Bratislava Brussels Bucarest Ljubljana Prague Zagreb

ALARO0

Contributors: Doina Banciu, Radmila Brozkova, Bart Catry, Jure Cedilnik, Dunja Drvar, Jean-François Geleyn, Luc Gerard, Janko Masek, Neva Pristov, Ivana Stiperski, Piet Termonia, Alena Trojakova, Martina Tudor, and Filip Vaña

Contents

1 Intro

We should do 2 things in parallel

- put all the different developements together,
- test individual pieces one by one,

The base for the new code is CY29T2.
It should contain the following modifications:

- GFL for pTKE, new GFL fields for prognostic convection and new GFL for CVGQ
- l(z) and MFSTEP modifications should be provided from Prague,
- modified versions of accoefk, acdifus and aplpar that contain developements neccesary for pTKE,
- modified versions of accoefk, acdifus and aplpar that contain developemens of Ivana on Luc's scheme,
- modifications of Bart in aplpar and microphysics (routines aplmphys, acacon, accoll and acevmel),
- radiation from Janko phased by Radmila,
- the historic/prognostic entrainment stuff from Doina not yet ready,

There are further modifications neccesary:

1.1 Turbulence

The l(z) and MFSTEP settings use Ayotte PBL height to compute mixing length using Jure's formula - it is hardcoded now, but it should be put under an option.
accoefk should use total water and equivalent temperature.
acdifus does not conserve energy since it does not diffuse q_t and s_L and we do not know if there is some condensation going on in the turbulence that would also be a source for microphysics. The turbulent flux of s is different than the turbulent flux of s_L:
$F_s = F_{s_L} + L_l F_{q_l} + L_i F_{q_i}$
$F_{q_l} = \alpha F_{q_t} + \beta F_{s_L} + \gamma$
$F_{q_i} = \alpha' F_{q_t} + \beta' F_{s_L} + \gamma'$
integral from 0 to P of $(L_l dF_{q_l} + L_i dF_{q_i}) = 0$

The turbulence may go into the closure or in to the initial conditions of the convection scheme. Contibution from turbulence goes into large scale and Smith cloudiness. Some contributions of acdifus should not go into microphysics (since they are non local).

1.2 Correction of negative values

The correction of negative values (that may happen due to advection or diffusion) is done by pumping from the q_v (for q_l, q_i, q_r and q_s) for the same level and if it is not enough then by pumping from below. But this involves a phase change (and consequently cooling) so we could get a cold spot if some point would systematicaly generate values below zero in each time-step. Therefore, these fluxes should not affect temperature, but only the moist variables. Also, at the time, these fluxes are a part of the vertical diffusion fluxes. But this should be changed, and these fluxes should enter cptend as separate fluxes. They sould enter as input to aplmphys. While correcting for the negative values, we may have fluxes that reach the surface, but should not, so this should be made clean too.

1.3 Microphysics

Variables comming from large scale condensation and updraft to microphysics are updated, but not using cptend. The temporary updated values chould be computed using $C_p(q)$ and $L_v(T)$ (instead of C_{p0} and L_{v0} as first planned).
Local values of q_v, q_l, q_i are updated in aplmphys to make computations sequential, but each of the microphysics routines under aplmphys gives out fluxes, and the updated values are not passed out of the routine. The idea is that each routine computes its bussiness not knowing if the input has been updated by other processes or not and only computes the fluxes. Afterwards, depending on some keys, the temporary variables are updated, before being passed to the next routine. We should consider updating variables only with the local phenomena like phase changes. Vertical diffusion does not enter this balance since it is non-local.

1.4 Closure for the prognostic convection

Ivana found that the advective part of the prognostic convective variables is not so important, but the historical parts of σ_u and σ_d are important, so the advection of these variables may be switched off to save memory and time of computations.
Closure assumption for convection should contain total tendency - physics and dynamics. q_t should be used in the computation of moisture convegence. However it is possible to use ether q_v or q_t to compute CVGQ.

$$CVGQ = \left(\frac{\partial q_t}{\partial t}\right)_{vdif} + \left(\frac{\partial q_t}{\partial t}\right)_{adv}$$

$$\left(\frac{\partial q_t}{\partial t}\right)_{adv} = -u\frac{\partial q_t}{\partial x} - v\frac{\partial q_t}{\partial y} + \frac{1}{2}\left(\dot{\eta}\frac{\partial \pi}{\partial \eta}_{(\tilde{l}-1)}\frac{q_{t(\tilde{l})} - q_{t(\tilde{l}-1)}}{\Delta p_{(\tilde{l}-1)}} + \dot{\eta}\frac{\partial \pi}{\partial \eta}_{(\tilde{l})}\frac{q_{t(\tilde{l+1})} - q_{t(\tilde{l})}}{\Delta p_{(\tilde{l})}}\right)$$

$$\left(\frac{\partial q_t}{\partial t}\right)_{vdif} = -\frac{1}{\Delta p_l}\left(J^{turb}_{q_v(\tilde{l})} - J^{turb}_{q_v(\tilde{l}-1)} + J^{turb}_{q_l(\tilde{l})} - J^{turb}_{q_l(\tilde{l}-1)} + J^{turb}_{q_s(\tilde{l})} - J^{turb}_{q_s(\tilde{l}-1)} + PREC_{(\tilde{l})} - PREC_{(\tilde{l}-1)}\right)$$

So far, horizontal derivatives of q_v have been used to compute CVGQ. If q_v is gridpoint, then we do not have its horizontal derivatives. If q_v is spectral and other q's are gridpoint, then we do not have the same SL advection for all q's and q_t is not conserved. LSPRT=T enables using RT instead of T in spectral space. For SISL with q_v we must use LSPRT=T.
Yves Bouteloup has developed a solution on CY30. The solution is implemented and upgraded with options to use q_t instead of q_v to compute CVGQ.
The problem with CVGQ using q_t is that it is used in the Kuo closure for the updraft as $\Delta p * CVGQ * L_h$ thus multiplied with the specific heat of condensation or sublimation (depending on temperature, function fonice). However, if CVGQ also contains condensates, the convergence of condensates should not be multiplied with L_h. The solution would be to create yet another GFL that would contain water species and compute separate CVGQC for convergence of condensates.

1.5 Summary

1. negative correction - pump from q_v, then from below, and decide if these fluxes are part of diffusion fluxes or not
2. initial state of updraft knows about the turbulent flux of dry static energy but not the moist fluxes (use Yves LSPRT and q_t as closure variable), while the initial state of downdaft knows about microphysics but not the dry static energy (do not update temperature in the aplmphys loop on levels).
3. Bergeron-Findestein process should be written in P, P', P'', P''' form and then we should se if it fits any of the existing boxes of aplmphys or it needs a new one

1.6 New variables

prognostic and advected 3D	q_l, q_i q_r q_s TKE
prognostic advected or not 3D	$w*_{up}$ $w*_{down}$ σ_{up} σ_{down}
diagnostic 3D	N_{cnv}, P_{tot} ?, CVGQ, CVGQT ?
historic 2D	α_1 and α_2
diagnostic 2D	SHF, z_0

Explanations:

- diagnostic means that the variables are simply computed disregarding the value from the previous time-step.
- α_1 and α_2 will be used to recompute the vertical profile of entrainment.
- SHF is the sensible heat flux of precipitation (at surface)

1.7 New switches

1. L3MT - shallow convection is computed using q_t and s_L, acnebn uses q_t instead of q_v, also convective cloud fraction should replace f(conv prec). We should have F_s, F_{q_v}, F_{q_l}, F_{q_i} of F_{q_t} and F_{s_L}
2. Yves switch NCOMP_CVGQ

3. switch for advection of w*'s and σ's is YGFLC(IGFL)%LADV
4. LPTKE switch to use pTKE scheme
5. LRSAT - from Janko - better optical depth of clouds
6. usage of Ayotte PBL and Jure's mixing length is put under a switch LPRGML.

