functions not essential for launching Glotaran itself but essential for providing functionality to the application. Some of the functionality provided by supporting modules is:

- reading in various data file-formats and converting them to objects Glotaran understands
- saving results to file for later inspection.
- provide access to the available computational engine(s).
- provide support for all custom charts not already provided by the plotting library JFreeChart.

3.1.2.4 External libraries

Glotaran depends on various external libraries for plotting, communicating with the R computing platform, reading in various file formats and to provide for various mathematical subroutines.

JFreeChart is the main plotting library used in Glotaran. It has support for many output types, including Swing components (on display graphics), image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG). A wide variety of standard plots is available from within the Library, and JFreeChart can be extended to support different types of plots as well.

Rengine is the library provided by the Rserve software package, to facilitate the communication between Java and the R server application provided by Rserve.

3.2 Graphical User Interface

The main graphical user interface of Glotaran is provided by the NetBeans Platform. The platform is responsible for window management, the file menu, toolbar, project management and persistence (user settings, window locations, opened projects, etc.). Everything else is contributed by the various modules that make up the Glotaran application. In the following sections the various parts of the user interface and their functions are further discussed.

3.2.1 Main window

Figure 9 shows the main window provided by the NetBeans Platform subdivided in 4 zones or modes: the center editor mode, the left and right explorer mode and the upper

Toolbar mode. All windows are (un)dockable, draggable and resizable. For instance: the Project Explorer window can be undocked from the left explorer mode and docked into the right explorer mode. However, windows cannot be moved between different modes, i.e. the project explorer window cannot be dragged into the editor mode.

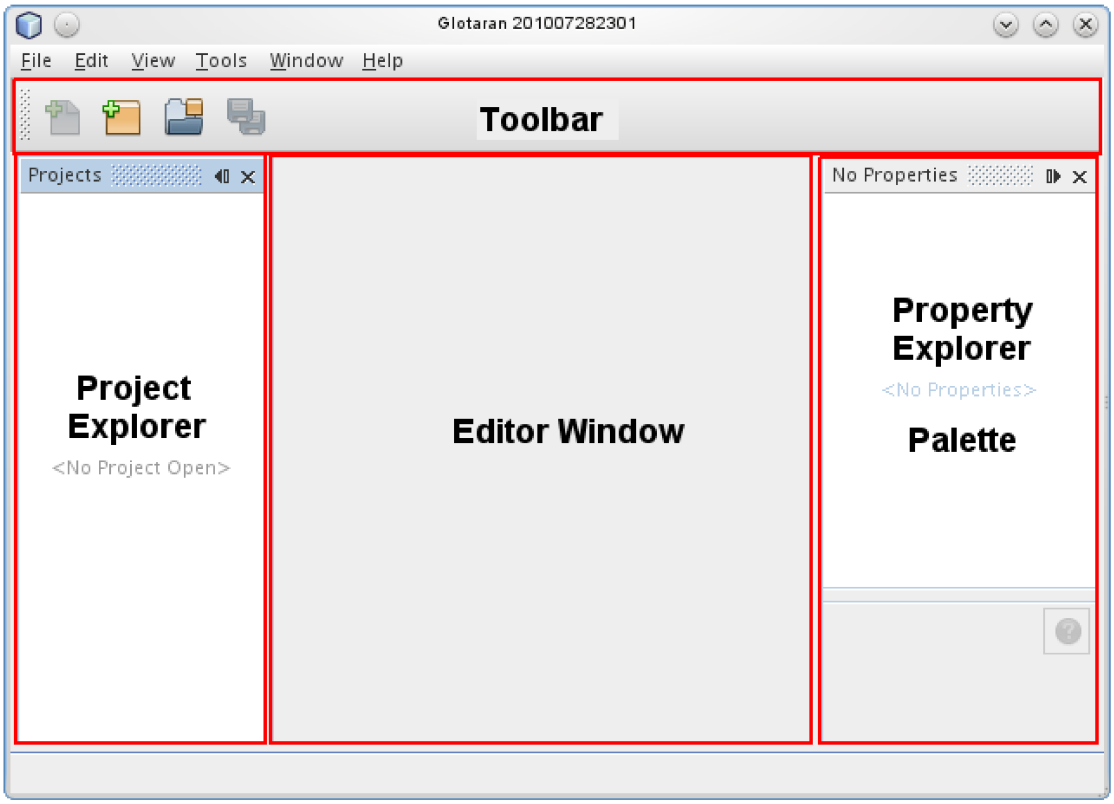


Figure 9: Glotaran main user interface window

3.2.1.1 Project explorer

The project explorer window provides a file-explorer like view of all currently opened projects. For every open project a project node is created. Each project node can be expanded to show the underlying structure. Figure 10 show an example of a completely unfolded project node showing the project structure. Each project contains the following subnodes: Datasets, Models, Results, Analysis Schemes and optionally Simulation Input.

The 'Datasets' folder node contains all datasets-files added to the project; for every file added, one subnode is created. Every dataset-file node can again contain subnodes for every dataset created from the original dataset file. A dataset created like this can be an exact copy or a sub-set of the original dataset or a dataset on which some pre-processing (like baseline subtraction) has been performed.

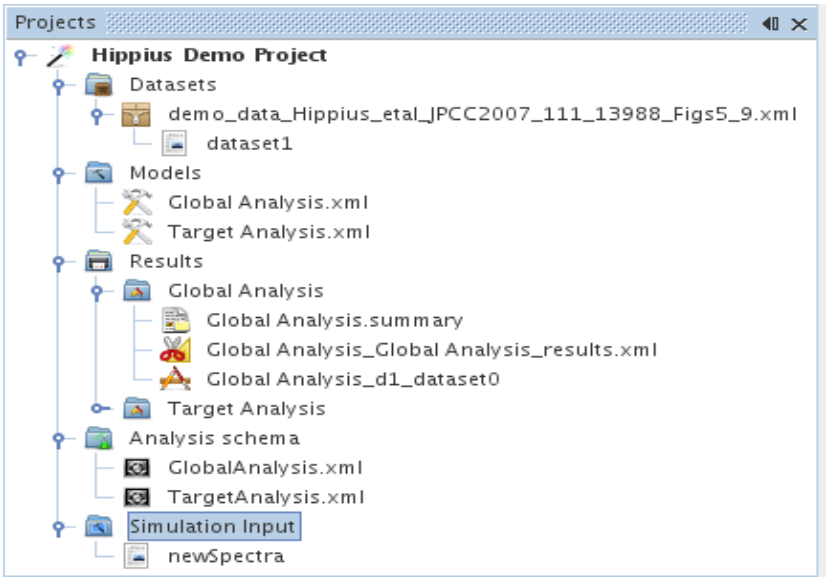


Figure 10: Project explorer window showing a project's structure

The 'Models' folder node contains all possible models that can be used in analysis. New (empty) models can be added to a new or existing analysis scheme from which they can be edited.
The 'Analysis schema' folder node contains the available analysis schemes. An analysis scheme file contains the analysis instructions for Glotaran: which datasets and what models are used and how these relate to each other. Once opened it opens in the visual analysis editor (see §3.2.3).
The 'Results' folder node contains a result folder for every analysis run. Every result folder contains a summary file (a plain text file with a brief summary of the results), a _results.xml file (further detailing the results from the analysis) and one result-file per analyzed dataset. The result-files can be opened and viewed in a separate viewer.
The 'Simulation Input' folder node contains input files detailing spectra to be used in the simulation of experiments.

3.2.2 Data exploration windows

Glotaran currently supports two types of data: time-resolved spectroscopy data and time resolved microscopy data. Both data-types differ in the way they should be presented to the user and therefore have 2 different visualization windows.

3.2.2.1 Time-resolved spectroscopy data

Time resolved spectroscopy data consists of some form of multiway data; typically 2-way data with intensity measured as a function of the experimental spectral variable wavelength λ and the independent experimental variable time t relative to the moment of excitation. Figure 11 shows the time-resolved spectroscopy data-editor with an example of such data.

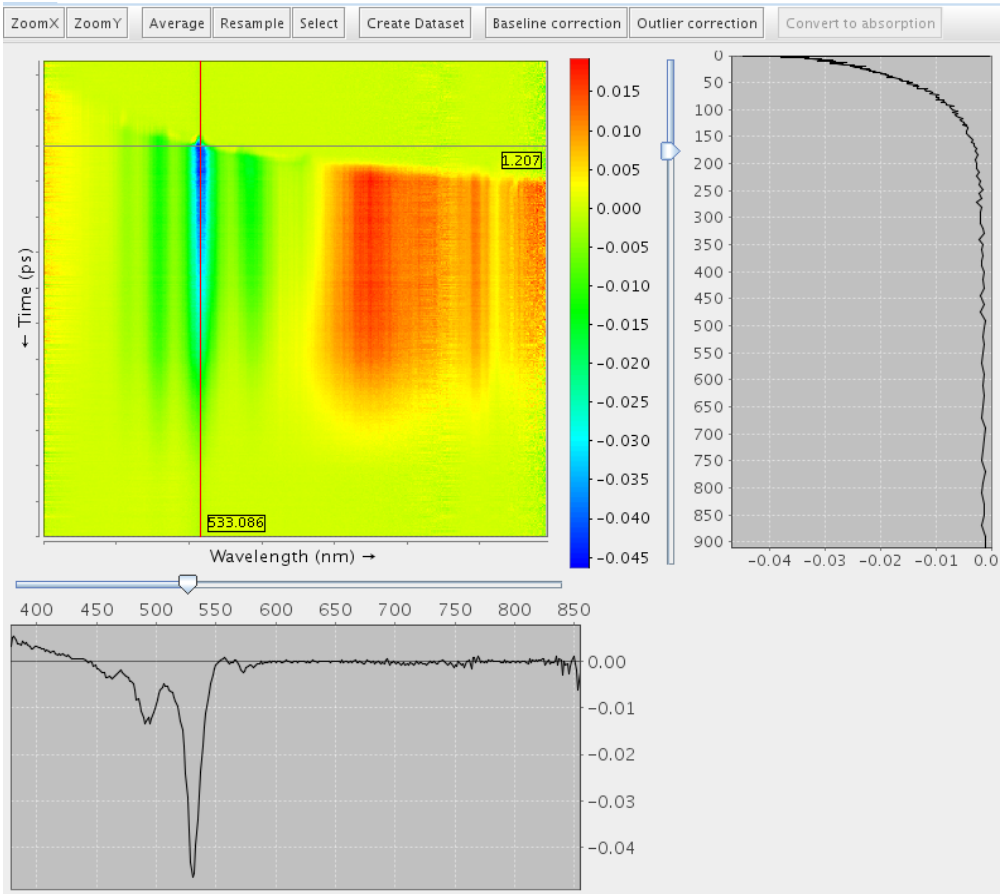


Figure 11: Time-resolved spectroscopy data editor; example with femtosecond transient absorption spectroscopy data

In this editor the value of the measurement (fluorescence intensity, absorption, etc.) is shown as a function of time (on the vertical axis) and wavelength (on the horizontal axis). The sliders can be manipulated to show one specific time-trace (at a certain wavelength) or spectrum (at a specific time) from the entire dataset as indicated by overlaid lines. In this specific example time-resolved spectroscopy data is displayed, with wavelength as the independent spectral variable. However other experimental variables like temperature, polarization dependence (anisotropy), acidity (pH or pD), excitation wavelength or quencher concentration are also possible. For the moment only the visualization of 2-way data (the most common type) is supported but support for multi-way data is conceivable in the form of multiple 2D plots where one of the independent variables is interchanged.

