Technical Note on:

High level description of the new functionalities to be implemented in MIPAS Level 2 processor

IROE - GS 98 - 11

15 February 1999

Issue 2

Delivery of the Study:

"Development of an Optimized Algorithm for Routine P,T and VMR Retrieval from MIPAS Limb Emission Spectra"

Prepared by:

Name	Institute
Piera Raspollini	IROE - CNR
Marco Ridolfi	IROE - CNR
Bianca Maria Dinelli	ISM - CNR

Approved by:

Name	Institute
Bruno Carli	IROE - CNR

CONTENTS

1. Introduction

Functionalities candidate for the implementation in MIPAS Level 2 processor were listed in the Technical Note entitled: *'New functionalities of ORM_SDC. Recommendations for implementation of the new functionalities in MIPAS Level 2 processor'* (1st September 1998), in which recommendation for their development was provided in the light of their usefulness and of the amount of work needed for their implementation. That document was used as starting point for discussions at the 8th Progress Meeting of MIPAS ORM Study and at the 31st MIPAS Scientific Advisory Group Meeting (ESTEC, 14th and 15th October 1998).

The final result has been the selection of a set of functionalities to be implemented in MIPAS Level 2 processor. The functionalities under consideration have been grouped into five sets depending on their impact on Level 2 processor:

1. The following set of functionalities will have impact on Level 2 Retrieval Components Library (RCL) and will be implemented internally to the ORM code.
   - Fit of VMR profiles to a user-defined sub-set of tangent altitudes
   - Handling of logical masks at the Jacobian and χ^2 computation level
   - Improvement of FOV convolution algorithm: definition of the simulation grid in the tangent altitude domain
   - Improvement of FOV convolution algorithm: definition of FOV shape
   - Profiles regularization
   - Implementation of a compressed frequency grid for cross-sections storage and radiative transfer computation. Consequent use of an optimized algorithm for interpolation / AILS convolution.

1. The following functionalities will affect the Framework part of the Level 2 algorithm. These functionalities will be implemented as self-standing modules of the scientific software package (i.e. will have a corresponding 'scientific' version but will not affect the ORM code):
   - Computation of temperature / pressure induced errors in VMR retrievals
   - Algorithm for construction of optimized Occupation Matrices (OMA)
   - Implementation of the new format of the MW databases

1. The following functionality will affect the Framework part of the Level 2 algorithm and will not have a corresponding 'scientific' version:
   - Handling of Initial Guess (IG) profile data with Optimal Estimation

1. The following functionalities will be implemented in the SDC (Scientific Development Code) version of the ORM, but not in Level 2 baseline:
   - Evaluation of retrieval error components and total error budget
   - Interpolation of the retrieved profiles to a user-defined vertical grid

1. The following functionalities which where considered in the mentioned TN of 1st September 1998 have subsequently been dropped:
   - Fit of temperature profile to a user-defined sub-set of tangent altitudes.

   It was decided to drop this functionality from the Level 2 processing because, since all chain of VMR retrievals uses temperature and tangent pressure profiles retrieved by pT retrieval, the decrease of the vertical resolution in pT retrieval leads to the decrease of the vertical resolution in all VMR retrievals.
   - Handling of logical masks at the level of tangent altitude dependent radiative transfer calculations (irregular fine grids): the handling of altitude dependent irregular grids

increases the complexity of the program making doubtful the computing time saving expected from this optimization (see TN mentioned above).

This note describes how the ORM code and its inputs and outputs have to be modified in order to introduce the functionalities belonging to set 1.
Besides it provides the high level description of the algorithms relating to the other considered functionalities.
Concerning the functionalities already discussed in the TN of 1st September 1998, we refer to that document for the discussion of the rationale that motivates their implementation. For the functionalities not mentioned in the above TN, a brief discussion of the rationale is provided here.

2. High level description of the functionalities that will be implemented internally to the ORM and will have impact on Level 2 Retrieval Components Library

2.1 Fit of VMR profiles to a user-defined set of tangent altitudes

Retrieval of VMR profiles in correspondence of a (sub-) set of the actual measurement tangent altitudes (in alternative to the whole set) is an ORM functionality discussed by the study team since the beginning of the ORM code development. This is the reason why ORM modules have been conceived in compliance with this functionality, even if, for technical and schedule reasons, during subsequent code evolution and refinements this functionality has not been considered any longer. In particular the code already contains an input logical vector (*lfit*) identifying the tangent altitudes at which the profile is fitted.
Please note that the implementation of this function (as for all other ORM functions) is independent from the presence of corrupted spectra in the actual scan. In fact the ORM always assumes as uncorrupted the measured spectra recorded in its input files. If a measured scan contains corrupted spectra, these must be filtered-out when constructing the ORM input files. The logical vector *lfit* refers therefore to the scan pattern defined in the ORM input files and not to the scan actually measured by MIPAS. *lfit* is a user-defined input parameter and must refer to the actual (uncorrupted) set of tangent altitudes.
For the implementation of the fit at a user-defined sub-set of tangent altitudes, all the ORM modules have been critically reviewed and modified whenever necessary. It has been found that only few code modifications were necessary.
The code parts to be modified are mainly contained in the routine '***concandcol***' calculating the values of vertical columns, concentrations and related variance-covariance matrices to be reported in the final output files.

Code modifications:

- Module ***concandcol***:
  - ⇒ modified algorithm for construction of perturbed VMR profiles required for calculation of concentrations and columns derivatives with respect to VMR: logical vector *lfit* is now taken into account.
  - ⇒ The vector *rxmod* has been dropped from the interface of this routine.
  - ⇒ Vector *rxmod* is now calculated at the beginning of ***concandcol*** by interpolating the *rxbase* profile to the *rzmod* altitude levels.
  - ⇒ Corrected factor in calculation of output concentration: $10^{-6} \rightarrow 10^{-11}$
- Module ***retr_vmr***:

⇒ Corrected call to ***concandcol***: dropped vector *rxmod.*
⇒ Dropped declaration of vector *rxmod*.
⇒ Corrected ***fwdmdl_vmr*** calls: dropped vector *rxmod.*

- Module ***fwdmdl_vmr***:
  ⇒ Dropped declaration of vector *rxmod*
  ⇒ Dropped vector *rxmod* from the interface
  ⇒ Dropped calculation of *rxmod* (formerly at the end of this module)

Interfaces modifications:

Modified interfaces of modules ***concandcol, fwdmdl_vmr*** as explained above (dropped vector *rxmod*).

2.2 Handling of logical masks at the Jacobian and χ^2 calculation level

In this section we describe the modifications enabling the ORM to handle microwindows defined in the recently released databases which are characterized by altitude-dependent size and shape (logical masks). Logical masks are handled only at Jacobian and χ^2 calculation level, i.e. Jacobian rows and residuals corresponding to skipped observations in the logical masks are forced to zero. Forward simulations are still calculated in correspondence of all the microwindow grid points, without introducing optimizations such as tangent altitude dependent irregular grids, which are of doubtful efficiency.

Code modifications:

- Created a new module (***read_masks_pt(vmr)***) for reading and initialization of logical masks files. Logical masks, preliminary extracted from the MW-database, are contained in MW-specific files (one ASCII file per MW). Logical masks files are contained in a dedicated directory defined by an environment variable (ORM_MASKS). This module is called in the retrieval main flow ***retr_pt(vmr)***.
- Created a new module (***jacob_modif_pt(vmr)***) which forces to zero the Jacobian rows corresponding to skipped observations in the logical masks. This module is called in the retrieval main flow ***retr_pt(vmr)***.
- Module ***difchi_pt(vmr)***: this subroutine calculating the residuals vector has been modified so that residuals corresponding to skipped observations in the logical masks are forced to zero. The interface of this subroutine has been modified as well: added integer variable *imask(imxi,imxgeo,imxmw)* containing logical masks definition, added variable *nmasked* representing the total number of masked observations.
- Modules ***chisq_pt(vmr)*** and ***output_pt(vmr)***: modified calculation of the number of degrees of freedom according to: *ifrede = iobs - imasked - itop*. Added variable *imasked* at the interface of both modules.
- Module ***newparest_pt(vmr)***: corrected call to ***chisq_pt(vmr)***. Added variable *imasked* at the interface.
- Minor changes have been implemented also in subroutines ***cross_pt*** and ***inigas_pt*** in order to avoid program crashes when dealing with p,T MWs with no CO_2 lines (no lines of the main gas). The interfaces of these subroutines have not been changed.

