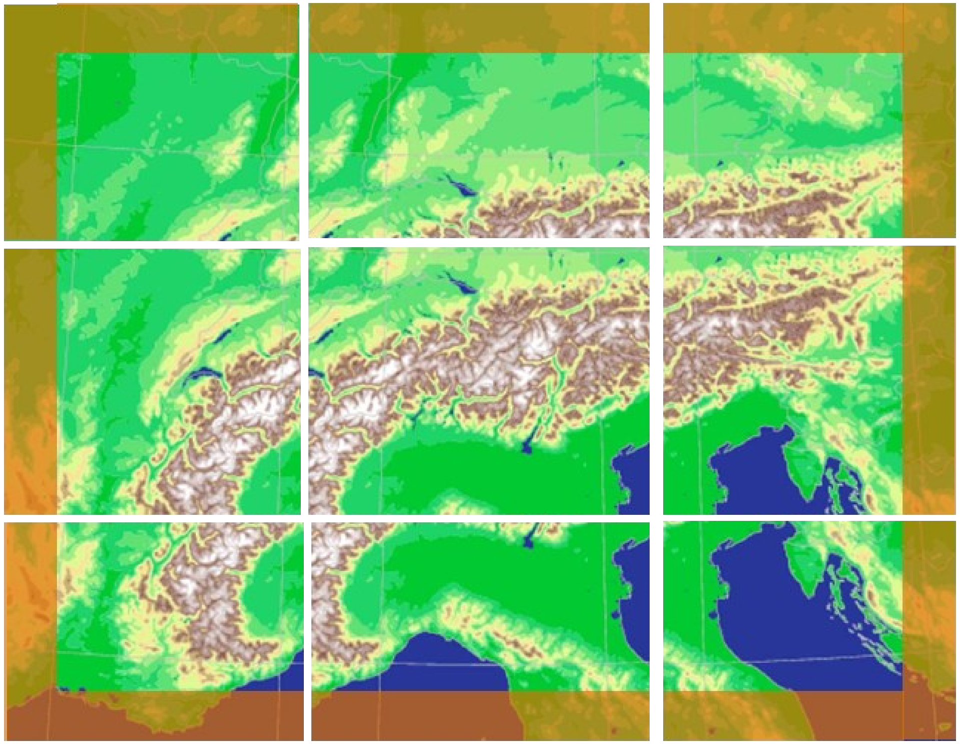
Cleaning up boundary conditions in COSMO

Carlos Osuna (C2SM), Oliver Fuhrer (MeteoSwiss)