3.2.2.2 Time-resolved microscopy data

Time-resolved microscopy data typically consist of spatially-resolved time-dependent fluorescence data. This kind of data is represented by a 2 dimensional spatial plot where every pixel consists of a time decay-trace as depicted in Figure 12.

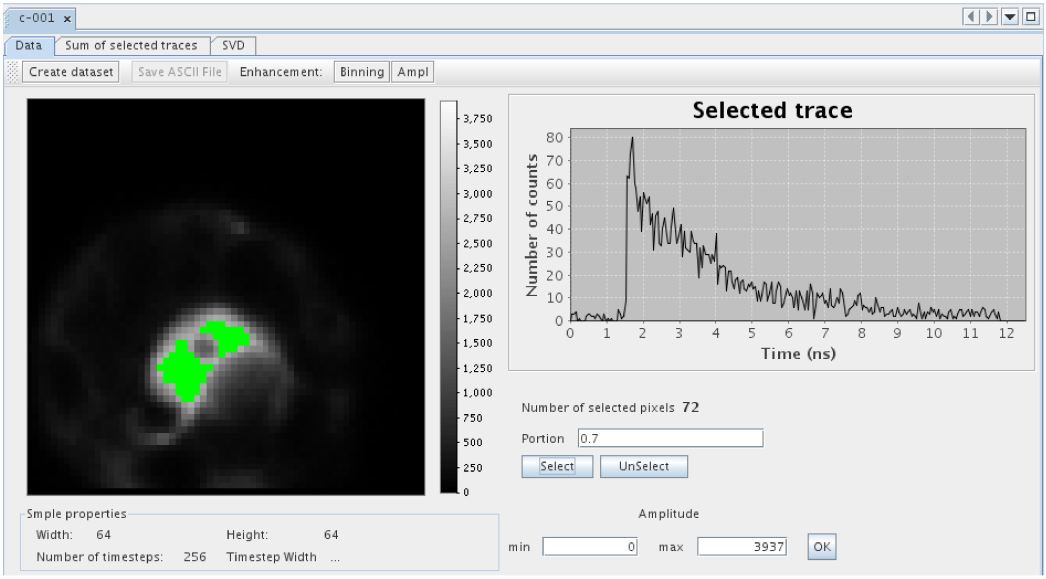


Figure 12: Time-resolved microscopy data editor; example with FLIM data.

In this editor the fluorescence intensity is shown as a function of two spatial coordinates, x and y. The time trace of the currently selected pixel is displayed to the right. For data-analysis a number of pixels is selected by clicking in the 2D plot and selecting pixels or simply by providing a certain threshold to automatically select a number of pixels; subsequently only the selected pixels serve as input for later analysis.

3.2.2.3 Other data types

Besides time-resolved spectroscopy data, and time-resolved microscopy data there is a wealth of data that could benefit from global or target analysis. The computational back-end TIMP was designed with the extension for other types of data in mind and Glotaran's modular design allows for easy support of visualization of new data-types (even by third party developers).

3.2.3 Analysis scheme editor

The analysis scheme editor is the editor for analysis files and provides the functionality to create and edit models (as part of the analysis scheme). Figure 13 shows a screenshot of the editor together with the Palette and Properties editor, which work in conjunction with the editor.

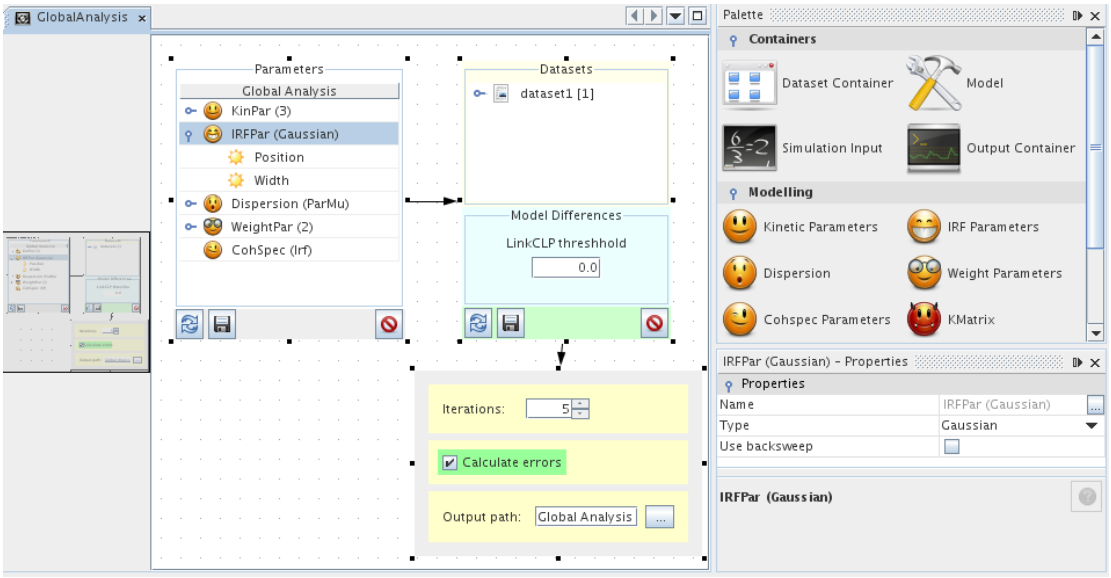


Figure 13: Analysis scheme editor

The Analysis scheme editor provides a visual view of the whole analysis process. A model consisting of a list of parameters is applied to one or more datasets and this is then linked to an output window. In the following sections the various components are explained in more detail.

3.2.3.1 Model specification window

The model specification consists of a so-called tree table view containing a list of parameters. Most parameters nodes can be expanded to view the underlying parameters, as indicated by the 'o-' symbol. The model specification is directly linked to a model specification file in the project's model directory in which the parameters of the models are stored. The various parameters can be dragged onto the view from the Palette and changed using the property explorer (see Figure 13). The KinPar node represents a number of kinetic rates. The IRFPar node represent the Instrument Response Function (IRF). This Dispersion node models dispersion of the IRF in case it has a wavelength dependent character. WeightPar can be used to give weights to certain regions of the data indicated by 2 number specifying the interval in a dimension and one number specifying the weight to be applied there. Finally, the CohSpec node holds the specification for the coherent artifact; typically modeled after the IRF.

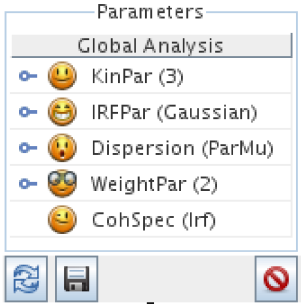


Figure 14: Model specification editor

3.2.3.2 Datasets specification window

The dataset specification window (Figure 15) is a relatively simple container that holds a reference to all the datasets (typically only one) that will be analyzed together. The number behind the dataset indicates to which group the dataset belongs.

Technically it is possible to analyze multiple datasets together, and for this purpose it is possible to free parameters from the originally specified model between datasets. These are so called 'model differences' between datasets. The LinkCLP threshold is a parameter that can link the conditionally linear parameters together if they fall within a certain interval, the value of this interval is controlled by this parameter. Figure 16 shows the different options for model and model differences specification.

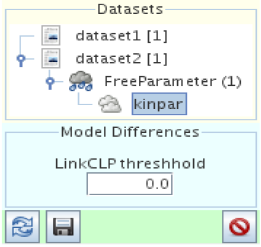


Figure 15: Dataset specification window with 2 datasets and model differences between datasets

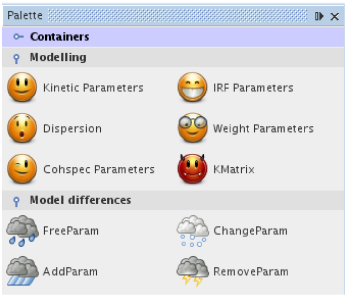


Figure 16: Palette with model differences shown

The FreeParam node allows for the possibility to free one or more parameter between datasets, in effect adding one or more degrees of freedom. The ChangeParam allows for the possibility to change a whole set of parameters. AddParam and RemoveParam can add or remove a parameter from the model for a dataset respectively. The freeing of parameters works relatively well but all other options should be used with caution and can lead to unexpected results.

3.2.3.3 Output specification window

The output specification windows is a simple window with the most important option being the number of iterations the model has to run for before it stops, even when convergence might not have been reached. Next is the options to calculate the errors in the resulting spectra. Note that the calculation of errors can increase the total time take for computation considerably. And finally there is the option to change the location where the results should be stored.

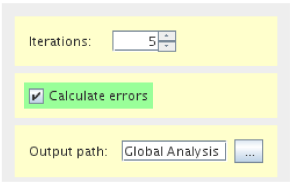


Figure 17: Output window

3.2.3.4 Simulation specification window

The simulation specification window is for all practical purposes identical to the data specification window, except that rather than a model being fitted to one or more datasets, instead a model is applied to one or more simulation input nodes which describe of one or multiple spectra and outputting a simulated dataset.

3.2.4 Results presentation windows

For the presentation of the results Glotaran makes use of the open source plotting library JFreeChart.

3.2.4.1 Time-resolved spectroscopy analysis results

The results window for the analysis of time-resolved spectrocopy data consists of 4 tabs. The first tabs provides a general overview of the fit in the form of spectra (EAS

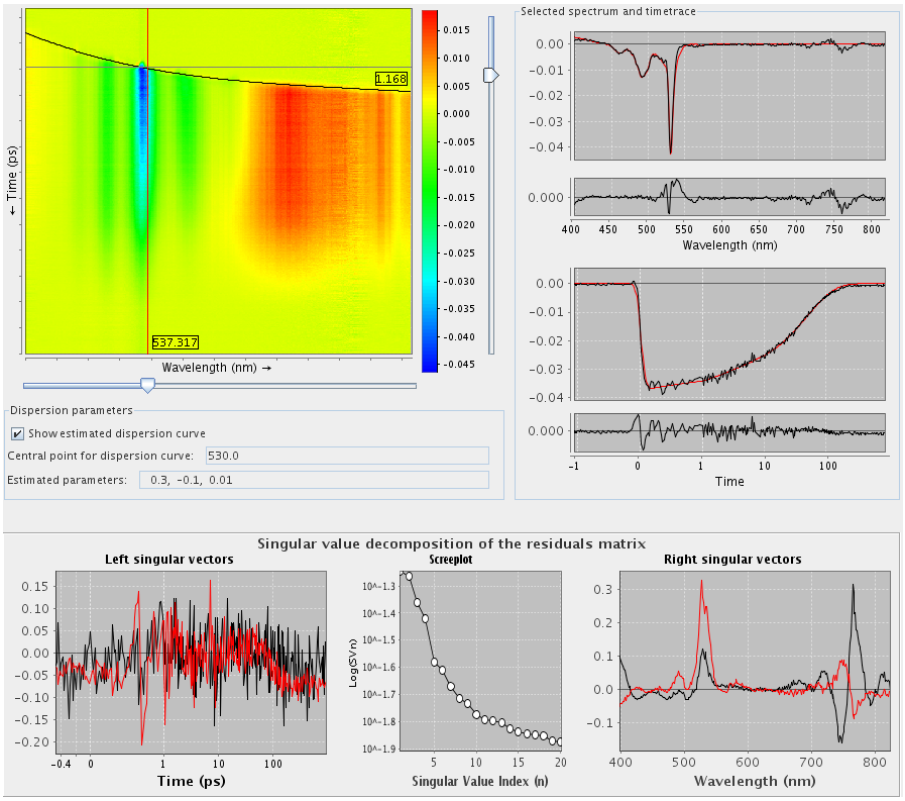


Figure 18: Overview of the fitted traces of a global analysis of time-resolved spectroscopy data

or SAS and DAS, as well as the normalized spectra) and the concentration profiles. In addition the SVD of the residual matrix is shown. The second tab shows the quality of the fitted traces. The last two tabs hold manually or automatically selected time and spectra traces to provide an overview of the data. The overview tab and the traces tab are shown in Figure 19 and Figure 18 respectively. Note that for both tabs it is possible to display the time traces on a linear-logarithmic scale where the linear parts of the axis is indicated by a time point (in this case 1).