1.7.1 The microphysics switches

yomphy/namphy

LCONDWT=.T. activates prognostic condensates

LPROCLD=.T. activates Lopez scheme of Meteo-France, should be set to false for alaro.

LEVAPP=.T. activates evaporation of precipitation in acprec (this is Luc's subroutine), used in advprc subroutine.

LCOLLEC=.T. activates the 3 collection processes in acprec (this is Luc's subroutine), used in advprc subroutine.

LSTRA=.T. activates diagnostic resolved precipitation scheme (acpluie subroutine is called), should be set to false for alaro.

LCVRA=.T. activates diagnostic deep convection scheme (accvimp and accvimpd subroutines) should be set to false for alaro.

LDIFCONS=.T. activates turbulent diffusion of cloud condensates (in acdifus).

yomcloud/namcloud

LPIL=.T. activates Luc's scheme with prognostic condensate.

LPHSPSH=.T. activates usage of pseudo-historic surface precipitation sensible heat (used in aplpar, it was used in the version of acdifus that I was using in Brussels - acdifus_ptke, but it is not used in the new version of acdifus).

LSMNIMBT=.T. prevents the melting of ice below the triple point of temperature in Smith's scheme (acqmesm subroutine)

LSMROT=.T. activates usage Rotstayn for ice fraction in acsmi1/acqmesm (this part is commented out in the current code).

LSMTPS=.T. activates smoothing of the temperature profile around the triple point in acsmi1 before the saturation moisture computation.

L1DRHCRI=.T. can be used in LAM only and activates usage of single vertical profile for RH in Smith's scheme - it is used only for the allocation of the array, not in the physics routines. default is true.

LGWRHCRI=.T. activates usage of GAW in the in the computation of critical RH, default is true.

NSMTBOT=0 - selects which temperature is used to compute saturation moisture at the bottom of the atmosphere in acsmi1,

- 0 to interpolate surface and the lowest model level,
- 1 to use temperature from the lowest model level. default is 0.

NSMDNEB=2 - selects the method of limiting vertical gradient of cloudiness in acsmi1:

- 0 applies no limitation,
- 1 applies smoothing and
- 2 applies limitation to the prescribed namelist value RSMDNEBX, default is 2.

NPRAG=1 used to activate all collection processes in acprec, used in sulocst, not elsewhere.

NPRAC=1 used to activate all collection processes in acprec, used in sulocst, not elsewhere.

NPRRI=1 used to activate all collection processes in acprec, used in sulocst, not elsewhere.

Local switch in aplpar:

LLMAC=LPIL.AND.LCVPRO.AND..NOT.LCVRA prognostic convection (updraft and downdraft routines accvud and acmodo) are called only if LLMAC is true. the old convection (accvimp) may be called only if not LLMAC. The convective cloudiness array is modified if LLMAC.

The prognostic convection switches:
namluc/yomluc

LLUC(50) - this is a non-local array of logical switches, the size of the array is 50, but not all of them are used. some switches are for updraft, some for downdraft and some for Luc's specific diagnostics. Only those that are used and have any effect on the model will be described. Default is false. If LLUC(1) is false, then PQICE and PQLI are set equal to the transient values PQIMP and PQLMP.

NLUC is a similar array of integers.

LCVPRO - activates prognostic convection

LDDPRO - activates prognostic convection - the downdraft part.

LLUC(1) - if .NOT.(LPIL.AND.LLUC(1)) PQLI and PQICE are set to zero in initaplpar.

LLUC(3)=.T. sets the trainsient value of temperatrue in aplpar equal to the input temperature (on several places).

updraft:

NLUC(10)=0 in accvud ZCFLUDOM=REAL(NLUC(10),JPRB) - I guess this one could be a namelist parameter or removed since it is set equal to zero in every namelist of Luc.

LLUC(10)=.F. is used in accvud:

```
LLNEWM=LLUC(10).AND.LNEIGE.AND.GCVMLT>0.0_JPRB
```

that is used later for the ice/liquid partition, for example:

```
IF (LLNEWM) THEN
   ZDELTA=MAX(0.0_JPRB,SIGN(1.0_JPRB,RTT-ZTDN(JLON,JLEV)))
ELSE
   ZDELTA=FONICE(ZTDN(JLON,JLEV))
ENDIF
```

LLUC(11)=.T. activates usage of cloud condensates since in the beginning of accvud LLCONRES=LLUC(11) that is later used in

```
IF (LLCONRES) THEN
   ZQC=PQI+PQL
ELSE
   ZQC=0.0_JPRB
ENDIF
```

LLUC(12)=.F. rescales ZS3 (entrainment) in accvud LLDIVENT=LLUC(12) and later:

```
IF(LLDIVENT.AND.LSCMF) THEN
  ZZ=1.0_JPRB/(1.0_JPRB-ZSIG9(JLON))
  ZS3(JLON)=ZENTR*ZZ
ELSE
 ZS3(JLON)=ZENTR
ENDIF
```

LLUC(13)=.F. controls the updraft mesh fraction computations, first LLSIGPROP=LLUC(13) in accvud and then:

```
IF (.NOT. LLSIGPROP) THEN
   DO JLEV=KTDIA,KLEV
      DO JLON=KIDIA,KFDIA
          PUDAL(JLON,JLEV)=ZSIG9(JLON)
      ENDDO
   ENDDO
ENDIF
```

if it is false, updraft mesh fraction does not vary in the vertical

```
IF (LLSIGPROP) THEN
   ZDMF=-PUDOM*PRDELP*KNACT
   ZS14=MAXVAL(ZDMF,DIM=2)
ELSE
   ZDMF=REAL(KNACT,JPRB)
   ZS14=1.0_JPRB
ENDIF
```

LLUC(14)=.F. controls the way the fluxes are computed, if it is true fluxes are explicit, first in accvud LLEXPL=LLUC(14) and afterwards there are different sets of computations for fluxes PDIFCS PDIFCQ PDIFCQI PDIFCQL PSTRCU PSTRCV depending on LLEXPL.

LSCMF=.T. is used for singificant mesh fractions, apart of the LLDIVENT part, it is used on another place in accvud:

```
IF (LSCMF) ZMIX=ZMIX/(1.0_JPRB-ZSIG9(JLON))
```

default is false.

dowdraught:

LLUC(15) cancels the convective fluxes of cloud condensates - I guess Luc used this in some particular tests, if it is true, the convective fluxes of cloud condensates are simply multiplied by zero (IDIFCQCD is set to zero).

LLUC(16)=.F. true value would reset dowdraught values and convective fluxes to zero since in the beginning of acmodo LLBO=LLUC(16), and if LLBO then the whole INND array is set to zero - and this INND array is further used to get the final values of PDDOM, PDDAL and convective fluxes.

