KOPRA architecture

M. Höpfner

Abstract: An overview of KOPRA's architecture is given starting from the coarse structure and ending with pseudo-code including all subroutines with a description of their tasks. The main program variables which are responsible for the data flow between the modules are described. An overview of the module tree is presented.

1 Program structure

1.1 Coarse structure

- Definition of input variables
- Forward model run
- Output of spectra and derivatives of spectra wrt retrieval parameters
- Deallocation of variables

1.2 More detailed structure with main modules

- Definition of input variables
  - Read main input and initialize variables.
    (input_m)
  - Read fixed parameters and initialize variables.
    (input_m)
  - Read atmospheric data and initialize variables.
    (input_m)
  - Read spectroscopic data and initialize variables.
    (inspec_m)
  - Read cross-section data on heavy molecules and initialize variables.
    (xinput_m)
  - Initialization of retrieval parameter vectors.
    (inipar_m)
- Forward model run
  - Initialization of derivative variables
    (inider_m)
  - Initialization of internal sub-microwindows
    (inismw_m)
  - Adding of additional geometries to the observed ones for simulation of field-of-view effects.
    (modgeo_m)
  - Adding of additional atmospheric levels to the input levels for more precise simulation of radiative transfer.
    (modlev_m)
  - Ray-tracing, calculation of path integrated values (Curtis-Godson values and column amounts) and the derivatives of path integrated values wrt retrieval parameters.
    (rayctl_m)
  - Calculation of absorption cross sections for each atmospheric path for hitran gases and heavy molecules. In case of nlte the cross sections are calculated for each nlte band. For T derivatives the derivatives of the cross sections wrt CG-T of the paths are determined.
    (abco_m)
  - Radiative transfer: the monochromatic spectra and their derivatives wrt the retrieval parameters are determined.
    (radtra_m)
  - Convolution of the monochromatic spectra and their derivatives wrt retrieval parameters with the AILS function and FOV calculation.
    (radtra_m,ilsfov_m)
  - Initialization of output.
    (iniout_m)
- Output of spectra and derivatives of spectra wrt retrieval parameters
  - Write convolved spectra and their derivatives wrt retrieval parameters into file.
    (wriout_m)

- Deallocation of variables.
  (deallo_m)
  - Deallocate all variables.

1.3 Detailed structure with main subroutines

- Definition of input variables (the following subroutines are called by (kopra))
  - (input@input_m)
    Control the input from files and define input variables
    - (input_main@input_m)
      Read from main input file:
      input file and directory names for fixed data
      input file and directory names for atmospheric/instrumental data
      - (input_hitmol@input_m)
        Read hitran info data on molecules
      - (input_isoprof@input_m)
        Read isotope abundance profiles

      Read from main input file:
      header for output files
      mode of observation and definition of geometry
      wavenumber discretization
      parameters for the adjustment of computational accuracy
      - (input_mwdef@input_m)
        Read microwindow definition section of main
        input file
      - (speciorder@input_m)
        Definition of the species and internal species numbering

      Read from main input file:
      instrumental parameters
      derivative definition
      profile definition
      - (input_pTprof@input_m)
        Read p,T profiles
      - (input_vmrprof@input_m)
        Read vmr profiles
      - (make_hydroequi@input_m)
        Bring input z or p profiles into internal hydrostatic equilibrium
      - (input_contprof@input_m)
        Read the aerosol absorption coefficient profiles for each mw
      - (input_pTgradprof@input_m)
        Read p,T - gradient profiles
      - (input_vmrgradprof@input_m)
        Read vmr - gradient profiles

      Read from main input file:
      line-mixing-parameters
      non-lte-parameters
      - (input_Tvibprof@input_m)
        Read Tvib-profiles

  · (input_Tvibgradprof@input_m)
    Read Tvib-gradient-profiles
  · (ils_radius@input_m)
    Determination of the radius (in multiples of the fine grid distance wgrid%fine) where the chosen apodization function 'accu%iapo' is decreased to some percentage of it's center value.
  · (extend_mw@input_m)
    Extend the microwindow boundaries by the ilsradius.