The overview tab of the results show the evolution associated spectra (EAS), or the specifies associated spectra (SAS) if a target scheme was used, the decay associated spectra, and the normalized version of both. A summary of the estimated parameters is shown in a scrollable text field and below the concentration profiles of the estimated components are displayed. If a coherent artifact was modeled it will be shown here and can also be shown in the EAS or SAS by toggling the 'ShowCohSpec' in the toolbar of the window (not shown here).

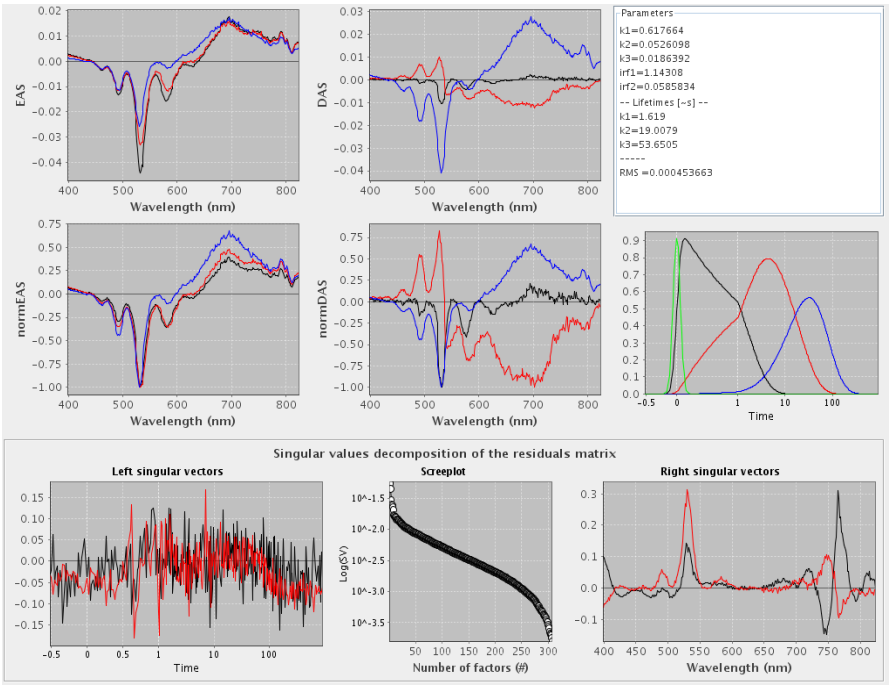


Figure 19: Overview of the results of a global analysis of time-resolved spectroscopy data

The overview of the fitted traces show the original data, possibly with an overlaid dispersion curve estimating the maximum of the IRF. The scrollbars to either side of the data window can be used to inspect the fitted time traces and spectra at various locations. Finally the SVD of the residual matrix is again displayed to inspect the quality of the fit and help determine if there are any trends left in the residuals that might need to be modeled.

3.2.4.2 Time-resolved microscopy analysis results

The results window for the analysis of time-resolved microscopy data consists of 3 tabs. The first tabs provides an overview of the analyzed pixels, the average lifetime per pixel the average lifetime histogram and the SVD of the residual matrix, see Figure 20. The second tab allows for the possibility to inspect the fitted traces per component, see Figure 21. In the overview of the fitted traces per component, per pixel a pixel can be selected to inspect the respective trace and corresponding fit.

Data analysis of FLIM data using TIMP and Glotaran was first demonstrated at the 2008 useR! Conference [34]. More information on data-analysis of FLIM data can be found in [13].

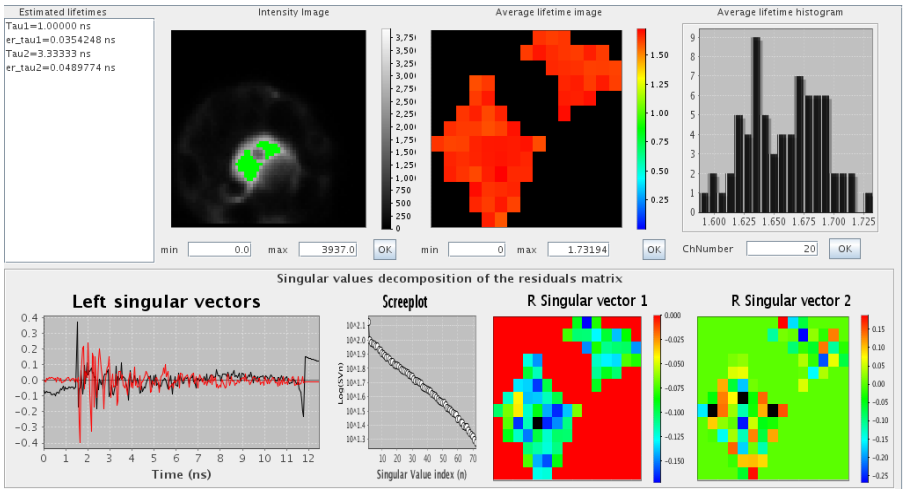


Figure 20: Overview of the global analysis of FLIM (time-resolved microscopy) data

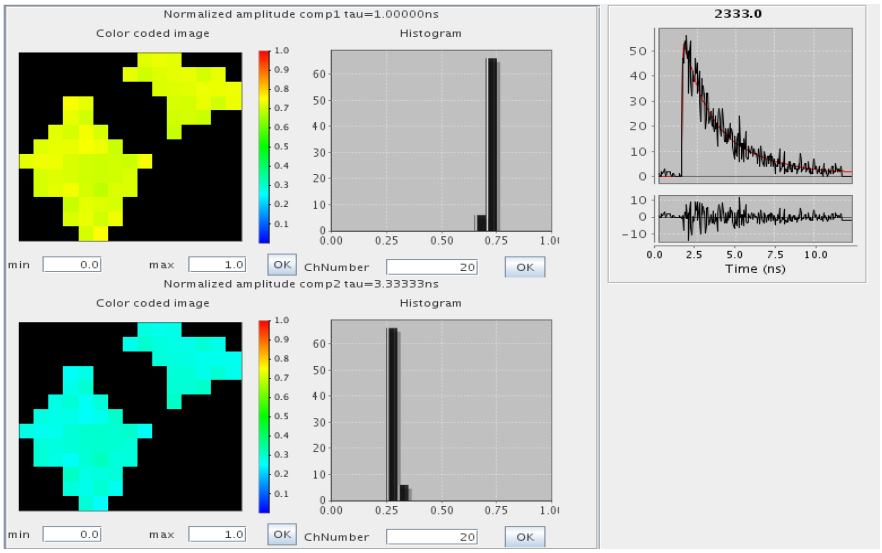


Figure 21: Overview of the fitted traces per component, per pixel for FLIM data

3.3 Screencast

In order to better demonstrate the working of Glotaran a screencast has been made showcasing the process of data-analysis in Glotaran. Provided with this thesis is a DVD containing a copy of this screencast in the open format ogg. Alternatively it can be viewed directly from the website: glotaran.org. The transcript for this screencast in included in Appendix D: Glotaran screencast.

4 Case study

4.1 Introduction

This chapter aim to demonstrate the use of Glotaran by means of a practical case study in which global and target analysis is applied to femtosecond transient-absorption spectroscopy data. The data in question originates from an experiment investigating a supramolecular building block consisting of a perylene bisimide chromophore (called **o**range) substituted with one **c**alix[4]arene unit, termed **oc [35]**. The orange chromophore is highly fluorescent with a quantum yield close to 1 and a lifetime of around 4 ns. In contrast, **oc** has a very low quantum yield. In the study femtosecond transient absorption spectroscopy is used to investigate a particular quenching process, likely to be attributed to a rapid photo-induced electron-transfer process. The data was analyzed using a combination of global and target analysis. This chapter focuses more on the intermediate steps and thought process required to arrive at the final results rather than the final interpretation of the resulting lifetimes and spectra.

4.2 Exploring the data

There are many different ways to visualize time-resolved spectroscopy data. In Glotaran the data is visualized in the form of a 2D intensity map where one dimension represents time whereas the other dimension represents wavelength; see Figure 22. The colors represent the relative intensity of the measured spectra (in arbitrary units) as represented by the colormap to the side of the plot.

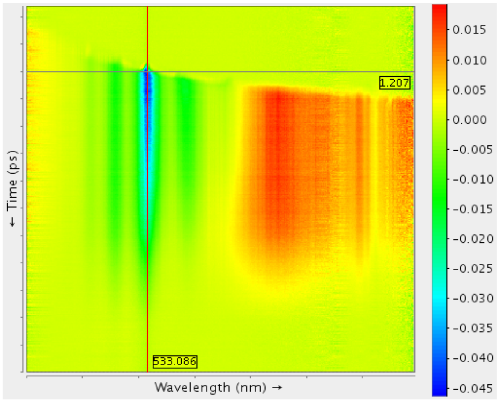


Figure 22: Raw data visualized as a 2D intensity map. The horizontal dimension is wavelength (in nm) and the vertical dimension is time (in ps)

There are two things that are immediately obvious from this representation of the data. First, the data become noisier towards the edges of the measured spectrum. And two, there is definitely some dispersion in these data, i.e. the maximum of the IRF at

higher wavelengths is slightly delayed. This is something to consider later in the analysis phase.

4.2.1 Data pre-processing

Data pre-processing is an often neglected but important step in the whole process of data analysis. Ideally, measurements would always results in high quality data where any amount of pre-processing would degrade the quality of the fit rather than improve it. In practice however the opposite is often the case. If there are outspoken outliers in the data, for instance due to measurements artifacts (e.g. cosmic background radiation) then these points needs to be discarded (or replaced by some average value of the surrounding pixels) before analysis. If certain regions are particularly noisy (often the edges of the spectra) then sometimes it is helpful to apply a window selection where these regions are excluded; provided the excluded region is not of particular interest. It is also possible to include the noisy region but subject these data points to a lower weight during analysis. This is particularly recommended if it is not yet clear if the excluded region holds information regarding the global dynamics. The Singular Value Decomposition of the data can often provide a handle to determine problematic features in the data, such as: extreme outliers, noisy regions or systematic experimental errors. In this particular dataset no significant outliers exist.