2 Dataflow between cpg and aplpar

cpg

GFL(YDAL%MP)	ZFPFPL	ZFCSQL	YTKE%LADV
GFL(YDOM%MP)	ZFPFPN	ZFCSQN	GFL(YTKE%MP)
GFL(YUAL%MP)	ZFQLI		
GFL(YUOM%MP)			
GFL(YUEN%MP)			
GFL(YUNEBH%MP)			

mf_phys

PDDAL	PFCQNG	PFPLSL	PFCSQL	LDLADVTKE
PDDOM	PFCQLNG	PFPLSN	PFCSQN	PTKET0
PUDAL	PFCQING	PFPLCL	ZFASL	ZTENDPTKE
PUDOM	PFCQRNG	PFPLCN	ZFASN	
PENTCH	PFCQSNG		ZFESL	
PUNEBH			ZFESN	

initaplpar

PFCQNG	PFPLCL	PFASL	
PFCQLNG	PFPLCN	PFASN	
PFCQING	PFPLSL	PFESL	
PFCQRNG	PFPLSN	PFESN	
PFCQSNG			

all set to zero

aplpar

PDDAL	PFCQNG	PFPFPL	PFASL	PTKE
PDDOM	PFCQLNG	PFPFPN	PFASN	PTENDPTKE
PUDAL	PFCQING		PFESL	ZXPTKEROV
PUDOM	PFCQRNG		PFESN	ZKNROV
PENTCH	PFCQSNG			
PUNEBH				

if LPIL

cptend_new

PFCQNG	PFPLSL	PFASL	PDIFTQ	
PFCQLNG	PFPLSN	PFASN	PDIFTQL	
PFCQING	PFPLCL	PFESL	PDIFTQI	
PFCQRNG	PFPLCN	PFESN	PDIFTQR	
PFCQSNG			PDIFTQS	

cputqy

PTENDQR	PTENDQL	PTKET1	
PTENDQS	PTENDQI	PTENDPTKE	

if not.LADV put to GFLT1

Prognostic convection variables (MUST BE 1:KLEV for GFL), Unless mf_phys has copied them into a 0:KLEV array. Up to now, we leave 1:KLEV for pentch: see later.
PDDAL(KLON,KLEV) INTENT(INOUT) : downdraft mesh fraction
PDDOM(KLON,KLEV) INTENT(INOUT) : downdraft vertical velocity
PUDAL(KLON,KLEV) INTENT(INOUT) : updraft mesh fracition
PUDOM(KLON,KLEV) INTENT(INOUT) : updraft vertical velocity
PUNEBH(KLON,KLEV) INTENT(INOUT) : (pseudo) historic convective cloudiness
PENTCH(KLON,KLEV) INTENT(INOUT) : prognostic/ pseudo historic entrainment
In cpg they are GFL(1,1,YDAL%MP,IBL), GFL(1,1,YDOM%MP,IBL), GFL(1,1,YUEN%MP,IBL), GFL(1,1,YUAL%MP,IBL
GFL(1,1,YUOM%MP,IBL), GFL(1,1,YUNEBH%MP,IBL), in the arguments to mf_phys.
PUNEBH, if LCVPRO PUDAL PUDOM and if LDDPRO PDDAL PDDOM are put to GFLT1 in mf_phys if not advected. But not PENTCH, since this array is not actually used yet.

PFHPS(KLON) INTENT(INOUT) : pseudo-historical array for surface precipitation sensible heat flux (1D) in mf_phys it is PGP(1,MVSPSH) used in acdifus and computed in aplpar

PFPFPN(KLON,0:KLEV)INTENT(INOUT) : flux of conversion of cloud ice to solid precipitation
PFPFPL(KLON,0:KLEV)INTENT(INOUT) : flux of conversion of cloud liquid to liquid precipitation

PFCCQL(KLON,0:KLEV) : convective condensation flux for liquid water.
PFCCQN(KLON,0:KLEV) : convective condensation flux for ice.
PFCSQL(KLON,0:KLEV) : stratiform condensation flux for liquid water.
PFCSQN(KLON,0:KLEV) : stratiform condensation flux for ice.
PFPLSL(KLON,0:KLEV) : stratiform precipitation (liquid).
PFPLSN(KLON,0:KLEV) : stratiform precipitation (solid).
PFPLCL(KLON,0:KLEV) : convective precipitation (liquid).
PFPLCN(KLON,0:KLEV) : convective precipitation (solid).
PFASL(KLON,0:KLEV) : autoconversion pseudo flux (LIQUID)
PFASN(KLON,0:KLEV) : autoconversion pseudo flux (SOLID)
PFESL(KLON,0:KLEV) : evaporation of rain
PFESN(KLON,0:KLEV) : evaporation of snow
ZFASL, ZFASN, ZFESL, ZFESN in mf_phys, computed in aplpar and used in cptend.
There are other local condensation and evaporation arrays inside aplpar.

Additional fluxes local to Aplpar:
ZFCSL(KLON,0:KLEV) : condensation flux from microphyscis(liquid)
ZFCSN(KLON,0:KLEV) : condensation flux from microphyscis(solid)
ZFCQL(KLON,0:KLEV) : adjustment condensation flux(liquid)
ZFCQN(KLON,0:KLEV) : adjustment condensation flux(ice)
ZFEQL(KLON,0:KLEV) : adjustment evaporation flux(liquid)
ZFEQN(KLON,0:KLEV) : adjustment evaporation flux(ice)
ZFEDL(KLON,0:KLEV) : downdraft evaporation flux(liquid)
ZFEDN(KLON,0:KLEV) : downdraft evaporation flux(ice)

Comments:

1. PENTCH (prognostic/pseudo historic entrainment) is not really used as a variable. Doina should develop something based on 2 2D fields to be used instead of this 3D one. So far the data flow related to this 3D variable exists in the code, except the storage in GFLT1 at the end of mf_phys. Should we keep it for possible future developements or remove it? not really urgent to decide.

3 The data flow of the prognostic cloud water, cloud ice, rain, snow and related variables in aplpar subroutine for ALARO0

Based on the document from Ivana, including some modifications introduced during the visit of JFG to Brussels.

if LPIL
aplpar 1
(correct negative advected)

input PQ (q_v), PQL (q_l), PQI (q_i), PQR (q_r), PQS (q_s)
upgrade ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_r), ZQSMP (q_s)
output PFCQLNG P'_l, PFCQING P'_i, PFCQRNG P'''_l, PFCQSNG P'''_i and PFCQNG (flux of q_v)

ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_r), ZQSMP (q_s) now contain the corrected values (no negative values).

actually the name of (q_r) in aplpar right now is PRR not PQR, and the name of q_s is PS not PQS (for the rain and snow variables).

a)

aplpar 2
if LNEBN
call acnebn
(cloudiness XR)

input PQSAT q_v^{sat}, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i)
upgrade
output PNEB, PQLI q_l^{rad}, PQICE q_i^{rad}

b)

aplpar 3
if LRAY
call acraneb
(radiation)

input PNEB, PQ (q_v), PQLI q_l^{rad}, PQICE q_i^{rad}
upgrade
output radiation fluxes PFRSO (J_{sw}^{rad}), PFRTH (J_{lw}^{rad})

c)

aplpar 4
if LVDIF
call acdifus
(diffusion)

input PT, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), PNEB
upgrade
output diffusion fluxes PDIFTS, PDIFTQ, PDIFTQL, PDIFTQN (J_s^{turb}, $J_{q_v}^{turb}$,$J_{q_l}^{turb}$,$J_{q_i}^{turb}$)

d)
aplpar 4
if LPIL
upgrade the transient
variables with
turb. diff. fluxes

input PT, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), PDIFTS, PDIFTQ, PDIFTQL, PDIFTQN (J_s^{turb}, $J_{q_v}^{turb}$,$J_{q_l}^{turb}$,$J_{q_i}^{turb}$)
upgrade by diffusion fluxes
output ZTMP (T), (q_v, q_l, q_i remain unchanged)

e)

aplpar 6
call acsmi1
if not LNEBN
call ancebpar
call acqmesm
(Smith cloudiness)

input ZTTR (T), ZQTR (q_v), ZQLTR (q_l), ZQITR (q_i), PUNEBH (N^c)
upgrade
output ZNEBS, PFCSQL (P'_l), PFCSQN (P'_i)

f)

aplpar 7
(update transient values
by cond. fluxes
phase partition and
update CVGQ)