∗ (input_spectroscopy@inspec_m)
  Controls the input of the spectroscopic data, i.e. the determination of the spectroscopic data type construction speci%iso%band%branch%line... This is performed for different options:
  no line mixing, no nlte
  no line mixing, nlte
  line mixing, no nlte
  line mixing, nlte
  · (numdata@inspec_m)
    Determine how many and which spectroscopy files have to be used
  · (readlines@inspec_m)
    Read spectroscopic data
  · (isoabun@inspec_m)
    Multiply line strength with isotope abundancy if species is single isotope

  If no line-mixing and no nlte:
  · (allocno_nlte@inspec_m)
    Allocate speci%iso and speci%iso%band in the case no nlte is considered
  · (allocno_lm@inspec_m)
    Allocate speci%iso%band%branch in the case no line mixing is considered
  · (allocopy1@inspec_m)
    Allocate speci()% vector and copy line data in case no nlte and no line mixing is considered

  If no line mixing, but nlte:
  · (alloc_nlte@inspec_m)
    Determine number of different isotopes for each species where nlte has to be considered and the number of nlte bands. Allocate speci%iso and speci%iso%band.
  · (allocno_lm@inspec_m)
    Allocate speci%iso%band%branch in the case no line mixing is considered
  · (allocopy2@inspec_m)
    Allocate the vectors speci()%iso()%band()%branch(0)%line and copy the line data from vector spe()%line into this vectors.

  If line mixing but no nlte:
  · (allocno_nlte@inspec_m)
    Allocate speci%iso and speci%iso%band in the case no nlte is considered
  · (read_lmdata@inspec_m)
    Read the line mixing data from file fil%linemix into vector 'lm'.

Only the branches which are 'near' a microwindow (distance w_linemix) are read.

- (alloc_lm@inspec_m)
  Allocate the line mixing branches of vector speci and copy the line data of the line mixing lines from the vector lm into
  speci()%iso()%band()%branch()%line and
  speci()%iso()%band()%branch()%lmline
- (allocopy2@inspec_m)
  Allocate the vectors
  speci()%iso()%band()%branch(0)%line and copy the line data from vector spe()%line into this vectors.
- (delete_lmlines@inspec_m)
  Deletes lines from the %branch(0)%line lines if they are also included in the line mixing branches, so that these lines are not calculated twice.

If line-mixing and nlte:

- (alloc_nlte@inspec_m)
  Determine number of different isotopes for each species where nlte has to be considered and the number of nlte bands. Allocate speci%iso and speci%iso%band.
- (read_lmdata@inspec_m)
  Read the line mixing data from file fil%linemix into vector 'lm'. Only the branches which are 'near' a microwindow (distance w_linemix) are read.
- (alloc_lm@inspec_m)
  Allocate the line mixing branches of vector speci and copy the line data of the line mixing lines from the vector lm into
  speci()%iso()%band()%branch()%line and
  speci()%iso()%band()%branch()%lmline
- (allocopy2@inspec_m)
  Allocate the vectors
  speci()%iso()%band()%branch(0)%line and copy the line data from vector spe()%line into this vectors.
- (delete_lmlines@inspec_m)
  Deletes lines from the %branch(0)%line lines if they are also included in the line mixing branches, so that these lines are not calculated twice.

End if

- (pointmw@inspec_m)
  Determine for each microwindow i the range of lines which will be used: speci()%...%mw_l1(i) speci()%...%mw_l2(i).

* (input_xsection@xinput_m)
Controls the input of the cross-section data for heavy molecules. Determination of the construct: speci%cross%...

- (meas_range@xinput_m)
  Determine laboratory measuring range for every microwindow/xsection-gas and tangent altitude
- (readx@xinput_m)
  Read heavy molecule cross section data

– (ini_para@inipar_m)
Initialization of parameter vector para&...

