Programming exercise in computational physics
Simo Tukiainen, 014007316

Retrieval of atmospheric profiles using dimension reduction

1 Introduction

Measurement of the Earth's atmosphere are crucial for monitoring and understanding the current state and possible time evolution of the atmosphere. Stratospheric ozone and greenhouse gases such as CO_2 and CH_4 are examples of the important constituents that require continuous global monitoring.

Often the data are collected using remote sensing instruments when interesting parameters like number densities are retrieved from the measured spectrum using a model that contains all the relevant physics. To retrieve key parameters from the data, an inverse problem must be solved. Typically, the inverse problem is ill-posed meaning that there are more unknown parameters than actual information in the data. Often the problem is non-linear too, at least in atmospheric remote sensing applications, and iterative solving methods are needed.

In this work I demonstrate a) how to simulate ground-based direct Sun measurements at short-wave infrared wavelengths and b) how to retrieve vertical information about the atmosphere from the (simulated) measurements using the dimension reduction method. The method is presented in a general form, so that it can easily be adapted for other applications. The simulator and the retrieval method are implemented using Fortran. All implementations are original code by the author. This study was done in 2015 for the course "Programming exercise in computational physics" at University of Helsinki.

2 Atmospheric transmission

The solar radiation entering the atmosphere experiences absorption and scattering from the molecules and particles in the atmosphere. In this work I consider an instrument that directly looks at the Sun and records spectrum at short-wave infrared (SWIR) wavelengths at around 1.6 μm. The scattering is negligible at these wavelengths and I will only deal with absorption from now on.

The absorption from the atmospheric molecules follows the famous Beer-Lambert law. Assuming a discretized atmosphere with n layers, and only one absorbing gas for simplicity, the transmittance in one wavelength is

$$\tau = \exp\Big(-\sum_{i=1}^{n}\sigma_i x_i l_i\Big), \tag{1}$$

where σ_i is the absorption coefficient, or cross section, of the layer i, and x_i and l_i are the corresponding number density and slant length of the layer. In matrix notation for the whole spectral window with m points I have the cross section matrix $\mathbf{S} \in \mathbb{R}^{m \times n}$, densities $\boldsymbol{x} \in \mathbb{R}^n$, and slant lengths $\boldsymbol{l} \in \mathbb{R}^n$. Now the slant densities are

$$\bar{\boldsymbol{x}} = \boldsymbol{l} \circ \boldsymbol{x}, \tag{2}$$

where $\circ$ is the point-wise product, and the transmittance

$$\boldsymbol{\tau} = \exp(-\mathbf{S}\bar{\boldsymbol{x}}), \tag{3}$$

which has the Jacobian

$$\mathbf{K} = \frac{\partial \boldsymbol{\tau}}{\partial \bar{\boldsymbol{x}}} = -\mathrm{diag}(\boldsymbol{\tau})\mathbf{S}, \tag{4}$$

where $\mathbf{K} \in \mathbb{R}^{m \times n}$. At the SWIR wavelengths, the absorption cross-section is a function of pressure and temperature. The fundamental theory behind the absorption line formation and the line shape are covered in numerous textbooks such as Goody and Yung [1995], Liou [2002]. The actual cross section calculation is not relevant for this study and I will take the cross sections as input data. It should be only noted that the retrieval of the vertical information is based on the temperature and altitude dependence, thus *altitude dependence*, of the line shape.

3 Profile retrieval

The ground-based and Sun looking instrument measures solar light modified by the whole atmospheric column of myriad trace gases. Our retrieval problem is then to estimate vertical information from the single spectrum. Given the measured spectrum $\boldsymbol{y} \in \mathbb{R}^m$, where m is the number of wavelengths, I wish to approximate the state vector $\boldsymbol{x} \in \mathbb{R}^n$, where n is the number of atmospheric layers. The state vector $\boldsymbol{x}$ represents the densities of the discretized forward model atmosphere, so that

$$\hat{\boldsymbol{x}} = \mathbf{R}(\boldsymbol{y}, \mathbf{C}_y), \tag{5}$$

where $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ is the retrieved state, $\mathbf{R}$ is the *retrieval method*, and $\mathbf{C}_y \in \mathbb{R}^{m \times m}$ the measurement error covariance. However the measured, or in this case simulated, SWIR spectrum does not contain enough information to resolve all ~50 altitude levels in any meaningful way (the inverse problem is said to be *ill-posed*). A commonly used solution is to regulate, or constrain, the problem using a priori information about the state vector $\boldsymbol{x}$, and use Bayes' theorem to combine information from the prior and measurement in the estimation process