input PT, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), PFCSQL P'_l, PFCSQN P'_i, q_t for CVGQ
upgrade ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i) with condensation fluxes from Smith scheme
modify ZQLMP and ZQIMP according to FONICE(PT) and compute the condensation fluxes ZFCQL and ZFCQN
add diffusion fluxes to PCVGQ
output ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), PCVGQ

g)

aplpar 8
if LLMAC
call accvud
(updraft)

input PT, ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), PTW, PQSAT, PQW
upgrade ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i) (by P'_l, P'_i)
output convective fluxes PDIFCS, PDIFCQ, PDIFCQL, PDIFCQN (J_s^{conv}, $J_{q_v}^{conv}$,$J_{q_l}^{conv}$,$J_{q_i}^{conv}$), PFCCQL P'_l, PFCCQN P'_i, ZNEBC, PUDAL (σ_u), PUDOM (ω_u)

accvud computes ZSIGDE, and if LLMAC, ZNEBC=ZSIGDE+PUDAL and it is stored to PUNEBH

h)

aplpar 9
call aplmphys
(microphysics)

input ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i),

ZQRMP (q_r), ZQSMP (q_s), PQW, max(PNEBC,PNEB*)
do jlev=ktdia, klev

autoconversion

input ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_r), ZQSMP (q_s), PQW
upgrade ZQLMP (q_l), ZQIMP (q_i)
output fluxes PFASL P''_l, PFASN P''_i

collection

input ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_r), ZQSMP (q_s), PQW
upgrade ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i)
output fluxes PFCSL P'_l, PFCSN P'_i, PFASL P''_l, PFASN P''_i, PFESL P'''_l, PFESN P'''_i

evaporation
melting

input ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_r), ZQSMP (q_s), PQW
upgrade ZQMP (q_v)
output fluxes PFESL P'''_l, PFESN P'''_i

upgrade P_r, P_s and trasfer to the next level
enddo
output PFCSL P'_l, PFCSN P'_i, PFASL P''_l, PFASN P''_i,
PFESL P'''_l, PFESN P'''_i, PFPLSL P_r and PFPLSN P_s

i)

aplpar 10
if LLMAC
if LDDPRO
call acmodo
(downdraft)

input ZTMP, ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_r), ZQSMP (q_s), PFPLSL P_r, PFPLSN P_s, PUDAL (σ_u), PUDOM (ω_u) and convective fluxes
upgrade ZTMP, ZQMP (q_v), ZQRMP (q_l), ZQSMP (q_i) (by P_l''', P_i''')
output convective fluxes (J_s^{conv}, $J_{q_v}^{conv}$,$J_{q_l}^{conv}$,$J_{q_i}^{conv}$) PFESL P_l''', PFESN P_i''', PFPLSL P_r, PFPLSN P_s, PDDAL (σ_d), PDDOM (ω_d)

j)

aplpar 11
correction of negative values and partition of convective and stratiform precipitation

input ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_l), ZQSMP (q_i), P_r and P_s
correct ZQMP (q_v), ZQLMP (q_l), ZQIMP (q_i), ZQRMP (q_l), ZQSMP (q_i)
compute P_r^c and P_s^c, P_r^s and P_s^s
output precipitation fluxes (P_r^c and P_s^c, P_r^s and P_s^s)

4 GFL attributes and consistency of their use

copyright: Jure Cedilnik (after discussion with Filip Vana, Karim Yessad, Gwenaëlle Hello, Claude Fischer and Yann Seity)

4.1 Input/output and coupling issues

LCLDPIN switch is removed. LCOUPLING is replaced by NCOUPLING and LREQIN with NREQIN. As a consequence there are nine options possible for initialisation and coupling.

option	NREQIN	NCOUPLING
Initialisation from file, no coupling	1	0
Initialisation from file, LBC coupling	1	1
Initialisation from file, coupling with reference value	1	-1
Initialisation with reference value, no coupling	-1	0
Initialisation with reference value, LBC coupling	-1	1
Initialisation with reference value, coupling with reference value	-1	-1
No initialisation, no coupling	0	0
No initialisation, LBC coupling	0	1
No initialisation, coupling with reference value	0	-1

In general, NREQIN and NCOUPLING are 1 if the field is to be read from file, -1 if it is not read but set to a reference value or 0 if it is neither read nor set to a certain reference value. For a global model, one does not need to care about NCOUPLING value, in this case it should be by default 0.

It is true, that the last two options in the table seem very unusual, but they should be preserved as possibilities. To make the matter even more complicated, there should be two reference values prescribed: one for coupling and one for initialisation. Again, these options are present to make things as general and consistent as possible and without having any particular use of them in mind. So there are two reference values as GFL attributes: REFVALI and REFVALC, for initialisation and coupling respectively.

Another attribute for the GFLs is the LREQOUT (field required in output) (similar to now already former LREQIN), which is by default TRUE for all variables, but enables that a field may not be in the initial file, but can be in the output.

There would be no consistency check necessary at this point since all options are in principle allowed.

4.2 Advection and choice of interpolators for GFLs

Not necessarily all GFL fields need to be advected, some do and others do not. This is already taken care of in the GFL structure itself (with LADV attribute), but this choice used to be more or less overwritten with value of LADVAMV from NAMDYN. This means that the setup routines got some cleaning: there is no need for LADVAMV anymore and the routines DEFINE_GFL_COMP and SET_GFL_ATTR are called with the proper GFL attributes' values.

Another issue is the choice of interpolators. There were many logical switches used in the setup governing the choice of interpolators for one or more GFLs (for instance LSLHD_O3 for ozone, LQMHV for other GFLs...). These are replaced by GFL attributes. The GFL attribute CSLINT (describing interpolation type) remains, but s not an attribute of the namelist counterpart of GFL. CSLINT is always computed in routine SUCSLINT (as it is already done for some GFLs) which uses the values of GFL namelist attributes.

New `*\_NL` attributes are:

LSLHD for diffusive interpolation,

LQM for quasi-monotonous interpolation,

LQMH for horizontal quasi-monotonous interpolation,

LRSPLINE for 4 points spline interpolation,

LHV for vertical Hermite interpolation and

LVSPLIP for full spline interpolation along vertical.

Along with the introduction of the new attributes, some new consistency checks have been introduced as well:

- we only allow LVSPLIP=T attribute to be used with ozone, and in that case, any other newly defined attribute must be set to F (for other GFLs or with another combination of swithces it is not yet coded)
- LHV=T can only be used for ozone (again because the code does not exist) and in this case LSLHD must be F and LRSPLINE must be F

If these requirements are not met, the model should complain and abort.

These newly defined attributes are a compromise solution. The most consistent solution would be to use fully GMV-like set of interpolator switches.This would mean LSLHD_TKE for use of semilagrangian interpolators for TKE, as we have LSLHD_T for use of semilagrangian interpolators for temperature. But such a set of switches would grow very quickly with the number of GFLs and there is no guarantee that for a newly introduced GFL all corresponding switches would be defined. What is more, there is a chance that with an introduction of a new GFL it might again come to a common switch for more than one GFL (like now LADVAMV).

EXAMPLE: Instead of logical switch LSLHD_TKE one would use a GFL structure switch YTKE%LSLHD.

EXAMPLE: We would like to have a TKE field, with SLHD and quasimonotonous interpolation, without advection, coupled and initialised with reference value 10**-6, without being present in the initial file. This would be defined in the namelist as

```
YTKE_NL%NREQIN=-1
YTKE_NL%LREQOUT=.T.
YTKE_NL%NCOUPLING=-1
YTKE_NL%REFVALI=0.000001_JPRB
YTKE_NL%REFVALC=0.000001_JPRB
YTKE_NL%LADV=.F.
YTKE_NL%LSLHD=.T.
YTKE_NL%LQM=.T.
YTKE_NL%LQMH=.F.
YTKE_NL%LRSPLINE=.F.
YTKE_NL%LHV=.F.
YTKE_NL%LVSPLIP=.F.
```

Of course many of these options can be set to default values according to some other logical switches in the setup and there would be no need to define them all in the namelist if one just uses defaults.

