Technische Universität München
Fakultät für Physik

Abschlussarbeit im Bachelorstudiengang Physik

Assessment of a Variable Projection Algorithm for Trace Gas Retrieval in the Short-Wave Infrared

Adelina Bärligea

22. Juli 2022

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Methodik der Fernerkundung
Atmosphärenprozessoren

Erstgutachter: Prof. Dr. Norbert Kaiser

Contents

Abstract v

1 Introduction 1

2 Atmospheric Remote Sensing 3
2.1 Radiative Transfer Model 4
2.2 Space-borne Measurements 7
2.3 Trace Gas Retrieval 11

3 Least Squares Approximation 15
3.1 Linear Least Squares 15
3.2 Nonlinear Least Squares 18
3.3 A Survey of Iterative Methods 21

4 Variable Projection Method 25
4.1 Separable Problems 25
4.2 Theoretical Background 26
4.3 Algorithm Outline 29
4.4 Implementation 31

5 Testing and Comparison 35
5.1 Nonlinear Solvers 36
5.2 Initial Guesses 40
5.3 Bound-Constraints 41

6 Conclusions and Outlook 45

Acknowledgements 49

Bibliography 51

Abstract

An important part of atmospheric remote sensing is the monitoring of its composition, which can be retrieved from radiance measurements, e.g. in the short-wave infrared (SWIR). For deriving trace gas concentrations in the SWIR spectral region a radiative transfer model is fitted to observations by least squares optimization. The aim of this thesis is to present the well-established variable projection method for solving separable nonlinear least squares problems and to examine and configure it for trace gas retrieval. For this, a *Python* implementation of the algorithm, called *varpro.py*, will be outlined and later utilized in retrievals with real satellite observations. These are meant to assess the efficiency, accuracy and robustness of three iterative algorithms for nonlinear least squares problems which have been built into *varpro.py*. Furthermore, a new feature — applying bounds to the nonlinear fit parameters — will be included in the implementation and evaluated for its quality and usefulness. As a result of these tests, a new 'default' configuration will be suggested based on the algorithm with the best performance for trace gas retrieval. Also, ideas for analysing and testing strategies which could lead to even more insights will be proposed. Finally, possible future applications for trace gas retrieval will be motivated and suggestions for further research and modifications of *varpro.py* will be made.

Chapter 1

Introduction

The measurement of geophysical quantities is of central importance for improving the understanding of the atmosphere and environment including phenomena like air pollution, ozone depletion or climate change. Especially for the latter, it is essential to monitor the concentrations of important greenhouse gases, such as carbon dioxide (CO_2) and methane (CH_4) on a global scale in order to quantify their impact on the earth's climate. These can be derived from satellite measurements of the incoming radiation. The resulting process is called trace gas retrieval which will be introduced in Chapter 2.

One essentially needs three things for it: a radiative transfer model, space-borne measurements and a retrieval method. As the required precision for retrieval products increases, all of these are subject of ongoing research and improvements. A radiation transfer model, for instance, can be formulated in many ways and considerably modified in different wavelength regions. In this thesis, the line-by-line code, Py4CAtS (PYthon for Computational ATmospheric Spectroscopy) [1], is utilized for this purpose. Regarding space-borne measurements, the Orbiting Carbon Observatory (OCO-2) [2] is introduced as an example in Chapter 2. It observes CO_2 in the short-wave infrared (SWIR) with an almost unrivalled high precision and coverage compared to satellites with the same goal, which is why its measurements are used for the retrievals in this thesis.

Instead of focusing on models or measurements, this thesis deals with the inversion method, which is the process of using both to retrieve the underlying atmospheric parameters, such as trace gas concentrations. For this, a fit between the model and observation has to be performed. Such an approximation is usually based on least squares optimization which will be motivated and described in Chapter 3. As the corresponding problems may not only differ heavily but also rise in complexity, countless variations of least squares algorithms have been developed and extended over the past decades (see [3] as an example). One of them is the Variable Projection algorithm introduced by Golub and Pereyra [4] in 1973, which is specialized for nonlinear problems that are separable. A thorough derivation of its theory, including how separability can be exploited to reduce the size of the problem and make the fit more efficient, can be found in Chapter 4. It is also reasoned why this algorithm can and should be applied to the mathematical problem posed by trace gas retrieval.

The idea of this thesis is to motivate the replacement of the currently used standard least squares solver by a more suitable and efficient one and to assess its properties. An implementation of the Variable Projection method based on [5], called *varpro.py*, was used for this. The goal of the conducted tests was to find the best configuration for its use and to explore new features. Chapter 5 holds the results of this assessment.

Ultimately, the purpose of this thesis was to lay the foundation for the establishment of a more suitable method for trace gas retrieval. It is argued in Chapter 6 that this could result in a considerable improvement of the inversion process (see Section 2.3) in the future. After all, such advancements are essential for the ongoing developments in atmospheric science and research.

Chapter 2

Atmospheric Remote Sensing

In order to monitor the state of the atmosphere on a global scale it is necessary to observe the earth from a far distance. These observations are accomplished by space-borne sensors on satellites that measure incoming radiation. Most instruments are passive, which means that they do not have their own source of radiation but instead use emission by the sun or thermal emission of the atmosphere itself. The characteristics of such measurements and exemplary satellites will be introduced in Section 2.2.

The principle of passive remote sensing is simple: light (or more generally: radiation) traversing the atmosphere is to some extent absorbed and scattered by molecules or other particles like clouds or aerosols before it reaches an observing satellite. The resulting change in measured intensity can thus be used to determine the atmospheric state including its composition [6, chapter 2]. For a quantification of parameters like the amount of trace gases present, it is usually necessary to have a **radiation transfer model** (see Section 2.1) which is able to mimic the observed radiance spectra. By comparing the measured signals to the simulated ones with approximative methods (see chapters 3 and 4), the sought atmospheric parameters can be determined.

The work for this thesis is based on of many areas of remote sensing: a process called **trace gas retrieval** (see Section 2.3), which has the goal to estimate trace gas concentrations in the atmosphere. It is just one of many areas of remote sensing. The applicability of this method depends very strongly on the wavelength region chosen for the molecule under scrutiny.

In this thesis the near and short-wave infrared (NIR and SWIR) regions are dealt with, which approximately reach from 14 000 cm^{-1} to 4000 cm^{-1} (0.8 µm to 2.5 µm)[1]. This spectral range allows the precise monitoring of important greenhouse gases such as CH_4 and CO_2 which play a significant role in global warming. In it, thermal emission is of little relevance and scattering is comparatively weak (except for heavy

[1] For the infrared spectral region it is advantageous to use the more natural units of cm^{-1} corresponding to the wavenumber ν which is the inverse of the wavelength λ. It can be calculated as $\nu[\mathrm{cm}^{-1}] = 10000/\lambda[\mu\mathrm{m}]$

aerosols loading or cloudy conditions). As a result, the light path can be assumed to be straight from the sun to the earth's surface and back to the satellite [7]. This considerably simplifies the model which will be described in the following section.

2.1 Radiative Transfer Model

In order to retrieve atmospheric state parameters (i.e. trace gas concentrations, temperature, or aerosol and cloud parameters), it is necessary to understand the sources of electromagnetic radiation and its transport through the atmosphere. According to Burrows et al. [6, chapter 1] there are three main sources of radiation in the atmosphere: external radiation from the sun, thermal emission by particles in the atmosphere or from the ground itself, and scattering, i.e. the radiation scattered into and out of the primary beam. On its way though the atmosphere this radiation is attenuated by processes of absorption and scattering (denoted by the subscripts *a* and *s*), which are commonly referred to as the extinction [6, chapter 1]. Combining all those processes, one can describe the change of intensity of the incoming radiation I at a certain wavenumber ν traversing the distance ds with the **radiation transfer equation** for a single species as [6, chapter 1]

$$\begin{aligned} \frac{\mathrm{d}I(\nu,\Omega)}{\mathrm{d}s} \quad = \quad & -\underbrace{\left[\epsilon_a(\nu)+\epsilon_s(\nu)\right]\, I(\nu,\Omega)}_{\text{attenuation}} \quad + \underbrace{\epsilon_a(\nu)\, B(\nu,T)}_{\text{thermal emission}} \\ & + \underbrace{\epsilon_s(\nu) \int \frac{S(\Omega',\Omega)}{4\pi}\, I(\nu,\Omega')\,\mathrm{d}\Omega'}_{\text{scattering}} . \end{aligned} \tag{2.1}$$

The parameters in equation (2.1) are:

$\epsilon_{a,s}(\nu) = n \cdot \sigma_{a,s}(\nu)$: absorption/scattering coefficient of particle

n : number density of absorbing/scattering particle

$\sigma_{a,s}(\nu)$: absorption/scattering cross section of particle

$B(\nu,T)$: Planck's function of black body radiation at temperature T

$S(\Omega',\Omega)$: dimensionless scattering phase function

As mentioned, an advantage of the NIR and SWIR spectral regions is that they allow to neglect thermal emissions and molecular (Rayleigh) scattering [6, chapter 2]. If only cloud-free measurement scenes are considered, the last term in (2.1) for scattering on clouds can be dropped, as well. This considerably simplifies[2] equation (2.1) to

$$\frac{\mathrm{d}I(\nu)}{\mathrm{d}s} = -n \cdot \sigma(\nu) \cdot I(\nu) \tag{2.2}$$

which is called the **Beer-Lambert law**. Thus, for the most simple solution of the radiation transport equation, the observed spectral signatures are directly related to the absorption spectra of atmospheric constituents. In the integrated form and for several absorbing molecular species[3] this means

$$I(\nu) = r\, I_{\text{sun}}(\nu)\, \exp\left[-\int\limits_{\text{path}} \left(\sum_m n_m(s)\, \sigma(\nu)\right) \mathrm{d}s\right] \tag{2.3}$$

which serves as a model for the intensity of the sun's radiation after having taken a path through the earth's atmosphere. The dependence of the cross section σ on the path will be explained later in this section. The factor r accounts for the surface reflectivity, often called albedo. It is defined by the ratio of reflected to transmitted radiance and usually depends on the wavenumber ν. However, this dependence differs strongly over the globe [6, chapter 2], which is why the ν-dependent function r must be fitted for each measurement.

For the purpose of this thesis, a code called Py4CAtS (PYthon for Computational ATmospheric Spectroscopy) [1] was used to model the radiation transfer described in (2.3). Py4CAtS is a *Python* re-implementation of an earlier *Fortran* code called GARLIC (Generic Atmospheric Radiation Line-by-line Infrared Code) [8], which — as the name suggest — is a line-by-line model. The strategy of such models is to start by computing the absorption cross section of a particular molecule m for a given pressure p and temperature T by summing up the contributions from many lines l (cf. *line-by-line*) as

$$\sigma_m(\nu, p, T) = \sum_l S_{ml}(T)\; g(\nu; \hat{\nu}_{ml}, \gamma_{ml}(p, T)), \tag{2.4}$$

where each line is characterized by its temperature-dependent strength S_{ml} and a normalized shape function g describing the broadening of the line [1]. The combined effect of pressure and Doppler broadening is represented by a Voigt line profile

[2] Since scattering is neglected only the absorption components of equation (2.1) are considered and therefore the subscripts a are dropped in this and later equations. This fact cancels out the only Ω-dependent part of the equation, namely the scattering phase function $S(\Omega', \Omega)$, which is why this dependency is also dropped in later expressions.

[3] The only change when considering several molecules appears in the absorption coefficient ϵ_a which turns into a sum of all molecular (absorption) cross sections multiplied by their number density.

[6, chapter 1] at a position $\hat{\nu}_{ml}$ with the broadening parameters γ_{ml}. The necessary molecular properties can be picked out of comprehensive line databases that list line parameters "for millions to billions of lines of some dozen molecules" [1]. The reason why the cross section in equation (2.3) is indirectly dependent on the path length s, is because p and T vary with it. Their atmospheric height profiles differ across the globe which is why such temperature and pressure profiles are necessary input data for a radiation transport simulation.

The next step in line-by-line codes, such as Py4CAtS, is to compute the absorption coefficients ϵ_m and with them the **total optical depth** τ_m, which is defined as

$$\tau_m(\nu) = \int\limits_{\text{path}} \epsilon_m(\nu, s, p, T)\, \mathrm{d}s = \int\limits_{\text{path}} n_m(s)\, \sigma_m(\nu, p(s), T(s))\, \mathrm{d}s. \tag{2.5}$$

This is only possible with apriori knowledge about the number density n_m of each molecule in the atmosphere on the specific path. Considering the geometry of the problem (see the double slant path in figure 2.1) and assuming a plane parallel atmosphere, one can split the integral (2.5) into two separate paths, parameterized by s' and s'', which are uniquely related to the altitude z by

$$s' = \frac{z}{\cos(\theta_{\text{vza}})} \equiv \frac{z}{\mu} \qquad s'' = \frac{z}{\cos(\theta_{\text{sza}})} \equiv \frac{z}{\mu_\odot} \qquad \begin{array}{l}\text{vza = viewing zenith angle}\\ \text{sza = solar zenith angle}\end{array} \;. \tag{2.6}$$

As most atmospheric data (such as pressure and temperature profiles) are specified in terms of altitude, this relation is necessary to rewrite the from (2.5) to

$$\tau_m(\nu) = \int\limits_{z_{\text{srf}}}^{z_{\text{toa}}} \left(\frac{1}{\mu} + \frac{1}{\mu_\odot}\right) \epsilon_m(\nu, z, p, T)\, \mathrm{d}z. \tag{2.7}$$

Here, the integral limits have been shifted to only include parts from the actual surface height (srf) to the top of the atmosphere[4] (toa), because all relevant molecular absorption processes take place within that range.

To model an actual observation $\hat{I}$ the incoming radiation I has to be convoluted with the instrumental spectral response S, which accounts for the fact that the actual signals are not fully resolved in a sensor. Including all mentioned considerations (i.e. optical depths, solar and viewing geometry, instrument resolution, etc.) a radiance measurement can be simulated by

$$\hat{I}(\nu) \;=\; r(\nu) \cdot \mu_\odot \cdot I_{\text{sun}}(\nu) \cdot \exp\left(-\sum_m \tau_m(\nu)\right) \quad \otimes \quad S(\nu) \tag{2.8}$$

[4] For the application of trace gas retrieval z_{toa} is usually taken to be between 80 km and 100 km.

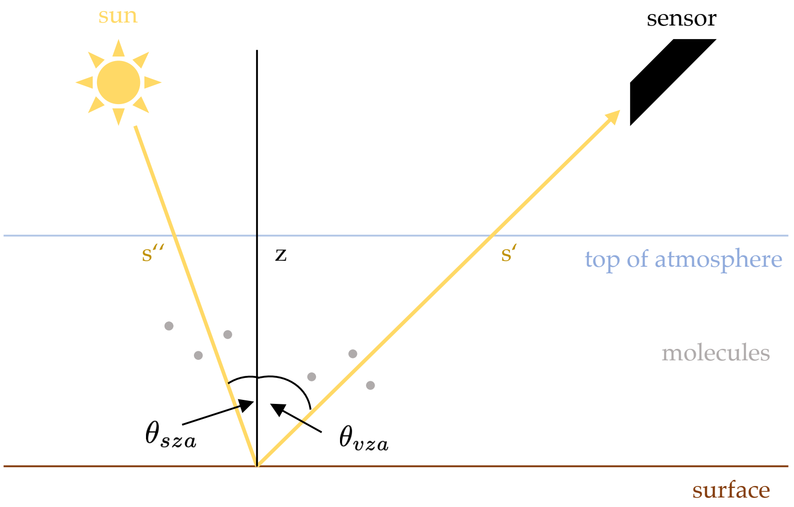

Figure 2.1: Schematic illustration of radiation transfer in passive remote sensing.

which forms the foundation of the **forward model** in a SWIR trace gas retrieval algorithm. The incident radiation of the sun $I_{\text{sun}}(\nu)$ that reaches the earth can be well approximated with Planck's law for black-body radiation using its effective temperature of 5778 K. The factor $\mu_\odot$ accounts for the fact that the solar radiation does not reach the earth's surface perpendicularly, but under an angle θ_{sza} (see figure 2.1), so that the radiant flux per area is reduced by a factor $\cos\theta_{\text{sza}}$, which is $\mu_\odot$. Most surface types (e.g. sand, snow, ice) are well described by the approximation of Lambertian reflection characteristics, which assume isotropic distribution of reflected light [6, chapter 2]. Therefore, the surface reflectivity $r(\nu)$ can be assumed to depend only on the wavenumber ν and not on the geometry of the problem. In fact, it can be fitted by a polynomial[5] in ν (see beginning of Section 2.3). The exponential term in equation (2.12) is called the transmission $\mathcal{T}$, which is defined as the ratio of incoming and outgoing radiation.