Please note that these changes do not affect the residuals values reported in the output files, which are always obtained from the actual differences between observations and simulations independently of the logical masks information.

In addition to the above modifications which are the only ones necessary for handling logical masks and the MWs defined in the recently released MW databases, some further modifications have been applied to the ORM mainly aiming at the minimization of work required for conversion of spectroscopic data released by Karlsruhe study team. With these modifications the ORM handles the database of spectroscopic data with the same approach used for the other databases (LUTs, irregular grids and logical masks).

These modifications consist in adapting the module ***finput_pt(vmr)*** and its sub-unit ***r_spect_pt(vmr)*** for reading the spectroscopic database in the same format used by Karlsruhe team: unused fields are discarded. The spectroscopic lines relating to the individual MWs are now contained in MW-specific files (one ASCII file per MW) with the same name of the MW to which they refer. These files are recorded in a dedicated directory whose name is defined by an environment variable (ORM_SPECT_DB).

The format of the spectroscopic line lists generated by the Karlsruhe study team is described in Table 5 of document (3) with the only modification that in the first record, the microwindow label must be read with the format '(A12,5X)' instead of '(A6,5X)'.

2.3 Improvement of FOV convolution algorithm: definition of the simulation grid in the tangent altitude domain

In the ORM, FOV convolution is performed analytically first interpolating the spectra calculated at the tangent pressures to determine the dependence of the spectra as a function of altitude and then integrating this analytical function with the FOV pattern.

In the old program the spectrum interpolation around a given tangent altitude is performed by drawing a parabola between three spectra at three contiguous tangent altitudes. Only in the case of H_2O retrieval in troposphere, additional spectra are simulated in correspondence of tangent altitudes located just in the middle of each couple of contiguous tangent altitudes and interpolation is built using 5 spectra and a fourth order polynomial.

The tests with RFM code showed that this approximation is too rough in some cases (tropospheric H_2O and CO_2, the latter being particularly affected by the strong variation of temperature in troposphere). Besides the program is not flexible in presence of non standard MIPAS limb-scanning sequences, and can become inaccurate when corrupted sequences are considered.

The aim of the planned modification is the implementation of more efficient and conservative criteria for the treatment of those cases in which extra simulated spectra are needed for the interpolation of the FOV.

Since FOV convolution is more critical in troposphere, it is necessary to distinguish between troposphere and the rest of the atmosphere. Above tropopause, the MIPAS standard limb scanning grid (equal to 3 km) is sufficient for an accurate FOV convolution. Additional simulated spectra have to be used in the case of corrupted sequence, or, in general, when the distance between two contiguous tangent altitudes is greater than a maximum value read from input. In troposphere, the maximum allowed difference between two tangent altitudes for accurately computing FOV convolution depends on the variation of VMR profile of the molecules contributing to the spectrum and hence depends on which retrieval is considered.

Three input parameters have been added: *rtropopause,* indicating the height level delimiting troposphere (a conservative high value can be used, valid for all the atmospheric models), *rint,* indicating the maximum allowed difference between the tangent altitudes of two simulated spectra in troposphere (on the light of the OFM - RFM test comparisons *rint*=2 km in p-T retrieval and O_3 VMR retrieval, *rint*=1.5 km for H_2O retrieval, *rint*= 3 km for the other VMR retrievals), *rintup*, indicating the maximum allowed difference between the tangent altitudes of two simulated spectra above tropopause (a reasonable value for variable *rintup* is provided by *rbase*, i.e. the extension of the FOV pattern).

Routine ***occusim_pt(_vmr)***, which defines the tangent altitudes, other than the ones corresponding to observations, in correspondence of which spectra have to be simulated, has been modified in order to fulfil the following criterion: two contiguous tangent altitudes must not be more distant than *rintup* above tropopause and than *rint* in troposphere.

The possibility of having corrupted sequences imposes also to review the hydrostatic equilibrium formula which is used in routines ***updprof_pt***, ***mkplev_*** and ***chbase_*** for updating pressures or altitudes. The approximated formula used also in the previous version of the ORM code introduces maximum errors smaller than 0.5 % when applied to layers with thickness equal or smaller than 4 km, but the error increases when the thickness of the layers is increased. In order to skip this problem, when layers are thicker than 4 km, pressure corresponding to the higher level is computed adding intermediate levels inside the layer for applying the hydrostatic equilibrium formula to altitude steps smaller than 4 km.

Code modifications:

- routines ***occusim_pt(_vmr)***: these routines have been strongly modified
  - ⇒ Modified criterion for determining the set of spectra to be simulated, in addition to the ones corresponding to the observations, as follows: two contiguous tangent altitudes must not be more distant than *rintup* above tropopause and than *rint* in troposphere.
  - ⇒ Added computation of the array *isimfov(imxlmb,imxmw,imxsfo)* that, for each tangent altitude corresponding to an observation and for each microwindow, defines how many and what spectra have to be used for computing FOV convolution. The parameter *imxsfo* has been added in the files *parameters_pt(_vmr)* and indicates the maximum number of spectra (plus 2) used for making the interpolation. The array *isimfov* is used by routines ***fov_pt(_vmr)***.
  - ⇒ Changes in the interface: dropped variables *imaingas* and *rbase*, added *isimfov*, *rtropopause, rint* and *rintup*.

- routine ***updprof_pt***

⇒ since the number of simulated geometries is no longer hardwired in the program, the call to routine ***occusim_pt*** has been added where the vector containing the tangent altitudes of all simulated geometries is updated.
⇒ Added call to routine ***tcgeo_pt*** where the computation of arrays *igeocder* and *igeotder* is updated.
⇒ Modified updating of vector *rzbase* using hydrostatic equilibrium in presence of layers characterised by a pressure variation equivalent to 4 km: in this case the height corresponding to the higher level is computed adding intermediate levels inside the layer for applying the hydrostatic equilibrium formula to altitude steps smaller than 4 km.
⇒ Changes in the interface: added variables *lfit, lfitgeo, iocsim, irowmw, iobs, isimfov, igeotder, igeocder, rtropopause, rint, rintup, rztang*

- routines ***fov_pt(_vmr)***: for the changes introduced in these routines, see Sect.2.4.

- routine ***intcon_ pt(_vmr)***: for the changes introduced in these routines, see Sect. 2.4.

- routine ***chbase_pt(_vmr)***:

⇒ modified updating of pressure profiles using hydrostatic equilibrium in presence of layers thicker than 4 km: in this case the pressure corresponding to the higher level is computed adding intermediate levels inside the layer for applying the hydrostatic equilibrium formula to altitude steps smaller than 4 km.

- routines ***mkplev_pt(_vmr)***:

⇒ modified updating of pressure profiles using hydrostatic equilibrium in presence of layers thicker than 4 km: in this case the pressure corresponding to the higher level is computed adding intermediate levels inside the layer for applying the hydrostatic equilibrium formula to altitude steps smaller than 4 km.

- routines ***input_pt(_vmr)***: see Sect. 2.4

- routines ***fwdmdl_pt(_vmr):*** see Sect. 2.4

- routines ***retr_pt(_vmr)*** : see Sect. 2.4

Affected I/O

Three new input variables, *rtropopause, rint* and *rintup* have been added in all input files settings_*.dat.

2.4 Improvement of FOV convolution algorithm: definition of FOV shape

Up to now FOV pattern was assumed to be a trapezium, defined by means of the input parameters *rbase* (its greatest base) and *rsl* (the half-difference between the two bases). In order to be safe in case that at a later stage it is found that FOV pattern deviates significantly from a trapezium, we decided to foresee in the ORM the option for reading as input any FOV shape, tabulated at discrete points to be linearly interpolated.
Even if this 'broken line' includes the trapezium as a particular case, we decided to keep the possibility of working as before with a trapezium, because this allows to exploit some optimisations in the computation of analytical convolution.
Also when the tabulated FOV pattern is used, FOV convolution of spectrum (as well as derivatives of the spectrum with respect to VMR, continuum parameters and temperature) is performed, for each spectral point, with an analytical convolution computed by routine ***intcon_pt(_vmr)***, called by routine ***fovn_pt(_vmr)*** inside routine ***fov_pt(_vmr)*** (see below)**.**
Also the derivative of the spectrum with respect to tangent pressure is computed analytically deriving the expression of the convoluted spectrum with respect to the tangent altitude and multiplying the result by the derivative of the tangent altitude with respect to tangent pressure.