NOTE: The particular physics packages' switches still remain (for instance LARPHY), but they only define the default values of `Y[X]%*` attributes, that are used if they aren't mentioned in the namelist. It is very important that such switches should never overwrite something one asked for in the namelist. If something is inconsistent, the model should complain and abort.

5 aplmphys

Computes the fluxes of cloud water and ice and rain and snow due to autoconversion, collection, melting, freezing and evaporation.

The routine computes contributions from autoconversion, collection and evaporation by calling a separate subroutine for each given process in a loop on levels in the vertical - meaning that we are calling subroutines in a loop and subroutines are using 1D arrays.

First, some usefull values are computed:

$$ZDQL = \min\left(q_l, (q_w - q_v)\max(0, sign(1, T - T_t))\right) \tag{1}$$

$$ZDQI = \min\left(q_i, (q_w - q_v)\max(0, sign(1, T_t - T))\right) \tag{2}$$

There are 3 sources of precipitation considered here:

1. the precipitation descending from the layer above, (precipitation flux)
2. the precipitaition available from the previous time-step (q_r and q_s)
3. the precipitation generated in a given layer due to autoconversion (from cloud liquid water q_l and cloud ice q_i)

$$ZPL1 = P_{ll\tilde{-}1} \qquad ZPN1 = P_{il\tilde{-}1} \qquad ZPL2 = q_r\frac{\Delta p}{g\Delta t} \qquad ZPN2 = q_s\frac{\Delta p}{g\Delta t} \tag{3}$$

The temporary arrays for q_l, q_i (ZQLH, ZQIH) are in the beginning set equal to the input values.

5.1 acacon

Computes autoconversion of liquid water to rain and cloud ice to snow. The autoconversion of cloud liquid water to rain and cloud ice to snow are computed using formulas

$$\left(\frac{\partial q_l}{\partial t}\right)_{auto} = -\frac{q_l}{\tau_l}\left(1 - e^{-\left(\frac{q_l}{q_r}\right)^2}\right) \qquad \left(\frac{\partial q_i}{\partial t}\right)_{auto} = -\frac{q_i}{\tau_n}\left(1 - e^{-\left(\frac{q_i}{q_s}\right)^2}\right) \tag{4}$$

The output arrays contain:
PFASL - autoconversion of cloud liquid water to rain.

$$\Delta F_{al} = q_l\left[1 - e^{-\left(\frac{q_l}{q_r}\right)^2}\right]\frac{1}{\tau_l}\frac{\Delta p}{g\Delta t} \tag{5}$$

PFASN - autoconversion of cloud ice to snow.

$$\Delta F_{an} = q_i\left[1 - e^{-\left(\frac{q_i}{q_s}\right)^2}\right]\frac{1}{\tau_n}\frac{\Delta p}{g\Delta t} \tag{6}$$

where τ_l and τ_n are new tuning parameters TAUL and TAUN introduced to NAMPHY0 (so far set equal to 1).

Afterwards, the precipitation fluxes generated in a given layer due to autoconversion from cloud liquid water q_l and cloud ice q_i is computed

$$ZPL3 = F_{al} \qquad ZPN3 = F_{an} \tag{7}$$

and the autoconversion flux at the bottom of the layer is

$$F_{all\tilde{}} = F_{all\tilde{-}1} + ZPL3 \qquad F_{anl\tilde{}} = F_{anl\tilde{-}1} + ZPN3 \tag{8}$$

the new values of q_l and q_i are computed, to be used later

$$q_l = q_l - \frac{F_{al}}{\frac{\Delta p}{g\Delta t}} \qquad q_i = q_i - \frac{F_{an}}{\frac{\Delta p}{g\Delta t}} \tag{9}$$

there is no need to recompute q_r and q_s (ZPL2 and ZPN2) since this is the third kind of precipitation we distingusih and treat separately.

5.2 accoll

Computes amount of liquid water and cloud ice collected by rain and snow. Since both rain and snow could collect both cloud liquid water and ice, there altogether 4 fluxes computed using the formula

$$\left(\frac{\partial P_l^{\frac{1}{5}}}{\partial p}\right)_{coll} = 0.0069 q_l \qquad \left(\frac{\partial P_l^{\frac{1}{5}}}{\partial p}\right)_{coll} = 0.0069 q_i \qquad \left(\frac{\partial P_i^{\frac{1}{5}}}{\partial p}\right)_{coll} = 0.0069 q_l \qquad \left(\frac{\partial P_l^{\frac{1}{5}}}{\partial p}\right)_{coll} = 0.0069 q_i \tag{10}$$

actually this formulas allow collection whenever q_l and q_i are bigger than zero, even when there is no precipitation, there could be some cloud water and ice collected.

The amount of cloud liquid water collected by falling rain:

$$PTENDL1 = \left[\left(P_r + \frac{1}{2} q_r \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{al}\right)^{\frac{1}{5}} + C_{oll} q_l \frac{\Delta p}{g\Delta t}\right]^5 - \left(P_r + \frac{1}{2} q_r \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{al}\right) \tag{11}$$

The amount of cloud ice collected by falling rain:

$$PTENDL2 = \left[\left(P_r + \frac{1}{2} q_r \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{al}\right)^{\frac{1}{5}} + C_{oll} q_i \frac{\Delta p}{g\Delta t}\right]^5 - \left(P_r + \frac{1}{2} q_r \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{al}\right) \tag{12}$$

The amount of cloud liquid water collected by falling snow:

$$PTENDN1 = \left[\left(P_s + \frac{1}{2} q_s \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{an}\right)^{\frac{1}{5}} + C_{oll} q_l \frac{\Delta p}{g\Delta t}\right]^5 - \left(P_s + \frac{1}{2} q_s \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{an}\right) \tag{13}$$

The amount of cloud ice collected by falling snow:

$$PTENDN2 = \left[\left(P_s + \frac{1}{2} q_s \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{an}\right)^{\frac{1}{5}} + C_{oll} q_i \frac{\Delta p}{g\Delta t}\right]^5 - \left(P_s + \frac{1}{2} q_s \frac{\Delta p}{g\Delta t} + \frac{1}{4} F_{an}\right) \tag{14}$$

the new values of q_l and q_i are computed, to be used later

$$q_l = q_l - (ZTENDL1 + ZTENDN1)\frac{1}{\frac{\Delta p}{g\Delta t}} \qquad q_i = q_i - (ZTENDL2 + ZTENDN2)\frac{1}{\frac{\Delta p}{g\Delta t}} \tag{15}$$

Afterwards, the precipitation generated in a given layer due to collection of cloud liquid water q_l and cloud ice q_i by rain and snow is computed so the precipitation generated in a layer is modified.

$$ZPL3 = F_{al} = F_{al} + (ZTENDL1 + ZTENDL2) \qquad ZPN3 = F_{as} = F_{as} + (ZTENDN1 + ZTENDN2) \tag{16}$$

and the autoconversion, condensation and evaporation fluxes at the bottom of the layer are

$$PFASL = F_{al\tilde{l}} = F_{al\tilde{l}} + ZTENDL1 \qquad PFASN = F_{an\tilde{l}} = F_{an\tilde{l}} + ZTENDN2 \tag{17}$$

$$PFCSL = F_{cl\tilde{l}} = F_{cl\tilde{l}} - ZTENDN1 \qquad PFCSN = F_{cn\tilde{l}} = F_{cn\tilde{l}} - ZTENDL2 \tag{18}$$

$$PFESL = F_{el\tilde{l}} = F_{el\tilde{l}} - ZTENDL2 \qquad PFESN = F_{en\tilde{l}} = F_{en\tilde{l}} - ZTENDN1 \tag{19}$$