2.2 Space-borne Measurements

There are several earth observing satellites operating in various spectral regions and thereby monitoring different trace gas species. In this thesis measurements from the

[5] Usually, in order to avoid over-fitting this is taken to be first- or second-degree. For a short analysis of problem conditioning in terms of finding the right amount of fitting parameters in SWIR trace gas retrieval, see [9, chapter 4].

OCO-2 (Orbiting Carbon Observatory-2) satellite are utilised in later calculations (see Chapter 5) which is why it is used as an example in this section. The OCO-2 satellite [2] was launched on July 2, 2014 and was NASA's first spacecraft dedicated to studying atmospheric CO_2 from space. Its mission statement [2] is the following:

> The OCO-2 Project primary science objective is to collect the first space-based measurements of atmospheric carbon dioxide with the precision, resolution, and coverage needed to characterize its sources and sinks and quantify their variability over the seasonal cycle.

A disadvantage of space-borne measurements used to be their coarse spatial resolution which is not fine enough to resolve individual power plants or even large cities in detail [7]. This could be crucial, however, to understand the atmospheric behaviour of important green house gases like CO_2. This is why OCO-2 among other measures significantly improved spatial resolution compared to other operating instruments. Another reason why densely-spaced samples are advantageous is that it increases the number of measurements that are not impaired by the presence of clouds or uneven terrain such as mountains. Clouds and optically thick aerosols can shield the atmosphere below them from the satellite view while topographic variations can also block the light and therefore cause incomplete measurements [7].

OCO-2 flies in a sun-synchronous orbit, which means that it samples each location at a constant local time [see 2, for Mission Overview]. Each sample covers an area of about $3\,\mathrm{km}^2$ when the instrument is looking straight down. This viewing geometry is called *nadir*[6] and it usually also means viewing directions up to 60° around the zenith (for a schematic illustration see figure 2.1). This way, the instrument can gather 72,000 soundings on the sunlit side of any orbit [see 10, for further specifications] which ensures quite a good number of high quality measurements. The OCO-2 instrument has three specific spectral bands in the NIR and SWIR where it measures intensity at the same location on the earth's surface simultaneously. These regions are specified in [10] as follows:

O_2 (A-band)	**CO_2 (weak-band)**	**CO_2 (strong-band)**	
0.758 to 0.773	1.590 to 1.622	2.043 to 2.083	in [µm]
12 937 to 13 193	6165 to 6289	4801 to 4895	in [cm^{-1}]

Table 2.1: Spectral range of OCO-2's measurement channels.

The designers of OCO-2 added the complementary absorption measurement of another atmospheric gas, O_2, because its concentration is constant, well-known and

[6] Note that, another common viewing geometry is *limb* which refers to horizontally sounding instruments.

uniformly distributed throughout the atmosphere. Therefore, atmospheric oxygen is the best candidate for providing reference measurements to assure the accuracy of CO_2 products [2].

For the measurement of carbon dioxide itself, two specific absorption bands in the SWIR are covered by the OCO-2 instrument (see yellow lines at the top of figure 2.2). The weak band measurements are said to be "relatively clear and unambiguous" [see 2, for Measurement Approach] because no other gas has significant absorption within this range. The strong channel serves the purpose of providing a second independent measure of the CO_2 abundance. However, observations in this band are more sensitive to the presence of aerosols and variations in atmospheric pressure and humidity, which is why it can be used to detect (and reduce) such impacts on the final results [2].

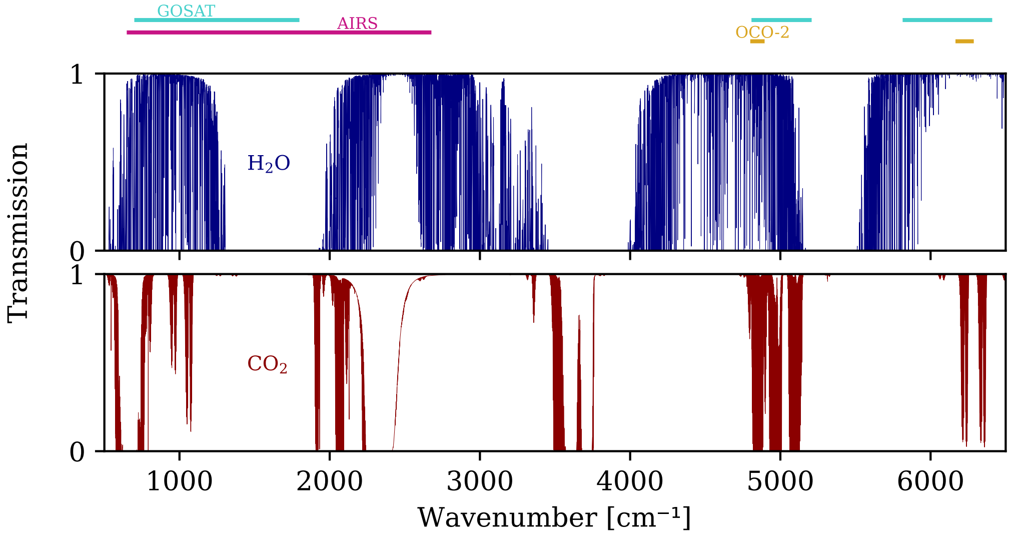

Figure 2.2: Monochromatic transmission spectrum of atmospheric CO_2 and H_2O in the TIR, NIR and SWIR spectral regions simulated with Py4CAtS [1]. The measurement ranges of the satellites AIRS, GOSAT and OCO-2 are indicated at the top.

In general, CO_2 concentrations can also be derived from radiance measurements in other spectral regions like the thermal infrared (TIR) with even longer wavelengths than the SWIR or NIR. Figure 2.2 shows the transmission spectrum of carbon dioxide in the atmosphere with clearly visible CO_2 absorption bands throughout the entire spectrum. Examples for two satellites that (also) measure in the TIR are GOSAT (Greenhouse Gases Observing Satellite) [11] and AIRS (Atmospheric Infrared Sounder) [12]. Their exact spectral regions are indicated in figure 2.2. The reason why they are so much broader than OCO-2's range is because they consider

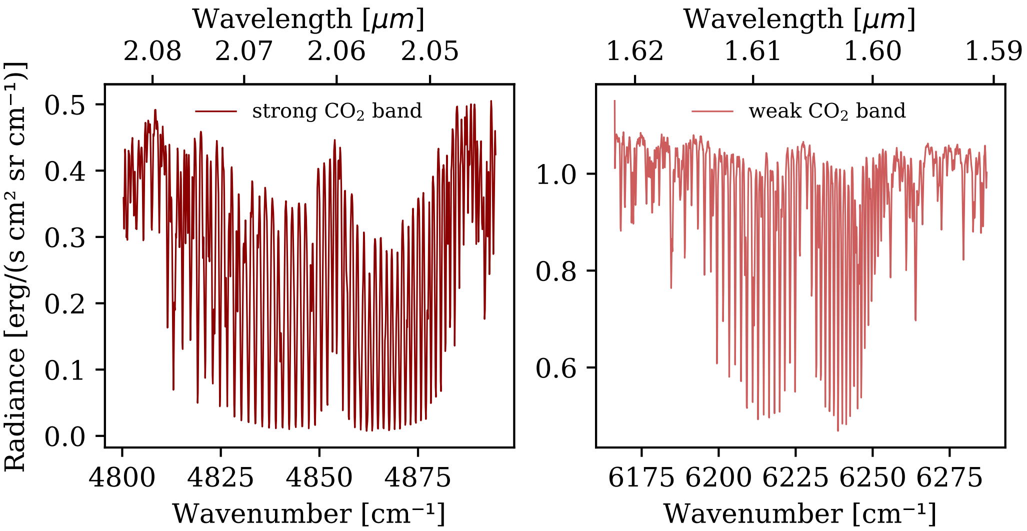

Figure 2.3: Radiance spectrum of atmospheric CO_2 in the strong and weak band measured by OCO-2 (data frame 1144 of orbit 31366a on the 05/25/2022).

other trace gases, as well, that have their absorption peaks in the vicinity of carbon dioxide. Especially in the TIR, a large number of different atmospheric trace gases can be observed [7]. However, a disadvantage of this region is that thermal emission of the surface and atmosphere itself have to be considered in the radiation transfer model which significantly complicates later retrievals. So in order to observe CO_2 with a high sensitivity across all atmospheric layers it is advantageous to use only the SWIR and NIR ranges as OCO-2 does.

In figure 2.2 it is also noticeable that the chosen OCO-2 ranges are relatively narrow (just one band peak) compared to the total width of the CO_2 absorption. Large fitting windows are unfavourable due to their increased noise and the fact that they include more overlapping lines of different trace gases [6, chapter 2]. It is also advantageous to choose windows with low water absorbance, because H_2O concentrations vary largely across the globe, so that the retrieval of it could pose as an additional source of error in the final CO_2 results. This is why the OCO-2 spectral ranges (as classified in table 2.1) can be used unaltered as retrieval windows.

An example for the observed spectral radiance measured by OCO-2 can be seen in figure 2.3. As was already mentioned in Section 2.1 it is important to consider the fact that the measured values are never fully resolved spectrally but taken at instrument resolution. For a grating spectrometer as OCO-2 the shape of the spectral response function (SRF) is well represented by a Gaussian. Still, OCO-2 [10] offers tabulated SRF values for each spectral pixel of every measurement in all bands. Other important information that is included in their product is data quantifying the viewing geometry, solar position, geographic position, surface elevation and quality flags for each measurement. These are all necessary to accurately simulate the measured spectrum with the model described in Section 2.1.

2.3 Trace Gas Retrieval

The essence of trace gas retrieval is to use space-borne measurements and a corresponding forward model to derive parameters that describe the molecular composition of the atmosphere. In the spectral region used for CO_2 retrievals the nadir viewing geometry does not allow to obtain information on the vertical distribution of the gases [13]. In this case the result of a retrieval is the **total column density** of the molecule m

$$N_m = \int_{z_{\text{srf}}}^{z_{\text{toa}}} n_m(z)\,\mathrm{d}z, \tag{2.9}$$

which is the integral over its number density n_m along the path[7]. A related, but even more common quantity used to express retrieval results for trace gases like CO_2, is the **total column averaged dry air mole fraction** defined as

$$x\mathrm{CO_2} = \frac{N_{\mathrm{CO_2}}}{N_{\text{air}} - N_{\mathrm{H_2O}}}, \tag{2.10}$$

which is also often denoted as the "concentration" of the molecule. Its current monthly average for CO_2 is approximately 417 ppm [14].

But how are the quantities described in equation (2.9) and (2.10) related to the forward model (2.8) and how can they be derived in a fit?

Comparing the total column density N_m to the previously introduced total optical depth τ_m from equation (2.5), one can see that they scale the same way. Now, both of these measures can only be calculated along a given path using apriori assumptions on the composition of the atmosphere, namely the number density $n_m(s)$. Thus, the actual total optical depth of a molecule m can be written as

$$\tau_m(\nu) = \alpha_m \cdot \tau_m^{\text{prior}}(\nu), \tag{2.11}$$

which is the reference τ_m^{prior} based on apriori assumptions multiplied by a **molecular scaling factor** α_m. With this relation the fitting function can be derived from the

[7] For this reason, it inherently depends on the surface height z_{srf} at which the light path is reflected. This dependence is not shown in the formulas for simplicity.

forward model (2.8) to be

$$\hat{I}(\nu) \;=\; r(\nu)\cdot\mu_{\odot}\cdot I_{\mathrm{sun}}(\nu)\cdot\exp\left(-\sum_m \alpha_m\cdot\tau_m^{\mathrm{prior}}(\nu)\right) \quad\otimes\quad S(\nu). \tag{2.12}$$

Here, the unknown (fitting) parameters are the reflectivity parameters from the surface albedo polynomial $r(\nu)$ and the molecular scaling factors α_m.

The only two molecules that must be considered for a retrieval in the OCO-2 spectral regions are CO_2 and H_2O (cf. absorption bands in figure 2.2). Using a first degree polynomial to describe the surface reflectivity, this results in the state vector

$$\boldsymbol{x} \;=\; (\,\alpha_{\mathrm{CO_2}}\,,\,\alpha_{\mathrm{H_2O}}\,,\,r_0\,,\,r_1\,)^{\top} \tag{2.13}$$

consisting of four unknown parameters. Inversion of a nonlinear model like (2.12) means to use an initial guess $\boldsymbol{x}_0$ for the state vector and then iteratively improve the values until the model spectrum they produce best approximate the measurements. Finally, the state vector $\hat{\boldsymbol{x}}$ that created the "best" spectrum is considered as the result.

Figure 2.4 shows a schematic illustration of the just described methodology. The necessary input data in trace gas retrieval (apart from an initial state vector) are the solar zenith angle, the solar irradiance, the optical depths based on atmospheric state data and the surface elevation, and tabulated instrument response values (see overview at the end of this section for more detail).

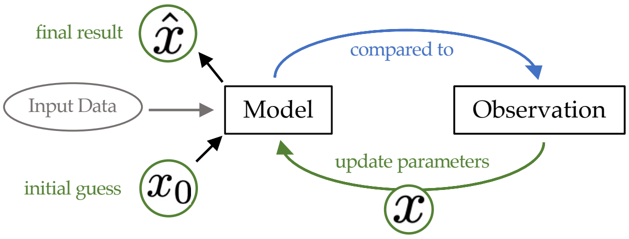

Figure 2.4: Schematic illustration of an iterative retrieval algorithm.

In order to accurately separate the effects of absorbing molecules and surface reflectance on the radiance measurement, it is necessary to use observations at several wavelengths [6, chapter 2]. In the strong CO_2 band of OCO-2 there are approximately 1000 spectral pixels, which are far more than the fitting parameters are. Such a system is called over-determined. Figure 2.5 shows an example of how a simulated spectrum can approximate a measured one in the strong band after a fit has been performed. The optimization method used for such a fit is called **least**

squares which is motivated and explained in Chapter 3. The idea is to minimize the squared differences between model and observation, called residuals. The lower plot in figure 2.5 shows the remaining residuals of the fit that come from errors in the measurement (such as noise) or inaccuracies in the model.

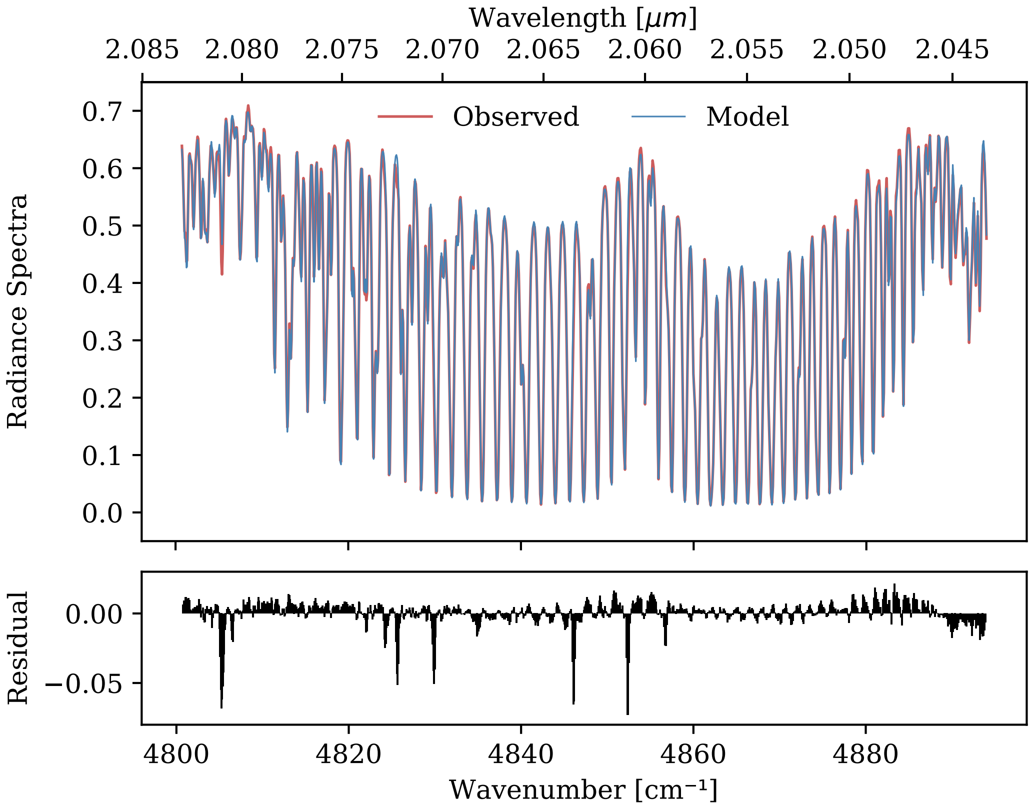

Figure 2.5: Example of measured and simulated radiance spectra after approximation has been performed in the strong CO_2 band. The lower plot displays the residuals of the fit. The y-axis of both plots has the radiance unit [erg/s cm^2 sr cm^{-1}].