Code modifications:

- routines ***fov_pt(_vmr)***: we report the modifications to be introduced in this routine for implementing the two new functionalities described in this section and in Sect. 2.3. Routines ***fov_pt(_vmr)*** have been strongly modified for including the possibility of interpolating the spectrum in the altitude domain using a general number *n* of spectra and the option of using a tabulated FOV pattern.

  ⇒ For each tangent altitude corresponding to an observation (*iactugeo*), the choice of the simulated spectra to be used for computing the interpolation is made using array *isimfov* computed by routine ***occusim_pt(_vmr):*** *isimfov(iactugeo,imw,1)* indicates the number *n* of spectra to be used for interpolation; *isimfov(iactugeo,imw,2)* indicates the index of the simulated geometries corresponding to *iactugeo*; *isimfov(iactugeo,imw, 3 -> n+2)* indicates the indices of the simulated geometries used for making the interpolation. Using this information, the vectors *igeofov(1 -> n)* and *rtangfov(1 -> n)* are filled with the indices of the simulated geometries used for the interpolation and the corresponding tangent altitudes.

⇒ Sub-routine ***fov3_pt(_vmr)*** has been replaced by sub-routine ***fovn_pt(_vmr)***

⇒ Sub-routines ***fov4_vmr*** and ***fov5_vmr*** have been dropped from ***fov_vmr***

⇒ Changes in the interface: added variable *isimfov, nfovinc,rfov,ranginc,lfovtab*.

- routine ***fovn_pt(_vmr):*** it returns the FOV convoluted spectrum, as well as the derivatives of the spectrum with respect to tangent pressure, continuum parameters and temperature (VMR), independently of the number of spectra to be used for the interpolation.

The only modifications introduced in routine ***fovn_pt(_vmr)*** with respect to the old ***fov3_pt(_vmr)*** are the following ones:

⇒ use of a general value *n* of spectra for the interpolation,

⇒ the computation of the derivative of the spectrum with respect to tangent pressure made using the value *rpt* computed in ***intcon_pt(_vmr)*** routine,

⇒ a different call to ***intcon_pt(_vmr),***

⇒ the correction of an error in the call to routine ***intcon_pt*** for computing the derivative of the spectrum with respect to temperature (the perturbed tangent altitude, instead of the non perturbed one, is provided)**.**

- routine ***intcon_pt(_vmr)***: these routines have been strongly modified.

⇒ if a n-points interpolation (with *n*=3, 4 or 5) has to be computed and FOV tabulated spectrum is not provided (*lfovtab*= .false.), the computation of FOV convolution is carried-out as in the old ***intcon_vmr***, if a n-points (with n > 5) interpolation has to be computed or the tabulated FOV pattern is provided, the calculation is made as follows: the contribution of each line segment *j* of the broken line delimited in altitude by z_1 and z_2 is computed by integrating the product of interpolated spectrum with the segment line:

$$\int_{z_1}^{z_2} \sum_{i=1}^{n} c(i) \cdot z^{i-1} \cdot (a + b \cdot z) = \sum_{i=1}^{n} c(i) \cdot z^{i} \cdot \left(\frac{a}{i} + \frac{bz}{i+1} \right) \Bigg|_{z_1}^{z_2}$$

where $z_1 = rzc + ranginc(j)$, $z_2 = rzc + ranginc(j+1)$, *rzc* being the central tangent altitude in correspondence of which FOV is computed, *ranginc(j)* and *ranginc(j+1)* being the x-values of two contiguous FOV tabulated points, $\sum_{i=1}^{n} c(i) \cdot z^{i-1}$ represents the interpolated polynomial of degree *n*, $(a + b \cdot z)$ is the line segment of the 'broken line' we are considering, *a* and *b* depend on z_1, z_2 and the y-values of FOV tabulated points.

⇒ It is assumed that even if the standard trapezium shape is used (and hence *lfovtab* is set to *.false.*), its tabulation as a 'broken line' is also provided.

⇒ Added computation of the derivative of the spectrum with respect to tangent altitude, *rpt*.

⇒ Changes in the interface: added variable *rpt, nfovinc,rfov,ranginc,lfovtab*.

- routines ***input_pt(_vmr)***:

⇒ added reading of variables *nfovinc, rfov, ranginc, nucl, lfovtab, rtropopause, rint* and *rintup* by subroutine ***r_settings_pt(_vmr)***. Added the new variables in common *settings_p(_v)*

- routines ***fwdmdl_pt(_vmr):***

⇒ modified calls to routines ***fov_pt(_vmr)*** as listed above

⇒ changes in the interface: added variables *isimfov, nfovinc, rfov, ranginc, nucl, lfovtab*

- routines ***retr_pt(_vmr)*** :

⇒ modified calls to routines ***fwdmdl_, occusim_*** and ***updprof_pt*** as listed above

I/O files modifications:

The following parameters have been added in all the input files settings_*.dat

- the logical variable *lfovtab*: if it is true, it means that FOV tabulated function is provided
- *nfovinc* (integer*4), equal to the number of points used for tabulating FOV pattern
- *rfov(1->nfovinc)*(Real*8): vector containing the y-values of FOV tabulated function
- *ranginc(1->nfovinc):* vector containing the x-values of FOV tabulated function, relative to the centre of FOV pattern

It is assumed that even if the standard trapezium shape is used, its tabulation as a 'broken line' is also provided. On the contrary, if FOV tabulated function is provided, *rbase* represents the vertical range in which FOV pattern is significantly different of 0.

Example of tabulation for the trapezium FOV pattern with *rbase=4* and *rsl=0.6*:

```
 F
 4
 0.d0,  1.d0 , 1.d0,  0.d0
-2.d0, -1.4d0, 1.4d0, 2.d0
```

2.5 Profiles regularization

In some cases the retrieved profiles are oscillating more than what can be reasonably expected from the physical point of view. As outlined in the ESA study 12055/NL/CN, the oscillations are confirmed by the shape of the Modulation Transfer Functions (MTF) which show values greater than 1 (1). Tikhonov regularization (2) can be adopted in MIPAS retrievals with the objective of constraining the MTF to values equal or smaller than 1. Test calculations are however needed in order to tune the strength of the Tikhonov constraint (analysis of the MTF shape as a function of the strength of Tikhonov constraint). The implementation of this new algorithm in the ORM requires limited changes and it is useful to have this option accessible. The inversion formula used at each retrieval iteration for deriving the new parameters estimate, in the Levenberg-Marquardt (LM) approach is:

$$\hat{\mathbf{y}} = \mathbf{A}^{-1} \cdot \mathbf{K}^T (\mathbf{V}^n)^{-1} \mathbf{n} \qquad (2.5.1)$$

with

$$\mathbf{A}_{i,j} = \left(\mathbf{K}^T (\mathbf{V}^n)^{-1} \mathbf{K}\right)_{i,j} \qquad \text{for } i \neq j \text{ and}$$

$$\mathbf{A}_{i,i} = \left(\mathbf{K}^T (\mathbf{V}^n)^{-1} \mathbf{K}\right)_{i,i} \cdot (1+\lambda) \qquad (2.5.2)$$

where λ is the LM damping factor and is adjusted during the iterations. Please refer to chapter 4 of document (2) for the definition of the other symbols appearing in equations (2.5.1) and (2.5.2).

When using Tikhonov regularization the inversion formula (2.5.1) must be modified accordingly to:

$$\hat{\mathbf{y}} = \left(\mathbf{A} + \gamma \mathbf{L}^T \mathbf{L}\right)^{-1} \cdot \mathbf{K}^T (\mathbf{V}^n)^{-1} \mathbf{n} \tag{2.5.3}$$

where **L** is a ‘regularization operator and γ is a parameter that can be used for tuning the strength of Tikhonov constraint.