5.3 acevmel

Computes evaporation or condensation and melting or freezing of falling precipitations. Evaporation is computed according to the formula

$$\left(\frac{\partial \sqrt{R}}{\partial \frac{1}{p}}\right)_{evap} = EVAP(1 - m_e(1 - REVGSL))(q_w - q_v) \tag{20}$$

where R stands for any of the 6 kinds of precipitati (liquid or frozen coming from any of the 3 sources). EVAP is a namelist parameter in NAMPHY0 equal to $4.8 \cdot 10^6$.
First, some usefull values are precomputed (using new values of q_l and q_i that have been updated with autoconversion and collection):

$$ZDQL = \min\left(q_l, (q_w - q_v)\max(0, sign(1, T - T_t))\right) \tag{21}$$

$$ZDQI = \min\left(q_i, (q_w - q_v)\max(0, sign(1, T_t - T))\right) \tag{22}$$

$$ZRME = m_i = \frac{P_s + q_s\frac{\Delta p}{g\Delta t} + F_{an}}{P_r + q_r\frac{\Delta p}{g\Delta t} + F_{al} + P_s + q_s\frac{\Delta p}{g\Delta t} + F_{an}} \tag{23}$$

evaporation of precipitation that arrived from the layer above

$$PTENDL1 = \left[P_l^{\frac{1}{2}} + EVAP(1 - m_i(1 - REVGSL))ZDQL\left(\frac{1}{p_b} - \frac{1}{p_h}\right)\right]^2 - P_l \tag{24}$$

evaporation of rain q_r

$$PTENDL2 = \left[\left(\frac{1}{2}q_r\frac{\Delta p}{g\Delta t}\right)^{\frac{1}{2}} + EVAP(1 - m_i(1 - REVGSL))ZDQL\left(\frac{1}{p_b} - \frac{1}{p_h}\right)\right]^2 - \frac{1}{2}q_r\frac{\Delta p}{g\Delta t} \tag{25}$$

evaporation of rain that was created in the layer due to autoconversion

$$PTENDL3 = \left[\left(\frac{1}{4}F_{al}\right)^{\frac{1}{2}} + EVAP(1 - m_i(1 - REVGSL))ZDQL\left(\frac{1}{p_b} - \frac{1}{p_h}\right)\right]^2 - \frac{1}{4}F_{al} \tag{26}$$

evaporation of frozen precipitation that arrived from the layer above

$$PTENDN1 = \left[P_s^{\frac{1}{2}} + EVAP(1 - m_i(1 - REVGSL))ZDQI\left(\frac{1}{p_b} - \frac{1}{p_h}\right)\right]^2 - P_s \tag{27}$$

evaporation of snow q_s

$$PTENDN2 = \left[\left(\frac{1}{2}q_s\frac{\Delta p}{g\Delta t}\right)^{\frac{1}{2}} + EVAP(1 - m_i(1 - REVGSL))ZDQI\left(\frac{1}{p_b} - \frac{1}{p_h}\right)\right]^2 - \frac{1}{2}q_s\frac{\Delta p}{g\Delta t} \tag{28}$$

evaporation of snow that was created in the layer due to autoconversion

$$PTENDN3 = \left[\left(\frac{1}{4}F_{an}\right)^{\frac{1}{2}} + EVAP(1 - m_i(1 - REVGSL))ZDQI\left(\frac{1}{p_b} - \frac{1}{p_h}\right)\right]^2 - \frac{1}{4}F_{an} \tag{29}$$

ZPL1, ZPL2, ZPN1 and ZPN2 are temporarily upgraded with the values coming from evaporation before melting and freezing computations.
Melting and freezing of precipitation is computed as a change in the proportion of snow in the total precipitation flux

$$\frac{dm_i}{d\frac{1}{p}} = FONT(1 - m_i(1 - REVGSL))\frac{T - T_t}{\sqrt{R}} \tag{30}$$

where R stands for any of the 6 kinds of precipitati (liquid or frozen coming from any of the 3 sources). FONT is a namelist parameter in NAMPHY0 equal to $2.4 \cdot 10^4$.
melting/freezing of precipitation that descended from the layer above

$$tmp = FONT(1 - m_i(1 - REVGSL))(T - T_t)\left(\frac{1}{p_b} - \frac{1}{p_h}\right)\sqrt{P_l + P_n} \tag{31}$$

$$PTENDN4 = \max(-P_n, tmp) * \delta + \min(P_l, tmp)(1 - \delta) \qquad where \quad \delta = \max(0, sign(1, -tmp)) \tag{32}$$

if the snow is melting, tmp and PTENDN4 are negative, and if the rain is freezing they are both positive.

Melting/freezing of precipitation from previous time-step

$$tmp = \left[FONT(1 - m_i(1 - REVGSL))\frac{(T - T_t)\left(\frac{1}{p_b} - \frac{1}{p_h}\right)}{\sqrt{(q_r + q_s)\frac{\Delta p}{g\Delta t}}} + \frac{q_s}{q_r + q_s}\right](q_r + q_s)\frac{\Delta p}{g\Delta t} - q_s\frac{\Delta p}{g\Delta t} \tag{33}$$

$$PTENDN5 = \max(-q_s\frac{\Delta p}{g\Delta t}, tmp) * \delta + \min(q_r\frac{\Delta p}{g\Delta t}, tmp)(1-\delta) \qquad where \quad \delta = \max(0, sign(1, -tmp)) \tag{34}$$

if the snow is melting, tmp and PTENDN5 are negative, and if the rain is freezing they are both positive.

Total evaporation of liquid and solid precipitation fluxes due to the evaporation, melting and freezing of precipitation are

$$F_{el} = PFESL = -ZTENDL1 - ZTENDL2 - ZTENDL3 + ZTENDN4 + ZTENDN5 \tag{35}$$

$$F_{en} = PFESN = -ZTENDN1 - ZTENDN2 - ZTENDN3 - ZTENDN4 - ZTENDN5 \tag{36}$$

the new values of q_v are computed

$$q_v = q_v - (ZTENDL1 + ZTENDL2 + ZTENDL3 + ZTENDN1 + ZTENDN2 + ZTENDN3)\frac{1}{\frac{\Delta p}{g\Delta t}} \tag{37}$$

the new values for the 6 kinds of precipitation are

$$ZPL1 = P_l = P_l + ZTENDL1 - ZTENDN4 \qquad ZPN1 = P_i = P_i + ZTENDN1 + ZTENDN4 \tag{38}$$

$$ZPL2 = q_r\frac{\Delta p}{g\Delta t} + ZTENDL2 - ZTENDN5 \qquad ZPN2 = q_s\frac{\Delta p}{g\Delta t} + ZTENDN2 + ZTENDN5 \tag{39}$$

$$ZPL3 = ZPL3 + ZTENDL3 \qquad ZPN3 = ZPL3 + ZTENDN3 \tag{40}$$

and finally the deposition is computed using probabilities for the three (or six) sources of precipitation. First the new R and ρ are computed.

$$R = R_d(1 - q_v - q_l - q_i - q_r - q_s) + R_v q_v \qquad \rho = \frac{p}{RT} \tag{41}$$

The ratio of the layer thickness and the distance that precipitaion might cross during one timestep:

$$Z_l = \frac{\frac{\Delta p}{g\Delta t}}{\rho w_l} \qquad Z_i = \frac{\frac{\Delta p}{g\Delta t}}{\rho w_i} \tag{42}$$