4.2.1.1 Window selection

Upon closer inspection of the right singular vectors the data below 390 nm and beyond 820 nm can be considered particularly noisy. Therefore a window selection in the spectral domain is done to include only the data between 390 nm and 820 nm. The left singular vectors seem reasonably well behaved over all spectra and thus no window selection is done in the time domain.

4.3 Interactive modeling

After pre-processing the data a first attempt can be made to model and fit the data. The best approach is often to start as simple as possible, choosing reasonable starting values for the minimum set of parameters that is expected to adequately describe the data. The chosen parameters should then be optimized by running the model for a few iterations and re-adjusting the starting values based on the results. Inspection of the fitted spectra and time traces, and by looking at the SVD of the residual matrix can then justify gradually increasing the complexity of the model to account for any misfit or misinterpretation of the physical spectra (for instance, negative bands where physically can be none). This process of modeling, inspection the results and adjusting the model accordingly is what is called interactive modeling and has been described in detail in §1.1.1, the same principles will be applied in this section.

4.3.1 Initial model

As is discussed extensively in §2.3.1, the first step in a global analysis typically involves looking at the singular value decomposition of the dataset. The singular

value decomposition (SVD) is a matrix factorization technique which can be used to explore the number of spectrally and temporally independent components in the data matrix, which is an important aspect of defining an initial model. The SVD of the data-matrix is shown in Figure 23. Looking at this decomposition it seems that the first four, possibly five singular vectors have enough structure to justify including as a component in an initial model.

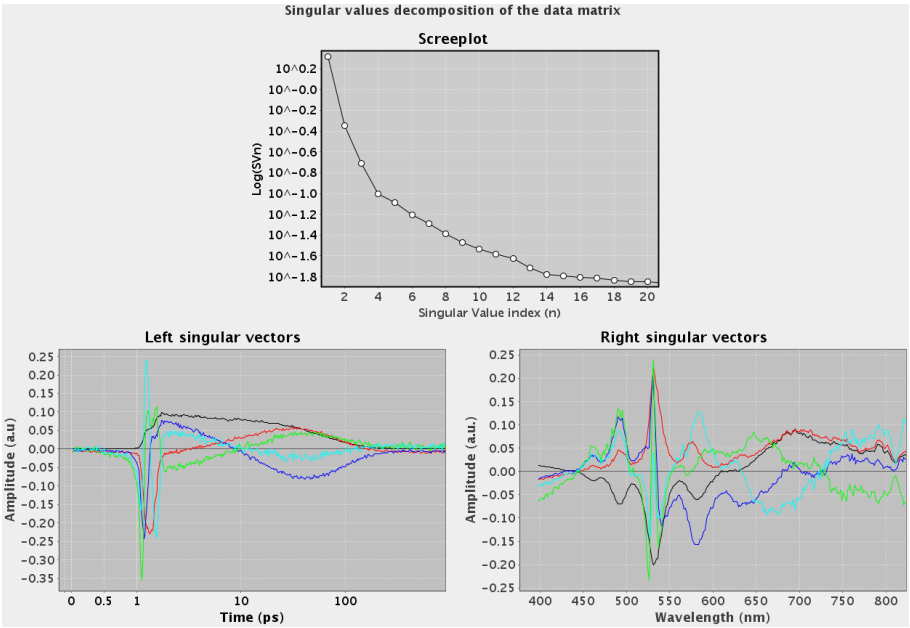


Figure 23: Singular Value Decomposition (SVD) of the data-matrix of the dataset from Figure 22. The first 5 singular vectors are plotted respectively in black, red, blue, green and cyan.

Motivated by the SVD of the data a 4 component sequential scheme with increasing lifetimes is chosen for the initial model. The lifetimes for the components are initially very roughly estimated to be 200 fs, 2 ps and 20 ps and one long lived component of more than 200ps, corresponding to rates of respectively 5, 0.5, 0.05 and 0.005. Naturally the time profile of the IRF is modeled as well. The starting values can often be estimated by the time profile of one of the first few left singular vectors. In this case the time profile of the red trace seems a good estimate, providing the starting values of 1.2ps as the position and 0.1ps as the width of the time profile. It turns out that this simple model can be used to globally fit the data and the results of this first analysis are shown in Figure 24.

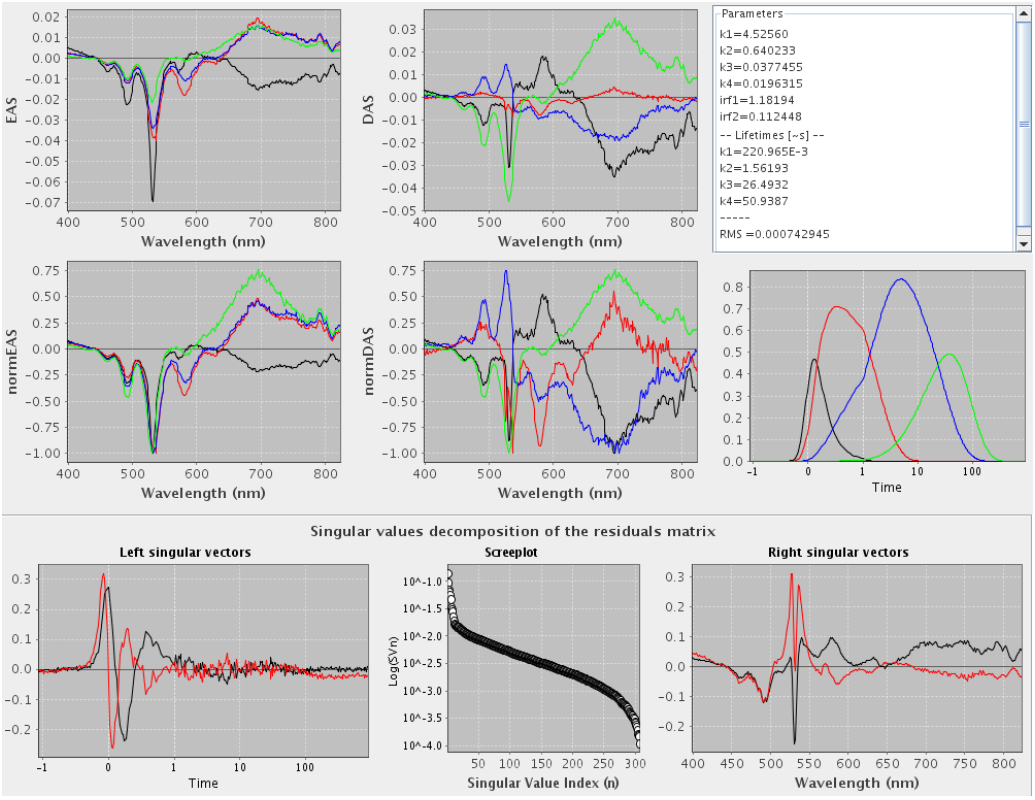


Figure 24: Overview of the results fitting the initial model

4.3.2 Extending the model

The model established in §4.3.1 describes the data already reasonably well. The final residual standard error after optimization - i.e. running it for a few iterations until convergence is reached - is 7.429E-4. Visual inspection of the fitted spectra and time traces reveal that his model agrees reasonable well with the data but there are some clear misfits as also indicated by the structure in the singular vectors from the SVD of the residual matrix. This suggests that the model can still be improved; extended with additional fitting parameters to better describe the data.

4.3.2.1 Adding dispersion

The first extension to the model should be the modeling of the dispersion already clearly observed when visualizing the data in Figure 22. Dispersion means that the IRF location has a wavelength dependent character. In this figure there is a clear delay visible between the moment of excitation at for instance 500 nm and 800 nm. If this is not correctly modeled it can actually appear as a short lived component and affect the estimated lifetimes of the other components or mask other short lived components entirely. This dispersion curve (the wavelength dependent curve to model the time of excitation) can be modeled by a quadratic polynomial function of wavelength, centered around a particular wavelength. The starting values for the coefficients of

this polynomial are specified to be 0.3, -0.1 and 0.01. The center wavelength is specified to be 530 nm. The results of the analysis with added dispersion are shown in Figure 25.

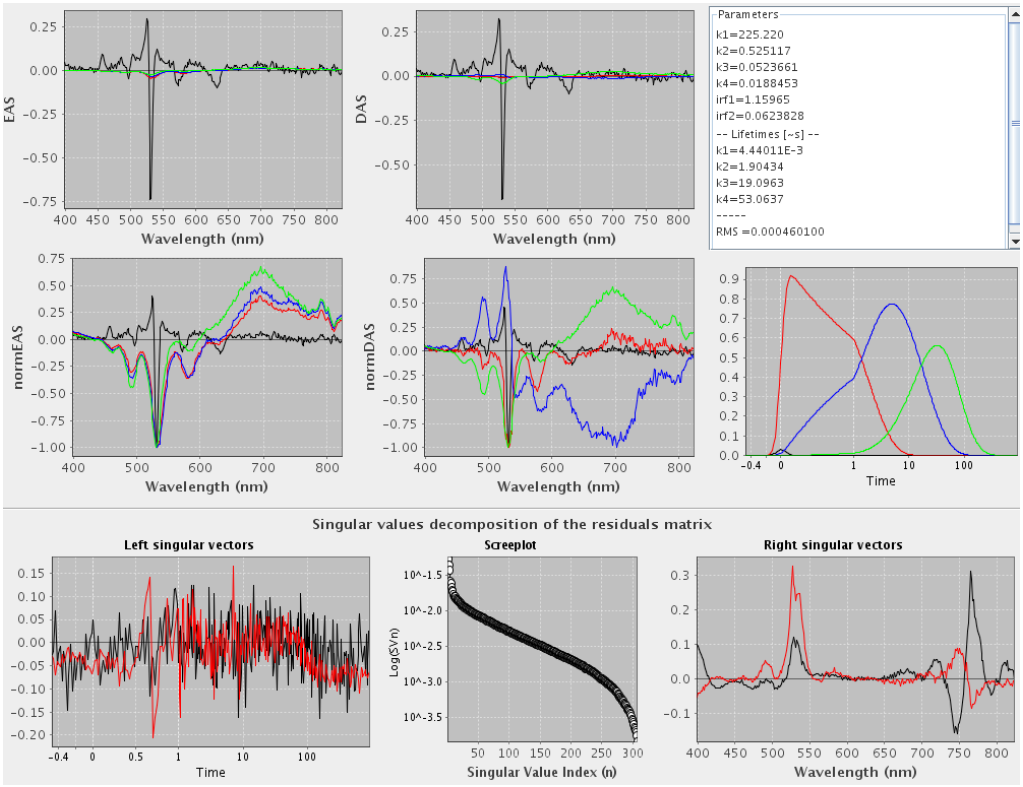


Figure 25: Results of the initial model extended with dispersion parameters

This model has reduced the total RMS of this fit to 4.601E-4, indicating a significant improvement in the overall quality of the fit. One of the most noticeable features of this analysis is the relatively large amplitude of the first EAS, corresponding to an ultra-fast lifetime of only 225 fs and with a maximum around the excitation wavelength of 530 nm. Before, this component was probably masked by the global dispersion. The IRF-like behavior of this ultra-fast component and its unphysical spectral contribution means that this component should be considered as a measurement artifact, as will be discussed further in §4.3.2.2. The quality of the fit of the dispersion can be observed directly in the data-inspection plot of the results (see Figure 26) where the dispersion curve is overlaid on top of the original data.