$$\boldsymbol{\pi}(\boldsymbol{x}|\boldsymbol{y}) \propto \boldsymbol{\pi}(\boldsymbol{y}|\boldsymbol{x})\boldsymbol{\pi}(\boldsymbol{x}), \tag{6}$$

where $\boldsymbol{\pi}(\boldsymbol{x}|\boldsymbol{y})$ is called the posterior, $\boldsymbol{\pi}(\boldsymbol{y}|\boldsymbol{x})$ is the likelihood term, and $\boldsymbol{\pi}(\boldsymbol{x})$ the prior term. In the CH_4 profile retrieval the prior is, in practice, our "best guess" profile $\boldsymbol{x}_0 \in \mathbb{R}^n$ having the error covariance $\mathbf{C}_x \in \mathbb{R}^{n \times n}$. Assuming that the prior and error distributions are independent Gaussian distributions and that the error has no bias component, the posterior density can be written in another form

$$\boldsymbol{\pi}(\boldsymbol{x}|\boldsymbol{y}) \propto \exp\left(-\frac{1}{2}\left(\|\boldsymbol{y} - f(\boldsymbol{x})\|^2_{\mathbf{C}_y} + \|\boldsymbol{x} - \boldsymbol{x}_0\|^2_{\mathbf{C}_x}\right)\right), \tag{7}$$

where $f\colon \mathbb{R}^n \to \mathbb{R}^m$ is the forward model. For linear models f, Eq. (7) would lead to a Gaussian distribution, but in practice the forward model f is usually non-linear, at least in atmospheric remote sensing problems. This means that the posterior distribution is non-Gaussian and the optimization problem of Eq. (7) must be solved in iterative way.

The prior term $\|\boldsymbol{x} - \boldsymbol{x}_0\|^2_{\mathbf{C}_x}$ is a widely used way to regulate the retrieval process. The solution is a smooth profile even though both $\boldsymbol{x}$ and $\boldsymbol{x}_0$ are defined in full state space. However, the solution depends significantly on $\boldsymbol{x}_0$ and the relationship between $\mathbf{C}_y$ and $\mathbf{C}_x$. Too loose prior distribution will easily lead to unstable solution, and too tight will skew the result if $\boldsymbol{x}_0$ is far from the truth. Hence, the prior is usually set relatively tight to substantially regulate the solution and to avoid any superfluous oscillation of the posterior.

3.1 Dimension reduction method

The dimension reduction method that I use in this work is based on the decomposition of the prior covariance [see e.g Marzouk and Najm, 2009, Cui et al., 2014]. The prior covariance is first defined in the full state space, i.e. the covariance is a $n \times n$ matrix where n is the number of layers in the model atmosphere. The idea is to truncate the prior covariance so that plausible profile candidates can be drawn from the reduced prior distribution with just a few parameters. Consider a Gaussian prior $\mathbf{X} \sim \mathcal{N}(\boldsymbol{x}_0, \mathbf{C})$, where $\boldsymbol{x}_0$ is the prior mean and $\mathbf{C} \in \mathbb{R}^{n\times n}$ is a positive-definite covariance matrix. The covariance $\mathbf{C}$ can be factorized with

$$\mathbf{C} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T = \sum_{i=1}^{n} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T, \tag{8}$$

$$\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n], \quad \boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \ldots, \lambda_n), \tag{9}$$

where $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$ are column vectors, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0$ are the singular values (or equivalently the eigenvalues) of $\mathbf{C}$. I can now define the *reduction mapping* of order k, $1 \leq k < n$, $\mathbf{P}_k = [\sqrt{\lambda_1}\boldsymbol{u}_1, \ldots, \sqrt{\lambda_k}\boldsymbol{u}_k]$, and then consider the matrix truncation of order k

$$\tilde{\mathbf{C}} = \mathbf{P}_k \mathbf{P}_k^T = \sum_{i=1}^{k} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T, \tag{10}$$

where $\mathbf{P}_k \in \mathbb{R}^{n\times k}$ is the so-called projection matrix, and $\tilde{\mathbf{C}} \in \mathbb{R}^{n\times n}$ approximates the original covariance. This allows as to parametrize the new profile with k parameters