With the final state vector $\hat{x}$ that is produced by the iteration method, physically meaningful results such as the distribution of trace gases can be derived. The final molecular scaling factors α_m can be used to calculate the actual total column density of a molecule m, in the form

$$N_m = \alpha_m \cdot N_m^{\text{prior}}. \tag{2.14}$$

In this thesis the code BIRRA (Beer InfraRed Retrieval Algorithm) [13] was used for the retrievals done (see Chapter 5, particularly). It has been validated for the trace gas retrieval of CO in the SWIR in [15] and was recently translated to *Python* from the original *Fortran* implementation [16]. This way it can be used together with the radiative transfer model Py4CAtS which calculates the optical depths for the forward equation (2.8). BIRRA performs a least squares fit to retrieve the atmospheric composition and surface parameters (cf. state vector (2.13)).

In fact, the special form of the model function (2.12) allows the use of an even more specific method within the least squares framework described in Chapter 3, called *Variable Projection*. In it, a separation between the nonlinear parameters, α_{CO_2} and $\alpha_{\mathrm{H}_2\mathrm{O}}$, and linear parameters, r_0 and r_1, is done to simplify the mathematical problem and increase the performance of the algorithm. This idea is subject of the assessment done in this thesis. Its underlying theory and a detailed description of it can be found in Chapter 4.

Finally, for overview, a summary is given of all auxiliary data that were used as input for the retrievals performed in this thesis:

- The molecular spectroscopic parameters S_{ml}, $\hat{\nu}_{ml}$ and $\gamma_{ml}(p, T)$ for the line-by-line calculations (cf. equation (2.4)) used in the forward model Py4CAtS were obtained from HITRAN (HIgh-resolution TRANsmission molecular absorption database) [17].
- The apriori number densities for the two relevant molecules n_{CO_2} and $n_{\mathrm{H}_2\mathrm{O}}$ were taken from the AFGL (Air Force Geophysical Laboratory) atmospheric constituent profiles [18].
- The temperature and pressure profiles $T(z)$ and $p(z)$ were taken from reanalysis data of the National Center for Environmental Prediction (NCEP).
- The radiance measurements $\hat{I}(\nu)$ and the corresponding geometry parameters θ_{sza} and θ_{vza}, surface elevation z_{srf}, and tabulated instrumental line shape values $S(\nu)$ were extracted from the level 1b products of OCO-2 [10].

Chapter 3

Least Squares Approximation

The purpose of this chapter is to explain and motivate a highly acclaimed approximation method used in scientific computing. Björck [19] formulated its essence as follows:

> A fundamental task in scientific computing is to estimate parameters in a mathematical model from observations that are subject to errors. A common practice is to reduce the influence of the errors by using more observations than the number of parameters.

From this follows a mathematical problem that arises in numerous disciplines within science and engineering. It is not surprising to find it in trace gas retrieval, as well. Here, the main idea is to find parameters (i.e. molecular scaling factors) such that the predicted model spectra match the measured ones to the best extent possible. In this chapter it is derived that under certain assumptions the "best linear unbiased estimate" is made by minimizing the sum of the squared difference between model and measurement, which is called least squares approximation. In order to do this, numerical algorithms have been developed some of which are going to be derived in the following. Furthermore, the underlying mathematical theorems and correlations will be outlined.

3.1 Linear Least Squares

Before diving into the theory of least squares problems some remarks on notation are necessary. Bold lower case letters denote (column) vectors, while bold upper case letters are used for matrices. As usual, the transpose of a matrix $\boldsymbol{A}$ will be written as $\boldsymbol{A}^\top$. In this thesis only the ℓ_2-vector norm

$$\|\boldsymbol{x}\|_2 = \sqrt{\boldsymbol{x}^\top \boldsymbol{x}} = \big(\sum_i |x_i|^2\big)^{1/2} \tag{3.1}$$

corresponding to the Euclidean length will be used and the subscript $_2$ will be omitted in the following.

Given a vector $\boldsymbol{b} \in \mathbb{R}^m$ and a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, the aim is to find a vector $\boldsymbol{x} \in \mathbb{R}^n$ such that the product of $\boldsymbol{Ax}$ is the best approximation to $\boldsymbol{b}$. The vector of observation $\boldsymbol{b}$ is assumed to be subject to an error $\boldsymbol{e}$ which is why no $\boldsymbol{x}$ satisfies $\boldsymbol{Ax} = \boldsymbol{b}$ exactly, but rather

$$\boldsymbol{Ax} = \boldsymbol{z} \qquad \text{with} \quad \boldsymbol{b} = \boldsymbol{z} + \boldsymbol{e}. \tag{3.2}$$

In order to reduce the influence of the error in the observations one would like to use a greater number m of measurements than the number of unknown parameters n in the model which leads to an over-determined system. Furthermore, a usual and necessary statistical preliminary is that the errors in the observations are random and uncorrelated, which means that their expectation value is zero, they all have the same variance and that their covariance matrix has entries only on the diagonal [1]. The stated case is called **Gauss-Markov** model which is the basis for the following Gauss-Markov theorem (cf. [19, chapter 2.1]):

Theorem. *Consider the linear Gauss-Markov model* (3.2)*, where the matrix* $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ *has full column rank n and the standard assumptions stated above hold. Then, the best linear unbiased estimator of* $\boldsymbol{x}$ *is the vector* $\hat{\boldsymbol{x}}$ *that minimizes the sum of squares*

$$\min_{\boldsymbol{x}} \|\boldsymbol{b} - \boldsymbol{Ax}\|^2. \tag{3.3}$$

This vector is unique and equal to the solution of the ***normal equation***

$$\boldsymbol{A}^\top \boldsymbol{Ax} = \boldsymbol{A}^\top \boldsymbol{b} \tag{3.4}$$

which represent a system of n linear equations in n unknowns.

This theorem has laid a sound theoretical basis for the method of least squares as the "best" approximation tool. By defining the residual vector

$$\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{Ax} \tag{3.5}$$

the normal equations (3.4) can be rewritten as

$$\boldsymbol{A}^\top (\boldsymbol{b} - \boldsymbol{Ax}) = \boldsymbol{A}^\top \boldsymbol{r} = \boldsymbol{0}. \tag{3.6}$$

In the full column ranked case, $\boldsymbol{A}^\top \boldsymbol{A}$ is a symmetric positive definite matrix, and so the solution of the normal equation can be formulated as

$$\hat{\boldsymbol{x}} = (\boldsymbol{A}^\top \boldsymbol{A})^{-1} \boldsymbol{A}^\top \boldsymbol{b} = \boldsymbol{A}^\dagger \boldsymbol{b} \tag{3.7}$$

[1] It is assumed that the reader has some familiarity with these mathematical and statistical properties. If not, [20, chapter 1] is recommended.

where the matrix $\boldsymbol{A}^\dagger$ denotes the pseudo-inverse[2] of $\boldsymbol{A}$.

There are three direct methods for solving the linear least squares problem (3.3). A more detailed survey of these can be found in [21, chapter 6]:

- A simple and intuitive method is **solving the normal equations** (3.4) by directly calculating the center part of solution (3.7) which includes calculating the inverse of $\boldsymbol{A}^\top \boldsymbol{A}$. However, this method yields the drawback of poor stability and robustness, especially for ill-conditioned problems, and it rapidly deteriorates with increasing conditions.

- A more stable way to proceed is provided by orthogonal transformations like the **QR decomposition**. It exploits the property that for any matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with full column rank n there exists an orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{m \times m}$ and a matrix $\left[\begin{smallmatrix}\boldsymbol{R}\\ \boldsymbol{0}\end{smallmatrix}\right] \in \mathbb{R}^{m \times n}$ where $\boldsymbol{R} \in \mathbb{R}^{n \times n}$ is upper triangular and $\boldsymbol{0} \in \mathbb{R}^{(m-n) \times n}$ is the zero matrix, such that
$$\boldsymbol{A} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{bmatrix}. \tag{3.8}$$
Using this decomposition the residual within (3.3) can be reformulated to a reduced form for which the solution can be expressed as
$$\hat{\boldsymbol{x}} = \boldsymbol{R}^{-1}(\boldsymbol{Q}^\top \boldsymbol{b}). \tag{3.9}$$

- Sufficiently more robust for problems with rank-deficient $\boldsymbol{A}$ is the **singular value decomposition** (SVD). It is based on the theorem that for any $\boldsymbol{A} \in \mathbb{C}^{m \times n}$ with rank r there exist the unitary matrices $\boldsymbol{U} \in \mathbb{C}^{m \times m}$ and $\boldsymbol{V} \in \mathbb{C}^{n \times n}$ such that
$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^H \qquad \text{where} \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma_r} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \tag{3.10}$$
Here, $\boldsymbol{V}^H$ denotes the conjugate transpose of the matrix $\boldsymbol{V}$ and $\boldsymbol{\Sigma_r} = \text{diag}(\sigma_1, ..., \sigma_r)$ with the unique singular values $\sigma_1 \geq ... \geq \sigma_r > 0$ of $\boldsymbol{A}$. This is a powerful tool for solving linear least squares problems, because the unitary matrices that transform $\boldsymbol{A}$ into a diagonal form do not change the Euclidean norm of vectors [20, chapter 1]. This way the solution of (3.3) can be written as the latter part in (3.7) containing the pseudo-inverse
$$\hat{\boldsymbol{x}} = \boldsymbol{V} \begin{bmatrix} \boldsymbol{\Sigma_r}^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \boldsymbol{U}^H \boldsymbol{b}. \tag{3.11}$$

[2] Another name for the pseudo-inverse is the Moore-Penrose inverse. The idea of these inverses is going to be used again in Section 4.2

3.2 Nonlinear Least Squares

A more common problem is the one that arises when trying to fit given data (y_i, t_i) with $i = 1 : m$ to a model function $g(\boldsymbol{x}, t_i)$ which does not depend linearly on its parameters. The error in the model prediction for the i^{th} observation can be denoted as

$$r_i(\boldsymbol{x}) = y_i - g(\boldsymbol{x}, t_i) \qquad \text{with} \quad i = 1 : m, \tag{3.12}$$

which leads to a residual vector of the form $\boldsymbol{r}(\boldsymbol{x}) = [r_1(\boldsymbol{x}), ..., r_m(\boldsymbol{x})]^\top$. The problem of finding a global minimizer for the sum of squares of the residuals can be formulated as

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) \qquad \text{with} \quad f(\boldsymbol{x}) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(\boldsymbol{x}). \tag{3.13}$$

This equation[3] naturally implies an approach which can be stated as in [19]: "The nonlinear least squares problem is a simple special case of the optimization problem to minimize a convex objective function." In this case the objective function in (3.13) is $f(\boldsymbol{x})$ which can also be written as

$$f(\boldsymbol{x}) = \frac{1}{2} \boldsymbol{r}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{r}(\boldsymbol{x})\|^2 \tag{3.14}$$

The factor $\frac{1}{2}$ is chosen for convenience which will become apparent in the following, where derivative information necessary for solving the nonlinear least squares problem (3.13) is introduced.

The first derivatives of the residual vector can be written within a matrix called the **Jacobian**

$$\boldsymbol{J}(\boldsymbol{x}) \in \mathbb{R}^{m \times n} \qquad \text{with} \quad \boldsymbol{J}(\boldsymbol{x})_{ij} = \frac{\partial r_i(\boldsymbol{x})}{\partial x_j} \quad \text{and} \; {}^{i=1:m}_{j=1:n} \tag{3.15}$$

and the second derivatives of the i^{th} element of $\boldsymbol{r}(\boldsymbol{x})$ can be stored in a matrix called the **Hessian**

$$\boldsymbol{H}_i(\boldsymbol{x}) \in \mathbb{R}^{n \times n} \qquad \text{with} \quad \boldsymbol{H}_i(\boldsymbol{x})_{jk} = \frac{\partial^2 r_i(\boldsymbol{x})}{\partial x_j \, \partial x_k} \quad \text{and} \; {}^{j=1:n}_{k=1:n} \tag{3.16}$$

which is by definition symmetric. A necessary condition for $\hat{\boldsymbol{x}}$ to be a critical point of the objective function $f(\boldsymbol{x})$ is that the gradient $\nabla f(\boldsymbol{x})$ vanishes at that point. Furthermore, a sufficient condition for $\hat{\boldsymbol{x}}$ to be a local minimizer is that the Hessian matrix $\nabla^2 f(\boldsymbol{x})$ evaluated at $\hat{\boldsymbol{x}}$ is positive definite. Fortunately, both the gradient and

[3] For simplicity, the over-determined case of $m \geq n$ will be considered in this section and $f(\boldsymbol{x})$ will be a twice continuously differentiable function.

the second derivatives of $f(x)$ can be written in terms of the just defined Jacobian (3.15) and Hessian (3.16) as

$$\nabla f(\boldsymbol{x}) = \boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{r}(\boldsymbol{x}) \tag{3.17}$$

$$\nabla^2 f(\boldsymbol{x}) = \boldsymbol{J}(\boldsymbol{x})^\top \boldsymbol{J}(\boldsymbol{x}) + \sum_{i=1}^{m} r_i(\boldsymbol{x}) \boldsymbol{H}_i(\boldsymbol{x}). \tag{3.18}$$

The essence of all methods for solving (3.13) is to reformulate the problem into a linear one which can be solved numerically. The simplest way to do this is to approximate the behaviour of $\boldsymbol{r}(\boldsymbol{x})$ in the neighborhood of a given point $\boldsymbol{x_k}$ by the linear model[4]

$$\boldsymbol{r}(\boldsymbol{x}) \approx \boldsymbol{r}(\boldsymbol{x_k}) + \boldsymbol{J}(\boldsymbol{x_k})(\boldsymbol{x} - \boldsymbol{x_k}). \tag{3.19}$$

By solving the linear least squares problem at what the point $\boldsymbol{x_k}$ within an iteration

$$\min_{\boldsymbol{x}} \|\boldsymbol{r}(\boldsymbol{x_k}) + \boldsymbol{J}(\boldsymbol{x_k})(\boldsymbol{x} - \boldsymbol{x_k})\|^2 \tag{3.20}$$

one can derive a new approximate solution.

In fact, all methods for nonlinear optimization are iterative and each iteration step usually requires the solution of a related linear problem. Their idea is to start from a sufficiently good initial approximation which is then successively improved until a sufficiently accurate solution is obtained.

The method described above is a well-established one, called the **Gauss-Newton** method. The original idea of it is to compute the correction $\boldsymbol{p_k}$ to the current approximation $\boldsymbol{x_k}$ by solving

$$\min_{\boldsymbol{p_k}} \|\boldsymbol{r}(\boldsymbol{x_k}) + \boldsymbol{J}(\boldsymbol{x_k})\boldsymbol{p_k}\|^2 \tag{3.21}$$

and then repeating the calculation at the new approximation point

$$\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + \boldsymbol{p_k}. \tag{3.22}$$

As described in the previous section, the Gauss-Newton step is the solution to the linear problem (3.21) and can therefore be derived as

$$\boldsymbol{p_k} = -\boldsymbol{J}(\boldsymbol{x_k})^\dagger \boldsymbol{r}(\boldsymbol{x_k}) \ \in \ \mathbb{R}^n \tag{3.23}$$

by using a stable direct method such as the QR decomposition or SVD of $\boldsymbol{J}(\boldsymbol{x_k})$. The only issue with the original method is that it generally has only a linear rate of

[4] This corresponds to the first degree Taylor polynomial of $\boldsymbol{r}(\boldsymbol{x})$ at $\boldsymbol{x_k}$.

convergence and may not even locally converge at all on problems that are very nonlinear or have large residuals[5]. This is the reason why out of this framework many modified methods have evolved, all known as "Gauss-Newton type methods" which improve the strategy into a globally convergent one. Ruhe and Wedin reported and confirmed in [22] that the asymptotic convergence of these algorithms is fast for problems with small final residuals which should apply whenever a good mathematical model is used. These methods are the so called line search, trust region and damped methods, some of which are going to be outlined in the following section.