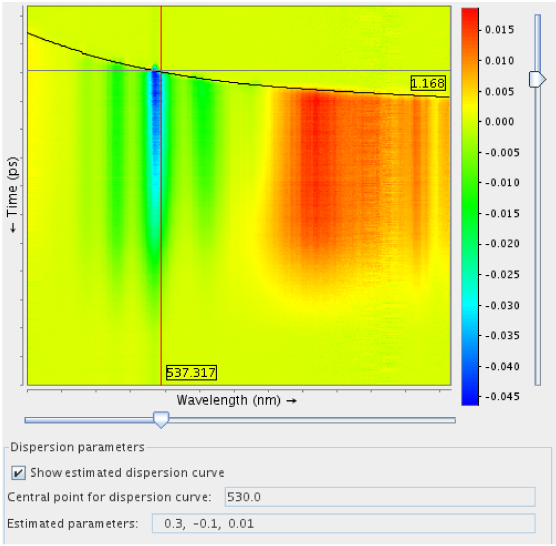


Figure 26: Fitted dispersion curve overlaid on the original data

4.3.2.2 Coherent artifact

The measurement artifact discussed in §4.3.2.1 is probably due to a so called coherent artifact, a common occurrence in femtosecond transient absorption spectroscopy [36]. Usually its time profile can be approximated by the time profile of the IRF and thus modeled. The results for the model from §4.3.2.1 where the ultra-fast lifetime is removed and modeled instead as coherent artifact are shown in Figure 27.

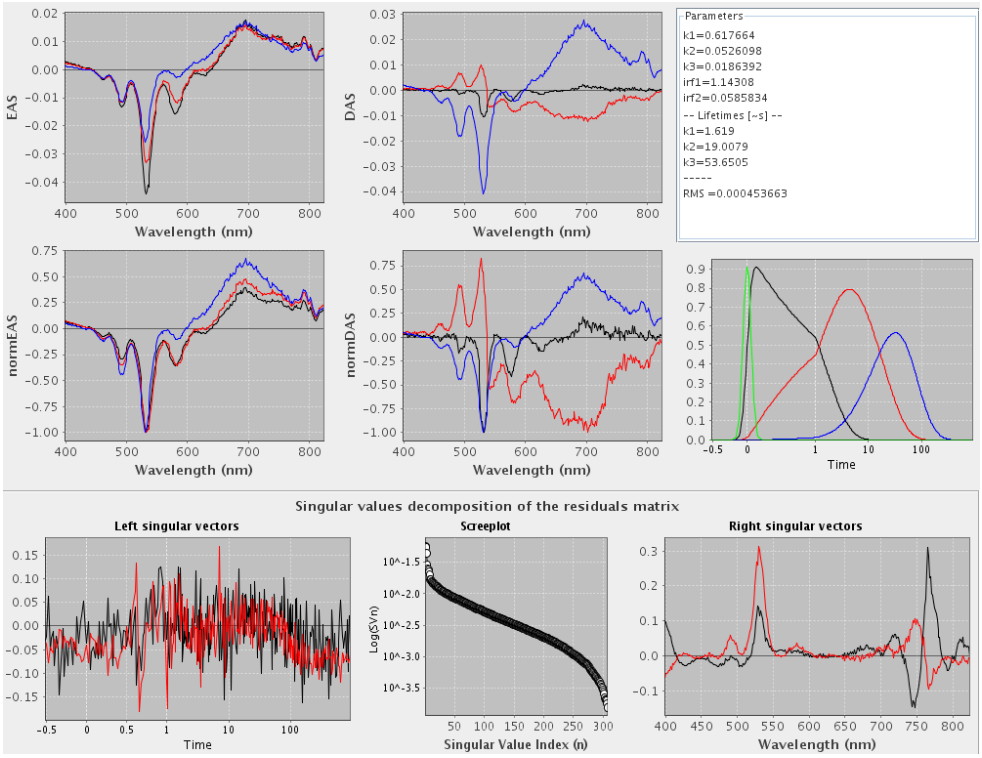


Figure 27: Overview of the results from fitting a model with coherent artifact

As a result of modeling the ultra-fast component as a coherent artifact the RMS was further decreased to 4.537E-4. There is still some trend left in the left singular vectors of the residual matrix but it is much more noisy than before and will be difficult to model.

4.3.2.3 Weighing the data

One aspect from the results of the analysis in 4.3.2.2 is the significance of the contributions of different parts of the spectrum. In the right singular vectors in Figure 25 it can be seen that most of the contribution to the residuals originate from those parts of the spectra that are of least interest with respect to the system under study. No significant spectral contribution are expected below 550nm (other than the excited state bleach around 530nm) or above 720 nm, yet this is where the residuals are highest. In order to reduce the effect that these regions have on the fitted lifetimes their weight is reduced. The region from 400nm to 550nm is assigned a weight of 0.1, and the region from 720 to 850 is assigned a weight of 0.2. Although the application of weights should not affect the estimated lifetimes and resulting spectra too much, it does directly affect the RMS value. In fact, applying a weight of 0.1 over the whole spectral region would result in an identical analysis, expect that the RMS value of the fit would be reduced by a factor of 10. Therefore, the results from two different models can only be compared based on the RMS value if they use identical weights.

Although it might be possible to improve the quality of the fit by applying weights in this case it would not add much to the story of this chapter. Instead the application of weights will be tested in the section on Target Analysis.

4.3.2.4 A comparison

At this point the results of the global analysis can be compared to the results from the analysis in the original publication describing these data, see Figure 5A and Figure 6A from Hippius et al. [35].
In the analysis described in this chapter the lifetimes corresponding to a sequential kinetic scheme with increasing lifetimes were found to be 1.8ps, 19ps and 53ps. The resulting EADS are shown in Figure 27 (labeled EAS). In Hippius et al. the resulting lifetimes from global analysis were 1.8ps, 22ps and 50ps and the resulting EAS are shown in Figure 6A of that publication. Note that in that figure the coherent artifact is not shown. Also it appears that the spectrum for the 50ps component is incorrect (compare with Figure 27), although the lifetime is correctly estimated. Most likely this is due to a plotting error from the original author and not an error in the data-analysis[6]. The lifetimes in this analysis and those from the Hippius paper are found to be in reasonable agreement with each other. In reasonable agreement here is defined as "lie within 10% of each other". There are always bound to be differences between similar analyses of different measurements, or even similar analyses of the same experiment. Differences between two similar analyses of the same dataset can be apart from differences in the kinetic model, due to differences in data-preprocessing (e.g. the spectral or temporal window selected), differences in applied weights, spectral constraints and whether certain parameters are fixed or free.
In the Hippius paper it was assumed that the very first lifetime (1.8ps) is due to very fast solvent reorganization processes. In 1.8ps the first EADS (black EAS spectrum in Figure 28) evolves into the red spectrum. Which then evolves in 19ps into the blue spectrum. This final component then decays in 50ps. It is suggested that the transition from the red to the blue spectrum suggests the formation of an additional excited state due to the almost complete loss of stimulated emission at 580nm, together with the broadening and apparent blue shift of the positive transient absorption band at around 700nm, which could be attributed to a non-emissive component (i.e., a charge transfer or CT state). However, due to the non-emissive character of this intermediate state a simple sequential model is no longer sufficient to model this schema. A more complex target analysis is required.

4.3.2.5 Target Analysis

The EADS resulting from the global analysis with a simple sequential kinetic scheme represent the global evolution of the sample accurately but as was found in §4.3.2.4 they cannot truly reproduce the spectra of the individual excited species that are involved in the processes occurring upon photo-exciation of the sample. This requires a target analysis with a specific model based on a compartmental scheme as described

6 This was confirmed by inspecting the original results from the data-analysis as kindly provided by Dr. Ivo H. M. van Stokkum

in §2.3.2.1. Here the first compartment represents an excited state population which is populated almost instantaneously from the IRF. From this excited state there is branching to two separate compartments. One compartment represents the population found in global analysis which evolves with roughly 19ps, and the other compartment decays much faster in roughly 2ps but exists in much lower concentrations (the branching ratio between compartments 2 and 3 is modeled to be roughly 9:1). Compartment 2 and 3 share the same SADS and evolve with their respective lifetimes in the final component, a non-emissive CT state, which then decays in roughly 50ps to the ground state. The data is weighted to decrease the influence of the outer regions of the spectra, applying a weight of 0.1 to the spectrum from 400nm to 550nm and a weight of 0.2 to the region of 720nm to 850nm. The results of the analysis are shown in Figure 28.

These results correspond quite well with the results from the analysis in the original publication, but they are different in the sense that here the oc compound was analyzed individually and not simultaneously with the oc2 compound as was the case in Hippius et al. Figure 6B. As a result the lifetimes are slightly different but still very similar. Note that the ultra-fast first component is plotted here, but not in Figure 6B of Hippius et al. The SADS of the final spectrum indicated by a green line in Figure 28 with a lifetime of 52 ps reveals the spectral features of a charge-transfer state: virtually no stimulated emission at 580 nm and a broadening of the band clearly visible below 700 nm. Finally the RMS is now 2.467E-4, which can no longer be compared with the Global Analysis due to the difference in weights but is still very reasonable.

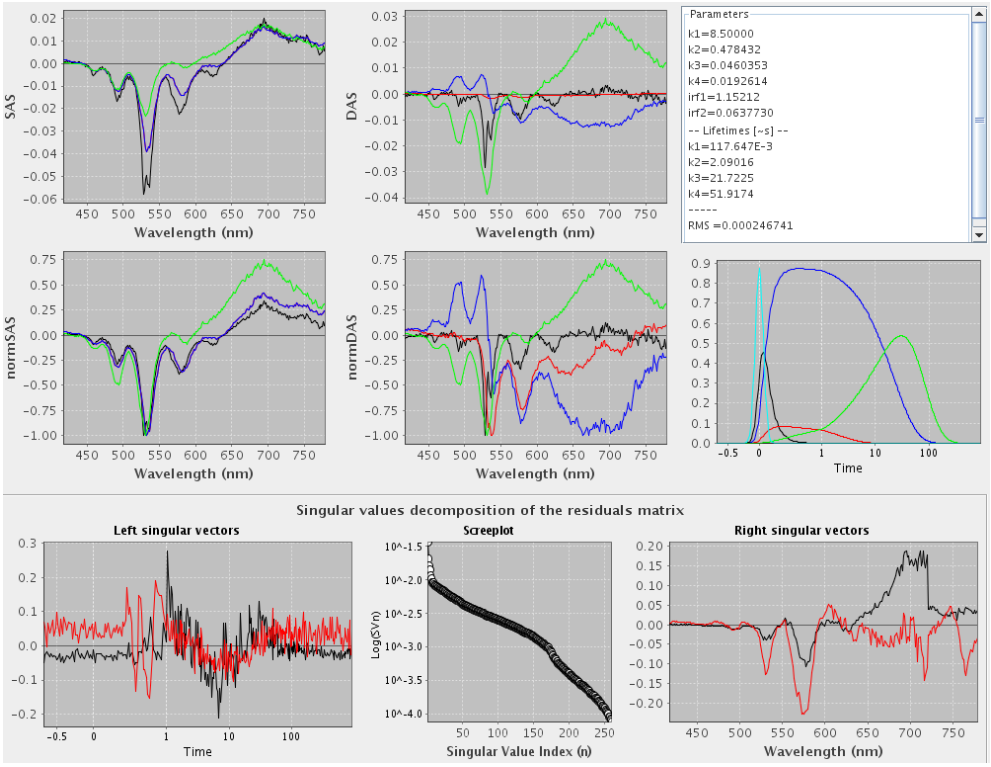


Figure 28: Results of the target analysis using a compartmental scheme

4.3.2.6 Further analysis

Looking at the SVD of the residual matrix of the target analysis it is clear that there is still room for improvement. There is a clear trend in the residuals for instance that could indicate the formation of a long lived triplet state. However, as with any extension to the model the questions is whether this will actually have a significant positive impact on the results as a whole. Every additional parameter adds another degree of freedom, and every parameter that needs to be fixed because of the additional freedom is a parameter lost. There will always exist a balance between the quality of the fit and the complexity of the model, but in the end all parameters should be traced back to some physical origins.