$$\tilde{\boldsymbol{x}} = \boldsymbol{x}_0 + \mathbf{P}_k \boldsymbol{\alpha}, \tag{11}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^k$ and furthermore $\boldsymbol{\alpha} \sim \mathcal{N}(0, \mathbf{I}_k)$. The expectation value

$$\mathbb{E}(\tilde{\boldsymbol{x}}) = \boldsymbol{x}_0 + \mathbb{E}(\mathbf{P}_k\boldsymbol{\alpha}) = \boldsymbol{x}_0 + \mathbf{P}_k\mathbb{E}(\boldsymbol{\alpha}) = \boldsymbol{x}_0, \tag{12}$$

and the covariance

$$\operatorname{cov}(\tilde{\boldsymbol{x}}) = \operatorname{cov}(\mathbf{P}_k\boldsymbol{\alpha}) = \mathbf{P}_k \operatorname{cov}(\boldsymbol{\alpha})\mathbf{P}_k^T = \tilde{\mathbf{C}}. \tag{13}$$

Hence, the profiles drawn using the reduced covariance approximate the original distribution accurately if the neglected singular values are small. Furthermore, the Jacobian in Eq. (4) is straightforward to transform for the parametrized profile

$$\mathbf{K}_k = \frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{\alpha}} = \mathbf{K}\mathrm{diag}(\boldsymbol{l})\mathbf{P}_k \tag{14}$$

being $\mathbf{K}_k \in \mathbb{R}^{m\times k}$.

In the dimension reduction based profile retrieval, I try to find α parameters that minimize the cost function. It is easy to see that the posterior distribution now becomes

$$\boldsymbol{\pi}(\boldsymbol{\alpha}|\boldsymbol{y}) \propto \exp\left(-\frac{1}{2}\left(\|\boldsymbol{y} - f(\tilde{\boldsymbol{x}})\|^2_{\mathbf{C}_y} + \|\boldsymbol{\alpha}\|^2_{\mathbf{I}_k}\right)\right), \tag{15}$$

where $\boldsymbol{y} \in \mathbb{R}^m$ is the measurement, $f\colon \mathbb{R}^n \to \mathbb{R}^m$ is the forward model, and $\mathbf{C}_y \in \mathbb{R}^{m\times m}$ is the measurement error covariance. $\mathbf{I}_k \in \mathbb{R}^{k\times k}$ is the prior covariance of the α parameters i.e. a diagonal unit matrix. In practice, the relative contribution of the prior term in Eq. (15) is quite small. Typically there are many more wavelengths in the spectral fitting than α parameters, and the state vector (profile shape) has a significant impact on the residuals. Thus, in most cases the likelihood term $\|\boldsymbol{y} - f(\tilde{\boldsymbol{x}})\|^2_{\mathbf{C}_y}$ dominates the cost function. While Eq. (7) and Eq. (15) are analogous, the difference is that in the dimension reduction retrieval the likelihood term receives only smooth realization of $\tilde{\boldsymbol{x}}$ but in Eq. (7) the state vector $\boldsymbol{x}$ has no such restriction.

4 Optimization

One of the key benefits of the dimension reduction is that the number of estimated parameters diminishes drastically (e.g. from 50 to 3). Thus, it would be feasible, and highly recommended, to statistically sample the full posterior distribution using the Markov chain Monte Carlo method (MCMC). However, this would be out of the scope of this work and I use derivative-based optimization instead.

4.1 Levenberg-Marquardt method

For the optimization of the α parameters in Eq. (15) I use the the Levenberg-Marquardt method. It is a derivative based, iterative optimization algorithm commonly used for the nonlinear least squares problems, initially proposed by Levenberg [1944]. The iteration is defined

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + (\mathbf{K}^T\mathbf{K} + \gamma_i \mathrm{diag}(\mathbf{K}^T\mathbf{K}))^{-1}\mathbf{K}^T R(\boldsymbol{x}_i) \tag{16}$$

where $\mathbf{K}$ is the Jacobian and $R(\boldsymbol{x}_i)$ is the residual function. In the original version of the method, the parameter γ_i was chosen in each iteration to minimize the cost function. This increases the computational burden and Marquardt [1963] proposed to find such γ_i that simply reduces the cost function, no matter how much. The value of γ_i is usually scaled (multiplied or divided) by the factor of ten, until the cost function decreases, but the choice depends on the application. Commonly, LM algorithms are chosen to stop with respect to some norms e.g. if the difference in residuals $\|R(\boldsymbol{x}_{i+1}) - R(\boldsymbol{x_i})\|$ is small.