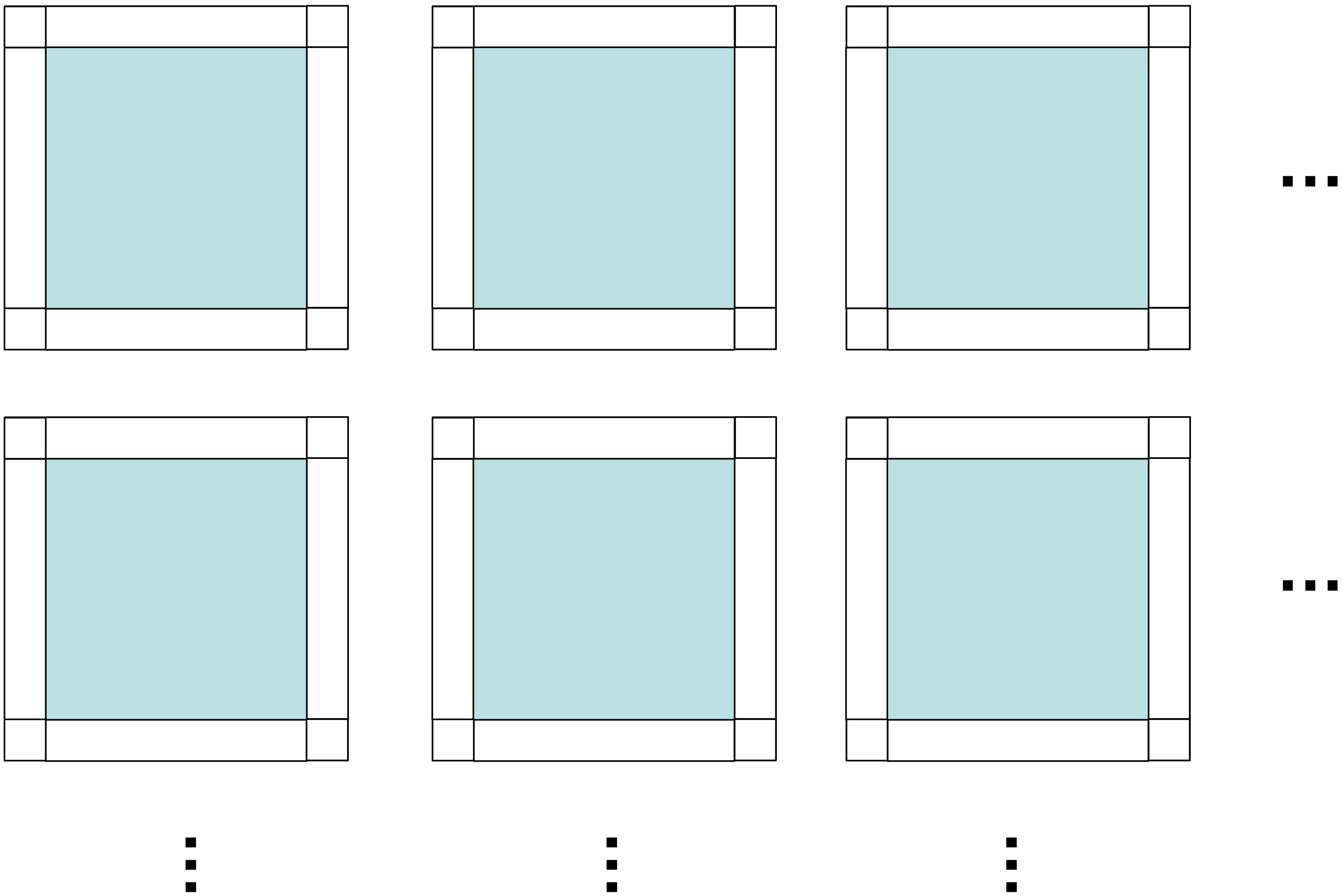
Introduction

Boundary conditions code in cosmo updates the boundaries of the global domain.



Halo exchanges & Boundary Conditions

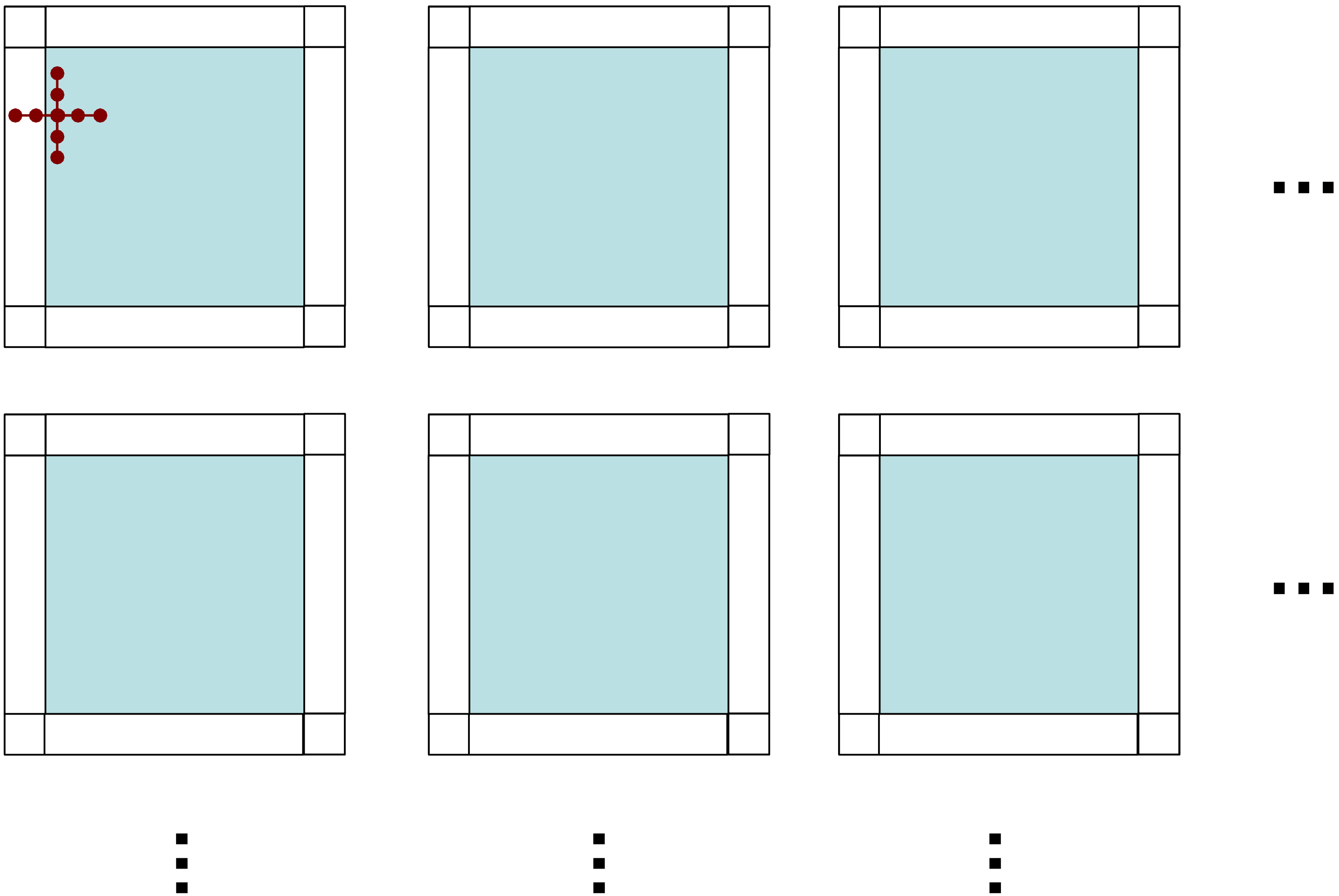
0. Computation (Stencil)



