Supplement of

The generic MESSy submodel TENDENCY (v1.0) for process-based analyses in Earth system models

R. Eichinger and P. Jöckel

Correspondence to: R. Eichinger (roland.eichinger@dlr.de)
MESSy TENDENCY User Manual

for the MESSy TENDENCY submodel

Roland Eichinger, Patrick Jöckel
Deutsches Zentrum für Luft- und Raumfahrt
Institut für Physik der Atmosphäre
Oberpfaffenhofen, 82230 Wessling, Germany
Roland.Eichinger@dlr.de, Patrick.Joeckel@dlr.de

This manual is part of the electronic supplement of our article “The generic MESSy submodel TENDENCY (v1.0) for process-based analyses in Earth System Models” in Geosci. Model Dev. Discuss. (2014), available at: http://www.geosci-model-dev-discuss.net

Date: February 19, 2014
Contents

1 Introduction 3

2 Subroutines called from BMIL 3
   2.1 The subroutine main_tendency_initialize........................................... 3
   2.2 The subroutine main_tendency_init_coupling.................................... 3
   2.3 The subroutine main_tendency_global_end........................................ 3
   2.4 The subroutine main_tendency_reset............................................... 3
   2.5 The subroutine main_tendency_free_memory..................................... 3
   2.6 The subroutine mtend_set_sqcst_scal............................................. 4

3 Subroutines and functions called from SMIL 4
   3.1 The function mtend_get_handle......................................................... 4
   3.2 The subroutine mtend_register........................................................ 4
   3.3 The subroutine mtend_get_start_l/g.................................................. 5
   3.4 The subroutine mtend_add_l/g.......................................................... 5

4 User interface 6
   4.1 TENDENCY CPL namelist........................................................................ 6
   4.2 The subroutine mtend_request............................................................ 6

5 Private subroutines 7
   5.1 The subroutine compute_eps_and_clear............................................... 7
   5.2 The subroutine tendency_read_namelist_cpl...................................... 7
   5.3 The subroutine tendency_parse_nml_cpl............................................. 7

References 7
1 Introduction

This document provides a more detailed description of the MESSy generic submodel TENDENCY. TENDENCY is a comprehensive and easily expandable submodel to access process-based tendencies of state variables (including tracers) for analyses and further usage.

The outsourcing of the time integration and the tendency accounting for the prognostic variables from the respective submodels to the TENDENCY submodel is implemented for all physical and dynamical processes, as well as for several chemistry related submodels within the version 2.42 (or later) of the ECHAM/MESSy Atmospheric Chemistry (EMAC) model. As a pre-processor directive is used for the implementation, and the conventions follow the MESSy coding standard, the procedure can easily be extended to other submodels attached to the system or when a submodel is replaced. Due to its independency of the time integration scheme it is also applicable to other base-models.

2 Subroutines called from BMIL

2.1 The subroutine main_tendency_initialize

This subroutine

- reads the coupling (CPL) namelist from file tendency.nml and broadcasts its entries (Sect. 4.1),
- assigns a unique identifier (“handle”) to the ECHAM5 basemodel specific processes (see Sect. 3.1), which have not (yet) been reimplemented as MESSy submodels,
- registers the state variables (including tracers), which are subject to modification by these processes, and
- assigns handles to two additional generic processes (mtend_diff and mtend_sum) used for the optional closure test and registers all state variables for those.

2.2 The subroutine main_tendency_init_coupling

With this subroutine the channels and channel objects for the full diagnostic output, the diagnostic sums and the closure test are defined, as requested by the user via the corresponding coupling (CPL) namelist (see Sect. 4.1). For further information on channels and channel objects see Jöckel et al. (2010).

2.3 The subroutine main_tendency_global_end

This subroutine displays the user requested tendency records in the log file. If the closure test is activated, the difference between the internal and the external tendencies (definition see corresponding article) are computed. With these differences the subroutine compute_eps_and_clear (see section 5.1) is called.

2.4 The subroutine main_tendency_reset

In this subroutine all the internal tendencies are reset to zero at the end of every time step.