5 Implementation

I have implemented all algorithms using Fortran (2003 standard). The code contains five modules in separate files and one main program for testing (Fig. 1).

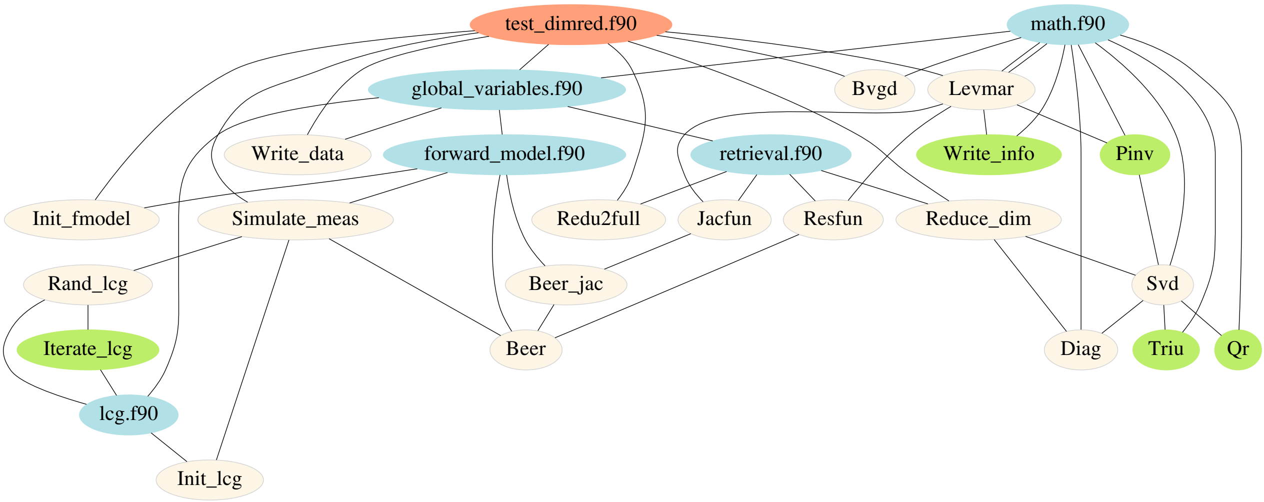

Figure 1: Modules (blue), public functions (light), private functions (green), main program (red), and their connections.

The implementation is only for educational purposes so the physics contain a few simplifications:

- Path lengths inside the atmospheric layers are not calculated nor taken account. I assume that all layers are 1 km thick (and the instrument points straight to the zenith).
- There is only one trace gas (CH_4) and neutral air density in the atmosphere.
- I have an ideal instrument, thus the instrument function is not taken account.

The simulated spectrum is shown in Fig. 2. I have used a narrow bandwidth that contains five CH_4 absorption lines.

The prior covariance is defined using two bivariate Gaussian distribution centered at 25 km (large variation) and 5 km (small variation). This is sufficient for the simulation experiment but with real data the covariance should be approximated using accurate measurements, e.g. balloon observations, or model data. The prior covariance is shown in Fig. 3.

5.1 Key algorithms

In this section I present in more detail the implementation of the algorithms needed in this work.

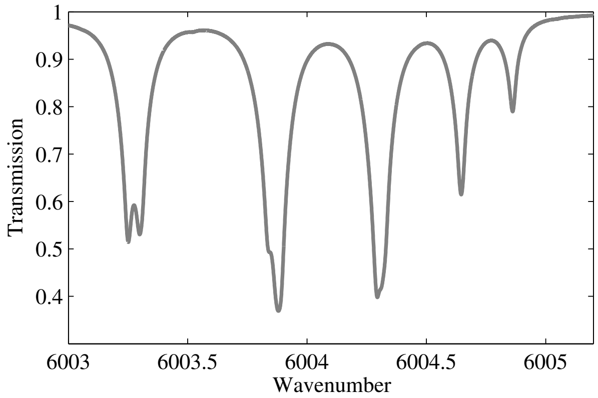

Figure 2: Simulated measurement.