Halo exchanges & Boundary Conditions

0. Computation (Stencil)

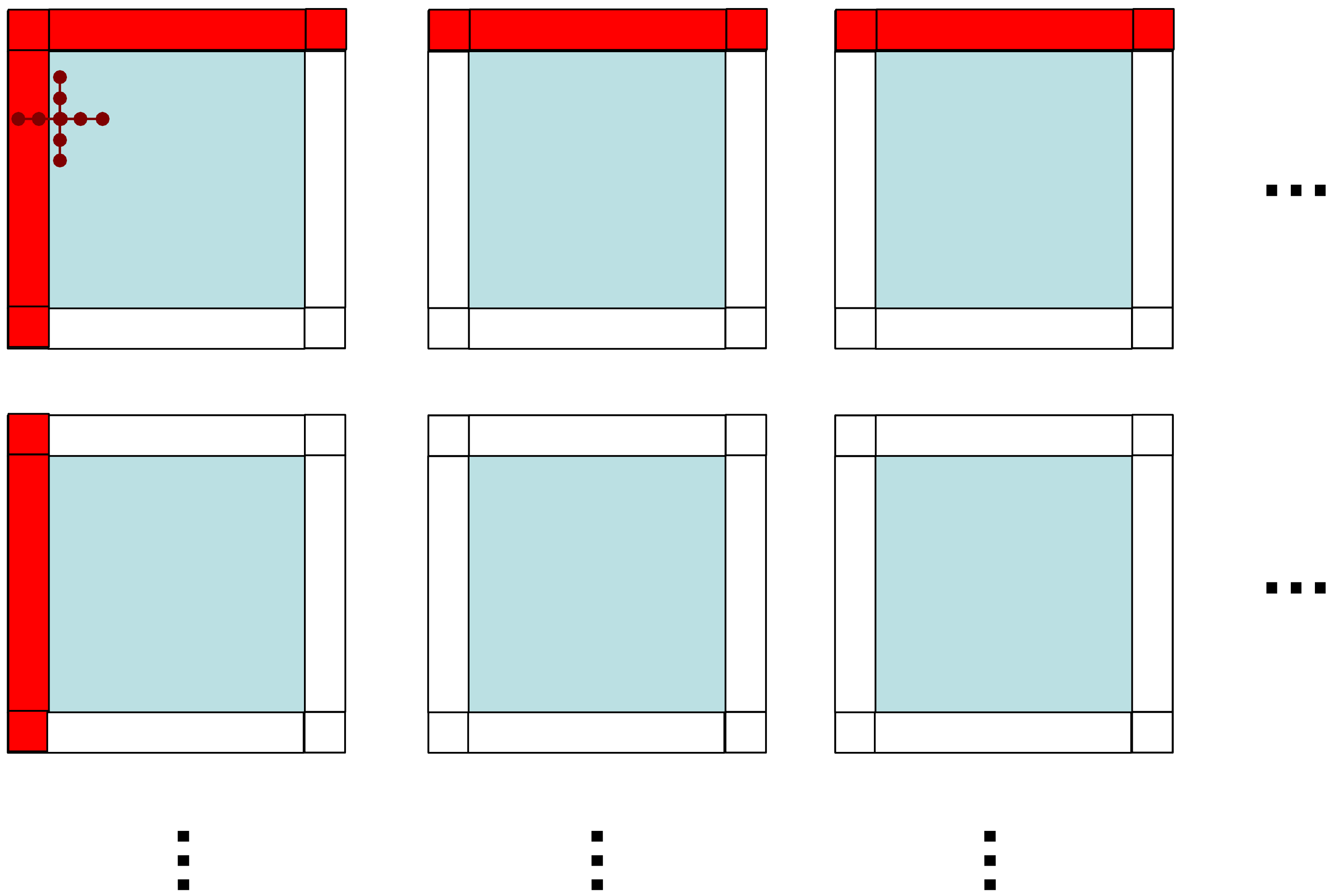
3. Computation (Stencil)



Halo exchanges & Boundary Conditions

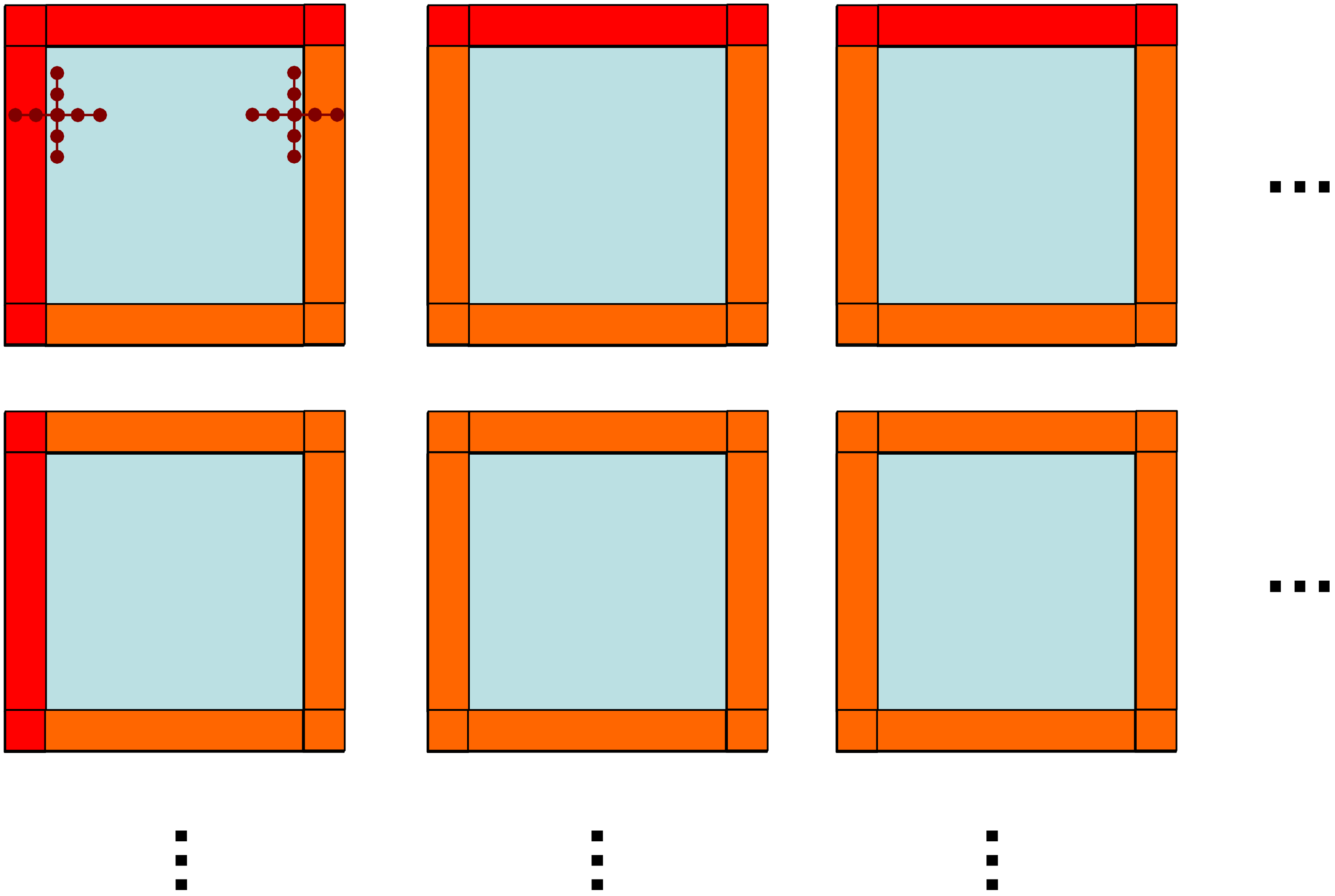
0. Computation (Stencil)