$\mathbf{L}^T\mathbf{L}$ is a square matrix whose dimension is equal to the number of fitted parameters, each diagonal element of this matrix corresponding to a fitted parameter. In the ORM implementation matrix $\mathbf{L}^T\mathbf{L}$ consists of three non-zero square blocks refering to the three different groups of parameters that are being fitted: VMR, atmospheric continuum and instrumental offset in VMR retrievals. In the case of p,T retrieval matrix $\mathbf{L}^T\mathbf{L}$ consists of four non-zero square blocks which refer to p,T, atmospheric continuum and instrumental offset. In both cases, matrix $\mathbf{L}^T\mathbf{L}$ is a strip-diagonal matrix whose elements different from zero are only the diagonal elements and the first off-diagonal ones:

$$\mathbf{L}^T\mathbf{L} = \left(\begin{array}{ccc} \begin{matrix} 1 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -1 & 0 \\ \vdots & \cdots & 0 & -1 & 2 & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{matrix} & 0 & 0 \\ 0 & \begin{matrix} 1 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -1 & 0 \\ \vdots & \cdots & 0 & -1 & 2 & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{matrix} & 0 \\ 0 & 0 & \begin{matrix} 1 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -1 & 0 \\ \vdots & \cdots & 0 & -1 & 2 & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{matrix} \end{array}\right)$$

Both diagonal and first off-diagonal elements of each individual block can be scaled by user-defined input parameters.

The ORM first builds matrix $\mathbf{L}^T\mathbf{L}$ taking into account the actual number of fitted parameters and the input scaling factors, subsequently this matrix is normalized to the initial guess value of the unknown parameters *par(i)* and tuned according to an input parameter η used for setting the overall strength of the constraint:

$$\gamma \left(\mathbf{L}^T\mathbf{L}\right)_{i,j} \Rightarrow \frac{\left(\mathbf{L}^T\mathbf{L}\right)_{i,j}}{\eta^2 \, par(i) \cdot par(j)}. \tag{2.5.4}$$

While implementing Tikhonov algorithm in the ORM, an additional modification has been done in order to cope with a different requirement. Since it has been decided that in Level 2 products the retrieved profiles must be consistent with the final value of the chi-square and of the residuals, the final profiles update, formerly performed after the iterations loop, has been removed from the ORM flow.

2.5.1 Code modifications

The following code / interfaces modifications were necessary for implementing the Tikhonov regularization functionality:

1. File **settings_xx.dat**. Entries have been added defining: a switch for enabling / disabling profiles regularization option, the weight η and the parameters used for scaling diagonal and off-diagonal elements of matrix $\mathbf{L}^T\mathbf{L}$ (6 parameters in VMR retrievals and 8 parameters in p,T retrieval).
2. Routines ***finput_pt(vmr)*** have been adapted for reading the new entries in files **settings_xx.dat**. Interfaces of ***finput_pt(vmr)*** have been modified in order to transfer to the main retrieval flow ***retr_pt(vmr)*** the new entries of **settings_xx.dat**: new common block 'regular_p(v)'.
3. A new program module (name = ***regsetup_pt(vmr)***) has been created. This module defines matrix (2.5.4) (program variable *reg(imxtop,imxtop)*) based on:
   - current set of parameters (*rxpar*)
   - input scaling factors
4. Routines ***amodif_pt(vmr)*** have been modified so that, if Tikhonov regularization is enabled (through the input switch *lifreg*) matrix *reg* (2.5.4) is added to matrix *ra*. Matrix *reg* and the logical switch *lifreg* have been added at the interface of these subroutines.
5. Modules ***retr_pt(vmr)***. Added common block 'regular_p(v)'. Added call to routine ***regsetup_pt(vmr)*** immediately after call to ***guesspar_pt(vmr)***. Modified calls to module ***amodif_pt(vmr)***. Dropped last parameters / profiles update. Matrix *ra* is now backed-up after evaluation in modules ***abcalc_pt(vmr)***. Expression for updating Marquardt damping factor *rlambda* in case of micro-iterations has been modified.

2.6 Compression of the irregular frequency grid for cross-sections storage and radiative transfer computation. Consequent use of an optimized algorithm for interpolation / AILS convolution

This modification exploits the fact that, after the introduction of the MW specific irregular spectral grids, only a reduced number of spectral grid points is actually required for cross section and radiative transfer calculations. Therefore, the dimension of cross-section array as well as high resolution spectrum and derivatives can be reduced accordingly.

Cross-sections, as well as spectra and derivatives (before AILS convolution), are all computed and stored on the compressed frequency grid (constisting of only the '1' points of the irregular grid). The reconstruction of the entire spectrum and derivatives is performed simultaneously with the convolution of the high resolution quantities with the AILS function, the result being the spectrum and the derivatives on the regular coarse grid.
For the implementation of the functionality mentioned in this section, please refer to document (7).

2.7 Other minor changes / corrections

Correction of an error in the derivatives of the spectra with respect to continuum, temperature and VMR

The spectrum at a given tangent altitude is not affected only by the value of the retrieved parameter (temperature, VMR or continuum) at its tangent altitude and the tangent altitudes above, but, because of FOV convolution, also by the value of the retrieved parameter at the tangent altitudes below which are interested by FOV convolution at the considered tangent altitude.

On the contrary, in the old program the derivatives of the spectrum at a given tangent altitude with respect to the retrieved parameter corresponding to the tangent altitudes below were set to 0.

In order to correct the program, the following modifications have to be made:

- routines ***tcgeo_pt (gcgeo_vmr)***:

⇒ the dimension of variables *igeotder(imxgeo,2), igeocder(imxgeo,2) (igeogder(imxgeo,2))* has to be modified as follows: *igeotder(imxgeo,3), igeocder(imxgeo,3) (igeogder(imxgeo,3)).*

⇒ for each spectrum derivative (with respect to temperature **=t*, continuum **=c*, gas VMR **=g*) the variable *igeo*der(jgeo,1)* and *igeo*der(jgeo,2)* represent, as in the old program, respectively the highest and lowest parameter to be considered for the considered derivative before FOV convolution, the new *igeo*der(jgeo,3)* represents the lowest parameter affecting the spectrum *jgeo* after FOV convolution. This variable is computed determining the highest parameter *j<mpar* which satisfies the condition:

$$(rztang(mpar) - rztang(j)) \geq rbase / 2 ,$$

where *rbase* is the vertical range where the FOV pattern is significantly different of 0, *mpar* is the lowest parameter affecting the spectrum before FOV convolution, *mpar=igeo*der(jgeo,2)* and *rztang* is the vector containing the tangent altitudes, from the highest to the lowest one.

⇒ Changes in the interfaces: added variables *rztang* and *rbase*.

- routines ***fov_pt(_vmr)***:

⇒ changed dimensions of variables *igeotder, igeocder (igeogder).*

- routines ***fovn_pt(_vmr):***

⇒ changed dimensions of variables *igeotder, igeocder (igeogder)*

⇒ the do-loops on parameters with respect to which temperature and continuum (VMR) derivatives are computed are defined between indices *igeo*der(jgeo,1)* and *igeo*der(jgeo,3).*

- routines ***retr_pt(_vmr)***:

⇒ changed dimensions of variables *igeotder, igeocder (igeogder)*

⇒ modified call to routine ***tcgeo_pt(_vmr)*** as described above.

- routines ***fwdmdl_pt(_vmr)***:

⇒ changed dimensions of variables *igeotder, igeocder (igeogder).*

3. High level description of the functionalities that will be implemented as self standing modules of the scientific s/w package and will affect the Framework part of Level 2 processor

3.1 Computation of temperature- and pressure-induced errors in VMR retrievals

A generic retrieved VMR profile **y** is obtained through the inversion formula:

$$\mathbf{y} = \left(\mathbf{K}^T \mathbf{V}_{obs.}^{-1} \mathbf{K}\right)^{-1} \mathbf{K}^T \mathbf{V}_{obs.}^{-1} \mathbf{n} \equiv \mathbf{D}\mathbf{n} \tag{3.1.1}$$

where **n** is the residuals vector, $\mathbf{V}_{obs.}$ is the VCM of the observed spectra and **K** is the jacobian of the VMR retrieval.

An uncertainty $\Delta(\mathbf{p},\ \mathbf{T})$ on the assumed tangent pressures and temperatures, translates into an error $\Delta\mathbf{n}$ on the simulated spectra and therefore into an error $\Delta\mathbf{y}$ on the retrieved profile equal to:

$$\Delta\mathbf{y} = \mathbf{D}\cdot\Delta\mathbf{n} = \mathbf{D}\cdot\mathbf{C}\cdot\Delta(\mathbf{p},\ \mathbf{T}) \tag{3.1.2}$$

where **C** is the matrix accounting for p,T error propagation in the simulated spectra of VMR retrieval and contains the derivatives:

$$C_{i,j} = \frac{\partial S_i}{\partial \begin{pmatrix} P \\ T \end{pmatrix}_j} \tag{3.1.3}$$

The index '*i*' identifies the fitted spectral points (as a function of frequency for all the microwindows and all the tangent altitudes) and the index '*j*' identifies the retrieved tangent altitudes.