2.5 The subroutine main_tendency_free_memory

In this subroutine the memory, which is used throughout the submodel and is not related to channel objects, is deallocated at the end of the model simulation.
2.6 The subroutine mtend_set_sqcst_scal

<table>
<thead>
<tr>
<th>SUBROUTINE mtend_set_sqcst_scal (scale_flag)</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>mandatory arguments:</td>
</tr>
<tr>
<td>scale_flag</td>
</tr>
</tbody>
</table>

This subroutine is specific for the ECHAM5 basemodel and it is called twice from within physc. It sets the internal LOGICAL scale_flag providing the information to TENDENCY, in which state the tendencies of the wind velocity components are: either scaled with the cosine of the latitude (scale_flag = .TRUE.), or unscaled (scale_flag = .FALSE.).

3 Subroutines and functions called from SMIL

3.1 The function mtend_get_handle

<table>
<thead>
<tr>
<th>INTEGER FUNCTION mtend_get_handle (name [,lnew])</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>mandatory arguments:</td>
</tr>
<tr>
<td>name</td>
</tr>
<tr>
<td>optional arguments:</td>
</tr>
<tr>
<td>lnew</td>
</tr>
</tbody>
</table>

With this subroutine a unique process identifier of type INTEGER (called “handle”) is assigned to the calling process by its module name (name). The handle is stored within an internal structure. Calling this function multiple times with the same name will yield the same handle. The optional parameter lnew (default: .TRUE.) can be used to suppress (lnew = .FALSE.) the assignment of a new handle, i.e., to probe for the handle corresponding to a specific process (name). The handle is −1, if the process is not present / not running.

3.2 The subroutine mtend_register

<p>| SUBROUTINE mtend_register (handle ,mtend_id [,idt_l | idt]) |
|---------------------------------------------|</p>
<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>intent</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mandatory arguments:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>handle</td>
<td>INTEGER</td>
<td>IN</td>
<td>process identifier</td>
</tr>
<tr>
<td>mtend_id</td>
<td>INTEGER</td>
<td>IN</td>
<td>variable identifier</td>
</tr>
<tr>
<td>optional arguments:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>idt_l</td>
<td>INTEGER, DIMENSION(:,),</td>
<td>IN</td>
<td>list of tracer indices</td>
</tr>
<tr>
<td>idt</td>
<td>INTEGER</td>
<td>IN</td>
<td>tracer index</td>
</tr>
</tbody>
</table>

Within this subroutine every calling process or submodel registers those prognostic variables, which it is going to alter. The information is stored as LOGICAL in an internal structure for each process - prognostic variable pair. This information is used for subsequent computations within TENDENCY.

The optional parameters define, if the submodel is used for a prognostic variable of the basemodel (no optional arguments), a single tracer (idt) or a list of tracers (idt_l). Further information on tracers and tracer identifiers can be found in Jöckel et al. (2008).

The identifiers for the prognostic variables are predefined as INTEGER PARAMETERS:

- mtend_id_t for the temperature,
- mtend_id_q for the specific humidity,
- mtend_id_xl for the liquid water content,
- mtend_id_xi for the ice water content,
- mtend_id_u for the u-wind component,
• \textit{mtend\_id\_v} for the v-wind component,

for the ECHAM5 specific prognostic variables, and

• \textit{mtend\_id\_tracer} for tracers.

### 3.3 The subroutine \texttt{mtend\_get\_start\_l/g}

<table>
<thead>
<tr>
<th>SUBROUTINE \texttt{mtend_get_start_l/g}^{*}</th>
<th>(\texttt{mtend_id [,v0] [,idt] [,v0t]})</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>type</td>
</tr>
<tr>
<td>\texttt{mtend_id}</td>
<td>INTEGER</td>
</tr>
<tr>
<td>\texttt{v0}</td>
<td>REAL(DP), DIMENSION(:,:), OUT</td>
</tr>
<tr>
<td>\texttt{idt}</td>
<td>INTEGER</td>
</tr>
<tr>
<td>\texttt{v0t}</td>
<td>REAL(DP), DIMENSION(:,:,:), OUT</td>
</tr>
</tbody>
</table>

\(^{*}\)Note: The two subroutines (\texttt{mtend\_get\_start\_l} and \texttt{mtend\_get\_start\_g}) differ by the rank of the parameters \texttt{v0} and \texttt{v0t}. The respective subroutine has to be chosen depending on the main entry point from where the corresponding process performs: some are called from the outer (global) loop and some from the inner (local) loop (see also Sect. 2.3 of the article).