2. Boundary Condition
3. Computation (Stencil)

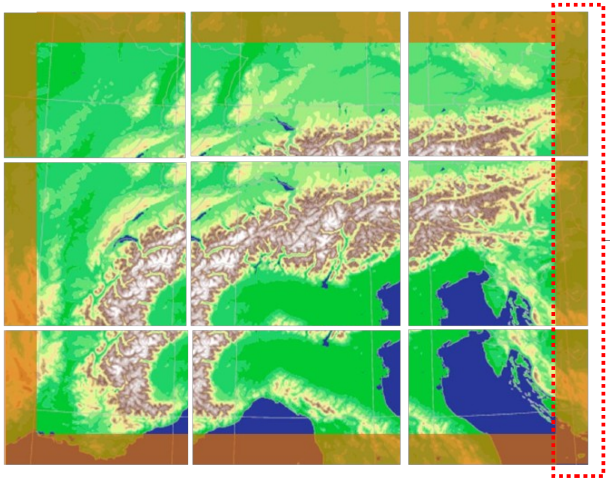


Halo exchanges & Boundary Conditions

0. Computation (Stencil)
1. Halo-Update
2. Boundary Condition
3. Computation (Stencil)



Introduction

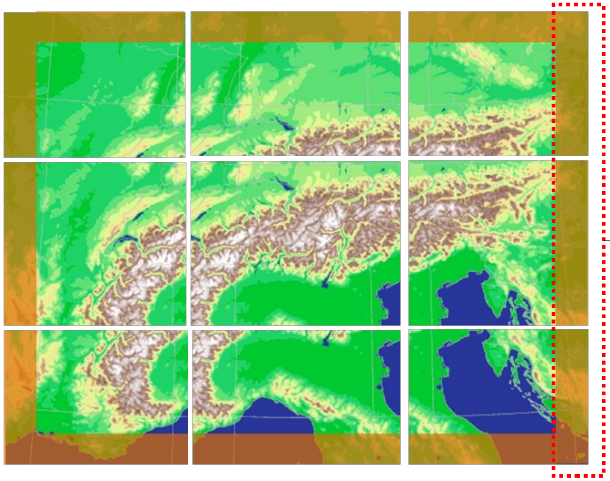


Typical example of boundary condition interpolating values from boundary fields at nnew:

```
IF( my_cart_neigh(3) = -1) THEN
  DO k=1, ke
     DO j=1, je
        DO i=iend+1, ie
           u (i,j,k,nnew) = z1 * u_bd (i,j,k,nbd1) + z2 * u_bd (i,j,k,nbd2)
        ENDDO
     ENDDO
  ENDDO
ENDIF
```

- **BC in cosmo code is not standarized.**

Introduction

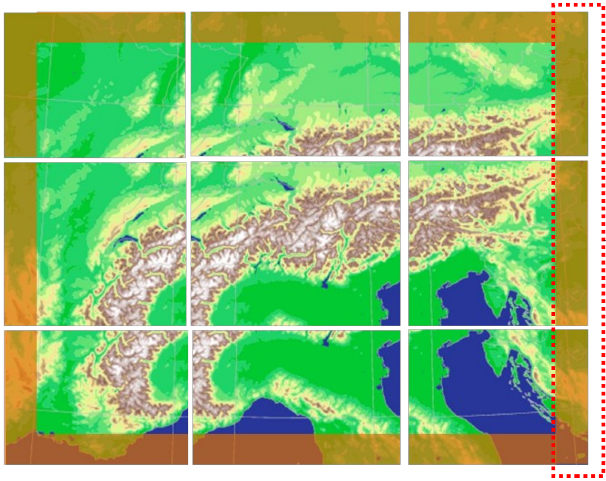


Typical example of boundary condition interpolating values from boundary fields at nnew:

```
IF( my_cart_neigh(3) = -1) THEN
  DO k=1, ke
     DO j=1, je
        DO i=iend+1, ie
           u (i,j,k,nnew) = z1 * u_bd (i,j,k,nbd1) + z2 * u_bd (i,j,k,nbd2)
        ENDDO
     ENDDO
  ENDDO
ENDIF
```