In equation (3.1.2) we have assumed **D** as locally independent from p,T (always true for small errors $\Delta(\mathbf{p},\ \mathbf{T})$).

As the error on the retrieved p,T is in our case described by a VCM $\mathbf{V}_{PT}$, the corresponding VCM $\mathbf{V}_y^{'}$ relating to **y** and due to p,T error is given by:

$$\mathbf{V}_y^{'} = \mathbf{D}\mathbf{C}\mathbf{V}_{PT}(\mathbf{D}\mathbf{C})^T \equiv \mathbf{E}\mathbf{V}_{PT}\mathbf{E}^T \tag{3.1.4}$$

where we have defined **E = DC**. **E** is the matrix transforming p,T error into VMR error. The dimensions of this matrix are:

E: (16 VMR retrieved points) x (32 p,T retrieved points) x (4 bytes/datum) = 2 Kb

In principle, matrix **E** depends on:

- current atmospheric status (p,T and VMR)
- set of adopted MWs in VMR retrieval (Occupation Matrix)

We discuss here below the different strategies that can be considered for handling matrix **E** in MIPAS Level 2 processor.

3.1.1 Strategy A: Creation of a database of derivatives (3.1.3) for construction of matrix C

This strategy consists in:

- assuming matrix **C** as p, T (and VMR) independent,
- creating in Level 2 framework a database of derivatives (3.1.3) required for reconstruction of matrix **C** as a function of the current set of used MWs (occupation matrix of VMR retrieval) at the actual tangent altitudes. The dimensions of that database can be estimated as:

  (100 coarse grid pts. / MW) x (100 MWs / database) x (5 databases) x (16 limb-views) x (32 p,T parameters) x (4 bytes / datum) ≅ 102 Mb

- after the VMR retrieval has been completed, matrix **E** can be evaluated based on matrix **D** which is always available at the end of VMR retrieval,
- $\mathbf{V}_y^{'}$ is then evaluated using equation (3.1.4).

3.1.2 Strategy B: Creation of a database of matrices E

This strategy consists in:

- assuming matrix **E** as p,T (and VMR) independent,
- creating in Level 2 framework a database of matrices **E**: since matrix **E** depends on the chosen set of MWs (occupation matrix), as many **E** matrices as many pre-defined occupation matrices must be stored in Level 2 framework. Given the relatively small size of matrix **E**, this should not be source of concern.
- $\mathbf{V}_y^{'}$ is then evaluated using equation (3.1.4).

3.1.3 Discussion A vs. B

Strategy A is more flexible and can be employed also in case occupation matrices are computed on-line. However A involves a more significant computing effort than B and the database to be stored in Level 2 framework is probably larger than in the case B.
The relative accuracy of the two methods must be assessed, however comparable accuracies are expected from the two methods.

3.1.4 Software tool for evaluation of p,T-induced errors in VMR retrievals

The self-standing software tool for evaluation of p,T- induced errors in VMR retrievals will be implemented as a sub-module of the tool for the evaluation of the various error components and of total error budget (see Sect. 5.1). This software has not been implemented yet. However, we foresee the following list of inputs / outputs of the sub-module evaluating p,T- induced errors in VMR retrievals:

Inputs:

- ORM input files '**mwoccmat_xx.dat**' (containing the actual occupation matrix)
- Results of p,T retrieval total error budget (are these data available in Level 2 Framework ?)
- p,T error propagation matrix (matrix **E** of Sect. 3.1)

Outputs:

- Matrix $\mathbf{V}_y^{'}$.

As a starting baseline, the sub-module evaluating p,T- induced errors in VMR retrievals will be built based on the approach B described above and its accuracy will be tested with respect to atmospheric variability.

3.2 Algorithm for construction of optimized Occupation Matrices (OMA)

For the description of the Algorithm for construction of optimized Occupation Matrices (OMA Vers.2.0, named also KOMA: Karlsruhe Optimized Occupation Matrix Algorithm), please refer to document (5).

3.3 Implementation of the new format of the MW databases

For the description of the new MW database format, please refer to document (6).

4. Functionality that will affect the framework part of Level 2 algorithm and will not have a corresponding 'scientific' version.

4.1 Handling of Initial Guess (IG) profile data

The scientific version of this function will not be implemented because the scientific code has only the capability of handling individual scans, whereas the elaboration of a suitable strategy for handling initial guess profiles requires consideration of the whole orbit. Data analysis at the 'orbit' level is handled only by the industrial Level 2 processor.
For the analysis of a given scan, the ORM code needs the following atmospheric profiles as input:

1. Pressure and temperature profiles
2. Continuum profiles for microwindows used in p,T retrieval
3. H_2O VMR profile
4. Continuum profiles for microwindows used in H_2O retrieval
5. O_3 VMR profile
6. Continuum profiles for microwindows used in O_3 retrieval
7. HNO_3 VMR profile
8. Continuum profiles for microwindows used in HNO_3 retrieval
9. CH_4 VMR profile
10. Continuum profiles for microwindows used in CH_4 retrieval
11. N_2O VMR profile
12. Continuum profiles for microwindows used in N_2O retrieval
13. VMR profiles for other contaminants,

which can be used in the different retrievals either as a first guess of the profiles that are going to be retrieved or as assumed profiles of the atmospheric model (profiles of interfering species and p,T profiles in the case of VMR retrievals).
For each of these profiles both the a-priori estimate (given by either the IG2 profiles or, preferably, if available, ECMWF profiles extended with IG2 profiles in the altitude ranges not covered by ECMWF data) and the result of the most recent measurement (obtained either from the retrieval of the previous scans or from a previous retrieval of the same scan) are available.
Up to now the strategy has been to use whenever possible the most recent measurement (in practice the a-priori estimate was used only for the retrieval of the first sequence or when previous retrievals were unsuccessful).
On the light of the fact that in some cases the errors of the retrieved profiles may be very large (VMR retrieval at very low altitudes, continuum retrieval), we now propose to change the concept of 'most recent measurement' into the concept of 'best estimate'. As it will be discussed later, the 'best estimate' coincides with the most recent measurement for the input files of type 1 (pressure and temperature profiles used as assumed profiles in VMR retrievals), with the a-priori

profiles for the continuum profiles associated to the microwindows used in the different retrievals, and for all the others is equal to the optimal estimation between the most recently measured profiles and the a-priori ones.
The optimal estimation method consists in weighting the retrieved profile, with its VCM, with the pre-stored profile, which will be characterized by a VCM with big diagonal values.
The optimal estimations of the profiles have to be determined not only at the beginning of each scan analysis, but also after each VMR retrieval, because the retrieved VMR profile is used as assumed profile for subsequent retrievals.
As discussed in the Technical Note on High Level Algorithms (3), the Optimal Estimation is not adopted for the determination of the results of Level 2 retrievals because it may introduce biases when averaging different results. Furthermore MIPAS limb scanning retrievals have redundancy of measurements and do not depend on the aid of a-priori information, therefore the choice of what a-priori information is added to the retrieved quantity can be left to the user. However, when selecting the assumed profile of interfering species and the initial guess profiles of the retrieved species, we are users of the results of previous retrievals and in this case we are authorized to apply optimal estimation without entering in contradiction with our stated principles.

Below the description of the algorithm which computes the Optimal Estimation between a-priori and retrieved profiles is provided.

Inputs:

- ⇒ Last retrieved profile x^r (profiles from retrievals up to *n* scans backwards can be used, where *n* is provided by the users)
- ⇒ simplified VCM of the retrieved profile $\boldsymbol{V}^r$, whose diagonal elements and the first off-diagonal ones are set equal to the corresponding elements of the VCM of the retrieved profile, the other elements are set to 0.
- ⇒ Interval of time elapsed since the scan in which the profile x^r was retrieved.
- ⇒ First guess profile x^f of the previous retrieval (used for extending the VCM $\mathbf{V}^r$ for the values of the profile above the highest tangent altitude)
- ⇒ ECMWF profile x^e (if available)
- ⇒ climatological profile IG2 x^i, selected according to the actual latitude/longitude position of a scan.
- ⇒ VCM $\boldsymbol{V}^c$ of the a-priori profile (i.e. an appropriate merging of the ECMWF and IG2 profiles), characterized by equal elements along the diagonal and equal values (which can be different from the diagonal ones) for the first off-diagonal elements, the other elements being 0.