The probability that the precipitation falling from the layer above will reach the bottom of the layer during the timestep:

$$P_1(Z_l) = \frac{1 - e^{-Z_l}}{Z_l} \tag{43}$$

The probability that the precipitation coming from previous timestep will reach the bottom of the layer during the timestep:

$$P_2(Z_l) = \frac{e^{-Z_l}(1 + Z_l)}{Z_l^2 + 3Z_l + 1} \tag{44}$$

The probability that the precipitation generated in autoconversion (or any other process) during this timestep will reach the bottom of the layer during the timestep:

$$P_3(Z_l) = \frac{1}{2Z_l} - \frac{e^{-Z_l}(1 + Z_l)}{Z_l^3 + 4Z_l^2 + 2Z_l} \tag{45}$$

and the final precipitation fluxes (at the bottom of the layer) are

$$P_l = P_1(Z_l)P_l + P_2(Z_l)q_r\frac{\Delta p}{g\Delta t} + P_3(Z_l)F_{al} \qquad P_i = P_1(Z_i)P_l + P_2(Z_i)q_s\frac{\Delta p}{g\Delta t} + P_3(Z_i)F_{an} \tag{46}$$

and the fall velocities are recomputed

$$w_l = \omega\left(\frac{P_l}{\rho^4}\right)^{\frac{1}{6}} \qquad w_i = \omega\left(\frac{P_i}{REVGSL\rho^4}\right)^{\frac{1}{6}} \tag{47}$$

where $\omega = 13.4$.

6 cptend and cputqy

6.1 cptend

cpfhpfs and cptedn are merged in one subroutine cptend_new that is called from mf_phys if LPIL. So far, it assumes that δ_m is always zero, does not use Lopez scheme (LPROCLD=.FALSE.) and there is a possibility of using LCONDWT. However, if LCONDWT is TRUE, all 4 additional moist species are used (q_l, q_i, q_r and q_s). The TKE is not yet included since the tendency is the direct output of aplpar.
First the total flux of enthalpy due to the radiative, convective and turbulent diffusion processes, phase changes and precipitation fluxes is computed

$$F_s^{tot} = F^{so} + F^{th} + F^{mh} + (c_w - c_{pd})(F^{plsl} + F^{plcl})\frac{T_{l+1}+T_l}{2} + (c_s - c_{pd})(F^{plsn} + F^{plcn})\frac{T_{l+1}+T_l}{2} - L_v(F^{ccql} + F^{csql} - F^{esl}) - L_s(F^{ccqn} + F^{csqn} - F^{esn}) \tag{48}$$

then the tendency of enthalpy is computed

$$\left(\frac{\partial h}{\partial t}\right)_{phy} = \frac{g}{\Delta p}\left[F^{tot}_{s(\tilde{l}-1)} - F^{tot}_{s(\tilde{l})}\right] \tag{49}$$

the tendency of water vapour due to physics is

$$\left(\frac{\partial q_v}{\partial t}\right)_{phy} = \frac{g}{\Delta p}\left[F^{tur}_{v(\tilde{l}-1)} - F^{tur}_{v(\tilde{l})} + F^{cnv}_{v(\tilde{l}-1)} - F^{cnv}_{v(\tilde{l})} + F^{ccl}_{v(\tilde{l}-1)} - F^{ccl}_{v(\tilde{l})} + F^{ccn}_{v(\tilde{l}-1)} - F^{ccn}_{v(\tilde{l})} + F^{scl}_{v(\tilde{l}-1)} - F^{scl}_{v(\tilde{l})} + F^{scn}_{v(\tilde{l}-1)} - F^{scn}_{v(\tilde{l})} - \left(F^{esl}_{(\tilde{l}-1)} - F^{esl}_{(\tilde{l})}\right) - \left(F^{esn}_{(\tilde{l}-1)} - F^{esn}_{(\tilde{l})}\right)\right] \tag{50}$$

the tendency of cloud water due to physics is

$$\left(\frac{\partial q_l}{\partial t}\right)_{phy} = \frac{g}{\Delta p}\left[F^{tur}_{l(\tilde{l}-1)} - F^{tur}_{l(\tilde{l})} + F^{cnv}_{l(\tilde{l}-1)} - F^{cnv}_{l(\tilde{l})} + \left(F^{asl}_{(\tilde{l}-1)} - F^{asl}_{(\tilde{l})}\right) - \left(F^{csql}_{(\tilde{l}-1)} - F^{csql}_{(\tilde{l})}\right) - \left(F^{ccql}_{(\tilde{l}-1)} - F^{ccql}_{(\tilde{l})}\right)\right] \tag{51}$$

the tendency of cloud ice due to physics is

$$\left(\frac{\partial q_i}{\partial t}\right)_{phy} = \frac{g}{\Delta p}\left[F^{tur}_{i(\tilde{l}-1)} - F^{tur}_{i(\tilde{l})} + F^{cnv}_{i(\tilde{l}-1)} - F^{cnv}_{i(\tilde{l})} + \left(F^{asn}_{(\tilde{l}-1)} - F^{asn}_{(\tilde{l})}\right) - \left(F^{csqn}_{(\tilde{l}-1)} - F^{csqn}_{(\tilde{l})}\right) - \left(F^{ccqn}_{(\tilde{l}-1)} - F^{ccqn}_{(\tilde{l})}\right)\right] \tag{52}$$

the tendency of rain due to physics is

$$\left(\frac{\partial q_r}{\partial t}\right)_{phy} = \frac{g}{\Delta p}\left[F^{tur}_{r(\tilde{l}-1)} - F^{tur}_{r(\tilde{l})} + \left(F^{esl}_{(\tilde{l}-1)} - F^{esl}_{(\tilde{l})}\right) - \left(F^{asl}_{(\tilde{l}-1)} - F^{asl}_{(\tilde{l})}\right) + \left(F^{plsl}_{(\tilde{l}-1)} - F^{plsl}_{(\tilde{l})}\right) + \left(F^{plcl}_{(\tilde{l}-1)} - F^{plcl}_{(\tilde{l})}\right)\right] \tag{53}$$

the tendency of snow due to physics is

$$\left(\frac{\partial q_s}{\partial t}\right)_{phy} = \frac{g}{\Delta p}\left[F^{tur}_{s(\tilde{l}-1)} - F^{tur}_{s(\tilde{l})} + \left(F^{esn}_{(\tilde{l}-1)} - F^{esn}_{(\tilde{l})}\right) - \left(F^{asn}_{(\tilde{l}-1)} - F^{asn}_{(\tilde{l})}\right) + \left(F^{plsn}_{(\tilde{l}-1)} - F^{plsn}_{(\tilde{l})}\right) + \left(F^{plcn}_{(\tilde{l}-1)} - F^{plcn}_{(\tilde{l})}\right)\right] \tag{54}$$

Some particular fluxes are also computed - to be used for diagnostic purposes (perhaps only if this particular diagnostics is required - under some if L...)

$$PFHSCL = F^{hscl} = F^{plcl}\frac{T_{l+1}+T_l}{2}(c_w - c_{pd}) \qquad PFHSCN = F^{hscn} = F^{plcn}\frac{T_{l+1}+T_l}{2}(c_s - c_{pd}) \tag{55}$$

$$PFHSSL = F^{hssl} = F^{plsl}\frac{T_{l+1}+T_l}{2}(c_w - c_{pd}) \qquad PFHSSN = F^{hssn} = F^{plsn}\frac{T_{l+1}+T_l}{2}(c_s - c_{pd}) \tag{56}$$

$$PFHPCL = F^{hpcl} = -L_v F^{ccql} \qquad PFHPCN = F^{hpcn} = -L_s F^{ccqn} \tag{57}$$

$$PFHPSL = F^{hpsl} = -L_v F^{csql} \qquad PFHPSN = F^{hpsn} = -L_s F^{csqn} \tag{58}$$

the total precipitation flux is

$$PFP = F^p = F^{plcl} + F^{plcn} + F^{plsl} + F^{plsn} \tag{59}$$

The total flux of enthalpy due to the phase changes and precipitation fluxes is

$$PFHP = F^{hp} = F^{hscl} + F^{hscn} + F^{hssl} + F^{hssn} + F^{hpcl} + F^{hpcn} + F^{hpsl} + F^{hpsn} \tag{60}$$