- BC in cosmo code is not standarized.
- **BC are hardcoded in every boundary update → code redundancy.**

Introduction



Typical example of boundary condition interpolating values from boundary fields at nnew:

```
IF( my_cart_neigh(3) = -1) THEN
  DO k=1, ke
    DO j=1, je
      DO i=iend+1, ie
        u (i,j,k,nnew) = z1 * u_bd (i,j,k,nbd1) + z2 * u_bd (i,j,k,nbd2)
      ENDDO
    ENDDO
  ENDDO
ENDIF
```

- BC in cosmo code is not standarized.
- BC are hardcoded in every boundary update → code redundancy.
- **Not always placed were expected (after exchange, before stencil that uses the field).**

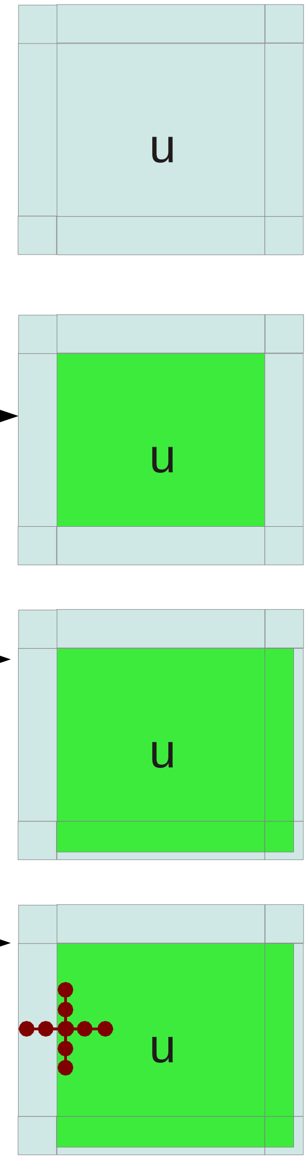
An example:

PE at the NW corner

initialize_loop
…
...
src_runge_kutta
…
...

exchg_boundaries(u(nnew), nlines=2)

hori_diffussion



Initializes u for all domain (including boundaries)

stencil compute u

u modified at compute domain
→ need a halo update
But no BC applied!

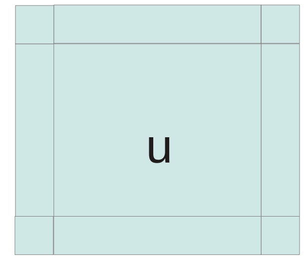
uses u at the boundaries

Risk: if some stencil modifies boundaries between initialize_loop & hori_diffusion.

PE at the NW corner

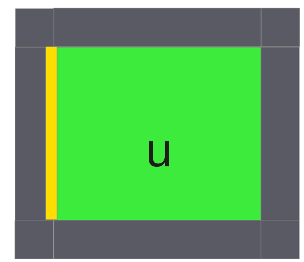
Actually fast_waves modifies the boundaries!

initialize_loop → Initializes u for all domain (including boundaries)



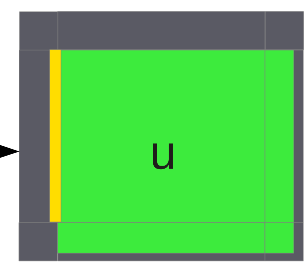
…
…
src_runge_kutta

fast_waves —1 line BC→
…



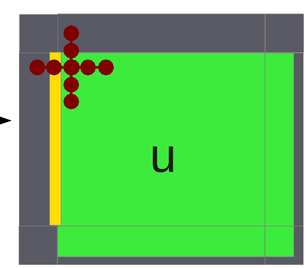
u = u(nnow)
Apply 1 line West/East BC

exchg_boundaries(u(nnew)) →



u modified at compute domain
→ need a halo update
But no BC applied!

hori_diffussion →

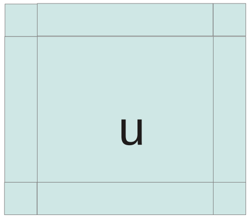


uses u at the boundaries

PE at the NW corner

Ideally:

initialize_loop



Initializes u for all domain (including boundaries)

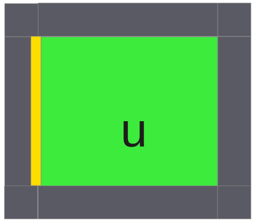
…

...

src_runge_kutta

fast_waves

1 line BC

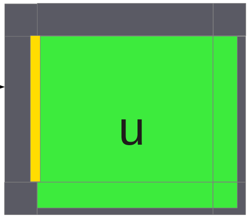


u = u(nnow)

Apply 1 line West/East BC

...

exchg_boundaries(u(nnew))

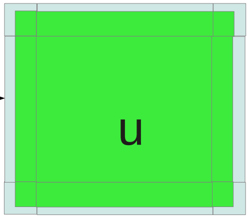


u modified at compute domain

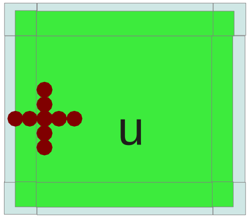
→ need a halo update

But no BC applied!