In addition, the idea of **Newton-type** methods which are contrary to the Gauss-Newton approach are briefly depicted now. It can be derived out of the conditions made for (3.17) and (3.18) at the solution. One starts with a quadratic model[6] of the objective function in a small neighbourhood around the current iterate

$$f(\boldsymbol{x_k} + \boldsymbol{h}) \approx f(\boldsymbol{x_k}) + \left[\nabla f(\boldsymbol{x_k})\right]^\top \boldsymbol{h} + \frac{1}{2}\, \boldsymbol{h}^\top \left[\nabla^2 f(\boldsymbol{x_k})\right] \boldsymbol{h} \tag{3.24}$$

which leads to the linear approximation of the gradient

$$\nabla f(\boldsymbol{x_k} + \boldsymbol{h}) \approx \left[\nabla f(\boldsymbol{x_k})\right] + \left[\nabla^2 f(\boldsymbol{x_k})\right] \boldsymbol{h}. \tag{3.25}$$

By demanding (3.25) to vanish at the minimizer of (3.24) one can find the current step $\boldsymbol{h_k}$ as the solution to

$$\left[\nabla^2 f(\boldsymbol{x_k})\right] \boldsymbol{h} = -\left[\nabla f(\boldsymbol{x_k})\right] = -\boldsymbol{J}(\boldsymbol{x_k})^\top \boldsymbol{r}(\boldsymbol{x_k}). \tag{3.26}$$

The minimizer of (3.24) will then be used as the next iterate

$$\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + \boldsymbol{h_k}. \tag{3.27}$$

An important advantage of this approach is that under usual circumstances it is quadratically convergent to a local minimizer rather that just linearly. A downside, however, is that it requires the calculation of the Hessian (3.18). This can be computationally expensive, which is why most methods only approximate it. These are then called **Quasi-Newton** methods, but they will not be discussed in this thesis.

[5] For a detailed analysis of convergence rates [20, chapter 9] and [19, chapter 2.8] are recommended.

[6] This corresponds to the second degree Taylor expansion of $f(\boldsymbol{x})$ around $\boldsymbol{x_k}$. It needs to be second degree because Newton's well-known method for finding the roots of a function is applied to its first derivative here. This way it instead finds the minima of that function.

3.3 A Survey of Iterative Methods

There have been many modifications to the previously mentioned methods to make them more stable and achieve better convergence. In this section three commonly used algorithms for solving nonlinear least squares problems are introduced. The reason for choosing those specific ones will become apparent in Chapter 5, when a thorough comparison between them will be pursued.

3.3.1 Levenberg-Marquardt Algorithm

A well-established method for unconstrained nonlinear optimization is the Levenberg-Marquardt method. Levenberg [23] in 1944 and Marquardt [24] in 1963 both independently proposed approaches to further stabilize the Gauss-Newton method and thereby overcome the possibility of failure by cleverly constraining the Gauss-Newton step. A first idea was to limit the size of $\boldsymbol{p_k}$ by introducing a damping parameter $\mu_k > 0$ which results in the extension of (3.21) to

$$\min_{\boldsymbol{p_k}} \; \{ \|\boldsymbol{r}(\boldsymbol{x_k}) + \boldsymbol{J}(\boldsymbol{x_k})\boldsymbol{p_k}\|^2 + \mu_k \|\boldsymbol{p_k}\|^2 \} \tag{3.28}$$

which is therefore called a **damped** method. This can be further stabilized by constraining the Gauss-Newton step instead of damping it. The resulting strategy is to solve the linear least squares problem (3.21) which is now subject to

$$\|\boldsymbol{D_k}\boldsymbol{p_k}\| \leq \delta_k. \tag{3.29}$$

with $\boldsymbol{D_k}$ being a diagonal scaling matrix[7]. This way the set of feasible vectors complying with (3.29) can be thought of as a trust region for the linear model (3.19), which is why this method is called a **trust region** method.

Even though this is a very elegant algorithm for the numerical solution of nonlinear least squares problems, early implementations of it lacked robustness and a solid theoretical justification. Moré [25] then proposed one that is efficient and has strong convergence properties. For this reason he implemented it as a *Fortran* package called MINPACK [26] which is a collection of quality optimization software. His algorithm can be outlined[8] as follows:

Starting with the initial values $\boldsymbol{x_0}$, $\boldsymbol{D_0}$ and δ_0 and a $\beta \in (0, 1)$. For $k = 0, 1, 2, ...$

1. Determine $\boldsymbol{x_k}$ as a solution to (3.21) subject to the constraints (3.29).

[7] The initial matrix $\boldsymbol{D_0}$ is chosen such that the algorithm is scale-invariant, meaning that it generates the same iterations if applied to $\boldsymbol{r}(\boldsymbol{D_k}\boldsymbol{x_k})$.

[8] This particular depiction can be found in [20, chapter 9] and has been chosen to be presented here for its clarity instead of the original formulation by [25] which uses a slightly different notation and specific values that have shown to be replaceable.

2. Compute the gain ratio

$$\rho_k = \frac{\|\boldsymbol{r}(\boldsymbol{x_k})\|^2 - \|\boldsymbol{r}(\boldsymbol{x_k} + \boldsymbol{p_k})\|^2}{\|\boldsymbol{r}(\boldsymbol{x_k})\|^2 - \|\boldsymbol{r}(\boldsymbol{x_k}) + \boldsymbol{J}(\boldsymbol{x_k})\boldsymbol{p_k}\|^2} = \frac{\textit{actual decrease of } f(\boldsymbol{x})}{\textit{model decrease of } f(\boldsymbol{x})} \tag{3.30}$$

   which indicates the agreement between the linear model (3.19) and the nonlinear function $\boldsymbol{r}(\boldsymbol{x})$.

3. If $\rho_k > \beta$ the step is considered successful and one can set $\boldsymbol{x_{k+1}} = \boldsymbol{x_k} + \boldsymbol{p_k}$. Otherwise, the iteration is considered unsuccessful and $\boldsymbol{x_{k+1}} = \boldsymbol{x_k}$.

4. Update[9] the scaling matrix $\boldsymbol{D_k}$ and the trust region boundary δ_k.

A detailed elaboration on all of these steps and different strategies for choosing the parameters can be found in the original paper [25].

3.3.2 Trust Region Reflective Algorithm

The next algorithm introduced is another adaptation of a trust region method. It is based on Branch et al. [28] who extended an interior and reflective Newton method [29], that can solve bound-constraint nonlinear least squares problems, to a subspace method which is additionally well-suited for large-scale and sparse problems. Because the size of the problem (2.12) posed by trace gas retrieval in the SWIR is quite manageable, this outline will focus more on the idea of the method being *interior* and *reflective* and less on the subspaces.

Another reason for this is, that reflection is a technique for dealing with bounds which have not yet been addressed. The bound-constraint problem can be written as

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \left\{ f(\boldsymbol{x}) : \boldsymbol{l} < \boldsymbol{x} < \boldsymbol{u} \right\} \tag{3.31}$$

with the lower and upper bounds $\boldsymbol{l}$ and $\boldsymbol{u}$. An algorithm being able to solve this can be called interior because the iterates $\{\boldsymbol{x_k}\}$ are in the strict interior of the feasible region $(\boldsymbol{l}, \boldsymbol{u})$. In the original method by [29], the subproblem

$$\min_{\boldsymbol{s} \in \mathbb{R}^n} \left\{ \Psi_k(\boldsymbol{s}) : \|\boldsymbol{D_k s}\| \leq \delta_k \right\} \tag{3.32}$$

is solved as in standard trust region methods (i.e. Section 3.3.1), except that the quadratic Newton approximation (3.24) is used for $\Psi_k(\boldsymbol{s})$ instead of a linear Gauss-Newton approximation like (3.19). Another difference is that a special *affine* scaling matrix is used for $\boldsymbol{D_k}$ which shows improved performance compared to other methods. Details on this can be found in [28].

[9] A summary of convenient strategies for choosing these measures can be found in [27].

Furthermore, both [29] and [28] use a reflection technique that is said to substantially reduce the number of iterations and accelerate convergence. Without it, the strategy is to determine the step $\boldsymbol{s_k}$ from either minimizing $\Psi_k(\boldsymbol{s})$ along $\boldsymbol{p_k}$ or along the scaled steepest descent direction[10]. Now, a (single) reflection step is defined in [28] as follows:

> Given a step $\boldsymbol{p_k}$, consider the first bound constraint crossed by $\boldsymbol{p_k}$, assuming it is the i^{th} (lower or upper) bound constraint. Then the reflection step becomes $\boldsymbol{p_k}^R = \boldsymbol{p_k}$, except in the i^{th} component where $p^R_{k,i} = -p_{k,i}$.

Since the reflected step $\boldsymbol{p_k}^R$ permits further descent, a new possibility for determining the step $\boldsymbol{s_k}$ — minimizing the value of $\Psi_k(\boldsymbol{s})$ along the "reflected"[11] direction $\boldsymbol{p_k}^R$ — could be added to the original strategy. Apparently, this addition significantly enhances the performance.

3.3.3 Powell's Dogleg Method

The last method outlined here is Powell's dogleg method [30] which was then modified for better performance in [31]. Again, this is a trust region algorithm allowing for bounds on the parameters. The essence of all trust region methods can be summarized as follows:

1. Approximate the objective function either by a Gauss-Newton or a Newton model.
2. Compute the corresponding step while constraining its norm to be within a certain trust region.
3. Compare the reduction of the model function to that in the objective function. If they agree to a certain extent, the step is accepted and the trust region can be expanded. Otherwise, the step is rejected and the trust region is contracted.

The original method of Powell was all about his special strategy of choosing the next step (called the *dog leg step*) to stay within the trust region. The authors of [31], however, focused more on the shape of the trust region. Defining a general trust region as

$$\mathcal{T}_k = \{\boldsymbol{x} \in \mathbb{R}^n \mid \|\boldsymbol{x} - \boldsymbol{x_k}\| \leq \Delta_k\}, \tag{3.33}$$

it is obvious that the shape of it depends on the norm used. While the Euclidean norm $\| \, \|_2$, which has been in use up until now, always corresponds to a hyper

[10] A calculation of this can be found in [28]. Generally, a descent direction is one that decreases the objective function value.

[11] As explained above the reflection (i.e. changing signs) is only done along one direction in which the first bound was crossed.

sphere in the n dimensional parameter space, the uniform norm[12] $\| \|_\infty$ defines a hyper box. This approach using a rectangular geometry is advantageous for it allows effortless adaptation to bound constraint problems. With this method, one can solve the more general bound constrained problem (3.31) using Newton's approximation and restricting the i^{th} component of the step $\boldsymbol{h_k}$ from (3.27) to

$$\max\,[l_i - x_i, -\Delta_k] \leq h_{k,i} \leq \min\,[u_i - x_i, \Delta_k]. \tag{3.34}$$

Without bounds[13], the shape of the trust region corresponds to a hyper cube while bounds transform it into a rectangle. This small change in norm and therefore geometry of the problem into a *dogbox* showed a dramatically improved performance for bound constrained least squares compared to Powell's original *dogleg* approach. In adaptation to the new shape, a modification of the strategy of choosing the step was made in [31] which further improved convergence. This is probably the reason why in most tests the rectangular trust region approach outperformed the original one even in the absence of constraints.

[12] The uniform norm is defined as $\|\boldsymbol{x}\|_\infty = \max_{1\leq i\leq n}|x_i|$, which corresponds to the absolute value of the largest element in $\boldsymbol{x}$.

[13] This means setting l_i to $-\infty$ and u_i to ∞.

Chapter 4

Variable Projection Method

In this chapter a special class of nonlinear problems is characterized in which the variables are separable into two sets. One will see that these types of problems are based on a powerful paradigm that allows more efficient ways of solving them. In 1973, Golub and Pereyra [4] described a method for such problems which they coined *Variable Projection* that is still in use today due to its many advantages. Even though they were not the first ones to come up with the idea, they thoroughly expounded the mathematical properties by justifying the concepts behind it. In this thesis, an algorithm based on this method is tested and applied to the problem of trace gas retrieval.

The reason why this algorithm is explored in this thesis is, that it has proven to not only be more efficient than other separable solvers[1], but also more versatile for different classes of problems (see [35] for a survey on successful applications of the variable projection method). Another motivation for using this method is that it is well suited for exponential problems such as (2.12). See [35] and [36] for a detailed analysis of this property.

4.1 Separable Problems

The main idea of separable approaches like variable projection is seen in least squares problems consisting of two sets of parameters, i.e. nonlinear and linear ones $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, where the linear ones can be explicitly eliminated such that a reduced problem arises:

$$\min_{\boldsymbol{\beta},\boldsymbol{\alpha}} \|\boldsymbol{y} - \boldsymbol{\eta}(\boldsymbol{\beta},\boldsymbol{\alpha})\|^2 = \min_{\boldsymbol{\alpha}} \|\boldsymbol{y} - \boldsymbol{\eta}(\boldsymbol{\beta}(\boldsymbol{\alpha}),\boldsymbol{\alpha})\|^2 . \tag{4.1}$$

This method has some apparent advantages. As Golub and Pereyra pointed out in their review paper [35] on 30 years of developments in the area of separable problems:

[1] Examples for other least squares algorithms that exploit separability are the continuation method [32], the alternating least squares method [33] and the Shen-Ypma algorithm [34] which is actually quite similar to the Variable Projection one.

> This procedure not only reduces the dimension of the parameter space but also results in a better-conditioned problem. The same optimization method applied to the original and reduced problems will always converge faster for the latter.

They mention two important aspects: efficiency and better convergence. Both properties have also been explored and proven by Ruhe and Wedin [22], who were able to show by comparing different algorithms that the ones based on separation always need a smaller number of iterations and therefore require considerably less time than when done without separation. In addition to that Sagara and Fukushima [32] pointed out that a considerable amount of storage can be saved by reducing the problem to one that depends on fewer parameters due to the smaller number of equations. Another important argument is that a smaller initial guess vector is needed for the problem as the most common method of solving (4.1) is to perform optimization with the nonlinear parameters $\boldsymbol{\alpha}$ first and then solve for the remaining linear parameters $\boldsymbol{\beta}$. Also, the reduced parameter space possibly causes a reduced number of local minimizers making it more likely to find a global minimum as Golub and Pereyra pointed out in [5].

All these advantages strongly hint that, whenever there is an apparent property of separability, it should be exploited in a least squares estimation. Especially large or complex problems like exponential fitting benefit from it as this method dramatically reduces computing time [35]. Considering the mathematical model (2.12) behind trace gas retrieval in the short-wave infrared a separable solver as the variable projection algorithm seems to be a very suitable candidate for exploration and testing. In the following the mathematical theories based on [4] and a possible algorithm outline by [5] are presented.

4.2 Theoretical Background

Most often the incentive of separable problems is to fit a nonlinear model[2]

$$\eta(\boldsymbol{\beta}, \boldsymbol{\alpha}, t) = \sum_{j=1}^{n} \beta_j \, \varphi_j(\boldsymbol{\alpha}, t) \tag{4.2}$$

which can be expressed as a linear combination of nonlinear functions φ_j, that need to be continuously differentiable with respect to $\boldsymbol{\alpha}$, to given data (t_i, y_i) with $i = 1, .., m$. One can therefore say that the variables separate into

[2] This problem was formulated more generally by Ruhe [22] who (as previously stated) considered the separation into two sets of variables, but in the case of the mathematical model (2.12) the separation into nonlinear and linear variables considered in [4] seems just as reasonable.