5.1.1 QR decomposition

- **name of the routine:** Qr
- **in file:** math.f90
- **used by:** Svd

The QR decomposition is the key procedure used in this work, and a fundamental matrix operation in general. In the QR decomposition, the input matrix $\mathbf{A}$ is decomposed into an orthogonal and triangular matrix:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1\mathrm{n}} \\ a_{21} & a_{22} & \cdots & a_{2\mathrm{n}} \\ \vdots & \vdots & \ddots & \vdots \\ a_{\mathrm{m}1} & a_{\mathrm{m}2} & \cdots & a_{\mathrm{mn}} \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1\mathrm{m}} \\ q_{21} & q_{22} & \cdots & q_{2\mathrm{m}} \\ \vdots & \vdots & \ddots & \vdots \\ a_{\mathrm{m}1} & a_{\mathrm{m}2} & \cdots & a_{\mathrm{mm}} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1\mathrm{n}} \\ 0 & r_{22} & \cdots & r_{2\mathrm{n}} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & r_{\mathrm{mn}} \end{bmatrix} \tag{17}$$

The QR decomposition is the basis of the iterative QR algorithm for finding eigenvalues, and it can be used for finding singular values, too. I have implemented the QR decomposition as the modified Gram-Schmidt (MGS) process [Golub and Van Loan, 1996]. Applying MGS to the column vectors of a full column rank matrix yields the QR decomposition.

5.1.2 Singular value decomposition

- **name of the routine:** Svd
- **in file:** math.f90
- **used by:** Pinv, Reduce_dim

SVD is the main algorithm in this work because it is used in the factorization of the prior covariance. SVD is also needed in the pseudo-inverse function used by the Levenberg-Marquardt algorithm. Singular value decomposition is a factorization of a $m \times n$ matrix $\mathbf{A}$ to the form

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T, \tag{18}$$

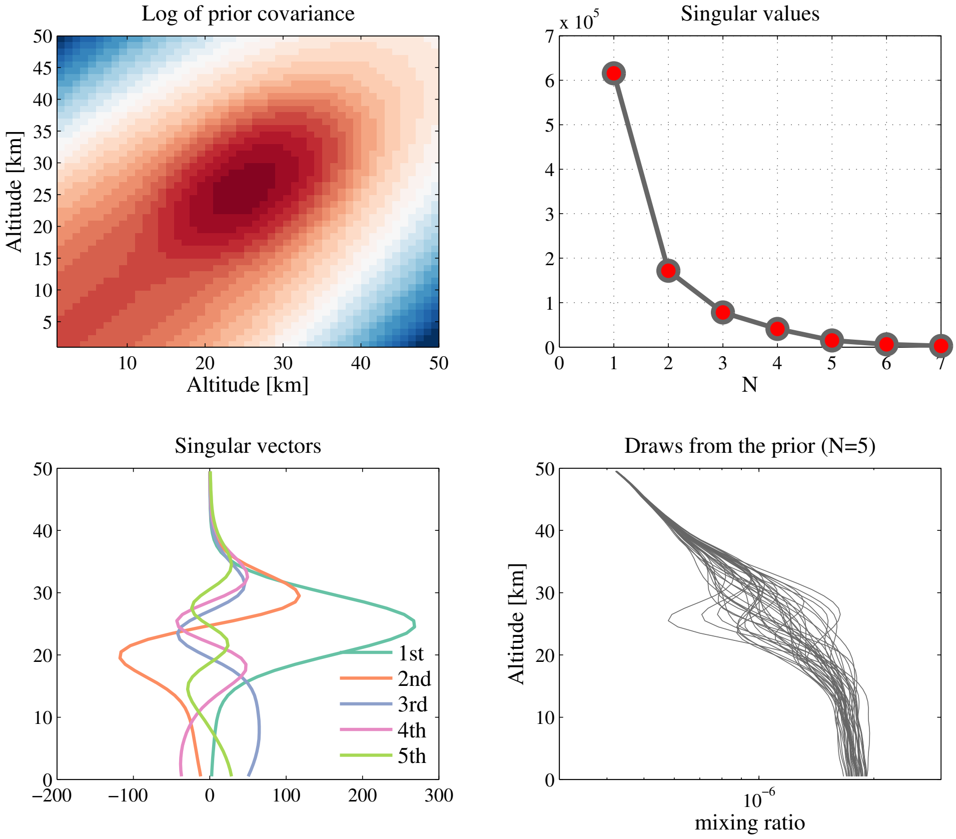

Figure 3: Logarithm of the CH_4 prior covariance (upper-left), seven largest singular values (upper-right), five largest singular vectors (lower-left), and random draws from the prior using the five singular vectors (lower-right).