This subroutine computes the initial values of the prognostic variable defined by its identifier \texttt{mtend\_id} (see Sect. 3.2). This is done by adding the product of the current tendency (i.e., the sum of all tendencies of prior processes in the same time step) with (two times) the length of the time step to the value of the respective variable of the time step before.

The optional arguments determine, if the subroutine is used for one of the basemodel specific prognostic variables (\texttt{v0}), a single tracer (\texttt{v0} and \texttt{idt}) or the complete tracer set (\texttt{v0t}). For usage with tracers the variable identifier \texttt{mtend\_id} must be \texttt{mtend\_id\_tracer}, for prognostic variables the respective identifier (see Sect. 3.2). Further information on tracer sets, tracers and tracer identifiers can be found in Jöckel et al. (2008).

### 3.4 The subroutine \texttt{mtend\_add\_l/g}

<table>
<thead>
<tr>
<th>SUBROUTINE \texttt{mtend_add_l/g}^{*}</th>
<th>(\texttt{handle ,mtend_id [,px] [,idt] [,pxt]})</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>type</td>
</tr>
<tr>
<td>\texttt{handle}</td>
<td>INTEGER</td>
</tr>
<tr>
<td>\texttt{mtend_id}</td>
<td>INTEGER</td>
</tr>
<tr>
<td>\texttt{px}</td>
<td>REAL(DP), DIMENSION(:,:), IN</td>
</tr>
<tr>
<td>\texttt{idt}</td>
<td>INTEGER</td>
</tr>
<tr>
<td>\texttt{pxt}</td>
<td>REAL(DP), DIMENSION(:,:,:), IN</td>
</tr>
</tbody>
</table>

\(^{*}\)Note: The two subroutines (\texttt{mtend\_add\_l} and \texttt{mtend\_add\_g}) differ by the rank of the parameters \texttt{px} and \texttt{pxt}. The respective subroutine has to be chosen depending on the main entry point from where the corresponding process performs: some are called from the outer (global) loop and some from the inner (local) loop (see also Sect. 2.3 of the article).

With these subroutines the individual process tendencies for all variables are collected. They are used to update the total tendency and the internal sum of tendencies. A copy of the tendency of every process - prognostic variable pair (as requested in the CPL namelist, see Sect. 4.1) is stored in a separate channel object. Additional messages, useful for development, are optionally printed to the log file (see section 4.1).

The optional arguments determine, if the subroutine is used for one of the basemodel specific prognostic variables (\texttt{px}), a single tracer (\texttt{px} and \texttt{idt}), or the complete tracer set (\texttt{pxt}). For usage with tracers the variable identifier \texttt{mtend\_id} must be \texttt{mtend\_id\_tracer}, for prognostic variables the respective identifier (see Sect. 3.2). Further information on tracer sets, tracers and tracer identifiers can be found in Jöckel et al. (2008).
4 User interface

4.1 TENDENCY CPL namelist

The coupling (CPL) namelist of the TENDENCY submodel comprises entries to control the creation of additional channels and corresponding channel objects, the conduction of an internal closure test and the output of additional information. All options are independent of each other and therefore not exclusive:

- **l_full_diag** is a logical switch, which enables the full diagnostic output. If set .TRUE. (default is .FALSE.), a channel object is created for each possible process - prognostic variable pair.

- **l_closure** is a logical switch for the internal closure test. The default is .FALSE., the test is enabled if l_closure is set to .TRUE.. If enabled, two extra pseudo process identifiers (I_HANDLE_SUM and I_HANDLE_DIFF) are defined (see Sect. 2.1).

- **l_clos_diag** is a logical switch, which causes additional output to the log file during the model execution, if set .TRUE. (default is .FALSE.). The output comprises information on the external and the internal tendencies and their differences, and is mostly used for debugging purposes, e.g., when including a new submodel for TENDENCY.