- linear parameters $\boldsymbol{\beta} = [\beta_1, ..., \beta_n]^\top \in \mathbb{R}^n$ and
- nonlinear parameters $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_k]^\top \in \mathbb{R}^k$

Consequently, minimizing the nonlinear functional

$$\begin{aligned} r(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \|\boldsymbol{y} - \boldsymbol{\eta}(\boldsymbol{\beta}, \boldsymbol{\alpha})\|^2 &= \sum_{i=1}^{m} \left[y_i - \sum_{j=1}^{n} \beta_j \varphi_j(\boldsymbol{\alpha}, t_i) \right]^2 \\ &= \|\boldsymbol{y} - \boldsymbol{\Phi}(\boldsymbol{\alpha})\boldsymbol{\beta}\|^2 \end{aligned} \tag{4.3}$$

can be rewritten by separating the parameters and defining $\{\boldsymbol{\Phi}(\boldsymbol{\alpha})\}_{ij} := \varphi_j(\boldsymbol{\alpha}, t_i)$ with $i = 1 : m$ and $j = 1 : n$ as a matrix function. A necessary condition for the following formalism is that minimizing (4.3) represents a fully or over-determined problem with $m \geq n + k$.

At this point it is necessary to introduce the notation $\boldsymbol{\Phi}^\dagger$, which is the Moore-Penrose generalized inverse[3] of a matrix $\boldsymbol{\Phi}$. A useful reference for details on the formalism of pseudo-inverses is [19, chapter 2.2]. Assuming that for any given $\boldsymbol{\alpha}$ there is a minimal least squares solution

$$\boldsymbol{\beta}(\boldsymbol{\alpha}) \equiv \boldsymbol{\Phi}^\dagger(\boldsymbol{\alpha})\boldsymbol{y} \tag{4.4}$$

to the linear problem, we can reformulate the original functional (4.3) as

$$\tilde{r}(\boldsymbol{\alpha}) = \|\boldsymbol{y} - \boldsymbol{\Phi}(\boldsymbol{\alpha})\boldsymbol{\Phi}^\dagger(\boldsymbol{\alpha})\boldsymbol{y}\|^2 \tag{4.5}$$

which is called the **variable projection** functional [4]. The reason for this name is that for each $\boldsymbol{\alpha}$ there is a linear operator

$$\boldsymbol{P}_{\boldsymbol{\Phi}(\boldsymbol{\alpha})} = \boldsymbol{\Phi}(\boldsymbol{\alpha})\boldsymbol{\Phi}^\dagger(\boldsymbol{\alpha}) \tag{4.6}$$

that is an orthogonal projection onto the column space of the matrix $\boldsymbol{\Phi}(\boldsymbol{\alpha})$[4]. Keeping the notation from [4], $\boldsymbol{P}^\perp_{\boldsymbol{\Phi}(\boldsymbol{\alpha})} := \boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{\Phi}(\boldsymbol{\alpha})}$ is the projector on the orthogonal complement of the column space of $\boldsymbol{\Phi}(\boldsymbol{\alpha})$. One can now rewrite the functional (4.5) and obtains

$$\tilde{r}(\boldsymbol{\alpha}) = \|\boldsymbol{P}^\perp_{\boldsymbol{\Phi}(\boldsymbol{\alpha})}\boldsymbol{y}\|^2. \tag{4.7}$$

One should mention at this point that the use of orthogonal projectors is justified due to their main property of norm preservation, which means that applying an

[3] Shortly after [4] established this formalism, Kaufman [37] showed that it is sufficient to use a symmetric generalized inverse $\boldsymbol{\Phi}^-$ satisfying the conditions $\boldsymbol{\Phi}\boldsymbol{\Phi}^-\boldsymbol{\Phi} = \boldsymbol{\Phi}$ and $(\boldsymbol{\Phi}\boldsymbol{\Phi}^-)^\top = \boldsymbol{\Phi}\boldsymbol{\Phi}^-$.

[4] It is assumed that the reader has some familiarity with orthogonal transformations and why they are important in numerical computations. Else, a useful reference can be [19, chapter 2].

orthogonal transformation in a floating point environment does (in principle) not amplify errors [21, chapter 4]. It is easy to see that the variable projection functional (4.7) translates into a purely nonlinear least squares problem of the form

$$\min_{\boldsymbol{\alpha}} \tilde{r}(\boldsymbol{\alpha}) = \min_{\boldsymbol{\alpha}} \|\boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}(\boldsymbol{\alpha})}\boldsymbol{y}\|^2 \tag{4.8}$$

which can be solved using familiar methods for nonlinear problems as described in Section 3.2. The idea of Golub and Pereyras' method is to first solve the reduced problem (4.8) to obtain the nonlinear minimizers $\hat{\boldsymbol{\alpha}}$, and then insert them into equation (4.4) to get the optimal linear parameters $\hat{\boldsymbol{\beta}}$.

In order to justify this procedure and to show equivalence of the solution for the original functional (4.3) as well as the reduced functional (4.7), Golub and Pereyra proved the following theorem [4]:

Theorem. *Assuming that the matrix $\boldsymbol{\Phi}(\boldsymbol{\alpha})$ has constant rank $r \leq \min(m,n)$ for $\boldsymbol{\alpha} \in \Omega \subset \mathbb{R}^k$ with Ω being an open set containing the solution $\hat{\boldsymbol{\alpha}}$. Further, let r denote $r(\boldsymbol{\beta},\boldsymbol{\alpha})$ as in* (4.3) *and $\tilde{r}$ denote $\tilde{r}(\boldsymbol{\alpha})$ as in* (4.5) *and* (4.7).

- *If $\hat{\boldsymbol{\alpha}}$ is a critical point (or a global minimizer in Ω) of $\tilde{r}$, and*

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\Phi}^{\dagger}(\hat{\boldsymbol{\alpha}})\boldsymbol{y} \tag{4.9}$$

  then $(\hat{\boldsymbol{\beta}},\hat{\boldsymbol{\alpha}})$ is a critical point of r (or global minimizer in Ω) and $r(\hat{\boldsymbol{\beta}},\hat{\boldsymbol{\alpha}}) = \tilde{r}(\hat{\boldsymbol{\alpha}})$.

- *If $(\hat{\boldsymbol{\beta}},\hat{\boldsymbol{\alpha}})$ is a global minimizer of r for $\boldsymbol{\alpha} \in \Omega$, then $\hat{\boldsymbol{\alpha}}$ is a global minimizer of $\tilde{r}$ in Ω and $\tilde{r}(\hat{\boldsymbol{\alpha}}) = r(\hat{\boldsymbol{\beta}},\hat{\boldsymbol{\alpha}})$. Furthermore, if there is a unique $\hat{\boldsymbol{\alpha}}$ among the minimizing pairs of r, then $\hat{\boldsymbol{\alpha}}$ must satisfy* (4.9).

This theorem shows the correspondence between the critical points of $r(\boldsymbol{\beta},\boldsymbol{\alpha})$ and $\tilde{r}(\boldsymbol{\alpha})$, and therefore proves the viability of this method.

Being familiar with critical points, one would expect that the proof of the theorem involves derivatives of the variable projection functional (4.7). The same applies for most iterate methods for finding a solution to the minimization problem (4.8). For these, it is necessary (cf. Section 3.2) to calculate derivatives. It is important to state the condition that the rank of $\boldsymbol{\Phi}(\boldsymbol{\alpha})$ must be locally constant at the point, where the derivative is calculated. Otherwise, the pseudo-inverse $\boldsymbol{\Phi}^{\dagger}(\boldsymbol{\alpha})$ is not a continuous function and therefore not differentiable. By using the property $\boldsymbol{P}_{\boldsymbol{\Phi}}\boldsymbol{\Phi} = \boldsymbol{\Phi}$ one can derive the following formula for the derivative of an orthogonal projector[5]

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\alpha}}\boldsymbol{P}_{\boldsymbol{\Phi}} = \boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}}\frac{\mathrm{d}\boldsymbol{\Phi}}{\mathrm{d}\boldsymbol{\alpha}}\boldsymbol{\Phi}^{\dagger} + \left(\boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}}\frac{\mathrm{d}\boldsymbol{\Phi}}{\mathrm{d}\boldsymbol{\alpha}}\boldsymbol{\Phi}^{\dagger}\right)^{\top}. \tag{4.10}$$

[5] Let $\boldsymbol{P}_{\boldsymbol{\Phi}}$ denote $\boldsymbol{P}_{\boldsymbol{\Phi}(\boldsymbol{\alpha})}$ for simplicity.

For a detailed proof and derivation of this formula see [4]. Knowing the definition of $\boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}(\boldsymbol{\alpha})}$ it is easy to see that

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\alpha}}\boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}} = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\alpha}}\left(\boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{\Phi}(\boldsymbol{\alpha})}\right) = -\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\alpha}}\boldsymbol{P}_{\boldsymbol{\Phi}} . \tag{4.11}$$

Using (4.10) and (4.11) one can derive the final formula for the Jacobian of the residual vector from the variable projection functional (4.7) as

$$\boldsymbol{J} = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\alpha}}(\boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}}\boldsymbol{y}) = -\left[\boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}}\frac{\mathrm{d}\boldsymbol{\Phi}}{\mathrm{d}\boldsymbol{\alpha}}\boldsymbol{\Phi}^{+} + \left(\boldsymbol{P}^{\perp}_{\boldsymbol{\Phi}}\frac{\mathrm{d}\boldsymbol{\Phi}}{\mathrm{d}\boldsymbol{\alpha}}\boldsymbol{\Phi}^{+}\right)^{\top}\right]\boldsymbol{y} \tag{4.12}$$

which is an $m \times k$ matrix necessary for most iterative methods of solving (4.8).

Shortly after this was approach formulated, Kaufman [37] considered a simplification by dropping the second term in (4.12) in order to save function and gradient evaluation costs and therefore reduce the computing time per iteration. However, this also marginally increased the number of iterations.

For many years, it was reasoned that this still takes less time than computing the extra term, and Kaufman's simplification became established for many variable projection algorithms. Recently, O'Leary and Rust [5] pointed out that the balance between the computing time of extra iterations vs. the second term in (4.12) can change.

One reason for this is that the efficiency of matrix computation on modern computers exceeds that of function evaluations. Another is that evaluating functions often requires a simulation that has a high computational cost. In their paper [5] they could demonstrate that including the full Jacobian matrix reduces the number of iterations. This is the reason for using the original formula (4.12) in their algorithm.

4.3 Algorithm Outline

Papers on the development of a variable projection algorithm focus mainly on the development of numerical methods for solving the reduced nonlinear or linear least squares problem. In this thesis the outline of a typical variable projection algorithm is presented without going into the details of these two procedures, since direct methods for linear systems have already been discussed in Section 3.1 and the ideas of common iterative methods for nonlinear problems were explained in Section 3.3.

The presented algorithm will be based on a *MATLAB* implementation developed by O'Leary and Rust [5]. In addition to solving the standard separable least squares problem of minimizing (4.3) they allow for weights on the residuals and for bounds on the parameters:

$$\begin{aligned}\boldsymbol{\beta} &= [\beta_1, ..., \beta_n]^\top \in \mathcal{S}_{\boldsymbol{\beta}} \subset \mathbb{R}^n \\ \boldsymbol{\alpha} &= [\alpha_1, ..., \alpha_k]^\top \in \mathcal{S}_{\boldsymbol{\alpha}} \subset \mathbb{R}^k \,.\end{aligned}$$

The so (constrained and weighted) reduced problem can be written as

$$\min_{\boldsymbol{\alpha} \in \mathcal{S}_{\boldsymbol{\alpha}}} \|\boldsymbol{W}(\boldsymbol{y} - \boldsymbol{\eta}(\boldsymbol{\beta}(\boldsymbol{\alpha}), \boldsymbol{\alpha}))\|^2 \tag{4.13}$$

using an approach analogous to the one described in the previous section. One defines $\boldsymbol{W}$ as a diagonal matrix containing weights, which are chosen to make the standard deviation of the errors in the weighted observations $\omega_i y_i$ approximately equal. Since most level 1b satellite data do not include apriori weights on their measurements, this property was not exploited in the translated *Python* implementation [38] that is used in the present thesis.

Returning to the algorithm, a general approach proceeds as follows:

1. The user supplies the values $\boldsymbol{y}$, an initial guess for the nonlinear variables $\boldsymbol{\alpha}$, the number of linear parameters n and a subroutine supplying the evaluation of the nonlinear function $\boldsymbol{\Phi}$ and its derivative at a certain $\boldsymbol{\alpha}$.

2. Generate the variable projection functional (4.5) dependent on $\boldsymbol{\alpha}$ by calculating $\boldsymbol{\beta}(\boldsymbol{\alpha})$ as in (4.4) and inserting it into (4.3).

3. Use the partial derivatives of $\boldsymbol{\Phi}$ with respect to $\boldsymbol{\alpha}$ (which have ideally been supplied by the user) in order to generate the Jacobian (4.12) of the variable projection functional dependent on $\boldsymbol{\alpha}$[6].

4. Solve (4.8) using an already existing nonlinear least squares solver (ideally allowing bounds) to get the optimal parameters of $\hat{\boldsymbol{\alpha}}$ as a result.

5. Once having the final nonlinear minimizers, solve the linear equation (4.4) for the optimal linear parameters $\hat{\boldsymbol{\beta}}$.

6. Optionally, compute statistical diagnostics from the solution to help validate the estimated parameters.

There will be further remarks on step 1, how the user is supplying the input in the following section. For step 4, different solvers for nonlinear problems (based on the ideas of Section 3.3) will be mentioned in Section 4.4 and tested in the following chapter. Regarding step 5 and the general calculation of (4.4) in step 2, there are two well-established algorithmic options for directly solving linear systems as outlined in Section 3.1.

[6] A detailed description of this procedure can be found in section 2.3 in [5].

A "standard" approach in general-purpose software is the QR decomposition since it is fast and reliable for well-conditioned problems. A modified approach based on this has also been used by Golub and Pereyra [4]. Another approach is SVD (Singular Value Decomposition) which is mostly used when there is rank deficiency in which case the QR approach may not be sufficiently robust. The SVD based approach is very robust, and therefore advised for ill-conditioned problems [21]. However, it is significantly more expensive in computing time. Still, O'Leary and Rust chose to use this method, because their implementation makes it appropriate for small- and medium-size problems.

Moving on to step 6, which is fairly common in least squares algorithms. There is a lot of statistical information that the implementation of O'Leary and Rust offers to the user. This particularly makes it attractive and suitable for the kind of testing that will be performed in this thesis. The important diagnostic features that occur in the *Python* translation and have been used for the assessment will be described in detail in Chapter 5.

4.4 Implementation

In the course of the past 40 years there have been several implementations of variable projection algorithms in many different programming languages. The first ones were *Fortran* implementations like VARPRO [39] based on [4] which was written in 1977 by Pereyra. At Stanford University, John Bolstad streamlined the code and improved the documentation of VARPRO under the guidance of Gene Golub. Later, a graduate student at Stanford, Randy LeVeque, wrote a modified version of the VARPRO code called VARP2, which extended the original code to problems with multiple right-hand sides [40]. These refer to special types of problems which were later refined in [41] and will also be mentioned in Chapter 6. Other well-established least squares implementations including the variable projection method and its advancements can be found in the *PORT Mathematical Subroutine Library* [42].

One could argue that these implementations are nowadays somewhat outdated. Therefore the newer implementation by O'Leary and Rust based on [5] from 2013, which has been proven to be effective, is examined in this thesis. In the corresponding paper, the authors argue that the *Fortran* implementations, though running efficiently, sacrifice readability and often have lengthy and cryptic documentations.

In contrast, their implementation uses the language *MATLAB* which reduces efficiency, but it consists of only about 160 executable lines and in addition provides statistical diagnostics. The brevity of their code is partially due to the fact that they use the already existing nonlinear solver *lsqnonlin.m* instead of implementing their own one, so it is modular. This way it can simply be exchanged if a better version becomes available. These features make the code *varpro.m* easily understandable and therefore well suited for translations into other languages.

Last year, Schreier exploited this property and wrote his own *Python* version of the code called *varpro.py* [38]. This is especially useful because the forward model Py4CAtS [1], described and mentioned in Chapter 2, and the retrieval algorithm BIRRA [13] itself are both *Python* implementations. Figure 4.1 shows schematically, how it works.