BC (u(nnew))



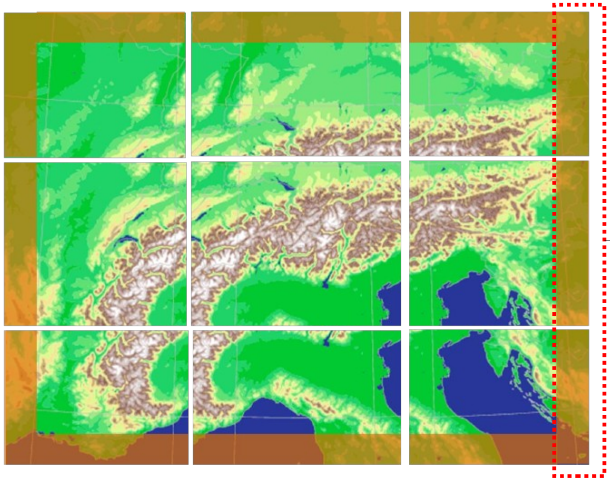
hori_diffussion



uses u at the boundaries

Introduction

Typical example of boundary condition interpolating values from boundary fields at nnew:



```
IF( my_cart_neigh(3) = -1) THEN
  DO k=1, ke
     DO j=1, je
        DO i=iend+1, ie
           u (i,j,k,nnew) = z1 * u_bd (i,j,k,nbd1) + z2 * u_bd (i,j,k,nbd2)
        ENDDO
     ENDDO
  ENDDO
ENDIF
```

- BC in cosmo code is not standarized.
- BC are hardcoded in every boundary update → code redundancy.
- Not always placed were expected (after exchange, before stencil that uses the field).
- **Different strategies are applied in different boundary updates.**

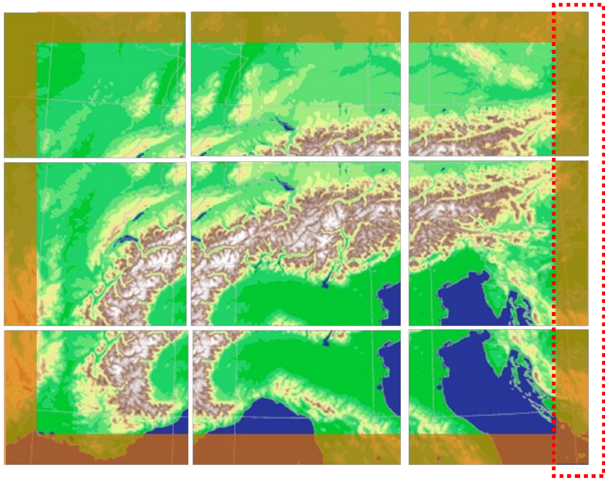
Different strategies are applied in different boundary updates.

Example in tracers (qi, qr, qs, qg):

If llb_qi = FALSE boundary conditions for qi are set to 0-value,

except for advection_pd, which uses zero_gradient

Introduction



Typical example of boundary condition interpolating values from boundary fields at nnew:

```
IF( my_cart_neigh(3) = -1) THEN
  DO k=1, ke
    DO j=1, je
      DO i=iend+1, ie
        u (i,j,k,nnew) = z1 * u_bd (i,j,k,nbd1) + z2 * u_bd (i,j,k,nbd2)
      ENDDO
    ENDDO
  ENDDO
ENDIF
```

- BC in cosmo code is not standarized.
- BC are hardcoded in every boundary update → code redundancy.
- Not always placed were expected (after exchange, before stencil that uses the field).
- Different strategies are applied in different boundary updates.
- **Sometimes BC code mixed with other code not related with BC.**
  Example: initialize_loop → mixing BC & initialization of fields.

Goals and Motivation for new BC implementation

- Standarize the boundary conditions code in COSMO.
- Avoid code duplication (reuse code, less errors, easy validation).
- Handle all types of boundary conditions in similar way.
- Better control of coupling of boundary conditions with halo exchanges.
- Reorganize BC & halo updates.
  Remove dependencies between code that applies BC and Stencil that uses boundary data.
- Clean code and easy to understand when applying boundary conditions.

Boundary condition types

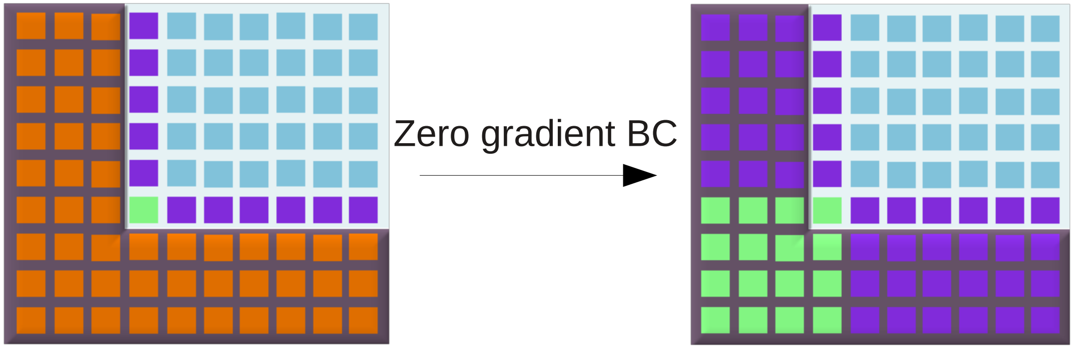
- **Interpolate boundary fields.**

u (i,j,k,nnew) = z1 * u_bd (i,j,k,nbd1) + z2 * u_bd (i,j,k,nbd2)