6.2 cputqy

first the change of c_p is computed (the rain and snow part not yet included in the code)

$$\frac{\partial c_p}{\partial t} = (c_{pv} - c_{pd}) \left(\frac{\partial q_v}{\partial t} \right)_{phy} + (c_w - c_{pd}) \left(\frac{\partial q_l}{\partial t} \right)_{phy} + (c_s - c_{pd}) \left(\frac{\partial q_i}{\partial t} \right)_{phy} +$$

$$(C_w - C_{pd}) \left(\frac{\partial q_r}{\partial t} \right)_{phy} + (C_s - C_{pd}) \left(\frac{\partial q_s}{\partial t} \right)_{phy} \tag{61}$$

the change of kinetic energy is also computed

$$\Delta E_k = \Delta t \left[u^t \left(\frac{\partial u}{\partial t} \right)_{phy} + v^t \left(\frac{\partial v}{\partial t} \right)_{phy} + \frac{1}{2} \Delta t \left(\left(\frac{\partial u}{\partial t} \right)^2_{phy} + \left(\frac{\partial v}{\partial t} \right)^2_{phy} \right) \right] \tag{62}$$

the new value for temperature is

$$T^{t+\Delta t} = T^{t+\Delta t} + \left[\frac{c_p^t T^t - \Delta E_k + \Delta t \left(\frac{\partial h}{\partial t} \right)_{phy}}{c_p^t + \Delta t \frac{\partial c_p}{\partial t}} - T^t \right] \tag{63}$$

should TKE enter here?

7 LSPRT

The problem has been described earlier, but the following could be repeated. LSPRT=.T. enables us to use virtual temperature $\frac{RT}{R_d}$ instead of temperature T as a spectral variable and q_v as a gridpoint one (NAMCT0). This is a desired situation since all q's would be gridpoint variables and therefore undergo the same advection and the total water content ($q_t = q_v + q_l + q_i + q_r + q_s$) would be perserved. Unfortunately, physics parameterization, demanding as they are, the convection in particular demands moisture convergence as input. The dynamical part of the moisture convergence is computed using horizontal derivatives of moisture (q_v actually) that are gained from the spectral transformations of q_v that are available only if q_v is spectral. Yves Bouteloup has overcome this difficulty by introducing a new spectral GFL called CVGQ. The sole purpose of this GFL array (so far) is to store moisture content that will undergo transformation to spectral space and in turn provide horizontal derivatives of this moist stuff in the atmosphere that we desire to use in computation of the moisture convergence.

Therefore, to say it as plainly as this, the CVGQ variable contains water, it could be water vapour q_v or total water q_t. Actually, it is possible to code any other option. And fortunately there is a very nice switch for this.

7.1 Namelist switch: NCOMP_CVGQ

First of all, there is a new namelist switch, it is NCOMP_CVGQ, and imagine, you should put it to NAMDYN.

- The default value for NCOMP_CVGQ is 0. It gives you the most familiar computation of moisture convergence from horizontal derivatives of q_v using q_v as spectral variable. No dirty tricks involved.
- The value of NCOMP_CVGQ equal to 1, the moisture convergence is computed using the horizontal derivatives of q_v, but q_v itself is gridpoint, so the new additional spectral variable called CVGQ is used to obtain them. Nothing but a mere additional cost of memory is involved.
- The value of NCOMP_CVGQ equal to 2, allows us to compute CVGQ in a semi-Lagrangian manner (Lagrangian tendency - Eulerian tendency), using q_v that is stored in the YCVGQ variable. This case is well designed for the case where q_v is a purely grid-point GFL, and where LSLAG=T.
- The value of NCOMP_CVGQ equal to -1, the moisture convergence is computed using the horizontal derivatives of q_t, while all the q's are gridpoint.
- The value of NCOMP_CVGQ equal to -2, allows us to compute CVGQ in a semi-Lagrangian manner, using q_t that is stored in the YCVGQ variable. This version is not yet properly coded.

7.2 Coding

CVGQ is put into GFL.

- in sufa and yomfa YFACVGQ is introduced,
- in suspeca and wrspeca reading and writing of a variable in a file is supressed for CVGQ spectral GFL.
- in lapineb, if CVGQ variable is gridpoint and advected, the semi-Lagrangian moisture convergence is computed if CVGQ GFL is gridpoint (using only q_v for the time being)

```
   IF (YGFLC(JGFL)%LADV) THEN
     IF ((YGFLC(JGFL)%CNAME == 'CVGQ            ') .AND. YGFLC(JGFL)%LGP) THEN
!    Computation of SL Moisture Convergence for Fench physics
        DO JLEV=1,NFLEVG
          DO JROF=KSTART,KPROF
              PGFLT1(JROF,JLEV,YGFLC(JGFL)%MP1)=&
              &ZGFL9(JROF,JLEV,JGFL) - PQ(JROF,JLEV)
          ENDDO
        ENDDO
      ELSE
         PGFLT1(KSTART:KPROF,1:NFLEVG,YGFLC(JGFL)%MP1)=&
         & PGFLT1(KSTART:KPROF,1:NFLEVG,YGFLC(JGFL)%MP1)+&
         & ZGFL9(KSTART:KPROF,1:NFLEVG,JGFL)
      ENDIF
    ENDIF
```

otherwise, if the variable is spectral, it contains total moisture for negative values of NCOMP_CVGQ, and water vapour for positive NCOMP_CVGQ.

```
IF (YCVGQ%LSP) THEN
  IF((NCOMP_CVGQ == -1) .OR. (NCOMP_CVGQ == -2)) THEN
    PGFLT1(KSTART:KPROF,1:NFLEVG,YCVGQ%MP1)=PGFLT1(KSTART:KPROF,1:NFLEVG,YQ%MP1)+ &
     & PGFLT1(KSTART:KPROF,1:NFLEVG,YL%MP1)+PGFLT1(KSTART:KPROF,1:NFLEVG,YI%MP1)+ &
     & PGFLT1(KSTART:KPROF,1:NFLEVG,YR%MP1)+PGFLT1(KSTART:KPROF,1:NFLEVG,YS%MP1)
  ELSE
    PGFLT1(KSTART:KPROF,1:NFLEVG,YCVGQ%MP1)=PGFLT1(KSTART:KPROF,1:NFLEVG,YQ%MP1)
  ENDIF
ENDIF
```

lattex, lapineb and call_sl should be modified to include other q's in the SL version.

- cpphinp computes the moinsture convergence (actual ZCVGQ array in mf_phys) using horizontal derivatives of q_v if NCOMP_CVGQ is equal to 0, horizontal derivatives of CVGQ GFL if NCOMP_CVGQ is equal to 1 or -1, or Semi-Lagrangian convergence if NCOMP_CVGQ is equal to 2 or -2.
- lattex fills the CVGQ GFL if it is purely gridpoint field with q_v for computation of SL moisture convergence (should also be modified for q_t).
- cpg just uses the new GFL and passes it to mf_phys as arguments (the field and its derivatives).
- mf_phys first calls cpphinp that computes the moisture convergence and then passes it to aplpar
- aplpar uses the moisture convergence unaware of how it was computed.

The problem with the total moisture convergence is in its usage. The convective closure uses the heat that would be released by condensation of the converged moisture. Since the additional moist species are already condensed, they should not be used.