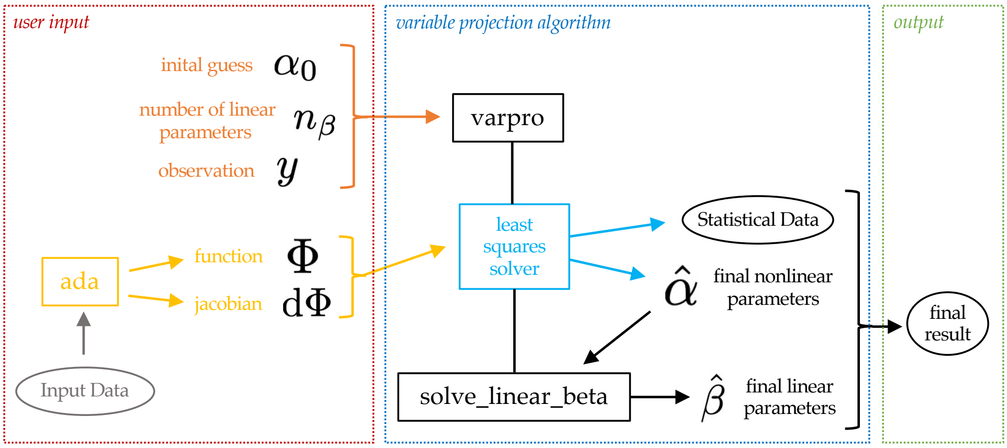

Figure 4.1: Simplified Illustration of data flows and calculations within *varpro.py*

In a first step, the user has to provide all the necessary input data concerning the satellite measurements $\boldsymbol{y}$ as mentioned in Section 2.2. Before the start of this thesis a *Python* program was specifically developed for this purpose, which derives all necessary input from OCO-2 data files and calculates optical depths with Py4CAtS (the radiation transfer code mentioned in Section 2.1). Additionally, a function *ada* has to be set up that models the nonlinear part $\boldsymbol{\Phi}$ of the forward function (2.12) and its derivatives $d\boldsymbol{\Phi}$. This was already implemented within BIRRA (retrieval algorithm mentioned in Section 2.3) [16]. Finally, initial guesses for the nonlinear parameters $\boldsymbol{\alpha}$ and the number of linear parameters n_β have to be supplied.

After this, the function *varpro* can start the approximation, which in a first step tests all the input parameters for consistency. Before the nonlinear least squares solver can be used, one needs two additional functions that use output from *ada* to generate values for the nonlinear functional (4.5) and its Jacobian (4.12), called *fct4nls* and *jac4nls*, respectively [38]. These and the initial guess $\boldsymbol{\alpha}_0$ are the input which the nonlinear least squares solver needs to derive the final nonlinear parameters $\hat{\boldsymbol{\alpha}}$. Existing least squares algorithms already provide some additional statistical data on the results, which can be used to calculate useful measures in order to asses the algorithm. Lastly, the linear parameters $\hat{\boldsymbol{\beta}}$ are calculated using the SVD approach within the function *solve_linear_beta* (cf. figure 4.1). The final output of the *varpro* function are together $\hat{\boldsymbol{\alpha}}$, $\hat{\boldsymbol{\beta}}$ and statistical diagnostics.

The original code has used the function *leastsq* [43] from the *scipy.optimize* module for solving the reduced purely nonlinear problem. This has been modified in this thesis for two reasons. The function *leastsq* is a wrapper around the *MINPACK* implementations LMDIF and LMDER for solving nonlinear least squares problems with the Levenberg-Marquardt algorithm [see 26, for details]. The original algorithm based on [25] as described in Section 3.3.1, however, does not allow constraints on the parameters and has some other limitations in its usability.

These drawbacks can be avoided using the function *least_squares* [44] which is part of the same module *scipy.optimize*. It offers not only two additional methods for solving nonlinear problems but these also come with the ability to handle bounds on the parameters. One of the additions is the trust region reflective algorithm based on [28], as outlined in Section 3.3.2 and the other is a rectangular trust region dogleg approach based on [31], as described in Section 3.3.3. This small modification gives the opportunity to test *varpro.py* for several methods and find out which is best for the application of trace gas retrieval. Further, constraints on the nonlinear parameters[7] (e.g. positivity) can be assessed for their benefits. Both features serve an important purpose in this thesis and will be utilized in Chapter 5.

[7] Since the linear least squares method used in this implementation does not allow bounds on the linear parameters this naturally implies that any variable that requires constraints should be included in the nonlinear parameters.

Chapter 5

Testing and Comparison

The objective of this chapter is the assessment of *varpro.py* for the application of trace gas retrieval in the SWIR as it was described in Chapter 2. Beforehand, a few remarks on the testing strategy are necessary:

There is no reason to conduct yet another comparison between a variable projection solver and a normal nonlinear one, since this has already been done several times (e.g. [22] for a general numerical comparison and parts in [9] for trace gas retrieval, in particular). Another reason why such a comparison would not be very meaningful for the retrieval setups in this thesis is, that the number of linear parameters used in the forward model (described in Chapter 2) is not significantly exceeding the number of nonlinear ones. This way, most of the benefits (mentioned in the beginning of Chapter 4) arising from the reduction of the problem size would not show up in the calculations suggested in Section 2.3.

However, the number of linear parameters may increase considerably in future applications (see Chapter 6 for more detail). In this respect, the idea of this thesis was to merely conduct tests within the *varpro.py* framework without including comparisons to "classical" least squares solvers or a verification of the algorithm (as [5] already did on *varpro.m*) itself.

According to [21], there are three criteria for assessing an algorithm: **Accuracy**, **efficiency** and **robustness**. The first can be indicated by the magnitude of error of the results which were calculated by the algorithm. Different measures to estimate this magnitude will be mentioned in Section 5.1. Intuitively, the measures for the efficiency of an algorithm are its rate of convergence[1], number of iterations and operation counts per iteration. About the second, the following was stated in [33]: "In fact, to compare iterative algorithms only on the basis of the number of iterations they require is not very meaningful, because the costs per iteration are very different." This is why in this thesis, the analysis of efficiency is focused on a comparison of the algorithms' running times which combines the above

[1] The rate of convergence is usually assessed by making theoretical statements regarding the algorithm (see [22] as an example) and not by actual calculations, because often it largely depends on the input data, instead of the underlying structure of the algorithm.

measures. The last criterion, robustness, refers to the algorithm's sensitivity to different conditions and therefore to its stability.

Taking these measures into consideration, there are three objectives of assessment in this thesis, all applied to the nonlinear solver function *least_squares* [44] which is utilized within the *Python* implementation *varpro.py*. The first is a comparison of the three nonlinear solving methods, that are offered by *least_squares*,

- *'lm'* - Levenberg-Marquardt Algorithm based on [25] (see Section 3.3.1)
- *'trf'* - Trust Region Reflective Algorithm based on [28] (see Section 3.3.2)
- *'dogbox'* - Powell's Dogleg Method based on [31] (see Section 3.3.3)

in terms of computational efficiency and accuracy of the retrieval results in Section 5.1. Secondly, these methods are tested for their robustness and accuracy when given different sets of initial guesses for the nonlinear parameters in Section 5.2. Lastly, *varpro.py* is examined for the use of bound-constraints on the nonlinear parameters (i.e. molecular scaling factors) in terms of their effects on the efficiency and robustness of the algorithm.

The following evaluations are all based on approximately 500 retrievals each using real measurements of radiance spectra from summer 2020 of the OCO-2 satellite mentioned in Section 2.2. For these retrievals of CO_2, the exemplary state vector (2.13) was chosen for the fit including 4 unknown parameters (two linear ones, r_0 and r_1, for the reflectivity polynomial and two nonlinear ones, α_{CO_2} and α_{H_2O}, as the relevant molecular scaling factors). All of the plots in the following two sections where generated from the same data set containing retrieval results from the strong CO_2 band of OCO-2.

5.1 Nonlinear Solvers

The first and most obvious idea of the assessment was to compare the computational times for all three methods *lm*, *trf* and *dogbox* within the *varpro* solver. Such comparisons are especially meaningful in scientific applications with real measurements, because the number of fits to be performed can reach up to several millions (cf. remarks in Section 2.2), then even small running time differences can have a large total impact.

Figure 5.1 shows a histogram of the time each method took for solving the nonlinear problem (cf. equation (4.8)) in the algorithm *varpro*. From this it can be concluded that the Levenberg-Marquardt algorithm used in *least_squares* is considerably slower than the other two methods. Even though the mean value of running time per

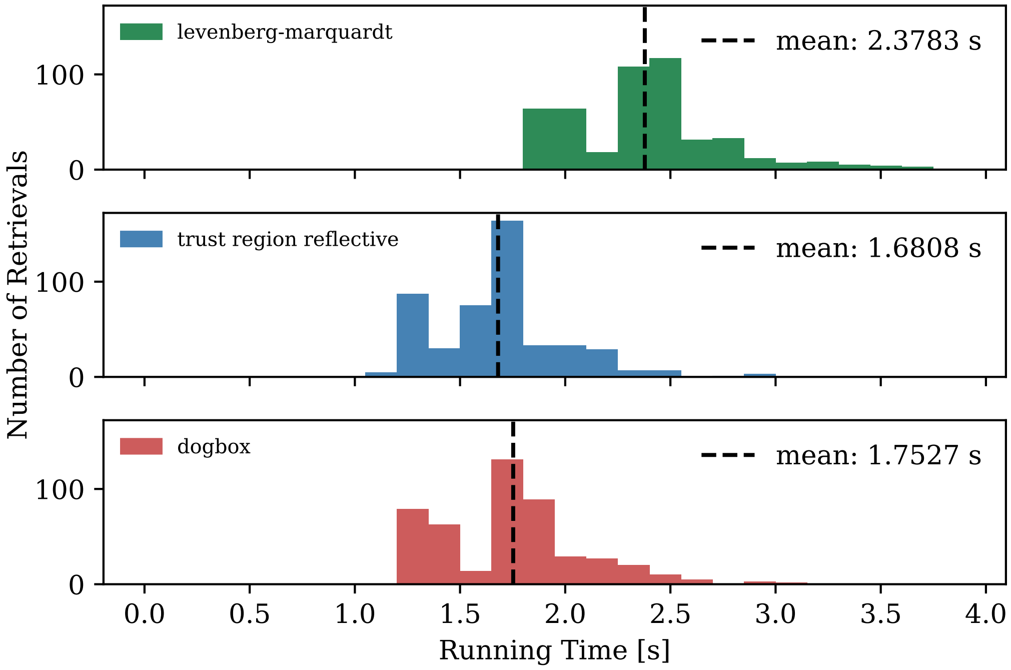

Figure 5.1: Histogram of the running times[2]and their mean values for three methods used to evaluate the *least_squares* function.

retrieval of the *lm*-method only differs by approximately half a second, this could have a significant effect when performing several thousands of such fits.

This is different for the even smaller time-difference between the dogbox and trust region reflective method. In order to better compare their performances, figure 5.2 shows the relative running time-difference $(t_\mathrm{x} - t_\mathrm{trf})/t_\mathrm{trf}$ for both methods *lm* and *dogbox* compared to *trf* in each single retrieval. One can derive that while *lm* performed consistently slower by an average of 43 %, the dogbox method performed rather comparatively only taking approx. 5 % longer[3] than *trf*.

Another interesting feature to compare is the termination reason of the algorithms. The user defines certain tolerance levels for changes of both the function to minimize, $\tilde{r}(\boldsymbol{\alpha})$ (cf. equation (4.7)), and the value it is minimized for, $\boldsymbol{\alpha}$. For the *least_squares* function these can be specified in the input arguments *ftol* and *xtol*[4]. As soon as an

[2] The absolute times displayed in this figure have no general informative value as they depend on external factors like the computing power of the device used for the retrievals.

[3] This difference is so small that it can be neglected, as background processes could be the cause of it.

[4] The currently demanded accuracy of $x\mathrm{CO}_2$ products is at 1 ppm which corresponds to 0.25 % of the

iteration produces values that differ from their previous ones by less than the given tolerances, it stops. This is to avoid over-fitting, meaning that the algorithm should produce results which are more precise than what is possible from the uncertainty of the input data.

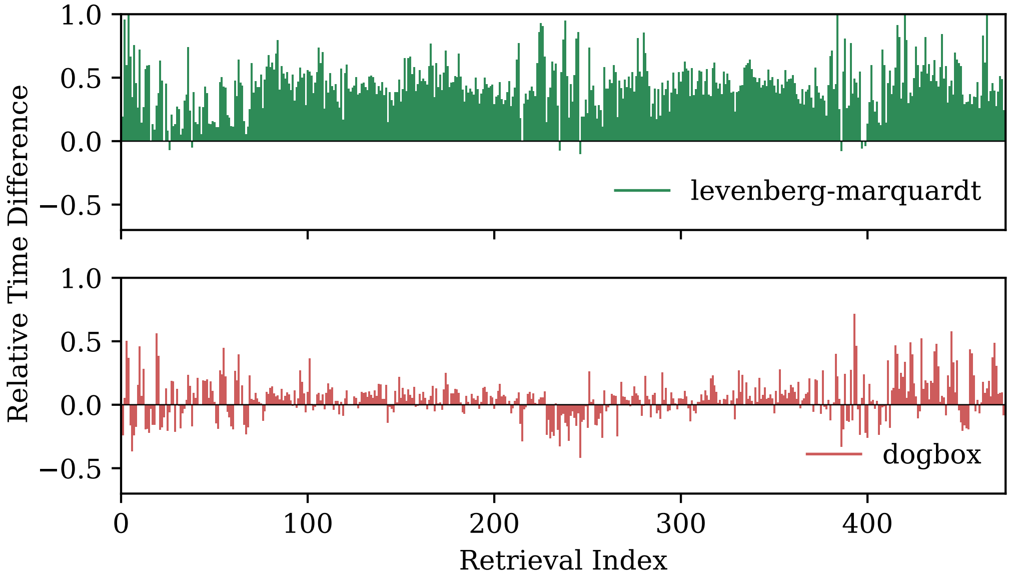

Figure 5.2: Relative time difference of the *lm* and *dogbox* method for each retrieval compared to the *trf* method.

Figure 5.3 shows a comparison of how often each method stopped because either "*ftol* was satisfied" or "both *ftol* and *xtol* were satisfied". Again, the results of the Levenberg-Marquardt algorithm seem to differ from the other two. Apparently it is less likely to end up satisfying both criteria (*ftol* and *xtol*). This means, the algorithm terminates more often, because the iterative change in the model spectrum has reached the uncertainty of the actual spectrum. In such cases, the least squares algorithm indicates that the noise of the spectrum does not allow for the desired accuracy of the $\hat{\boldsymbol{\alpha}}$ values, e.g. $\alpha_{\mathrm{CO_2}}$. Actually, the dogbox and trust region reflective method seem to have this "problem" fewer times (cf. figure 5.3).

Lastly, to judge the accuracy of results, it is necessary to introduce some statistical measures. A standard output for any least squares routine is the value of the cost

initial guess of 400 ppm. Therefore, $\Delta\alpha_{\mathrm{CO_2}} = 10^{-3}$ was used for *xtol*. In the level 1b product of OCO-2 [10] the signal-to-noise ratio (SNR) is specified for each spectrum ranging from about 100 to 1000. As the value used for *ftol* should approximately correspond to 1/SNR, 10^{-3} was chosen.