    * (ini_para_vmr@inipar_m)
      initialization of parameter vector for vmr parameters
    * (ini_para_T@inipar_m)
      initialization of parameter vector for temperature parameters
    * (ini_para_Tvib@inipar_m)
      initialization of parameter vector for nlte-vibrational temperature parameters
    * (ini_para_aerabs@inipar_m)
      initialization of parameter vector for aerosol absorption coefficient parameters
    * (ini_para_Tgrad@inipar_m)
      initialization of parameter vector for temperature gradient parameters
    * (ini_para_vmrgrad@inipar_m)
      initialization of parameter vector for vmr gradient parameters
    * (ini_para_p@inipar_m)
      initialization of parameter vector for pressure parameters
  - (kopra_forwrd@kopfwd_m)
    Performs the forward model run (see below '* Forward model run')
    Calculates numerical pressure derivatives

- Forward model run
  (the following subroutines are called by (kopra_forwrd@kopfwd_m))

  - (ini_deri@inider_m)
    Define deri% variable
  - (ini_sub_mw@inismw_m)
    Determines internal forward model sub-microwindows
  - (make_modelgeo@modgeo_m)
    Control of adding of additional geometries to the observed ones for simulation of field-of-view effects
    If no field-of-view is calculated:
    define the sim% variable exactly the same as the obs% variable from the input
    * (make_occusim@modgeo_m)
      Determine the microwindow occupation matrix for the simulated geometries

    If field-of-view is calculated:
    In the case the tangent altitudes are given:
    determine a rough estimate for the observation angle (without refraction and earth ellipsoid)
    Add geometries
    * (addsim_a@modgeo_m)
      Add simulated geometries to the observed ones
      for criterion accu%ifov <= 0 (criterion 1 in input-file)
    * (addsim_a@modgeo_m)
      Add simulated geometries to the observed ones
      for criterion accu%ifov > 0 (criterion 2 in input-file)
    * (make_occusim@modgeo_m)
      Determine the microwindow occupation matrix for the simulated geometries

In the case the tangent altitudes of the observations are given:
calculate tangent altitudes of the additional simulation geometries

- (make_modelgrid@modlev_m)
  Define the model altitude levels for the forward calculation. Starting from a base grid, levels are added in order to fulfill criteria on the T difference and half-width change between levels. Then a criterion for smaller levels distances directly over tangent altitudes is applied. At the end all levels which are less distant than a threshold are deleted.
  - (grid_t_hw@modlev_m)
    Fine-level gridding using T-differences and half-width changes

  Add model fine levels above simulated tangent altitudes
  If the tangent altitudes are given:
  - (grid_tang@modlev_m)
    Fine-level gridding due to levels above tangent points

  If the observation angles are given:
  - (calc_ztang@modlev_m)
    Determination of the tangent altitudes if the nadir angles of a limb scan are given
    - (tangalt@ray_m)
      calculate estimate for tangent altitude with refraction
  - (grid_tang@modlev_m)
    Fine-level gridding due to levels above tangent points

  End if
  - (min_distance@ray_m)
    Levels which are less distant than accu%dmin are selected out (only the ones that do not belong to the base-grid)
- (raytrace_ctrl@rayctl_m)
  Controls ray-tracing, calculation of path integrated values (Curtis-Godson values and column amounts) and the derivatives of path integrated values wrt retrieval parameters.
  For homogeneous path (cuvette) calculation:
  - (homog_path@rayctl_m)
    Determine the path parameters for homogeneous path calculation

  For atmospheric calculations:
  For each geometry :
  - (raytra@ray_m)
    Calculation of ray-tracing in inhomogeneous atmosphere and path integration
    - (nmax_calc@ray_m)
      Determination of the max. number of integration variables per layer