5 Conclusions

Biophysics is a field in science that applies the principles of physics and chemistry and the methods of mathematical analysis and computer modeling to biological systems, with the ultimate goal of understanding at a fundamental level the structure, dynamics, interactions, and ultimately the function of biological systems. Nowadays in biophysics, data are often acquired using advanced techniques in optical spectroscopy and microscopy generating large amounts of complex data. These data require sophisticated data analysis in order to extract the physically significant parameters underlying the data (e.g. the decay rate of the concentration of a component, or the location of the maximum of the spectrum of a component). Glotaran has been introduced as an open-source, cross-platform application to facilitate the process of interactive global and target data-analysis, required for the analysis of these data. Glotaran adds in functionality compared to existing software: interactive data-visualization, visual modeling and interactive inspection of analysis results. In addition its well defined 'project' facilitate collaborative analysis. Finally, the modular architecture induced by development on top of the NetBeans Rich Client Platform allows for easy extension of functionality. As of version 1.0.0 Glotaran is capable of handling complex kinetic models and has been shown to be capable of analyzing real experimental data. In the next chapter some ideas for future improvement and extension of the functionality of Glotaran are given.

5.1 Current usage

Although the functionality of Glotaran is still fairly modest compared to its full potential, already Glotaran enjoys a reasonably sized user base. At the time of writing, several weeks after the release of version 1.0.0 this is estimated between 30 to 40 users[7].

Various individuals from different parts of the world have taken the time to report using Glotaran, and more often than not have reported bugs or suggested various improvements, helpful to their particular situation. Some of those suggested are already implemented in a future update. One specific user-testimonial stands out as it perfectly grasps the essence of the idea behind the project that started the development of TIMP and Glotaran in the first place:

“I will definitely do some advertising for Glotaran/TIMP in my work group, it is something I searched for, but never found. As I already said, I think the ultrafast community is wasting a lot time (and quality sometimes) by not sharing their fitting and processing software. It seems these routines are the trade-secrets of the groups, which is in my opinion the wrong way of science.”

7 This estimate is based on the number of unique downloads from the project's hosting site kenai.com and user feedback.

6 Future work

In this final chapter the key areas in which Glotaran could be improved are identified and for each area various ideas or concepts of improvement are listed. For each idea/concept a description is given and a score with respect to the following three categories is assigned:

- Complexity – how difficult would it be to implement in terms of programming and theoretical groundwork.
- Applicability – how relevant is it to functionality of the Glotaran application as a software tool for global and target analysis of biophysical data.
- Desirability – how much would the feature probably be appreciated by users of the Glotaran application.

Each concept is graded in these qualifications from 1 to 3 points (●), where ● indicates low complexity, limited applicability and low desirability and where ● ● ● indicates (very) high complexity, high applicability and high desirability. Hopefully this allows future work to be prioritized accordingly.

6.1 Graphical user interface

6.1.1 User assisted dispersion estimation

Currently when dispersion is modeled in the data this has to be estimated by specifying a number of user estimated parameters. These parameters do not directly relate to the displayed data. However, after the analysis is done and the parameters are fitted to the data the dispersion curve is overlaid on top of the data. In the same way, in principle, it is also possible for the user to visually estimate the parameters of a polynomial function describing the dispersion by clicking on a few points in the data plot.

Complexity - ● – Applicability - ● ● – Desirability - ● ●

6.1.2 Visual compartmental modeling

Compartmental kinetic models are represented in Glotaran by a combination of a transfer matrix K, a vector representing the kinetic parameters (the rate constants), a vector representing the branching parameters and finally a vector of inputs (representing the IRF). Intuitively however, it makes more sense to represent a compartmental model as a target kinetic scheme, this is typically how biophysicists think. In Glotaran this could be (easily) represented using the NetBeans Visual Library (the same system in place to design the analysis scheme).

Complexity - ● ● – Applicability - ● ● ● – Desirability - ● ● ●

6.1.3 Integration with iReport

iReport is the free, open source report designer for JasperReports. It allows the user to create very sophisticated layouts containing charts, images, subreports, crosstabs and much more. Data can be accessed through a wide variety of method and support for Glotaran as a data source could be implemented. It would then be possible to

'compose' an analysis results report and publish it as PDF, RTF, XML, XLS, CSV, HTML, XHTML, text, DOCX, or OpenOffice.
Complexity - ● ● ● – Applicability - ● ● ● – Desirability - ● ● ●

6.2 Computational back-end

6.2.1 TIMP

With respect to the computational back end for Glotaran still a lot of work has to be done. As of version 1.10 of TIMP there are still some bugs in the analysis and model parsing, especially when analyzing multiple datasets with model differences between datasets. Before this can be properly functioning in Glotaran the underlying bugs first need to be fixed in TIMP.
In addition to this there is still a lot of room for faster computation within TIMP. Currently a lot of time is spent on overhead. Once this is fixed TIMP is expected to run an order of magnitude faster in some cases.
Complexity - ● ● ● – Applicability - ● ● ● – Desirability - ● ● ●

6.2.2 Interchangeability with other computational back end

The current computational back end for Glotaran is TIMP. However, conceivably the computational back end could be replaced by any other application that can do similar computations. The modular architecture of the NetBeans platform on which Glotaran has been built allows for this. One potential computational back end is the application TIM, which was discussed in chapter TIM. The other possibility is to perform some computations within Java to exploit the multi-threading nature and greater flexibility of Java as opposed to the single-threaded nature of R.
Complexity - ● ● ● – Applicability - ● ● – Desirability - ●

6.3 Application core

6.3.1 Integration with databases

One of the development criteria that was put forward for the development TIM was a graphical user interface supporting collaborative research and enabling distributed interactive modeling, where an expert in modeling and parameter estimation can analyze the data and an experimental scientist can contribute to the interactive modeling by discarding unrealistic models and suggesting model improvements. The first step for this was made in Glotaran by introducing the system of project-management, e.g. a project – consisting of datasets, models and analysis schema can easily be 'zipped up' and sent to a collaborator. The next step is the integration of Glotaran with a database system for easy models storage and potentially the storage of (selected) measurements as well. The NetBeans platform already offers the integration with databases 'for free' so all that is left is the implementation within Glotaran.
Complexity - ● ● – Applicability - ● ● ● – Desirability - ● ● ●

6.3.2 Systems biology

Systems biology is a rapidly growing inter-disciplinary field of science and becoming increasingly important. The Systems Biology Markup Language (SBML) has been developed for representing models of biological processes. The elementary events in (bio)photo-physics studied in the field of biophysics are not unrelated to processes studied in systems biology; a photo-physical event occurring in a protein membrane can be the start of a signal transduction pathway as studied in systems biology. However, as both processes occur on different scales, both spatially as well a temporally a bridge between the two does not yet exist. However, by implementing support for the SBML language and implementing some form of simple modeling the first step towards the combined analysis of elementary event in photo-physics and processes in computational systems biology can be made.

Complexity - ● ● ● – Applicability - ● – Desirability - ● ● ●

7 Appendix A: The NetBeans platform architecture

The NetBeans platform is a so called Rich Client Platform, providing a reliable and flexible application architecture.

Among the features of the platform are:

- User interface management (e.g. menus and toolbars)
- User settings management
- Storage management (saving and loading any kind of data)
- Window management (tabs, dockable windows)
- Wizard framework (supports step-by-step dialogs)
- NetBeans Visual Library (visualization of models)

All this functionality is practically 'free' and very little programming is required to make it work. Development on the application can take place within the NetBeans Integrated Development Environment (IDE), which integrates seamlessly with most versions control software allowing for easy collaboration between developers. The NetBeans IDE is also built on top of the NetBeans platform. Consequently all platform-related improvements automatically benefit all applications developed on top of the platform.

The NetBeans platform consists of a collection of modules. Each module provides some well defined functionality and contributes this functionality to the platform. This makes the platform highly extensible and makes it possible for third party developers to develop their own modules and integrate them into a RCP application.

Because Glotaran was built as a collection of modules on top the of NetBeans platform and because it is open-source under the terms of the GPLv2 license it provides the possibility for external developers to easily extend the functionality of Glotaran to include features Glotaran is lacking and if willing contribute these features back to the community as a whole. Appendix B goes into more detail regarding the possibility to contribute to Glotaran.

8 Appendix B: Contributing to Glotaran

There are many ways to contribute to Glotaran. The best way it to simply use it, and provide feedback by e-mail, phone or in person. Tell the developers what works well and what improvements could be made. Another way is to directly file bug reports if you find bugs or file a feature request if you feel something is really missing. It's possible to file bug reports or feature request in the Issue Tracker called Bugzilla[8]. Finally it's of course also possible to join the ranks of the developer yourself, check out the sources of the project and compile it yourself and start programming extension and fixing bugs. Below you can find more information on how this works.

Development requirements

- Java Development Kit 6 (JDK6), which can be downloaded from: https://jdk6.dev.java.net/
- NeBeans IDE 6.9 or higher, which can be downloaded from: http://netbeans.org/.
- A kenai.com or java.net account to register with the kenai.com project, located at: http://kenai.com/projects/glotaran/
- A computer with a modern CPU and at least 1GB of RAM is recommended.

Setting up the development enviroment

NetBeans 6.9 and up has built in support for Team Servers like the one from kenai at which the sources for Glotaran are hosted. Simply create an account with kenai.com, join the project and login to your accout from NetBeans. Then select 'Open Team Project' and search for glotaran and select “Open from Team Server”, see Figure 29.