Outputs:

- ⇒ the best estimation of the profile to be used as first guess or as assumed profile of the retrieval

Procedure:

1. Definition of the a-priori profile x^c.

if x^e is not available, then:

$x^c = x^i$
else
$x^c = x^e$ (in the altitude range covered by the ECMWF profiles)
$x^c = x^i * \alpha$ (in the altitude range not covered by the ECMWF profiles)
end if

The grid of x^c is defined by the grid of x^e in the altitude range covered by the ECMWF profiles and by the grid of x^i in the altitude range not covered by the ECMWF profiles; the number of points of this new grid is named *igrid*.
In order to avoid discontinuities, the merging of the ECMWF and IG2 profiles is matched scaling the values of the IG2 profile in correspondence of the highest point of the ECMWF profile and the points above by a factor α equal to the ratio between the value of the highest point of the ECMWF profile and the corresponding value of the IG2 profile (that may be obtained interpolating in the IG2 profile).

2. Correction of the VCM of the retrieved profile $\boldsymbol{V}^r$ due to latitude variation.

If the profile x^r has been retrieved in a previous scan (profiles from retrievals up to *n* scans backwards can be used), it means that it has been measured at a different latitude, and hence the estimated total retrieval error associated with this profile may underestimate its real error which is given by the total estimated retrieval error plus the error due to the latitudinal variation. In this case, the elements of the matrix $\boldsymbol{V}^r$ will be multiplied by a factor $\left(1+\frac{\boldsymbol{t}}{\boldsymbol{t}_0}\right)$ proportional to the time interval elapsed since the measurement of the profile x^r (the constant t_0 is a user input).

3. Interpolation of the retrieved profile x^r on the grid of the a-priori profile x^c (the new profile is named x^{ri})

The interpolation rules are the same as used in the ORM code, i.e. logarithmic interpolation with pressure between the nearest values for temperature and VMR, linear interpolation with pressure for continuum; altitude is re-built using hydrostatic equilibrium starting from the lowest altitude of the a-priori profile x^c.
The interpolated profile x^{ri} is related to the retrieved profile x^r by the following equation:

$$\boldsymbol{x}^{ri} = \boldsymbol{J}\boldsymbol{x}^r ,$$

where $\boldsymbol{J}$ is the matrix (of dimensions *igrid x ibase*) which transforms the retrieved profile (on a grid of *ibase* points) to the interpolated one (on a grid of *igrid* points): the element (*i, k*) of matrix $\boldsymbol{J}$ is given by:

$$\boldsymbol{J}_{ik} = \frac{\partial \boldsymbol{x}_i^{ri}}{\partial \boldsymbol{x}_k^{r}}$$

4. Extension of the VCM of the retrieved profile above the highest retrieved tangent altitude.

$\boldsymbol{V}^r$ provides the variance and covariance of the retrieved profile only for the retrieved 0tangent altitudes. The values of the retrieved profile above the highest retrieved tangent altitude are obtained by scaling the first-guess profile x^f with a factor equal to the ratio between the value of the retrieved profile at the highest tangent point and the corresponding value of the first-guess profile. Therefore the quadratic errors of these points are obtained by scaling the quadratic error at the highest tangent altitude ($\boldsymbol{V}_{11}^{r}$) with the square of the same factor:

$$\boldsymbol{V}_{ii}^{r\_ext} = \boldsymbol{V}_{11}^{r} \cdot \left(\frac{\boldsymbol{x}_i^f}{\boldsymbol{x}_n^f} \right)^2 ,$$

n is the index of the highest tangent altitude in the retrieved grid (dimension *ibase*) and $i < n$.

Correlations of the points of the profile above the highest tangent altitude, which, strictly speaking, should be set to 1, can be set to 0.5, in order to reduce a possible source of instability in the determination of the profile.

5. Determination of the VCM of the profile x^{ri}, $\boldsymbol{V}^{ri}$.

Matrix $\boldsymbol{V}^{ri}$ is obtained performing the following matrix product:

$$\boldsymbol{V}^{ri} = \boldsymbol{J}\, \boldsymbol{V}^{r\_ext}\, \boldsymbol{J}^T$$

6. Optimal estimation between profiles x^{ri} and x^c.

The optimal estimation (OE) of the profile to be used as first guess or as assumed profile of the retrieval is obtained as follows:

$$\boldsymbol{x}_{OE} = \left(\boldsymbol{V}^{c^{-1}} + \boldsymbol{V}^{ri^{-1}} \right)^{-1} \left(\boldsymbol{V}^{c^{-1}} \boldsymbol{x}^c + \boldsymbol{V}^{ri^{-1}} \boldsymbol{x}^{ri} \right)$$

In the case of temperature and pressure profiles, OE method is used only for the determination of the first-guess profiles for p, T retrieval, while the assumed p, T profiles in VMR retrievals are the p, T values retrieved in the current scan.

The reason of the different treatment of p, T profiles according to the different use as first-guess or as assumed profiles is that the retrieval of minor constituents from MIPAS measurements is based on the simultaneous and reliable measurement of the temperature profile. For this reason it is not possible to rely on any a-priori information. Indeed, if pressure and temperature retrieval fails or if these profiles are measured with too big uncertainties, VMR retrievals will not be performed.

On the contrary, in the case of VMR retrievals, where effects of interference should be limited by the optimized choice of microwindows, if the previous VMR retrieval fails or returns a profile with big uncertainties, the subsequent retrievals will be performed anyway. In this case, the use of a-priori information for the definition of the VMR assumed profiles of the interfering key species allows to minimize the effects of interference. Therefore, after each VMR retrieval the OE between the retrieved profile and the a-priori profile of that VMR is computed again and the obtained profile is used in the subsequent retrievals of the actual scan.

Continuum profiles do not have to be treated with the optimal estimation method, because these profiles are strongly undetermined and their contribution to the spectrum is very low (even if the fit of these profiles ‘helps’ the retrieval procedure). Therefore, the a-priori continuum profiles, and not the retrieved continuum profiles, are used as first-guess for the subsequent scans.

5. Functionalities that will be implemented only in the Scientific Development Code (ORM_SDC)

5.1 Evaluation of retrieval error components and total error budget

The recommendations arising from the 30th MIPAS Scientific Advisory Group meeting (ESTEC, 25th and 26th June 1998) regarding total retrieval error (see SAG minutes MIN-MIPAS-0030), can be summarized by the following statements:

- the random error affecting the retrieved profiles has three components:
  - ⇒ The first random component is due to measurement noise. This component is evaluated as part of the inversion procedure (Gauss-Newton method) by the retrieval program. The ORM output files already contain the retrieved profile(s) variance-covariance matrix obtained from measurement noise (derived in Level 1b processing) without scaling by the actual χ^2 of the fit. This variance-covariance matrix will be included in Level 2 products.
  - ⇒ The second random component of profiles error is due to temperature error propagation in VMR retrievals. This component will be evaluated, on-line, by a function implemented in Level 2 processor (Framework) and included in Level 2 products (see Sect.3.1).
  - ⇒ The third random component of VMR profiles error is due to the errors on retrieved tangent pressure. Even if there are no explicit recommendations from the SAG regarding this error component, the ORM study team recommends to use tangent pressure error for defining the actual size of pressure error bars relating to the retrieved points in the VMR profiles. For particular applications (such as building global maps on pressure surfaces) where pressure error can be hardly handled, the pressure error itself must be propagated in the VMR domain using the same formalism adopted for temperature error propagation (see later).
- Systematic error on the retrieved profiles has several components:
  - ⇒ **Contaminants**: this error component is due to imperfect assumption of the contaminants profiles in the forward model.
  - ⇒ **Spectroscopic errors**: are due to errors in spectroscopic data used by the forward model.
  - ⇒ **Gain error**. Is due to imperfect calibration of the observed spectra.
  - ⇒ **Model errors**. These are purely systematic errors biasing the simulated observations. For the moment only errors arising from neglecting Non-LTE in the forward model are expected to belong to this group.

  Systematic errors in contaminants profiles, spectroscopic data and gain have some 'random' character in the sense that, in principle, their value may vary from retrieval to retrieval. For this reason the rule with which they combine may vary as a function of the application. This prevents the calculation of a general purpose error budget.

  Systematic errors affecting the retrieved profiles will not be calculated by the on-line Level 2 processor. A database will be developed containing systematic error profiles relating to VMR profiles retrieved using different (pre-stored) occupation matrices. Level 2 products will contain a reference to that database. Systematic errors will not be available for generic occupation matrices which eventually could be constructed on-line by a function implemented in the Level 2 processor.