    Case of satellite tangent altitudes:
    determine index of lowest layer
    - (observer@ray_m)
      Calculate position and viewing direction at tangent altitude in cartesian coordinates

    for both parts of the geometry:
    - (latlon@ray_m)
      Calculate geographic latitude and longitude of cartesian point r

```
  · (tnew@ray_m)
    Calculation of new tangent vector along LOS
determine new point of LOS using old and new tangent vector
if a level boundary of the atmospheric model levels is crossed:
  · (leveltrans@ray_m)
    Find exact level positions along LOS and perform integration for
    the actual layer
    + (integrate@ray_m)
      Explicit integration for layer values and derivatives (columns,
      Curtis-Godson-T, -p, -Tvib, ...) with frequent calls to give_...@give_m
      and para_..._change@parchk_m
overwrite old tangent vector with new one
begin again with (tnew@ray_m)
End case of satellite tangent altitudes
Case of satellite observer altitude and elevation angle:
  · (observer@ray_m)
    Calculate position and viewing direction of observer in cartesian
    coordinates
  · (findtop@ray_m)
    Find (coming from outside) the crossing point r with top level
    of the atmosphere
  · (latlon@ray_m)
    Calculate geographic latitude and longitude of cartesian point r

Now go through the atmosphere from top to top:
  · (tnew@ray_m)
    Calculation of new tangent vector along LOS
determine new point of LOS using old and new tangent vector
  · (latlon@ray_m)
    Calculate geographic latitude and longitude
if the altitude is decreasing/increasing monotonously and
if a level boundary of the atmospheric model levels is crossed:
  · (leveltrans@ray_m)
    Find exact level positions along LOS and perform integration for
    the actual layer
    + (integrate@ray_m)
      Explicit integration for layer values and derivatives (columns,
      Curtis-Godson-T, -p, -Tvib, ...) with frequent calls to give_...@give_m
      and para_..._change@parchk_m
else if the altitude is increasing again:
  · (leveltang@ray_m)
    perform integration for tangent layer and find
    exactly tangent position
overwrite old tangent vector with new one
begin again with (tnew@ray_m)
End case of satellite tangent altitudes
Cases for other observation modes are handled equivalently.
Write output (integrated values) in geo%... vector
  · (ray_out@ray_m)
    Prepare output variable geo()%... which contains all path pa-
    rameters
```

  + (alloc_geo@ray_m)
    Allocate geo()%...%lay and geo()%...%lay%speci geo%...%lay%speci%iso , geo%... %iso%state
  + (pathcopy@ray_m)
    Copy path parameters into vector geo%...

If nlte should be considered:

* (calc_nlte_ratios@rayctl_m)
  The derivative of the ratio of nlte/lte with respect to Tkin is calculated.

If vibrational temperature derivatives should be calculated:

* (para_dTvib_ne0@rayctl_m)
  Determine the parameters which do not influence the Tvib's i.e. for which the derivative dTvib_cg_dT = 0

If vmr derivatives are calculated:

* (para_dcol_ne0@rayctl_m)
  Determine the parameters which do not influence the partial columns i.e. for which the derivative dcol = 0

If aerosol absorption derivatives are calculated:

* (para_dabsopt_ne0@rayctl_m)
  Determine the parameters which do not influence the aerosop absorption optical depth i.e. for which the derivative daeropt = 0

If p derivatives are calculated:

* (para_dp_ne0@rayctl_m)
  Determine the p parameters which do not influence the cg-p of air i.e. for which the derivative dp = 0

If T derivatives are calculated:

* (para_dT_ne0@rayctl_m)
  Determine the T parameters which do not influence the cg-T of air i.e. for which the derivative dT = 0

If T-gradient derivatives are calculated:

* (para_dTgrad_ne0@rayctl_m)
  Determine the Tgrad parameters which do not influence the cg-T of air i.e. for which the derivative dTgrad = 0

If vmr derivatives are calculated:

* (para_dcolgrad_ne0@rayctl_m)
  Determine the vmr gradient parameters which do not influence the partial columns i.e. for which the derivative dcol = 0

– (absco_calc@abco_m)
  Calculation of absorption cross sections ('absorption coefficients') [cm^2/molecule] for each atmospheric path for hitran gases and heavy molecules. In case of nlte the cross sections are calculated for each nlte band. For T derivatives the derivatives of the cross sections wrt CG-T of the paths are determined.