Boundary condition types

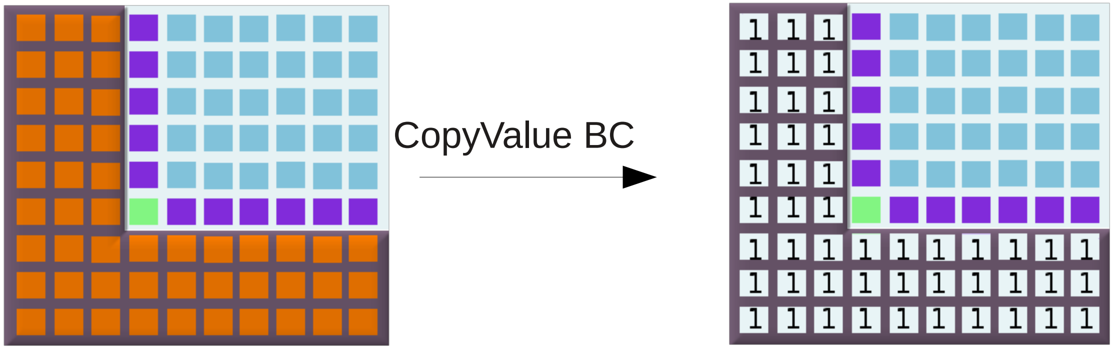
- Interpolate boundary fields.

- **Zero gradient**



Boundary condition types

- Interpolate boundary fields.
- Zero gradient
- **Copy value**



Boundary condition types

- Interpolate boundary fields.
- Zero gradient
- Copy value
- **Mirror**

West boundary	North Boundary
u(nboundlines,:,:) = -u(istart,:,:)	u(:,nboundlines,:) = u(:,jstart,:)
v(nboundlines,:,:) = v(istart,:,:)	v(:,nboudlines,:) = -v(:,jstart,:)

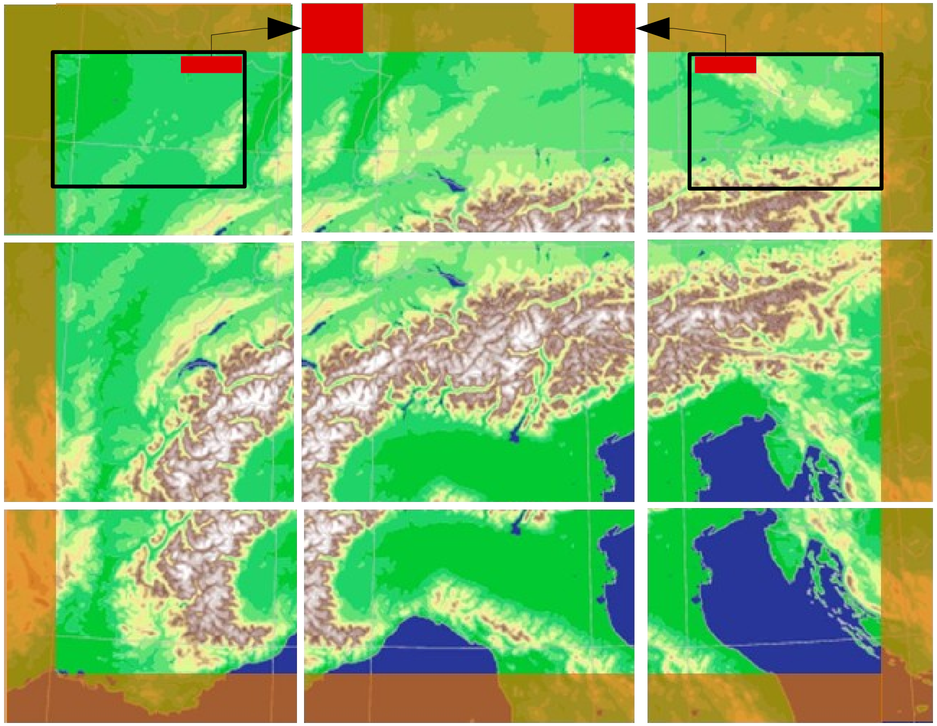
Boundary condition types

- Interpolate boundary fields.
- Zero gradient
- Copy value
- Mirror
- **Constant boundary condition** It does nothing!!!

Coupling to Halo Exchanges

In some BC types , like zero gradient, data update of corners of boundaries should come from neighbours.

If halos of a PE are not updated, this boundary condition will apply incorrect values at the corners.