8 Located at http://kenai.com/bugzilla/buglist.cgi?product=glotaran

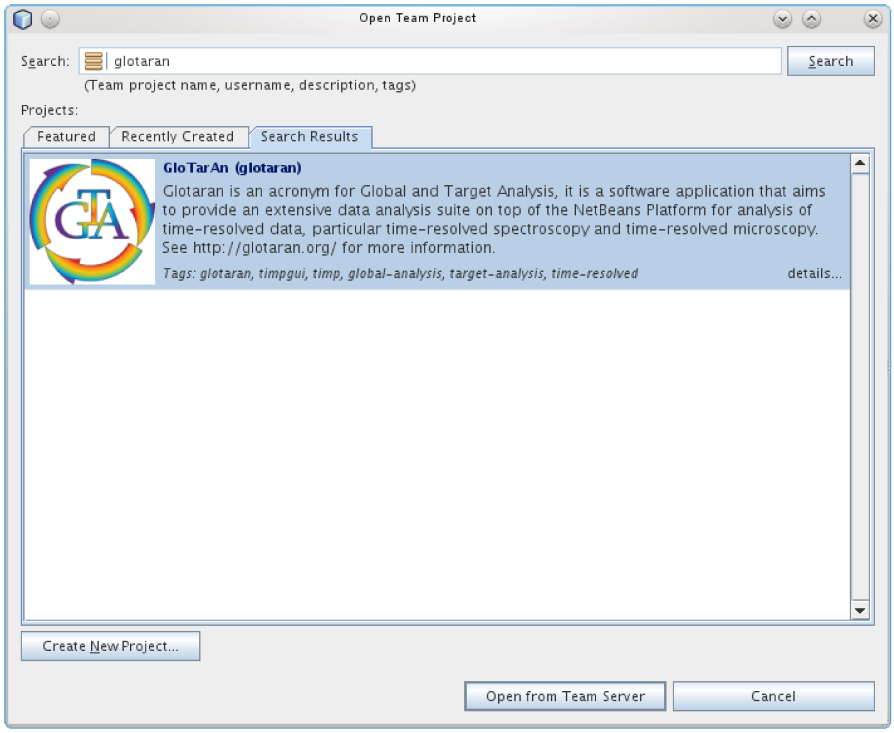


Figure 29: NetBeans Open Team Project wizard with Glotaran selected

From there on onward the project handles like any other NetBeans platform application. A few good starting points for working with the NetBeans platform are:

The book "The Definitive Guide to NetBeans™ Platform" by Heiko Boeck, ISBN13: 978-1-4302-2417-4.

The book "Rich Client Programming: Plugging into the NetBeans Platform" by Tim Boudreau, Jaroslav Tulach and Geertjan Wielenga, ISBN-13: 978-0132354806.

The official NetBeans platform website, located at:

http://netbeans.org/features/platform/

The blog of Geertjan Wielenga, located at:

http://blogs.sun.com/geertjan/

Support

Any questions can be directly addressed to the developers of Glotaran who can be contacted via the support mailing list support@glotaran.kenai.com

9 Appendix C: JFreeChart library

Introduction

JFreeChart[9] is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

- a consistent and well-documented API, supporting a wide range of chart types;
- a flexible design that is easy to extend, and targets both server-side and client-side applications;
- support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);
- JFreeChart is open source or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

LinLogAxis an extension to JFreeChart

JFreeChart is an extensive library with support for a great number of different types of plots. On the project website a list of examples is give. However, one type of plot frequently used in the visualization of time-resolved spectroscopy data, the linear-logarithmic plot is not included in the standard set of plots. Because this type of plot is essential to the working of a global and target analysis tool it was developed by the author of this work as part of the project. The results is the Java class LinLogAxis and implementation of the standard JFreeChart class org.jfree.chart.axis.ValueAxis. Because the LinLog functionality was implemented as a ValueAxis it integrates seamlessly with the standard JFreeChart chart library and can be reused with any XY Plot. The improvement for certain types of data is impressive when compared side by side, as demonstrated in Figure 30 and Figure 31.

9 More information available at: http://www.jfree.org/jfreechart/

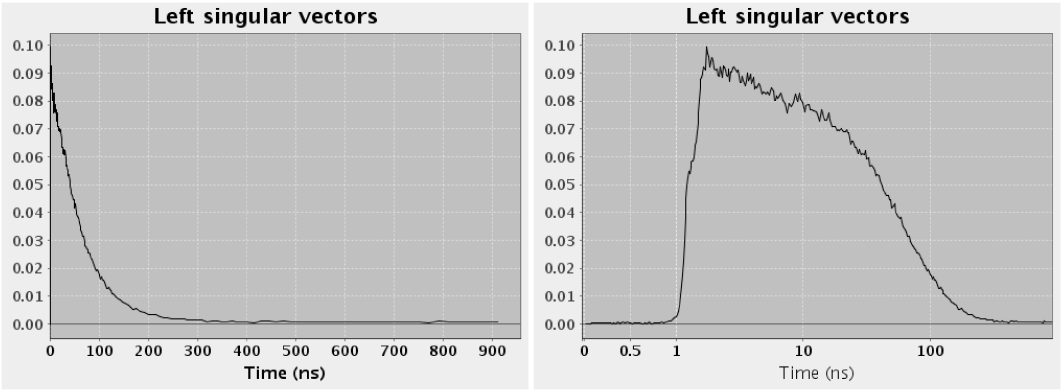


Figure 30: Comparison of the standard JFreeChart NumberAxis vs LinLogAxis

The left plot in Figure 30 uses the standard linear NumberAxis only to display a time trace, and the right plots uses the new LinLog axis instead. Figure 31 displays the estimated concentration profiles of a sequential model with lifetimes ranging from 1.76 till 54 ps. The additional information that the use of the LinLogAxis provides in these cases should be obvious.

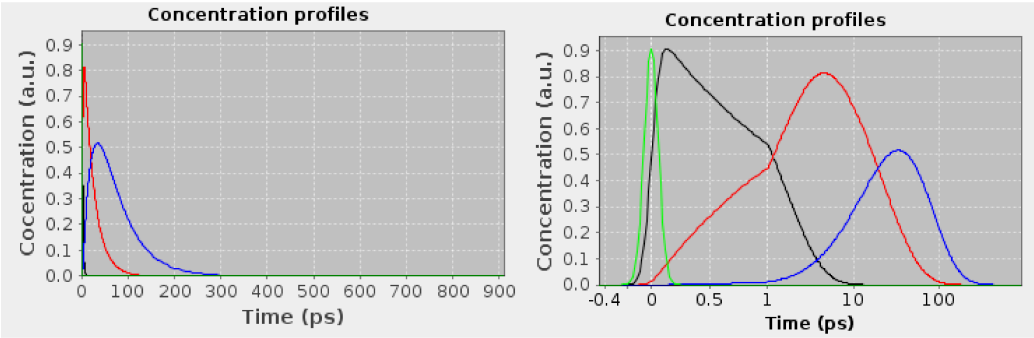


Figure 31: Two charts showing the estimated concentration profiles of a sequential model, the left chart uses the standard linear axis and the right charts uses the linear-logarithmic axis. Lifetimes: 1.76 ps (black), 18.7 ps (red), and 54 ps (blue). Green represents the shape of the coherent artifact, and is equal to the IRF.

10 Appendix D: Glotaran screencast

Hello and welcome to this screencast demonstrating the workings and some of the features of Glotaran.

Glotaran is an open-source and cross-platform application designed to facilitate simplified global and target analysis of spectroscopy and microscopy data from experiments particularly in the field of biophysics. In this screencast we will work through a simple analysis case-study of a dataset from an experiment by Hippius et. al. In this experiment energy transfer is observed from an artificial photosynthetic system.

[...]

The very first time you start Glotaran your window will look something like this.

Creating a new project
The first step we take is creating a new project, by selecting from the menu: File - New Project (CTRL + SHIFT + N). Alternatively we can press the “New Project” button.

In the New Project Wizzard that follows select the *General* category select '*Empty Project*'. Press next and specify a name, for instance 'Demo Project'. Press '*Finish*' to create the project.

The newly created project directly opens in the project tab and a folder structure is automatically created.

Adding a dataset to the project
Before we can start analysis we first need to import the data to analyze. We do this be right clicking on the *Datasets* folder and selecting 'Open dataset file'. Then we locate the spectroscopy data file from the experiment. The file-type is an ASCII formatted TIMP spectroscopy data file.

If the file is recognized by Glotaran a new data file node is created under the dataset folder with the file name of the added dataset. Now double clicking this node opens the data editor for this particular type of data.

Inspecting the data and creating a new dataset for analysis
The data editor allows us to inspect the spectroscopy data in various ways. To start we are presented with a general overview of the entire dataset. Here wavelength is displayed on the horizontal axis and time is displayed on the vertical axis. The intensity of the measured spectrum is represented by the color coding. Controlling the sliders allows us to looks at individual traces in the dataset. For instance we can look at the suspected moment of time-zero, which is the time where the instrument response function peaks. Zooming in allows us to take a closer look at a particular region of the data. In addition this allows us to select a subset of the data for analysis.

In order to create a dataset for analysis we select a region and then press 'Create

Dataset'. We specify a name for the dataset and press 'ok'. Now dataset node has appeared under the data-file node in the project management. This is the type of dataset we can later use for data-analysis.

Starting a new analysis

In order to start analysis of the dataset we first need to create a new analysis scheme. We do this by right clicking the Analysis Schema folder in our project and selecting 'New → New Analysis Schema'. We specify a file name and press 'Finish' and the schema file is created as a sub-node under the analysis folder and opens automatically. This is now our analysis canvas.

First we want to add a 'Dataset Container' to hold the datasets in the analysis. Once added we can drag and drop the dataset we created earlier into the container.

The next step is to add a model to apply to the dataset container. We can do this by dragging the 'Model ' onto the canvas and specifying a new name. A model file will automatically be created in the Models folder. The model is currently empty but we can drag and drop nodes from the 'Modelling' category on the Palette to specify our model.

Available are: Kinetic Parameters, Dispersion, Coherent Artifact, Instrument response function, Weighting and finally the (advanced) option of specifying the entire kinetic-matrix ourselves. Once a node is dragged onto the model we can change its properties in the Properties window.

Now, before we proceed we will take a small intermezzo to take another close look at our dataset in order to postulate a model that can describe the data.

Intermezzo: re-examining the dataset

The goal of this intermezzo is to take another look at the dataset and look for clues as to what the underlying kinetic model could be.

One of the first things we notice from the overview is dispersion: time-zero is dependent on the wavelength. Dispersion can be parametrized in our model, for this we need to determine the central wavelength and – depending on the order of the dispersion modeled – a number of starting values. In this case we see quadratic dispersion so we need 3 starting values that we can roughly estimate from the data.

Another important aspect of the analysis is to model the Instrument Response Function (IRF). Typically this is modeled by a Gaussian shape with a specific position and width. Again we can roughly estimate these values from the data.

Intermezzo: singular value decomposition

The next step is to estimate the number of components we want to attempt to resolve. A very useful diagnostic tool for this is the singular value decomposition or SVD. Switching to the SVD tab present us with a screeplot with the singular values, as well as a plot for the left and right singular vectors.

The screeplot can help to estimate the number of spectral components hidden in the data. In this case it looks like there is room for 3 to 4 components. To test this, the

number of displayed singular values can be increased to inspect the singular vectors. In this case it looks like 1, 2, 3, 4 … 3 is the optimal number of components as the 4th singular vector seems to be rather noisy.

For every component we need to specify a starting value. In this case we will simply pick some typical starting values for the decay rates for this particular kind of data. Alternatively a well educated guess of the decay rates of the right order of magnitude often leads to convergence to the right values.

Back to the analysis

Going back to the analysis schema we fill in the parameters we just investigated:

We add 'Kinetic Parameters' from the palette, change the number of components to 3, and expand the 'KinPar' node. Typical starting values for the decay rates are 0.5, 0.02 and 0.001.