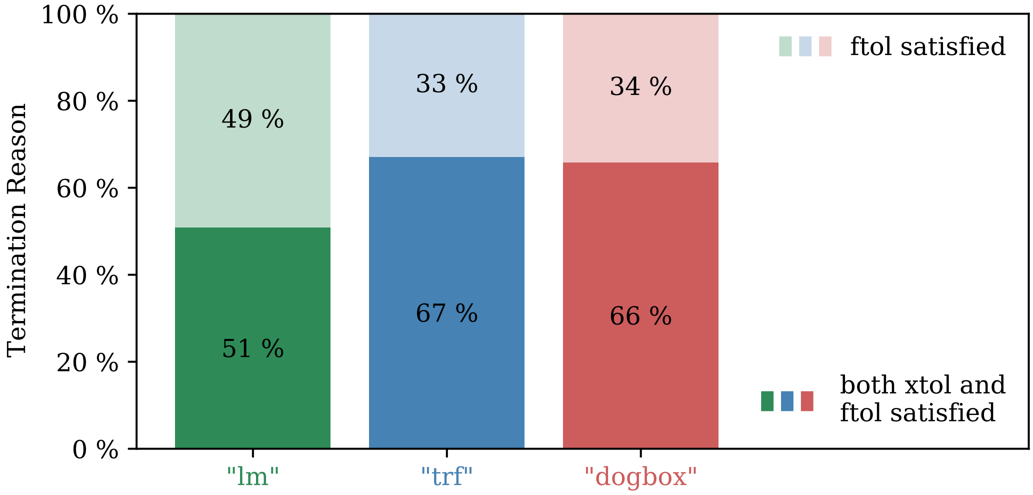

Figure 5.3: Bar diagram comparing the incidences of different termination reasons for all three methods.

function[5] at the solution, which in the case of *least_squares* means the variable projection functional (4.7) evaluated at the final $\boldsymbol{\alpha}$, namely $\tilde{r}(\hat{\boldsymbol{\alpha}})$. This is commonly referred to as the **residual mean squared** which can be seen as a general measure to compare the final error between model and observation after a fit has been performed. One calculates the **sigma of regression**

$$\sigma = \sqrt{\frac{\tilde{r}(\hat{\boldsymbol{\alpha}})}{m-n-k}} \quad , \tag{5.1}$$

which is the square root of the variance of the model (residual mean squared) divided by the number of degrees of freedom (number of data points minus number of linear and nonlinear variables). Finally, in unconstrained uses of the function, *least_squares* returns the **uniform norm of the gradient**, $\|\nabla\tilde{r}(\hat{\boldsymbol{\alpha}})\|_\infty$, which is called *optimality* in the documentation [44]. It can be viewed as a measure of how close the algorithm has got to the "actual" minimum of the cost function (4.7) before it terminated.

As they are closely related to one another, all three measures behave similarly as diagnostics for the accuracy of an algorithm. Figure 5.4 shows the regression sigma (5.1) and the gradient norm for all test retrievals done with the three algorithms. One can see that with respect to accuracy they all performed quite comparably.

[5] This generally refers to the function which is minimized, namely the squared euclidean norm of the residual vector.

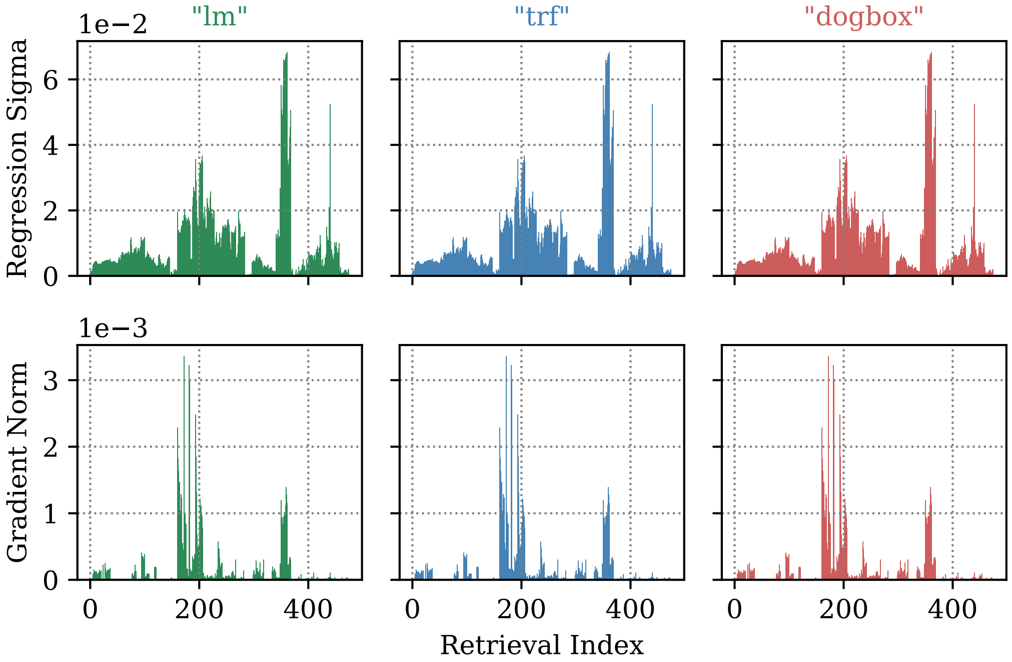

Figure 5.4: Histogram of the regression sigma and gradient norm of each retrieval for three methods.

5.2 Initial Guesses

The next step of the assessment was to compare the three nonlinear solving methods within *varpro* with respect to their robustness in using different sets of initial guesses for the nonlinear parameters α_{CO_2} and α_{H_2O}. In optimization routines it is usually advised to use initial values that are as close to the actual result as possible. Fortunately, the concentration of CO_2, as defined in (2.10) in Chapter 2, only varies by a few percent on a global scale [14]. Using current atmospheric data for the model, setting the initial α_{CO_2} to 1 will already fulfill this criterion.

In this thesis, four additional first guesses were tested (both for α_{CO_2} and α_{H_2O}) with the three algorithms *lm*, *trf* and *dogbox* corresponding to ±50 % and ±100 %. The evaluation showed that the accuracy of their retrieval results were still quite comparable. Only the gradient norm turned out to have small discrepancies for the three methods (see figure 5.5). Especially for the initial guess $\alpha_{CO_2} = 0$, the Levenberg-Marquardt algorithm got better results than the other two. In order to rule out the possibility that this discrepancy is sheer coincidence, the same tests

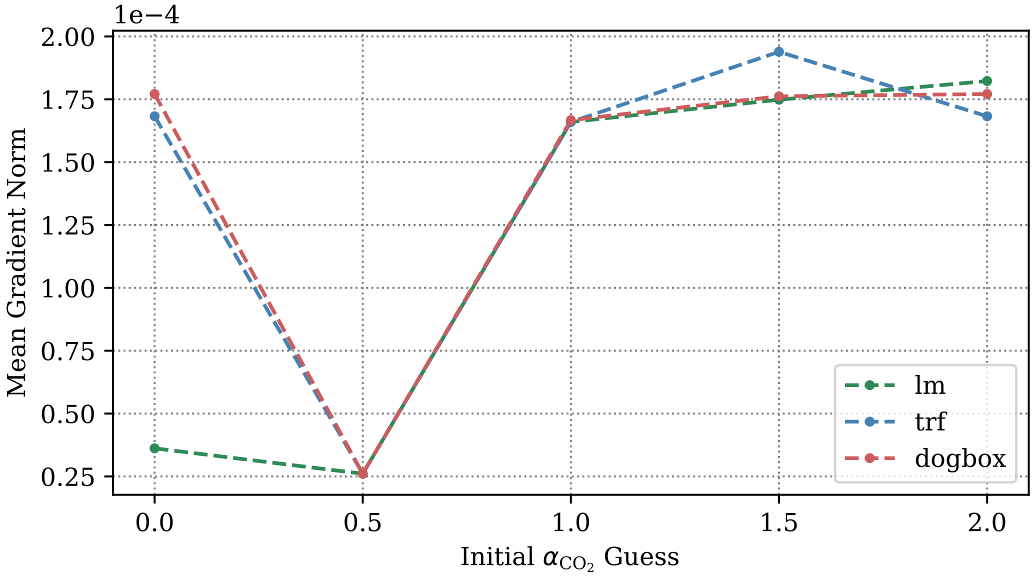

Figure 5.5: Mean gradient norm of retrievals using different initial guesses of α_{CO_2}.

were performed using the weak CO_2 band which showed the same behaviour. This finding agrees with [44], calling the *lm* method "very robust".

In order to really be able to compare the robustness of the methods on a larger scale, it would be necessary to conduct further tests with even "worse" initial guesses. These would not have any physical meaning, and therefore it is sufficient to stay with the present results. However, it might be interesting to find out in future analysis why all methods had their minimal mean gradient norm at $\alpha_{CO_2} = 0.5$ (see figure 5.5) instead of at the "best" guess 1.

5.3 Bound-Constraints

Lastly, the (in comparison to the initial version of *varpro.py* by [16]) novel feature was assessed, which was the use of bound-constraints on the nonlinear parameters. As already mentioned at the end of Chapter 4, in the implementation of *least_squares* only the trust region reflective and dogbox method come with this opportunity. The idea was to find out whether the application of nonlinear bound-constraints have any effect on the performance of the algorithms in terms of efficiency and accuracy.

The following sets of bounds $(l; u)$ were tested for α_{CO_2}[6]:

- Setting 0 : (0 ; ∞)
- Setting 1 : (0 ; 2)
- Setting 2 : (0.9 ; 1.1)

Setting 0 corresponds to the physical fact, that (trace gas) concentrations are always positive. This can be considered the most simple setting, as it only incorporates a lower bound. The second setting is stepping this up by adding a physically meaningful upper bound. At least for CO_2, a doubling of concentrations is not realistic in the next few years. The last setting is motivated by the most upper and lower limits of typical retrieval results from OCO-2, which for an initial guess of $x\text{CO}_2 = 400$ ppm would correspond to a range from 360 ppm to 440 ppm. Although this approach can not be regarded as scientifically justifiable in trace gas retrieval, as it might prevent important findings for the atmosphere (i.e. ozone hole), it still seemed numerically interesting whether such harsh restrictions on the parameter α_{CO_2} would have any effect on the algorithms speed or number of iterations.

Figure 5.6 shows the running time for both methods *trf* and *dogbox* in all three settings. In spite of the small difference between the two algorithms, they both show the same result, namely that setting 1 is slightly faster than the other two. Setting 2 seemed to be the slowest, therefore the previously mentioned possibility of narrow bound-constraints having a positive effect on an algorithm's speed is refuted.

When it comes to accuracy, again all retrieval results were comparable within uncertainty. The only noticeable difference was in the development of their mean gradient norms (see figure 5.7). For better comparability, a setting with no bounds was included into this analysis. While the *optimality* measure kept getting worse for each narrower setting for the *trf* method, it stayed on a quite low[7] level and even decreased a little with the *dogbox* method.

Finally, to answer the question of "bounds or no bounds" new retrievals[8] were performed with both *trf* and *dogbox*, one without bound-constraints and one using setting 1 as bounds.

[6] Different values were used for α_{H_2O}, as the concentration of water in the atmosphere varies much more strongly than $x\text{CO}_2$.

[7] The different magnitudes of order in the *optimality* measure for both methods in figure 5.7 are due to the fact, that in constrained uses of *least_squares* the mean gradient norms are calculated differently for each of those methods [44].

[8] In order to decrease the possibility of bias from the data and to make the evaluation more conclusive, different radiance measurements (from other orbits on a different day) were taken for this last calculation. The number of final retrievals stayed in the same magnitude.

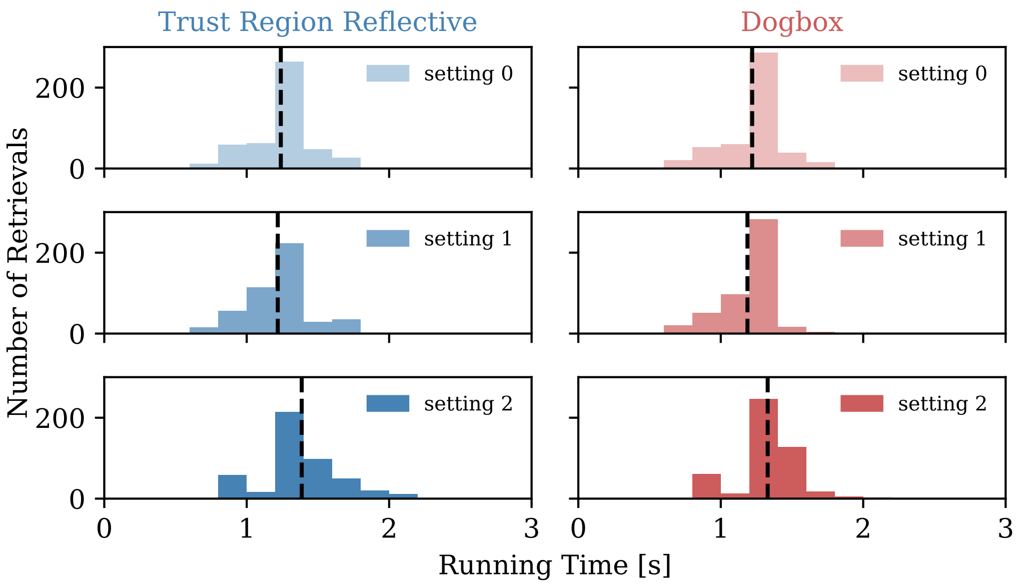

Figure 5.6: Histograms of the running times for the methods *trf* and *dogbox* for all bounds settings. The dashed lines indicate the mean values of these times.

The resulting mean running times were:

	trf	*dogbox*
no bounds	1.135 s	1.159 s
bounds	1.160 s	1.135 s

Even though these are all very close to each other, it leaves the impression that while *trf* is faster than *dogbox* without bounds, it is the opposite when bounds are applied. This tiny difference also showed up in the retrievals done for figure 5.6. A comparison of the number of function evaluations for each method can be seen in figure 5.8, which shows that the trust region reflective algorithm with bounds was the only one that needed more than 3 iterations for some retrievals. Such a behaviour is also mentioned by [44] stating about the *dogbox* method: "The algorithm often outperforms 'trf' in bounded problems with a small number of variables."

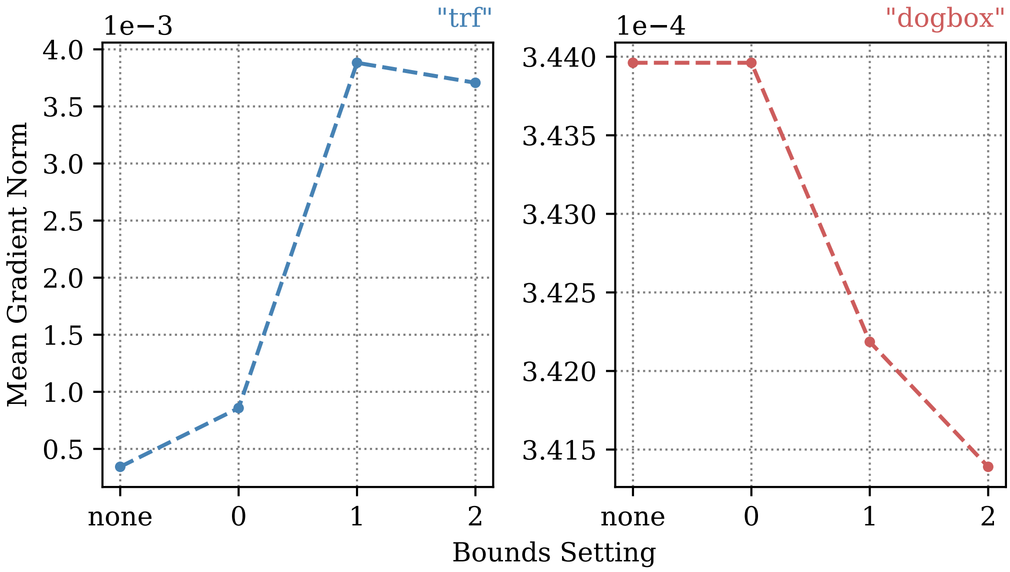

Figure 5.7: Mean gradient norms of methods *trf* and *dogbox* for all bounds settings (including no bounds).

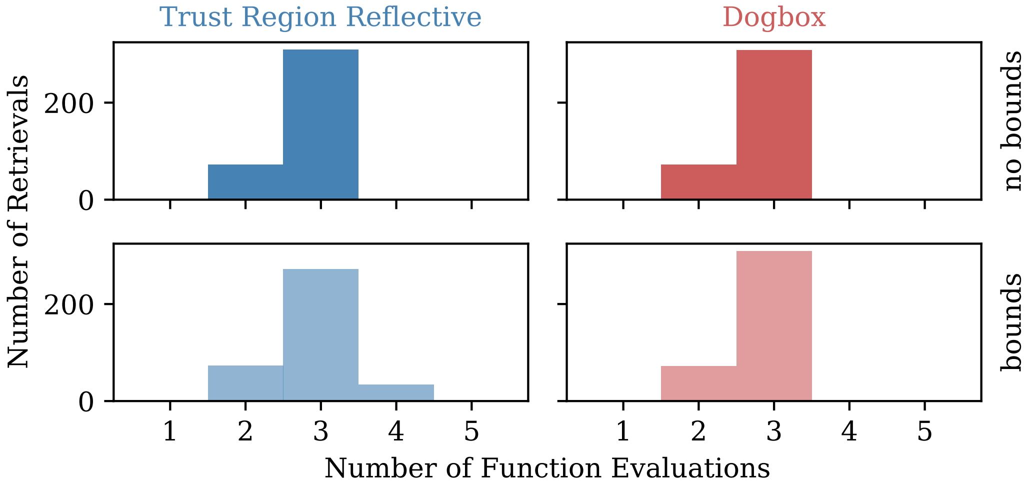

Figure 5.8: Histograms of number of function evaluations for both methods using "no bounds" and "bounds".