  * (allocate_cutoff@addlin_m)
    Allocates memory and reads cutoff-files "cutdop.dat" and "cutlor.dat"
  * (allocate_geo_mw@abco_m)
    Allocates geo%...%mw part of geo% variable (part where absorption coefficients will be stored)

Begin loop on microwindows

- ∗ (allocate_grid@addlin_m)
  Allocate and initialize the absco grid
- ∗ (allocate_x@xintpl_m)
  Allocate and initialize the cross-section grid

Begin loop on simulation geometries
Begin loop on geometry-parts for which absorption coefs. will be calculated explicitly
Begin loop on layers
Begin loop on line-data species for line-by-line calculation
Calculate lte absorption coefficients:

- ∗ (absco_branch@abco_m)
  Calculates the absorption coefficient of a branch
  - · (line_strength@abco_m)
    Calculates the line intensities and optionally the T-derivatives for a bundle of lines

  In case of line-mixing calculation with direct diagonalization:
  - · (y_calc_dd@linmix_m)
    Calculate the y-coefficients for direct diagonalization

  In case of line-mixing calculation with Rosenkranz-approximation
  - · (y_calc_rk@linmix_m)
    Calculates the y-coefficients for the Rosenkranz-approximation

  In both cases:
  - · (corr_y_coefs@linmix_m)
    Correct the y-coefficients

  In case of lines without chi-factor:
  - · (add_lines@addlin_m)
    Add the individual lines on the irregular grid
  - · (add_lines_lm@addlin_m)
    Add the individual lines on the irregular grid with line mixing

  In case of lines with chi-factor:
  - · (add_lines_chi@addlin_m)
    Add the individual lines on the irregular grid
  - · (add_lines_chilm@addlin_m)
    Add the individual lines on the irregular grid with line mixing

Store absorption coefficients in geo%...%absco:

- ∗ (init_absco@transf_m)
  Initialize absco-variable
- ∗ (interpolate_grid@addlin_m)
  Interpolates the intervals of the lower grid to the
  Calculate non-lte absorption coefficients: (equivalent to calculation of lte absorption coefficients above, only that the band index is now /=0)
  Calculate gas continua for line-data species:
- ∗ (n2calc@transf_m)
  - · (calc_n2cont@gascon_m.f90)
    Calculates the absorption cross section of the N_2-continuum in units of [cm^2/molec] as a function of wavenumber, N_2 density and temperature. The derivative of the N_2-continuum with respect to temperature is calculated optionally.

- (o2calc@transf_m)
  - (calc_o2cont@gascon_m.f90)
    Calculates the absorption cross section of the O_2-continuum in units of [cm^2/molec] as a function of wavenumber, O_2 density and temperature. The derivative of the N_2-continuum with respect to temperature is calculated optionally.
- (h2ocalc@tansf_m)
  - (calc_h2ocont@gascon_m.f90)
    Calculates the absorption cross section of the H_2O-continuum in units of [cm^2/molec].

End loop on line-data species for line-by-line calculation
Begin loop on cross-section species for heavy-molecule cross section calculation

- (interpl_xpt@xintpl_m)
  Pressure, temperature and wavenumber grid interpolation of heavy molecule cross-section measurements

Store absorption coefficients in geo%...%absco
End loop on cross-section species for heavy-molecule cross section calculation
For geometries higher than the lowest one:
use already calculated cross sections from the lowest geometry depending on accuracy parameter accu%iexpath
End loop on layers
End loop on geometry-parts for which absorption coefs. will be calculated explicitly
For a geometry-part for which the absorption coefs. are not calculated explicitly:
copy geometry part 1 absorption coefficients to geometry part 2
End loop on simulation geometries

- (deallocate_x@xintpl_m)
  Deallocate the cross-section grid
- (deallocate_grid@addlin_m)
  Deallocate the absco grid

End loop on microwindows

- (deallocate_cutoff@addlin_m)
  Deallocates space used for the cutoff-tables

– (radtrans@radtra_m)
Calculation of the radiative transfer through the atmosphere and determination of the fine-grid spectra and their derivatives. Convolution of the fine-grid spectra with the ails to get the spectra on the coarse-grid (measurement grid) and calculation of the field-of-view weighting.