Another aspect we can control is whether we want to enforce a parallel or sequentially decaying scheme. In this case we postulate a sequential scheme with increasing lifetimes. Furthermore we restrict the rates to be positive.

Next we add the IRF Parameters node from the palette. By default the Gaussian IRF type is selected for which we need to specify two parameters: position and width. The values can be roughly estimated from the data.

Next up we specify the dispersion. We change the order of the polynomial to 3 in order to specify quadratic dispersion as could clearly be seen from the data. Again the parameters can be guessed from the data.

Finally we would also like to model the coherent artifact. For now only a coherent artifact modeled after the Instrument Response function is supported, so we will enable that.

Connecting the model

Now that the model specification is finished we need to link our model to our dataset container. This means that the model is applied to all datasets in the dataset container. In this case just our one dataset.

Before we can start our analysis we still need to connect our 'Dataset Container' to an 'Output Container' where we can specify the number of Iterations and the Output Location for the results of the analysis. In addition we can toggle the calculation of errors to show the errors in the resulting spectra.

Now we wait …

… after a few seconds to minutes, depending on the size of the data and the complexity of the model a new folder with results will appear in the Results folder.

Inspection of results

The first file we can take a look at is the summary file. It provides us with a textual summary of the analysis process. Given is: the number of iterations the analysis was run for, the calls that were made to R in order to perform that analysis, the final

residual standard error and values of the fitted parameters with their standard errors. If the analysis failed, for instance because the estimated parameters were too far off from the real values this file will contain a diagnostic error message rather than the parameters.

The next file we can look at is the results-file named after that dataset. Every analyzed dataset will have a results file so if you analyze multiple datasets you will see multiple files appear.

The first tab in the results view provides us with an overview of the data. Here we see the Evolution Associated Spectra, the Decay Associated Spectra and the normalized versions of both as well as the corresponding time traces. In addition a short summary of the final values of the estimated parameters is provided. On the bottom the singular value decomposition of the residual data matrix is depicted.

This first indication of the quality of the fit comes from this SVD of the residual matrix. If the singular vectors are clearly structured that could indicate an extra component would be needed. In this case we can clearly see the effect of the degrading quality of the data at the edges of the measured spectrum. Something that could be solved by preprocessing the data or applying weights.

Some of the plots are hard to analyze because of the compression of the data in the earlier time-points, even if you zoom in. For this the plots have a helpful property that allows them to change to a linear-logarithmic representation of the data. Activating this feature changes the time axis to be linear around time zero, and logarithmic otherwise. This representation of the data gives us much more information regarding the earlier time-points in the data, where most of the dynamics takes place.

Another way to investigate the quality is to switch to the second tab … and look at the estimated dispersion curve overlaid on the data. Clearly there is still some room for improvement here.

By scrolling the sliders along the plot we can actually inspect the fit to the data at every timepoint and wavelength. … Naturally it's possible to also zoom in even more closely …. and again we can activate the LinLog functionality to get a better view of the earlier time-points.

In this tab the singular value decomposition of the residuals matrix is again displayed, however in this tab, when we zoom in on a particular region of the data we can actually look at the SVD for just that part.

Finally we can take snap shots of our analysis in the form of selected traces. This can be done automatically, where we simply need to specify how many traces we would like to see in each dimension. Or we can select the trace at where the sliders are currently located and add them to the list. The selected traces can then be viewed on the next few tabs.

The last file we can look at is the _results.xml file. This file presents an overview into the progression of the residual sum of squares (RSS) vs. the number of iteration. If the

values of the estimated parameters converge this is represented by a flattening of the curve. If there is no convergence, or the plot fluctuates this could be an indication to try and run the analysis for more iterations or to choose different starting values alltogether.

The next step would be to take the results of this analysis and try to give meaning and physical interpretation to the results, but that is something for another video ...

With that we conclude this screencast. I hope you enjoyed it and are now able to use Glotaran for your own data analysis. If you find any bugs, or you find you are missing certain features, please contact us and we will get on it as soon as possible.

Acknowledgments

First and foremost I would like to thank my supervisor Ivo van Stokkum for his support, academically as well as personally, and for his seemingly endless patience. Without his advise and continued support this thesis would never have been finished. I would like to thank Erwin Peterman for reviewing te thesis and for helpful suggestion on improving the text.
Secondly I would like to thank Sergey Laptenok, who has been working with me on the development of Glotaran on and off for many months already, and for hopefully many more to come. In this context, I would also like to thank Volha Chukhutsina who worked for some time with me to implement some of the simulation capabilities of Glotaran.
Last but not least I would like to thank Katharine Mullen who developed TIMP, the computational engine for Glotaran and whose dissertation provided much valuable insights into modeling spectroscopy data. Her work made the development of Glotaran a lot easier.

Bibliography

1. Lakowicz, J. Introduction to Fluorescence. *Principles of fluorescence spectroscopy* 6 (2006).
2. van Stokkum, I.H.M. & Bal, H.E. A Problem Solving Environment for interactive modelling of multiway data. *Concurrency Computat.: Pract. Exper.* **18**, 263-269 (2006).
3. van Wilderen, L.J.G.W. et al. Ultrafast infrared spectroscopy reveals a key step for successful entry into the photocycle for photoactive yellow protein. *Proceedings of the National Academy of Sciences* **103**, 15050-15055 (2006).
4. van Stokkum, I., Larsen, D.S. & van Grondelle, R. Global and target analysis of time-resolved spectra. *Biochim. Biophys. Acta* **1657**, 82-104 (2004).
5. Sytina, O.A. et al. Conformational changes in an ultrafast light-driven enzyme determine catalytic activity. *Nature* **456**, 1001-1004 (2008).
6. *IMSL Numerical Libraries - Visual Numerics is now Rogue Wave*. http://www.vni.com/products/imsl/
7. NWO - A PSE for hierarchical modelling of time-resolved spectra. http://nwo.nl/projecten.nsf/pages/1500108794
8. Mullen, K.M. & van Stokkum, I.H.M. TIMP: an R package for modeling multi-way spectroscopic measurements. *Journal of Statistical Software* **18**, (2007).
9. Team, R.D.C. *R: A Language and Environment for Statistical Computing*. (Vienna, Austria, 2010).http://www.R-project.org
10. Mullen, K.M. Separable nonlinear models: theory, implementation and applications in physics and chemistry. (2008).http://hdl.handle.net/1871/12926
11. Laptenok, S.P. et al. Global analysis of Förster resonance energy transfer in live cells measured by fluorescence lifetime imaging microscopy exploiting the rise time of acceptor fluorescence. *Phys. Chem. Chem. Phys.* **12**, 7593 (2010).
12. Krumova, S. et al. Monitoring Photosynthesis in Individual Cells of Synechocystis sp. PCC 6803 on a Picosecond Timescale. *Biophysical Journal* **99**, 2006-2015 (2010).
13. Laptenok, S. et al. Fluorescence Lifetime Imaging Microscopy (FLIM) Data Analysis with TIMP. *Journal of Statistical Software* **18**, 1–20 (2007).
14. TIMP: An R package for modeling multi-way spectroscopy data. http://timp.r-forge.r-project.org/
15. Joris J. Snellenburg TIMPGUI: A graphical user interface for the package TIMP. (2008).http://www.statistik.uni-dortmund.de/useR-2008/
16. *Applied Photophysics: Pro-Kineticist II - 2nd Order Global Analysis*. (Applied Photophysics:).http://www.photophysics.com/prokii.php
17. Scientific Software Technologies Center, Belarusian State University *FFS Data Processor; TRFD Fitting Software; TRFA Data Processor; BIFL Data Analyser*. (Scientific Software Technologies Center, Belarusian State University: Minsk, Belarus,).http://sstcenter.com/software/index.phtml?pageid=idProducts
18. Jobin Yvon *Fluorescence Decay Analysis Software | DAS6*. (HORIBA, Ltd:).http://www.horiba.com/scientific/products/fluorescence-spectroscopy/software/das6/das6-838/
19. Evaldas Katilius *ASUFIT*. http://www.public.asu.edu/~laserweb/asufit/asufit.html
20. Golub, G.H. & LeVeque, R.J. Extensions and Uses of the Variable Projection Algorithm for Solving Nonlinear Least Squares Problems. *Proc. Army Num. Anal. Comp. Conf. ARO Report* **79-3**, 1–12 (1979).

21. *VARPRO*. http://www.netlib.org/opt/varpro
22. Komura, M. & Itoh, S. Fluorescence measurement by a streak camera in a single-photon-counting mode. *Photosyn. Res* **101**, 119-133 (2009).
23. van Stokkum, I.H.M. et al. (Sub)-picosecond spectral evolution of fluorescence in photoactive proteins studied with a synchroscan streak camera system. *Photochem. Photobiol* **82**, 380-388 (2006).
24. van Stokkum, I., van Oort, B., van Mourik, F., Gobets, B. & van Amerongen, H. Femtosecond Time-Resolved Infrared Spectrocopy. *Biophysical Techniques in Photosynthesis* **26**, 233-240 (2008).
25. van Amerongen, H. & van Grondelle, R. Transient absorption spectroscopy in study of processes and dynamics in biology. *Meth. Enzymol* **246**, 201-226 (1995).
26. Chang, C., Sud, D. & Mycek, M. Fluorescence lifetime imaging microscopy. *Methods Cell Biol* **81**, 495-524 (2007).
27. Lakowicz, J. Time-correlated single-photon counting. *Principles of fluorescence spectroscopy* 103-107 (2006).
28. Mullen, K.M., Vengris, M. & van Stokkum, I.H.M. Algorithms for separable nonlinear least squares with application to modelling time-resolved spectra. *J Glob Optim* **38**, 201-213 (2007).
29. Bates, D.M. *Nonlinear regression analysis and its applications*. (Wiley: New York, 1988).
30. Golub, G.H. & Pereyra, V. The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate. *SIAM Journal on Numerical Analysis* **10**, pp. 413-432 (1973).
31. Mullen, K.M. & van Stokkum, I.H.M. The variable projection algorithm in time-resolved spectroscopy, microscopy and mass spectrometry applications. *Numer Algor* **51**, 319-340 (2008).
32. van Stokkum, I.H.M. Global and target analysis of time-resolved spectra. (2005).http://www.nat.vu.nl/~ivo/lecturenotes.pdf
33. Urbanek, S. Rserve - A Fast Way to Provide R Functionality to Applications. *Proc. of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)* (2003).http://rosuda.org/Rserve
34. Laptenok, S. New possibilities for interactive specification and validation of models for Fluorescence Lifetime Imaging Microscopy (FLIM) data with the TIMP package. (2008).http://www.r-project.org/conferences/useR-2008/abstracts/Laptenok+Mullen+Borst+van_Amerongen.pdf
35. Hippius, C., van Stokkum, I., Zangrando, E., Williams, R. & Würthner, F. Excited State Interactions in Calix[4]arene-Perylene Bisimide Dye Conjugates: Global and Target Analysis of Supramolecular Building Blocks. *J. Phys. Chem. C* **111**, 13988-13996 (2007).
36. Dietzek, B., Pascher, T., Sundström, V. & Yartsev, A. Appearance of coherent artifact signals in femtosecond transient absorption spectroscopy in dependence on detector design. *Laser Phys. Lett.* **4**, 38-43 (2007).

Notes

Back Cover