Chapter 6

Conclusions and Outlook

The objective of this thesis was the assessment of a least squares solver for trace gas retrieval. To this end the fundamentals of remote sensing of the atmospheric composition and the theory behind solving the nonlinear least squares problem arising in this context were explained. In particular, a separable least squares algorithm called *variable projection* was motivated and the implementation *varpro.py* [38] was described and evaluated for the application of trace gas retrieval in the short-wave infrared. The goal of the assessment was to find a nonlinear least squares solver for *varpro.py* that is best and most useful for this particular application and to work out suggestions for improvements and future research.

In this thesis three algorithms (all outlined in Section 3.3) were compared and tested acting as the nonlinear solvers within *varpro.py*. Because of the very specific problem treated, the results of the assessment are not automatically transferable to other atmospheric retrievals. However, they were able to show certain tendencies, based on which recommendations for future uses of *varpro.py* in trace gas retrieval in the SWIR can be made.

It is advised to exchange the originally used function *leastsq* [43] for solving the nonlinear problem with *least_squares* [44], as the *leastsq* algorithm is identical to the Levenberg-Marquardt (*lm*) method implemented in the latter, which continuously showed worse efficiency (see Section 5.1) than the other two options, the trust region reflective algorithm (*trf*) and the dogbox approach (*dogbox*). However, from the calculations done in this thesis it was not possible to determine which one of these two is generally more efficient, but this might be studied in future tests with more or different data (for example from other satellites, like GOSAT [11]). A comment made by the *scipy.optimize* documentation [44], claiming that *trf* and *lm* are very similar and have generally comparable performance, must be refuted. This is to say that in the conducted retrievals *trf* actually outperformed the Levenberg-Marquardt method.

Instead of following the advise of [44] that the Levenberg-Marquardt algorithm should be the first choice for unconstrained problems, it is rather recommended to stick with the default method *trf* or use *dogbox*, which showed even more comparable performance. Regarding accuracy all methods performed equally well, but in terms of robustness it could be interesting to further analyse the conditioning of the problem.

When it comes to the use of different sets of bounds, these do not show significant changes to the unbounded use of *varpro* in terms of efficiency or accuracy[1]. However, it turned out that whenever bounds were used, the dogbox algorithm slightly outperformed the *trf* method (as it was also stated in [44]). If, the model described in Section 2.1 is advanced in the future to also incorporate scattering by aerosols in the atmosphere, it might simplify the retrieval when physically meaningful bounds are applied to the increased number of fit parameters.

All tests and the theoretical background given in this thesis have focused mainly on the nonlinear part of the problem posed by trace gas retrieval. Therefore an idea for future research could be the modification of the linear least squares solver in *varpro.py*.

At present, the linear problem is solved via the SVD approach (cf. Section 3.1), and it is not possible to apply bound-constraints to the linear parameters (such as $r_0 \in [0,1]$). It would be interesting to analyse the effects that different linear solvers such as QR or even iterative algorithms have on the performance of *varpro*, especially if the complexity of the problem increases in future advancements. Such a change could create the possibility to also have bound-constraints for the linear parameters, which might be advantageous when their number increases.

All of these modifications and possible improvements of *varpro.py* are aimed to be ultimately incorporated into the *Python* version of the BIRRA retrieval algorithm [13] for future trace gas retrieval applications. There are several problems, in which one could really profit from using the variable projection structure instead of a regular nonlinear least squares solver (due to the advantages described in Chapter 4):

One of them would be multi-window fitting, which essentially means that several spectral windows (from one spectrum) are retrieved for one α_{CO_2} simultaneously. In 1979, Golub and LeVeque [40] published a paper introducing the mathematical formalism of such problems. This was explored further in 1992 by Kaufman and Sylvester [41] who called the problem "separable nonlinear least squares with multiple right-hand sides". The idea was later taken up again and expanded in [45]. These papers describe the theoretical procedure for solving separable problems with multiple data sets, in which "the linear parameters are to be specified for each data

[1] This could mean that the used retrieval formalism of scaling total column densities, as proposed in Section 2.3, is already sufficiently well conditioned and therefore does not need any further constraints.

set, but the nonlinear parameters have to minimize the least-squares function for all the data sets" [41].

For trace gas retrieval this would mean fitting the nonlinear parameters, such as the molecular scaling factors α_{CO_2} and α_{H_2O}, for several radiance spectra simultaneously, while the linear variables, e.g. reflectivity coefficients r_0 and r_1, vary for each data set. An implementation of such an algorithm could therefore not only be used for a multi-window fit of one data set, but also for an one window fit of many data sets. Especially for a trace gas like CO_2, which has fairly constant concentrations throughout the atmosphere, this could be a way to retrieve a kind of mean value for certain areas across the globe. Such problems would therefore have a significantly higher number of linear parameters ($\sim$ hundreds to thousands of spectra in one orbit) than nonlinear ones ($\sim$ trace gas species).

Final note: During the research on this thesis many different approaches for solving separable least squares problems have been explored with the intention to find out, whether one might perform best for the specific problem of trace gas retrieval. In all papers (including recent ones) it was always the variable projection algorithm that had the most arguments in favor. For this reason none of the others were described in this thesis and *varpro.py* is considered as the new default solver for the currently used *Python* implementation of BIRRA [13].

Acknowledgements

First of all, I would like to express my gratitude to Prof. Thomas Trautmann for letting me work as a student research assistant at his department at the DLR (German Aerospace Center), giving me the opportunity to write my Bachelor's thesis there, and for reviewing it.

Secondly, I would like to thank my former theoretical physics professor at TUM, Prof. Norbert Kaiser, for his unreserved interest and support of this thesis and for taking the time to engage in my research. Without his contribution, this work would not have been possible.

Furthermore, I am very grateful to my supervisor Dr. Franz Schreier, for motivating me to write my thesis at the DLR. I am much obliged to his many research incentives (not to mention all the books and papers that ended up on my desk because of him) and the continuing support and review of my work during the past months, which really brought me closer to the scientific way of thinking and working.

Also, I really have to thank Dr. Philipp Hochstaffl for the frequent scientific discussions and all the competent explanations, every time I needed help. These discussions clearly helped to deepen my knowledge of the subject.

Lastly, I want to express my gratitude to the Hanns-Seidel-Stiftung for supporting me throughout my entire bachelor studies by granting me a scholarship.

Bibliography

[1] F. Schreier et al. »Py4CAtS—PYthon for Computational ATmospheric Spectroscopy«. In: *Atmosphere* 10.5 (2019), p. 262. DOI: 10.3390/atmos10050262. URL: https://atmos.eoc.dlr.de/tools/Py4CAtS/.

[2] California Institute of Technology Jet Propulsion Laboratory. *Orbiting Carbon Observatory-2*. URL: https://ocov2.jpl.nasa.gov/ (visited on 30/05/2022).

[3] J. Moré and S. Wright. *Optimization Software Guide*. Philadelphia (USA): SIAM, 1993. ISBN: 9780898713220.

[4] G. Golub and V. Pereyra. »The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Variables Separate«. In: *SIAM Journal on Numerical Analysis* 10.2 (1973), pp. 413–432. DOI: 10.1137/0710036.

[5] D. O'Leary and B. Rust. »Variable projection for nonlinear least squares problems«. In: *Computational Optimization and Applications* 54.3 (2013), pp. 579–593. DOI: 10.1007/s10589-012-9492-9.

[6] J. Burrows, U. Platt and P. Borrell. *The Remote Sensing of Tropospheric Composition from Space*. Physics of earth and space environments. Heidelberg (Germany): Springer, 2011. ISBN: 9783642147906.

[7] A. Richter. »Satellite remote sensing of tropospheric composition – principles, results, and challenges«. In: *EPJ Web of Conferences* 9 (2010), pp. 181–189. DOI: 10.1051/epjconf/201009014.

[8] F. Schreier et al. »GARLIC — A general purpose atmospheric radiative transfer line-by-line infrared-microwave code: Implementation and evaluation«. In: *Journal of Quantitative Spectroscopy and Radiative Transfer* 137 (2014), pp. 29–50. DOI: https://doi.org/10.1016/j.jqsrt.2013.11.018.

[9] P. Hochstaffl. »Trace gas concentration retrieval from short-wave infrared nadir sounding spaceborne spectrometers«. PhD thesis. Ludwig-Maximilians-Universität München, 2022. DOI: 10.5282/EDOC.29404.

[10] D. Crisp et al. *Level 1B Algorithm Theoretical Basis: Orbiting Carbon Observatory – 2 & 3 (OCO-2 & OCO-3)*. Ed. by National Aeronautics and Space Administration (NASA). Pasadena, California, 2021.

[11] A. Kuze et al. »Thermal and near infrared sensor for carbon observation Fourier-transform spectrometer on the Greenhouse Gases Observing Satellite for greenhouse gases monitoring«. In: 48.35 (2009), pp. 6716–6733. DOI: `10.1364/AO.48.006716`.

[12] M. Chahine et al. »AIRS: Improving Weather Forecasting and Providing New Data on Greenhouse Gases«. In: 87 (2006), pp. 911–926. DOI: `10.1175/BAMS-87-7-911`.

[13] S. Gimeno García et al. »Near infrared nadir retrieval of vertical column densities: methodology and application to SCIAMACHY«. In: *Atmospheric Measurement Techniques* 4.12 (2011), pp. 2633–2657. DOI: `10.5194/amt-4-2633-2011`.

[14] California Institute of Technology Jet Propulsion Laboratory. *Vital Signs of the Planet - Carbon Dioxide*. URL: `https://climate.nasa.gov/vital-signs/carbon-dioxide/` (visited on 01/06/2022).

[15] P. Hochstaffl et al. »Validation of Carbon Monoxide Total Column Retrievals from SCIAMACHY Observations with NDACC/TCCON Ground-Based Measurements«. In: *Remote Sensing* 10.2 (2018). DOI: `10.3390/rs10020223`.

[16] F. Schreier. personal communication. 2022.

[17] I. Gordon et al. »The HITRAN2016 molecular spectroscopic database«. In: *Journal of Quantitative Spectroscopy and Radiative Transfer* 203 (2017), pp. 3–69. DOI: `10.1016/j.jqsrt.2017.06.038`.

[18] G. Anderson et al. *AFGL atmospheric constituent profiles (0.120 km)*. Tech. rep. Air Force Geophysics Lab Hanscom AFB MA, 1986.

[19] Å. Björck. *Numerical Methods in Matrix Computations*. Vol. 59. SpringerLink Bücher. Springer International Publishing, 2015. ISBN: 9783319050898.

[20] Å. Björck. *Numerical Methods for Least Squares Problems*. Vol. 51. Other titles in applied mathematics. Philadelphia, Pa: SIAM Society for Industrial and Applied Mathematics, 1996. ISBN: 0898713609.

[21] U. Ascher and C. Greif. *A first course in numerical methods*. Vol. 7. Computational science & engineering. Philadelphia: SIAM Society for Industrial and Applied Mathematics, 2011. ISBN: 9780898719987.

[22] A. Ruhe and P. Wedin. »Algorithms for Separable Nonlinear Least Squares Problems«. In: *SIAM Review* 22.3 (1980), pp. 318–337. DOI: `10.1137/1022057`.

[23] K. Levenberg. »A method for the solution of certain nonlinear problems in least squares«. In: *Quarterly of Applied Mathematics* 2.2 (1944), pp. 164–168.

[24] D. Marquardt. »An Algorithm for Least-Squares Estimation of Nonlinear Parameters«. In: *Journal of the Society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441. DOI: `10.1137/0111030`.

[25] J. Moré. »The Levenberg-Marquardt algorithm: Implementation and theory«. In: *Numerical Analysis*. Springer, Berlin, Heidelberg, 1978, pp. 105–116. DOI: `10.1007/BFb0067700`.

[26] J. Moré, B. Garbow and K. Hillstrom. *User guide for MINPACK-1*. USA, 1980. DOI: `10.2172/6997568`.

[27] Madsen, K. and Nielsen, H. and Tingleff, O. »Methods for Non-linear Least Squares Problems«. In: Informatics and Mathematical Modelling, 2004.

[28] M. Branch, T. Coleman and Y. Li. »A Subspace, Interior, and Conjugate Gradient Method for Large-Scale Bound-Constrained Minimization Problems«. In: *SIAM Journal on Scientific Computing* 21.1 (1999), pp. 1–23. DOI: `10.1137/S1064827595289108`.

[29] T. Coleman and Y. Li. »An Interior Trust Region Approach for Nonlinear Minimization Subject to Bounds«. In: *SIAM Journal on Optimization* 6.2 (1996), pp. 418–445. DOI: `10.1137/0806023`.

[30] M. Powell. »A New Algorithm for Unconstrained Optimization«. In: *Nonlinear Programming*. Elsevier, 1970, pp. 31–65. ISBN: 9780125970501.

[31] C. Voglis and I. Lagaris. »A Rectangular Trust Region Dogleg Approach for Unconstrained and Bound Constrained Nonlinear Optimization«. In: *WSEAS International Conference on Applied Mathematics* Vol. 7 (2004). DOI: `10.1201/9780429081385-138`.

[32] N. Sagara and M. Fukushima. »A continuation method for solving separable nonlinear least squares problems«. In: *Journal of Computational and Applied Mathematics* 10.2 (1984), pp. 157–161. DOI: `10.1016/0377-0427(84)90052-9`.

[33] P. Comon, X. Luciani and A. de Almeida. »Tensor decompositions, alternating least squares and other tales«. In: *Journal of Chemometrics* 23.7-8 (2009), pp. 393–405. DOI: `10.1002/cem.1236`.

[34] Y. Shen and T. Ypma. »An Efficient Algorithm for the Separable Nonlinear Least Squares Problem«. In: *Algorithms* 10.3 (2017), p. 78. DOI: `10.3390/a10030078`.

[35] G. Golub and V. Pereyra. »Separable nonlinear least squares: the variable projection method and its applications«. In: *Inverse Problems* 19.2 (2003), R1–R26. DOI: `10.1088/0266-5611/19/2/201`.

[36] V. Pereyra and G. Scherer, eds. *Exponential data fitting and its applications*. Bentham eBooks, 2010. ISBN: 9781608050482.

[37] L. Kaufman. »A variable projection method for solving separable nonlinear least squares problems«. In: *BIT Numerical Mathematics* 15.1 (1975), pp. 49–57. DOI: 10.1007/BF01932995.

[38] F. Schreier. *varpro.py*. Version 2. 2021.

[39] *SUBROUTINE VARPRO*. 9.05.1985. URL: http://www.netlib.org/opt/varpro.

[40] G. Golub and R. LeVeque. *Extensions and Uses of the Variable Projection Algorith for Solving Nonlinear Least Squares Problems*. Stanford, CA, 1979.

[41] L. Kaufman and G. Sylvester. »Separable Nonlinear Least Squares with Multiple Right-Hand Sides«. In: *SIAM Journal on Matrix Analysis and Applications* 13.1 (1992), pp. 68–89. DOI: 10.1137/0613008.

[42] J. Dennis, D. Gay and R. Welsch. »Algorithm 573: NL2SOL— An Adaptive Nonlinear Least-Squares Algorithm«. In: *ACM Transactions on Mathematical Software (TOMS)* 7.3 (1981), pp. 369–383.

[43] SciPy v1.8.0 Manual. *scipy.optimize.leastsq*. 6.02.2022. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.leastsq.html.

[44] SciPy v1.8.0 Manual. *scipy.optimize.least_squares*. 6.02.2022. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html.

[45] L. Kaufman. »Solving separable nonlinear least squares problems with multiple datasets«. In: *Exponential data fitting and its applications*. Ed. by V. Pereyra and Godela Scherer. Bentham eBooks, 2010, pp. 94–109. DOI: 10.2174/978160805048211001010094.