A scientific software tool will be developed for evaluating the different error contributions explained above and the resulting total error budget. Purpose of the present section is to describe the algorithms which will be implemented in this software tool. The tool will evaluate separately:

1. Temperature / pressure induced errors in VMR retrievals
2. Systematic error components

3. Total error budget
as a function of:

- error quantifiers reported in the microwindow database,
- current occupation matrix, p-,T-error propagation matrix
- current accuracy of temperature / pressure retrieved profiles
- user input parameter defining the error components to be taken into account for the evaluation of total error.

The part of this tool calculating temperature and pressure induced errors in VMR retrieval will constitute a scheme for the implementation of the corresponding function in Level 2 processor framework. If necessary, the tool can be used for the generation of a database of systematic errors to be attached to profiles retrieved with different (pre-stored) occupation matrices.

5.1.1 Temperature and pressure induced errors in VMR retrievals
For the evaluation of this type of errors, please refer to Sect. 3.1.

5.1.2 Systematic error components
The microwindow database contains quantifiers for the evaluation of systematic errors affecting the retrieved profiles. All the quantifiers have the form:

$$d_{\varepsilon_k} = \frac{1}{VMR(z)} \cdot \frac{\partial VMR}{\partial \varepsilon_k} \cdot \Delta\varepsilon_k \qquad (5.1.1)$$

where ε_k identifies the systematic error source. In case of a multi-MW retrieval in which N_{MW} microwindows are used at altitude z, the VMR error $\sigma_{\varepsilon_k}(z)$ induced by source ε_k at altitude z is given by:

$$\frac{\sigma_{\varepsilon_k}(z)}{VMR(z)} = \frac{\sum_{j=1}^{N_{MW}} \frac{\left(d_{\varepsilon_k}(z)\right)_j}{\left(\sigma_N^2(z)\right)_j}}{\sum_{j=1}^{N_{MW}} \frac{1}{\left(\sigma_N^2(z)\right)_j}} \qquad (5.1.2)$$

In the case of **model errors,** which have known sign, $\left(d_{\varepsilon_k}(z)\right)_j$ is the summation of all model error components relating to MW j at altitude z. However, we do not write explicitly here this summation because, for the moment, the only model error for which a quantifier is present in the MW database is Non-LTE error and therefore the mentioned summation contains only one term.

5.1.3 Total error budget
Assuming as 'independent' the various sources of error affecting the retrieved profiles, the total error is obtained from the summation of the variance covariance matrices (VCM) connected with the different error sources. As for the construction of the MW database the correlation among different altitudes is neglected, the VCM connected with error sources different from measurement noise and p,T error are diagonal matrices.
In conclusion, the total retrieved profile error is represented by the VCM $\mathbf{V}_{TOT}$ given by:

$$\mathbf{V}_{TOT} = \mathbf{V}_N + \mathbf{V}_y^{'} + \sum_{k=1}^{N_{SYST}} \mathbf{V}_{\varepsilon_k} \qquad (5.1.3)$$

where:
the summation extends over all the (N_{SYST}) systematic errors, $\mathbf{V}_N$ is the VCM due to measurement noise and is directly provided by the retrieval algorithm (Gauss-Newton method), $\mathbf{V}_y^{'}$ is the VCM connected with p,T error propagation in VMR retrieval (provided by the algorithm described in Sect. 3.1) and $\mathbf{V}\varepsilon_k$ is the VCM due to error source ε_k. $\mathbf{V}\varepsilon_k$ are diagonal matrices, with diagonal elements equal to:

$$\mathbf{V}_{\varepsilon_k}(i,i) = \sigma^2_{\varepsilon_k}(z_i) \qquad (5.1.4)$$

where σ_{ε_k} is provided by equation (5.1.2).
In equation (5.1.3), the first two terms represent the random component of the profile error, of course, in the case of p,T retrieval $\mathbf{V}_y^{'}$ does not exist and will not be included in the total error evaluation. The summation appearing in the last term of expression (5.1.3) represents the systematic component of the profile error.

5.1.4 Software tool for evaluation of various error components and total error budget

The self-standing software tool for evaluation of the various error components and of total error budget (with the algorithms explained in the above sections) has not been implemented yet. However, we foresee the following list of inputs / outputs of this tool:

Inputs:
- MW database,
- ORM input files '**mwoccmat_xx.dat**' (containing the actual occupation matrix)
- ORM main output files '**xx_out.dat**'
- p,T error propagation matrix (matrix **E** of Sect. 3.1)

Outputs:
- Matrix $\mathbf{V}_{TOT}$ resulting from equation (5.1.3)
- Matrix $\mathbf{V}_y^{'}$
- The summation of systematic errors (in Equ. (5.1.3)) will be broken into four partial summations representing the contributions of:
  1. contaminants error,
  2. spectroscopic error,
  3. gain error,
  4. model error,

  The results of these four partial summations (4 vectors) will be an output of the algorithm

5.2 Interpolation of the retrieved profiles to a user-defined vertical grid

5.2.1 Description of the problem

The analysis of the MIPAS atmospheric spectra performed with the ORM program gives as results Temperature and VMR profiles of the first priority molecules H_2O, O_3, CH_4, N_2O, and HNO_3 at the tangent pressures levels (the so called retrieval grid) of the spectra. It is possible that the pressure values of the retrieval grid do not coincide with the pressure levels needed by the users of the ENVISAT measurements. So the problem of giving reliable values of Temperature and VMR on this new pressure grid (called user-defined grid) arises.
To obtain the VMR and temperature values at the user-defined pressure grid we can follow the internal interpolation rules of the ORM program, making the problem of interpolation an easy one. The problem becomes non-trivial if we include the constraint that the vertical column does not change. The constraint of leaving the value of the columns unchanged arises from the fact that the quantity really determined by the analysis of the atmospheric spectra is the column of each gas between two tangent altitudes. And the users, rather than a mathematical interpolation, may wish to have a profile that satisfies the constraints provided by the measurements.
In other words the column calculated for the interpolated points should be equal, within some pre-defined tolerance, to the column obtained with the original data profile.
We recall that the vertical column at a particular altitude $rz1$ is defined as

$$Col = Cost \int_{rz1}^{rulatm} X_{gas} \frac{P(z)}{T(z)} dz \qquad (5.2.1)$$

where *rulatm* is the altitude of the upper boundary of the atmosphere and X_{gas}, $P(z)$ and $T(z)$ are the gas VMR, pressure and temperature respectively.
The quantities X_{gas}, $P(z)$ and $T(z)$ are measured on a discrete altitude grid, and hence in order to have a realistic result we must calculate the total column as the sum of partial columns between the altitudes of this grid. In turn these partial columns can be calculated interpolating the pressure, temperature and VMR profiles between the two limiting altitudes: linear interpolation is used for Temperature and VMR, and exponential interpolation for the pressure.

Therefore the problem of the interpolation of the VMR profiles becomes one of finding an appropriate transformation of the VMR profile that satisfies the constraint of an unchanged vertical column.

5.2.2 Strategy

The strategy adopted consists of constructing a new grid of VMR points and to impose a one-to-one correspondence of partial columns with the original profile. By finding the VMR values that satisfy this condition, a profile is identified at the new grid points. The latter will be different of the one obtained with a classical interpolation rule.
A ‘classical’ interpolation law is defined in order to identify from the discrete output of the ORM a continuous representation of the atmosphere. For each value of the independent variable pressure, the corresponding altitude and temperature can be obtained by means of hydrostatic equilibrium equation, assuming a linear dependence of temperature on altitude.
In turn, the VMR values at each user-defined altitude can be obtained by linear interpolation between the two nearest retrieved values.

On the basis of this continuous representation partial columns and hence total columns can be defined with respect to both the retrieval grid and the user defined grid in the following way:

1. First partial columns are calculated on the retrieval-grid

2. Then the VMR values are calculated at the user-defined grid using linear interpolation with the altitude.

3. Then the partial columns at the retrieval-grid are calculated following a spline that joins the user-defined grid points only.

4. We then vary the VMR values at the user-defined grid until the difference between the partial columns of the new profile and of the original profile are minimum. The resulting profile by definition has the property of maintaining the vertical column unmodified.