- (alloc_Sails@radtra_m)
  Allocate data vector where the coarse-grid spectrum and derivatives are stored

Begin loop on microwindows and sub-microwindows
Allocate radiance and derivative - variables for storage on non-equidistant fine grid
Calculate radiances and derivatives for each microwindow:

- (radtrans_mw@radtra_m)
  Begin loop on geometries

```
    Begin loop on geometry-parts and on layers
    Calculate radiative transfer layer-by-layer:
      · (tausrc@radtra_m)
        Calculation of layer transmission, layer source function (nlte considered) and some derivatives
        Initialize the numerator of the source function
        Initialize the variable where the optical depths are added with the aerosol extinction
        Begin loop on species
        Add the absorption coefficients for all bands of one species and multiply by the partial column of the species in the path to determine the optical depth of the path. Determine also the source function numerator.
        For vmr derivatives calculate d(optical thickness)/d(partial column) of the vmr-derivative-species.
        For vmr-gradient derivatives calculate d(optical thickness)/d(partial column) of the vmr-gradient-derivative-species.
        End loop on species
        In case of continuum derivatives calculate derivative of source function wrt aerosol optical depth
        Calculate source function
        Calculate layer transmission
    Calculate radiance at end of layer
    Calculate layer-derivatives
    End loop on geometry-parts and on layers
    Begin loop on geometry-parts and on layers
    Multiply layer-derivatives by total transmission between layer and observer
    End loop on geometry-parts and on layers
    Calculate derivatives with respect to parameters:
      · (derivmr_calc@radtra_m)
        Calculation of derivatives with respect to vmr parameters
      · (deriaer_calc@radtra_m)
        Calculation of derivatives with respect to aerosol absorption parameters
      · (deriT_calc@radtra_m)
        Calculation of derivatives with respect to T parameters
      · (deriTvib_calc@radtra_m)
        Calculation of derivatives with respect to Tvib parameters
      · (deriTgrad_calc@radtra_m)
        Calculation of derivatives with respect to T-gradient parameters
      · (derivmrgrad_calc@radtra_m)
        Calculation of derivatives with respect to vmr-gradient parameters
    End loop on geometries
  * (ilsapofov_calc@radtra_m)
    Convolution of the fine-grid spectra and their derivatives wrt retrieval parameters with the AILS function and FOV calculation
    If field-of-view is considered:
    Interpolate spectrum of each simulated geometry to the fine grid and store result:
      · (fin@transf_m)
      Interpolate non-equidistant fine grid spectrum to equidistant fine
      grid spectrum
    · (sub_mw_minmax@radtra_m)
      Determine the wavenumber indices (in the result of fin@transf_m)
      which are equal to the sub-microwindow boundaries
  Begin loop on observation geometries
  Calculate ails convolution and field-of-view and determine deriva-
  tives with respect to shift, phase and/or linear apodization, (or ESA
  ils-parameters), elevation angle:
  For spherical aperture:
    · (fovils1@ilsfov_m)
      Calculate ails only if it is necessary:
      + (makeifg@ilsfov_m)
          ⋮

  For ESA-ils:
    · (envfovils@ilsfov_m)
      ⋮

  End loop on observation geometries
  For vmr, T, Tvib, vmr-gradient, T-gradient, aerosol - derivatives:
  Begin loop over parameters
    · (lsimobs@radtra_m)
      Determine lsim and lobs, the 'activated' simulated and observed
      geometries i.e. the ones which are influenced by the actual re-
      trieval parameter lpara
    · (finewrk@radtra_m)
      Interpolate spectrum of each 'activated' simulated geometry to
      the fine grid and store result:
      + (fin@transf_m)
        Interpolate non-equidistant fine grid spectrum to equidistant
        fine grid spectrum
      + (sub_mw_minmax@radtra_m)
        Determine the wavenumber indices (in the result of fin@transf_m)
        which are equal to the sub-microwindow boundaries
  Begin loop on observation geometries
  Calculate ails convolution and field-of-view of derivatives:
  For spherical aperture:
    · (fovils1@ilsfov_m)
      Calculate ails only if it is necessary:
      + (makeifg@ilsfov_m)
          ⋮

  For ESA-ils:
    · (envfovils@ilsfov_m)
      ⋮

  End loop on observation geometries
  End loop over parameters
```

```
        If no field-of-view is considered:
        Begin loop on observation geometries
        Interpolate spectrum to the fine grid and store result:
          · (fin@transf_m)
            Interpolate non-equidistant fine grid spectrum to equidistant fine
            grid spectrum
          · (sub_mw_minmax@radtra_m)
            Determine the wavenumber indices (in the result of fin@transf_m)
            which are equal to the sub-microwindow boundaries
        Calculate ails convolution and determine derivatives with respect
        to shift, phase and/or linear apodization, (or ESA ils-parameters),
        elevation angle:
        For spherical aperture:
          · (ilsapo@ilsfov_m)
            Calculate ails only if it is necessary:
           + (makeifg@ilsfov_m)
               ⋮

        For ESA-ils:
          · (envils@ilsfov_m)
            ⋮

        For vmr, T, Tvib, vmr-gradient, T-gradient, aerosol - derivatives:
        Begin loop over parameters
        Interpolate spectrum to the fine grid and store result:
          · (fin@transf_m)
            Interpolate non-equidistant fine grid spectrum to equidistant fine
            grid spectrum
          · (sub_mw_minmax@radtra_m)
            Determine the wavenumber indices (in the result of fin@transf_m)
            which are equal to the sub-microwindow boundaries
        Calculate ails convolution of derivatives:
        For spherical aperture:
          · (ilsapo@ilsfov_m)
            Calculate ails only if it is necessary:
           + (makeifg@ilsfov_m)
               ⋮

        For ESA-ils:
          · (envils@ilsfov_m)
            ⋮

        End loop over parameters
        End loop on observation geometries
    Deallocate radiance and derivative - variables for storage on non-equidistant
    fine grid
    End loop on microwindows and sub-microwindows
  – (offset_scale@offsca_m)
    Add offset and scale spectra. Calculate offset and scale derivatives.
```