where $\mathbf{U}$ is a $m \times m$ unitary matrix and $\mathbf{V}^T$ is a $n \times n$ unitary matrix, so that

$$\mathbf{U}^T\mathbf{U} = \mathbf{I} \tag{19}$$

and

$$\mathbf{V}^T\mathbf{V} = \mathbf{I} \tag{20}$$

The matrix $\mathbf{S}$ is a $m \times n$ rectangular diagonal matrix and its diagonal values are called the singular values of $\mathbf{A}$. SVD is closely related to the eigenvalue decomposition of a symmetric matrix and the numerical implementations of both decompositions are similar, too. The general idea of all SVD algorithms is to find square roots of eigenvalues of $\mathbf{A}^T\mathbf{A}$ without actually having to have to compute it.

The concept of the SVD implementation in this work is to use the QR decomposition on $\mathbf{A}$ to iteratively "pull" $\mathbf{U}$ out from the left and then use QR on $\mathbf{A}^T$ to "pull" $\mathbf{V}$ out from the right. This process makes $\mathbf{A}$ lower triangular and then upper triangular alternately, and eventually $\mathbf{A}$ becomes diagonal matrix with the singular values on the diagonal.

5.1.3 Pseudo inverse

- **name of the routine:** Pinv

- **in file:** math.f90
- **used by:** Levmar

Pseudo inverse is the general inverse of a $m \times n$ matrix $\mathbf{A}$, where the result is a $n \times m$ matrix $\mathbf{B}$ so that $\mathbf{ABA} = \mathbf{A}$, $\mathbf{BAB} = \mathbf{B}$, and $\mathbf{AB}$ and $\mathbf{BA}$ are Hermitian. Pseudo inverse is convenient to implement using the SVD:

$$\mathbf{A} = \mathbf{USV}^T \tag{21}$$

so it follows

$$\mathbf{A}^{-1} = \mathbf{VS}^{-1}\mathbf{U}^T, \tag{22}$$

where $\mathbf{S}^{-1}$ is trivial to calculate because $\mathbf{S}$ is a diagonal matrix.

5.1.4 Levenberg-Marquardt

- **name of the routine:** Levmar
- **in file:** math.f90
- **used by:** test_dimred (main program)

The idea of the Levenberg-Marquardt method is explained in Sect. 4.1. I have tried to implement the LM algorithm as generally as possible. It takes the residual and Jacobian functions and the data structure as input. The variable “theta” contains initial, and finally optimized, values of the fitted parameters. The error covariance of the fitted parameters is estimated from the optimum with $(\mathbf{KK}^T)^{-1}$, where $\mathbf{K}$ is the Jacobian at the optimum.

5.1.5 Bivariate Gaussian distribution

- **name of the routine:** Bvgd
- **in file:** math.f90
- **used by:** test_dimred (main program)

The prior covariance is generated using two Gaussian bivariate distributions. A bivariate Gaussian distribution can be defined in 2-dimensional nonsingular case

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_x)^2}{\sigma_x^2} + \frac{(y-\sigma_y^2)^2}{\sigma_y^2} - \frac{2\rho(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y}\right]\right), \tag{23}$$

where μ_x and μ_y describe the mean point of the distribution, σ_x and σ_y are the deviations in x and y direction, and ρ is the correlation length.

5.1.6 Measurement noise

- **name of the routine:** Simulate_meas
- **in file:** forward_model.f90
- **used by:** test_dimred (main program)

In this work the measurement noise model is additive, normally distributed noise. Random numbers can be drawn from the standard normal distribution using the simple method

$$r_n = \sum_{i=1}^{N}(r_i) - \frac{N}{2} \tag{24}$$

where r is an ordinary (pseudo) random number between 0-1. For $N \to \infty$, the distribution of the sum becomes Gaussian. This is a famous rule in statistics and the proof is not relevant here.

5.1.7 Random number generator

- **name of the routine:** Init_lcg, Rand_lcg
- **in file:** lcg.f90
- **used by:** Simulate_meas

As a random number generator I use the the linear congruential generator. It is a simple method for generating pseudo random numbers. The number are generated using the recurrence relation:

$$X_{n+1} = (aX_n + c) \mod m \tag{25}$$

where $m, m > 0$ is called the "modulus", $a, 0 < a < m$ the "multiplier" and $c, 0 \leq c < m$ the "increment". The starting value $X_0, 0 \leq X_0 < m$ is called the "seed". Here $X_0 = 123123$ and the other parameters are taken from Press et al. [1992]: $m = 4294967296$, $a = 1664525$, and $c = 1013904223$.