   The variation of the VMR values on the user-defined grid are calculated using a non-linear least-squares fitting procedure. The quantities to be fitted are the partial columns on the retrieval grid. The parameters to be fitted are the VMR values at the user-grid pressure levels. The fitting method is the Gauss-Newton method: if **n** is a vector containing the difference between the observed partial columns and the calculated partial columns, then the correction **y** to be applied to the values of the VMR at the user-grid to minimise this difference is:

$$\mathbf{y} = \mathbf{D}\mathbf{n} \qquad (5.2.2)$$

   where $\mathbf{D} = (\mathbf{K}^T(\mathbf{V}_{col})^{-1}\mathbf{K})^{-1}\ \mathbf{K}^T(\mathbf{V}_{col})^{-1}$ and **K** is the Jacobian matrix and $\mathbf{V}_{col}$ is the Variance Covariance Matrix (VCM) of the measured columns.
   Since the values of the partial columns are numbers that change by several orders of magnitude from the highest altitude to the lowest, instead of fitting their values we fit the logarithm of their values. So $\mathbf{V}_{col}$ becomes the VCM of the logarithm of the partial columns $\mathbf{V}_{log(col)}$.
   The ORM code gives as output the VCM of the measured columns *col(i)* at the various tangent altitudes *i*. To get the VCM of the logarithm of the columns we apply the following transformation:

$$\left(\mathbf{V}_{\log(col)}\right)_{i,j} = \frac{\left(\mathbf{V}_{col}\right)_{i,j}}{col(i) * col(j)} \qquad (5.2.3)$$

   The errors associated with the solution of the fit are given by the squared root of the diagonal elements of the VCM of the solution (**V**) given by:

$$\mathbf{V} = (\mathbf{K}^T(\mathbf{V}_{\log(col)})^{-1}\mathbf{K})^{-1} \qquad (5.2.4)$$

   After solving equation (5.2.2) the VMR profile on the user-defined grid will be given by the new vector VMR_{int}:

$$VMR_{int} = VMR_{linint} + \mathbf{y} \qquad (5.2.5)$$

The fitting procedure is repeated until the sum of the squares of the differences between the new columns and the measured ones doesn't change of more than 1%. In the ORM code the retrieved profile above the highest fitted altitude is obtained by scaling the initial-guess profile of the same quantity used for the highest fitted point. The same procedure has to be applied to the interpolated points above the highest fitted altitude or to the highest point of the user-defined grid if it happens to be below the highest fitted altitude.

5. The errors associated with the new VMR values are characterised by the variance-covariance matrix **V** provided by equation (5.2.4).

6. Summary table of ORM modifications affecting Retrieval Components Library

#	Description of change	Affected / new ORM modules	Modified interface ?	Modified Sect's. In doc. (6)	List of modified I/O files	Modified TEPs
1	Fit of VMR profiles to a user-defined sub-set of tangent altitudes	concandcol retr_vmr fwdmdl_vmr	Y N Y	3.2.28 3.1 3.2.11	-	-
2	Handling of logical masks at the Jacobian and χ^2 computation level	read_masks_pt(vmr) jacob_modif_pt(vmr) retr_pt(vmr) difchi_pt(vmr) chisq_pt(vmr) newparest_pt(vmr) output_pt(vmr) cross_pt inigas_pt finput_pt(vmr) r_spect_pt(vmr)	Y Y Y Y Y Y Y N N N N	New section New section 2.1, 3.1 2.2.13, 3.2.13 2.2.13.1, 3.2.13.1 2.2.15, 3.2.15 2.2.19, 3.2.19 2.2.11.11 2.2.1.10 2.2.1 2.2.1.8	Introduced a database of spectral masks, 1 masks file per MW. Spectroscopic database file format changed.	-

#	Description of change	Affected / <u>new</u> ORM modules	Modified interface ?	Modified Sect's. In doc. (6)	List of modified I/O files	Modified TEPs
3	Improvement of FOV convolution algorithm: definition of the simulation grid in the tangent altitude domain & definition of FOV shape	occusim_pt(vmr)	Y	2.2.5, 3.2.5	settings_pt(vmrxx).dat	-
		updprof_pt	Y	2.2.16		
		fov_pt(vmr)	Y	2.2.11.19, 3.2.11.10		
		fov3_pt(vmr), fov4_vmr and fov5_vmr replaced by <u>fovn_pt(vmr)</u>	Y	The description of fovn_pt(vmr) will be contained in 2.2.11.20 (3.2.11.11), sections 3.2.11.12 and 3.2.11.13 have been removed		
		intcon_pt(vmr)	Y	2.2.11.22, 3.2.11.15		
		chbase_pt(vmr)	N	2.2.7, 3.2.7		
		mkplev_pt(vmr)	N	2.2.11.1, 3.2.11.1		
		input_pt(vmr)	N	2.2.1.2, 3.2.1.2		
		fwdmdl_pt(vmr)	Y	2.2.11, 3.2.11		
		retr_pt(vmr)	N	2.1, 3.1		
4	Profiles regularization	finput_pt(vmr)	Y	2.2.1, 3.2.1	settings_pt(vmrxx).dat	-
		retr_pt(vmr)	Y	2.1, 3.1		
		regsetup_pt(vmr)	Y	New section		
		amodif_pt(vmr	Y	2.2.14, 3.2.14		

#	Description of change	Affected / new ORM modules	Modified interface ?	Modified Sect's. In doc. (6)	List of modified I/O files	Modified TEPs
5	Implementation of a compressed frequency grid for cross-sections storage and radiative transfer computation. Consequent use of an optimized algorithm for interpolation / AILS convolution. Writing of continuum parameters in the output files. Run-time optimisations in the computation of product between matrices. These modifications have been described in TN-IROE-RSA9602, Issue 3, revision A.	retr_pt(vmr) guesspar_pt(vmr) fwdmdl_pt(vmr) mkplev_pt cross_pt(vmr) spectrum_pt(vmr) abcalc_pt(vmr) newparest_pt(vmr) updprof_pt(vmr) output_pt(vmr) mwcont_pt(vmr) ficarra_pt(vmr) jacloscalc read_irrgrid_pt(vmr) read_lookup_pt(vmr) decompr_pt(vmr) cont_char_pt(vmr)	N Y Y N Y Y Y Y Y Y Y Y N Y Y Y Y	2.1, 3.1 2.2.10, 3.2.10 2.2.11, 3.2.11 2.2.11.1 2.2.11.11, 3.2.11.16 2.2.11.18, 3.2.11.26 2.2.12, 3.2.12 2.2.15, 3.2.15 2.2.16, 3.2.16 2.2.19, 3.2.19 2.2.26 2.2.27 2.2.28 2.2.30 2.2.31 2.2.32 2.2.34, 3.2.33	-	-
6	Other changes / corrections	tcgeo_pt gcgeo_vmr fov_pt(vmr) fovn_pt(vmr) retr_pt(vmr) fwdmdl_pt(vmr)	N N N N N N	2.2.6 3.2.6 2.2.11.19, 3.2.11.10 2.2.11.20, 3.2.11.11 2.2.1.2, 3.2.1.2 2.2.11.1, 3.2.11.1	-	tep01_vmr tep02_pt tep05_vmr tep06_pt

References

1. Final Report of ESA Contract 12055-96-NL-CN, 'Study of the retrieval of atmospheric trace gas profiles from infrared spectra' (July 1998).

2. See e.g. Sect.6.4 of the book: C.D.Rodgers, 'Inverse Methods for Atmospheric Sounding: Theory and Practice' 6 Feb. 1998 available at *http://www.atm.ox.ac.uk/user/rodgers/*.

3. Technical Note TN-IROE-RSA9601 entitled: 'High level algorithm definition and physical and mathematical optimizations', Issue 2 Revision A, 20th October 1998.

4. Technical Note: 'Microwindow Database Definition, Selection Rules to Optimize Choice of Spectroscopic Data, and Spectroscopic Data Quality Assessment', Prog. Doc. No.: PO-TN-IMK-GS-001.

5. T.v.Clarmann and G.Echle, Technical Note on 'The Karlsruhe Optimized Occupation Matrix Algorithm' ESA contract n. 11717/95/NL/CN (28 August 1998).

6. T.v.Clarmann et all., Final report of the ESA 'Study on the Simulation of Atmospheric Infrared Spectra', Contract n. 12054/96/NL/CN (August 1998).

7. Technical Note on 'Software Architecture and Algorithms Definition', Issue 3, Prog. Doc. TN-IROE-RSA9602, ESA contract n. 11717/95/NL/CN (23 March 1998).