    * (off_deri@offsca_m)
      Calculate offset derivative
    * (sca_deri@offsca_m)
      Calculate scale derivative
    * (sca@offsca_m)
      Multiply spectrum by scale factor
    * (off@offsca_m)
      Add offset to spectrum

  - (ini_output@iniout_m)
    Initialization and determination of the output variable outdat%... (Pointer to Sails%...)
    Allocate data vector outdat%.. where the output spectrum is stored. The sub-microwindow index is now on the external microwindows. The spectra in outdat% are pointers to the spectra stored in Sails% where the sub-microwindow index is on the internal microwindows.

- Output of spectra and derivatives of spectra wrt retrieval parameters (the following subroutines are called by (kopra))

  - (writeout@wriout_m)
    Write convolved spectra and their derivatives wrt retrieval parameters (variable: outdat%) into file

- Deallocation of variables (the following subroutines are called by (kopra) or (kopra_forwrd@kopwd_m))

  - (deallocate_main@deallo_m)
    Deallocate variables.

2 Dataflow and interfaces

The data exchange for the main variables and parameters inside kopra is based on the use of variable/parameter-modules. Parameters and fixed data are made accessible by the modules: param_m, precis_m, xdata_m, and xparam_m. The main variables of the program are contained in the three modules inpdat_m, modat_m, and outdat_m. Whereas in modat_m only the forward-model internal variables are made visible, inpdat_m and outdat_m are used for the interface to the exterior. (Though parts of variables of inpdat_m are also determined inside the forward model.)

In order not to be forced to handle too many different single variables in the forward model user-defined-type variables are used for all the main data of kopra. These structures allow to sort the data in a systematic way. The totally extended variables are described explicitly in the variable description sector of this document.