- Individual output switches for the prognostic variables serve to diagnose individual tendencies from specific processes (or tailor made sums of those). With this approach, only the minimum memory required is used: For every state variable (or tracer), one channel object is created for each requested “sum over processes”, plus one additional channel object summing all “unaccounted” processes, which are not part of the user specified list. The syntax is

\[ \text{TDIAG}(i) = 'x', 'p1;p2+p3;...;pn', \]

where \( i \) is an arbitrary but unique number, \( x \) is the name of the prognostic variable (or tracer), and \( p1 \) to \( pn \) are the names of the processes. For the example

\[ \text{TDIAG}(1) = 't', 'vdiff; cloud + rad4all + convect; surf + dyn', \]

four channel objects are created with temperature tendencies: One for the tendency caused by \( vdiff \), one for the sum of \( cloud \), \( rad4all \) and \( convect \), one for the sum of \( surf \) and \( dyn \), and one for the sum of all not listed processes, i.e., for the “unaccounted”.

4.2 The subroutine mtend_request

```fortran
SUBROUTINE mtend_request (status ,process_string ,mtend_id ,ptr_out [,idt])

name             type         intent   description

mandatory arguments:
status            INTEGER      OUT      name of requested process
process_string   CHARACTER(LEN=32)   IN      variable identifier
mtend_id         INTEGER      IN       pointer to requested tendency array
ptr_out          REAL(DP), DIMENSION(:,,:)   POINTER

optional arguments:
idt              INTEGER      IN       tracer index
```

This subroutine can be called during the coupling phase from any submodel to access the individual tendency (of the variable defined by \( \text{mtend_id} \)) from any other submodel (with name \( \text{process_string} \)). One channel object is generated for the tendency of each requested process - prognostic variable pair within the TENDENCY submodel. The pointer \( \text{ptr_out} \) to its corresponding memory is returned to the calling submodel for further access. By calling the subroutine \( \text{mtend_add} \) (Sect. 3.4) from the requested submodel (with name \( \text{process_string} \)), TENDENCY stores a copy of the corresponding individual tendency in the channel object memory accessible by the pointer \( \text{ptr_out} \).
The optional parameter \texttt{idt} (a tracer identifier) determines the individual tracer, if a tracer tendency is requested \((\texttt{mtend}_\text{id} = \texttt{mtend}_\text{id\_tracer}, \text{see Sect. 3.2})\).

Further information on tracers and tracer identifiers can be found in Jöckel et al. (2008).

5 Private subroutines

5.1 The subroutine \texttt{compute} \texttt{eps} \texttt{and} \texttt{clear}

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>intent</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>CHARACTER(Len=*)</td>
<td>IN</td>
<td>variable string</td>
</tr>
<tr>
<td>array</td>
<td>REAL(DP),DIMENSION(;;;:;)</td>
<td>POINTER</td>
<td>pointer to external tendency array</td>
</tr>
<tr>
<td>array_diff</td>
<td>REAL(DP),DIMENSION(;;;:;)</td>
<td>POINTER</td>
<td>pointer to difference between internal and external tendency array</td>
</tr>
</tbody>
</table>

If \texttt{l\_closure} (see section 4.1) is set \texttt{.TRUE.}, this subroutine computes an \(\varepsilon = (\text{max} |x_{te}|) \cdot 10^{-10}\), where \(x_{te}\) denotes the external tendency, for each state variable at the end of each time step. The difference between the internal and the external tendency should be smaller than \(\varepsilon\). If this condition is not fulfilled, an error message is displayed in the log file and the model execution is terminated.

5.2 The subroutine \texttt{tendency\_read\_namelist\_cpl}

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>intent</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>INTEGER</td>
<td>OUT</td>
<td></td>
</tr>
<tr>
<td>iou</td>
<td>INTEGER</td>
<td>IN</td>
<td></td>
</tr>
</tbody>
</table>

With this subroutine the coupling (CPL) namelist is read from the file \texttt{tendency.nml}.

5.3 The subroutine \texttt{tendency\_parse\_nml\_cpl}

This subroutine parses the namelist strings. At first the strings are separated by the semicolons into the diagnostic sum strings. In the next step, the latter are separated by the plus signs into the individual process strings. This allows to assign the process identifiers to generate the required channel objects.

References