5.2 Input data

The program needs following input data in order to simulate the measurement:

- atmosphere (file **atmos.dat**). It contains altitudes, "true" density profile that I use in the simulation of the measurement, prior profile, and neutral air density profile.
- cross sections (file **abs_coef.dat**)

5.3 Output

Running the program will write information about simulation set-up and the retrieval results to several text files in the output/ folder. The output files are shown in Table 1. If the program is launched multiple times, the old files will be always overwritten. The results can be visualized with the Matlab function **plot_dimred_results.m** which is included in the root directory.

file name	dimensions	description
d.dat	1	number of used principal components
alt.dat	nalt	altitude vector
dens.dat	nalt	“true” profile used in simulation
prior.dat	nalt	prior profile
air.dat	nalt	air density profile
profile.dat	nalt	retrieved profile
t.dat	nwl	simulated transmission
wl.dat	nwl	wl vector
r.dat	nwl+d	residual
C.dat	nalt×nalt	prior covariance matrix
P.dat	nalt×d	projection matrix
q.dat	nalt	singular values of C

Table 1: Output files

5.4 Example run

This is a example session how to obtain the code, compile, run, and visualize main results with Matlab. Red text is commentary.

```
$ wget https://dl.dropboxusercontent.com/u/27553525/dim_red_retrieval.tar.gz
$ tar -xzvf dim_red_retrieval.tar.gz                unpack
$ cd dim_red_retrieval/                             move to the root folder
$ make                                              compile all modules
$ ./test_dimred                                     run the program

 Give std of noise. Signal is between 0-1 and the noise is additive ~ N(0,std^2)
 Common value is e.g. 0.001 but you can try different values..
0.0001                                              set noise level..
  Give number of principal components (~2-5 is good)
5                                                   give number of components..
 ----------------------------------
 Iteration      RSS          lam
 ----------------------------------
     0    5632.5160474      1.0E-03
     1       2.2121584      1.0E-03
     2       1.1082433      1.0E-04
     3       1.0207569      1.0E-05
     4       1.0039673      1.0E-06
     5       1.0019448      1.0E-07
     6       1.0019454      1.0E-06

$ matlab -nodesktop                               launch matlab
$ plot_dimred_results                             plot some results
```

6 Results

The shape of the proposal profiles depend on the covariance itself and the number of principal components that are used to approximate the original covariance. The available information content in the signal is limited by the measurement noise. Figure 4 shows example retrievals using different number of principal components (from top row to down: 1, 2, 3, 5) and noise $\sigma = 0.001$. The panels on the right show the corresponding residuals. It can be easily seen from the residual where the information about the vertical structure is coming. Furthermore, 1-2 principal components is not enough to find the true profile, but no more than 3 pieces of information can be retrieved due to the noise.

7 Discussion

In this study I implemented and tested a dimension reduction based retrieval method. This kind of technique can be used to decrease the number of estimated parameters in ill-posed inverse problems. For testing the method I generated a noisy transmission signal and created a prior covariance that was decomposed using the SVD. How well the method can find the true atmosphere depends on the covariance structure (and marginally on the prior profile itself), number of used principal components, and the measurement noise.

Because in this work the main motivation was the actual implementation of the used algorithms, I somewhat simplified the physics of the problem and did not perform very comprehensive analysis on the results. However, the minimization seem to work as it should and the retrieved profiles looks reasonable.