The variables in the three data-exchange modules are: (in brackets the modules are given where the variables are determined or modified)

In inpdat_m:

accu%	definition of the computational accuracy of the forward model (input_m)
deri%	definition of the derivatives which should be calculated during the forward model run (input_m,inider_m)
fil%	file and directory names (input_m)
inprof%	atmospheric input profiles (input_m)
inst%	instrumental specifications (for ails and fov) (input_m)
mol%	HITRAN molecule/isotope information (input_m)
mw%	definition of microwindows and sub-microwindows (input_m, inismw_m, modgeo_m)
n%	numbers (input_m, modlev_m)
nlte%	non-lte definition (input_m)
obs%	definition of the observation geometry (input_m, modgeo_m)
outheader	header for output data (input_m)
para%	definition of the atmospheric data which are handled as parameters (inipar_m)
speci%	definition of the species containing the spectroscopic data and the cross-section data on heavy molecules (input_m, xinput_m, inspec_m)
sw%	switches (input_m)
wgrid%	definition of the wavenumber grid (input_m)

In modat_m:

geo%	information on the paths along all simulated geometries, i.e. path length, partial column amounts, Curtis-Godson-values, derivatives of Curtis-Godson-values wrt retrieval parameters, and absorption coefficients per microwindow (rayctl_m, ray_m, abco_m)
modprof%	forward model altitude grid (modlev_m)
sim%	definition of the simulated geometries (=observation geometries + geometries necessary for fov-calculation) (modgeo_m, modlev_m)
Sails%	result-spectra and derivatives with internal sub-microwindow indices (radtra_m, offsca_m)

In outdat_m:

outdat%	output spectra and derivatives with external sub-microwindow indices (iniout_m)

3 Module tree

In the following module architecture the dependence on module precis_m is omitted since this describes the precision and is needed by quasi every other modules.
The following dependencies are not written explicitly in the general overview:

```
types_m
          param_m
inpdat_m
          types_m
modat_m
          types_m
outdat_m
          types_m
varsub_m
          param_m
give_m
          inpdat_m
          param_m


kopra
       inpdat_m
       input_m
                 param_m
                 types_m
                 inpdat_m
                 recipe_m
                 varsub_m
                 inspec_m
                           param_m
                           inpdat_m
                           varsub_m
                           recipe_m
                 xinput_m
                           types_m
                           inpdat_m
                           xparam_m
                           xdata_m
                                     xparam_m
                           varsub_m
       inipar_m
                 inpdat_m
       wriout_m
                 inpdat_m
                 outdat_m
       deallo_m
                 inpdat_m
                 modat_m
                 outdat_m
                 transf_m
       varsub_m
       kopfwd_m
                 param_m
                 inpdat_m
                 outdat_m
                 varsub_m
                 inider_m
                           inpdat_m
                 inismw_m
```

```
            inpdat_m
modgeo_m
            param_m
            types_m
            inpdat_m
            modat_m
            varsub_m
modlev_m
            types_m
            inpdat_m
            modat_m
            varsub_m
            give_m
rayctl_m
            inpdat_m
            modat_m
            ray_m
                        param_m
                        inpdat_m
                        modat_m
                        give_m
                        recipe_m
                        parchk_m
                                    param_m
                                    inpdat_m
                                    give_m
abco_m
            param_m
            inpdat_m
            modat_m
            addlin_m
            xintpl_m
                        types_m
                        inpdat_m
                        xparam_m
                        xdata_m
                        recipe_m
            transf_m
                        param_m
                        types_m
                        gascon_m
                                    ckdcoe_m
            linmix_m
                        linpac_m
radtra_m
            param_m
            inpdat_m
            modat_m
            transf_m
                        param_m
                        types_m
                        gascon_m
                                    ckdcoe_m
            ilsfov_m
```

```
                param_m
                types_m
                inpdat_m
iniout_m
        inpdat_m
        modat_m
        outdat_m
offsca_m
        inpdat_m
        modat_m
deallo_m
        inpdat_m
        modat_m
        outdat_m
        transf_m
                param_m
                types_m
                gascon_m
                        ckdcoe_m
ifnlte_m
        varsub_m
        inpdat_m
        modat_m
        param_m
        give_m
nltder_m
        inpdat_m
        modat_m
```