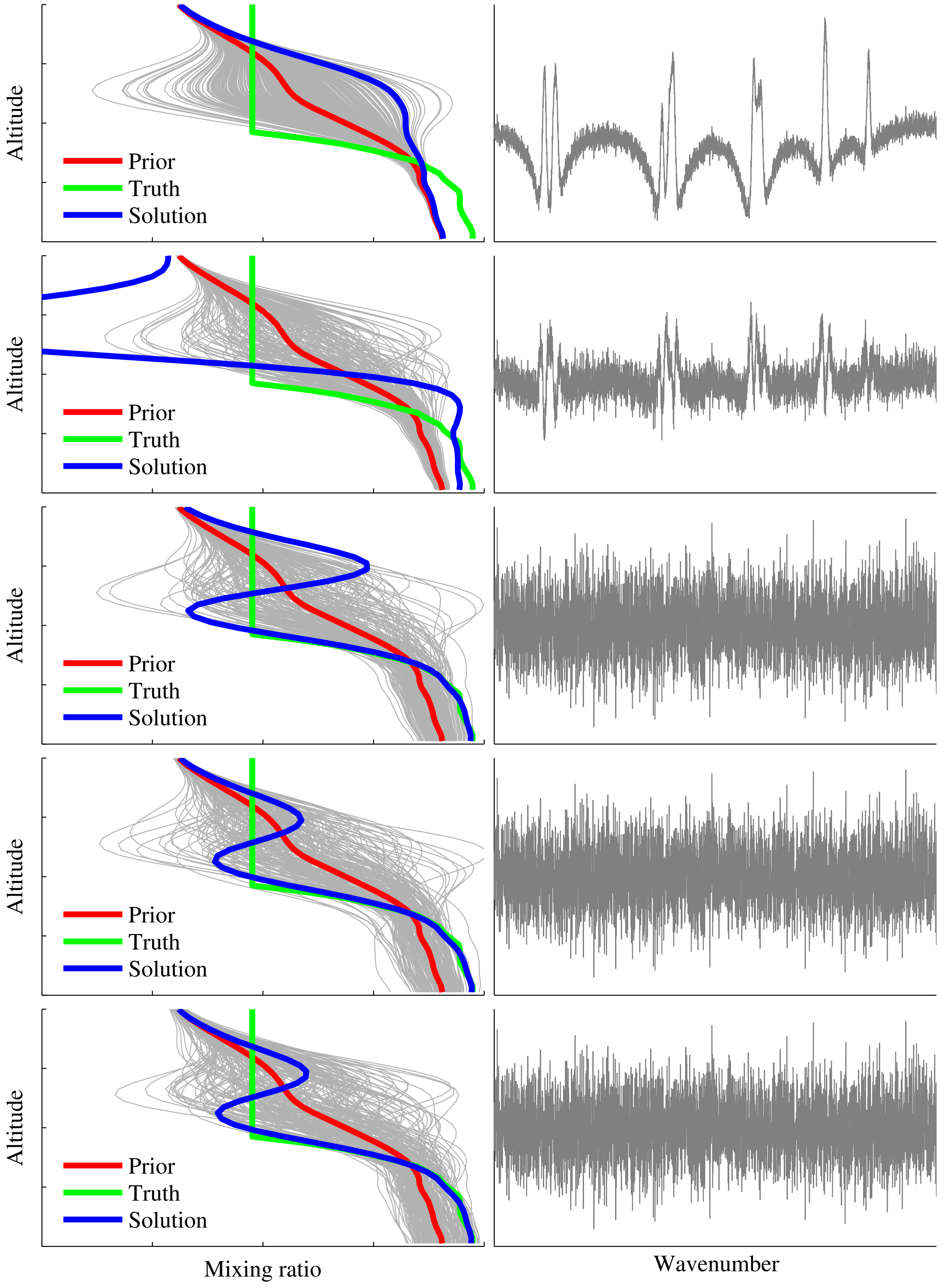

Figure 4: Left panels: prior profile (red), solution (blue), truth (green) and random draws from the prior (grey). Right panels: corresponding residuals. Different rows represent different number of principal components: 1 (uppermost), 2, 3, 4, and 5 (lowermost).

References

T. Cui, J. Martin, Y. M. Marzouk, A. Solonen, and A. Spantini. Likelihood-informed dimension reduction for nonlinear inverse problems. *Inverse Problems*, 30(11):114015 pp. 28, 2014. ISSN 0021-9991. doi: doi:10.1088/0266-5611/30/11/114015. URL `http://iopscience.iop.org/0266-5611/30/11/114015/`.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore and London, 3 edition, 1996.

R. M. Goody and Y. L. Yung. *Atmospheric radiation: Theoretical basis*. Oxford University Press, New York, 2 edition, 1995.

K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.

K. N. Liou. *An introduction to Atmospheric Radiation*. Elsevier, New York, 2002.

D. W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *J. Soc. Indust. Appl. Math.*, 11:431–44, 1963.

Y. M. Marzouk and H. N. Najm. Dimensionality reduction and polynomial chaos acceleration of bayesian inference in inverse problems. *Journal of Computational Physics*, 228(6):1862 – 1902, 2009. ISSN 0021-9991. doi: http://dx.doi.org/10.1016/j.jcp.2008.11.024. URL `http://hdl.handle.net/1721.1/59814`.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in FORTRAN, The Art of Scientific Computing*. Clarendon Press, Oxford, 1992.