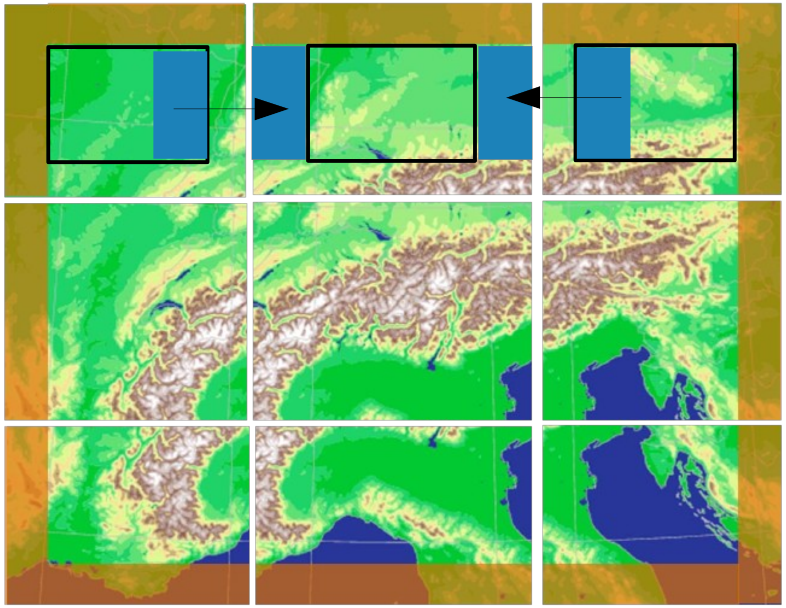
Coupling to Halo Exchanges

For a stencil that reads data at the boundary fixing the sequence

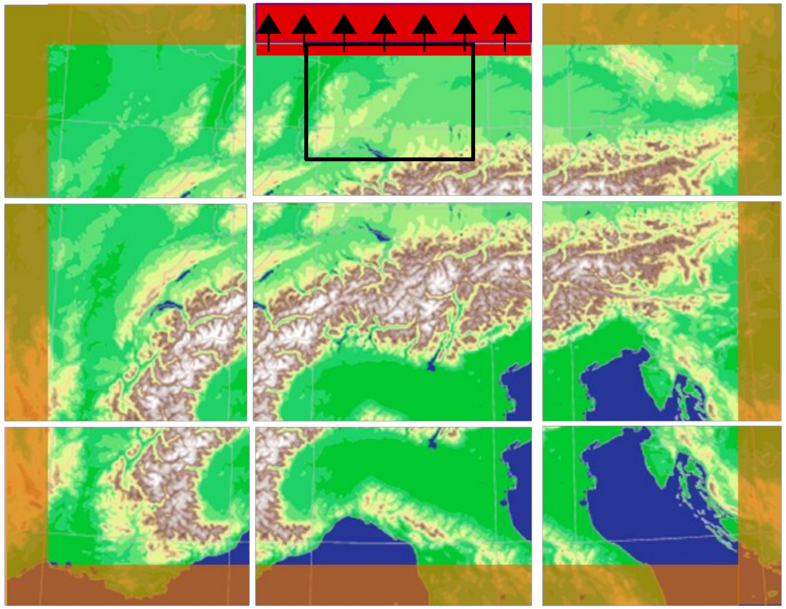
Halo Exchange → Boundary Condition → Stencil

always provides consistent data at the boundary for all the boundary condition types.

1st Step: Halo Exchange



2nd Step: Boundary Condition



New BC Implementation

API

lbc_masspoint | **lbc_upoint** | **lbc_vpoint**

Set coordinates of boundaries

lbc_compute_tendency

↓

lbc_backend

Check consistency of arguments.
Precomputes boundary regions to be computed.
Calls specific BC implementations.

↓

lbc_zerogradient | **lbc_interpol** | **lbc_copyvalue**

Actual implementation of BC algorithm

API example:

```
SUBROUTINE lbc_upoint_3d(lbc_type, field, field_type, nlines, doEW, doNS,      &
  doCorners, value, src, bd1, bd2, nincbound, nlastbound, tend, dt, mask )
```

lbc_type: specify BC type to be applied (zerogradient, interpolation, etc.)

field_type: whether scalar, horizontal component of vector, vertical component of vector. (some algorithms like Mirror may depend on it)

nlines: number of lines for region where to apply BC.

doEW, doNS: whether apply to EastWest (NorthSouth) boundaries or not.

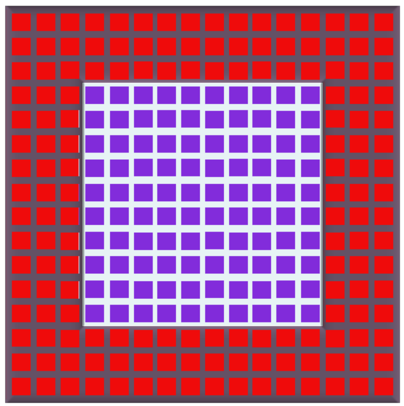
doCorners: whether apply BC to corners.

mask: field with logicals used as a mask to BC algorithm.

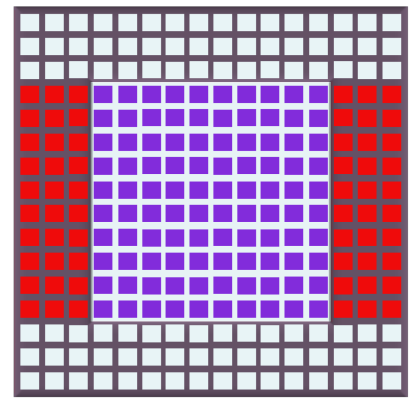
BC patterns allowed

Implemented all possible patterns for BC observed in COSMO.

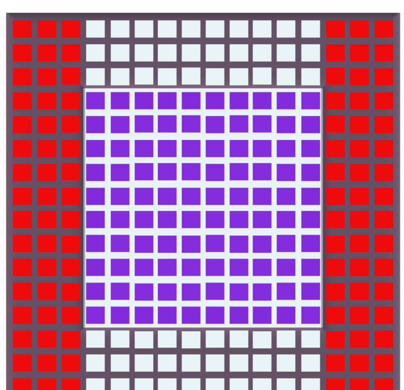
doEW, doNS, doCorners
(eg. initialize_loop)



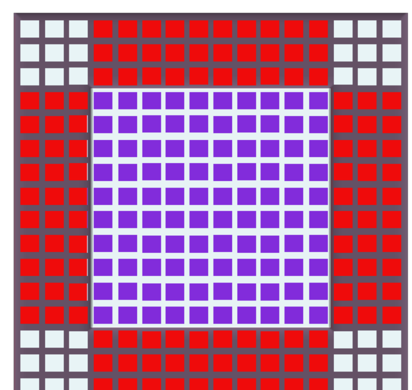
doEW, No NS, No Corners
(fast_waves)



doEW, No NS, doCorners



doEW, doNS, No Corners



Compute Boundary from BD fields or tendencies

We provide two working modes for doing Interpolation of boundaries.

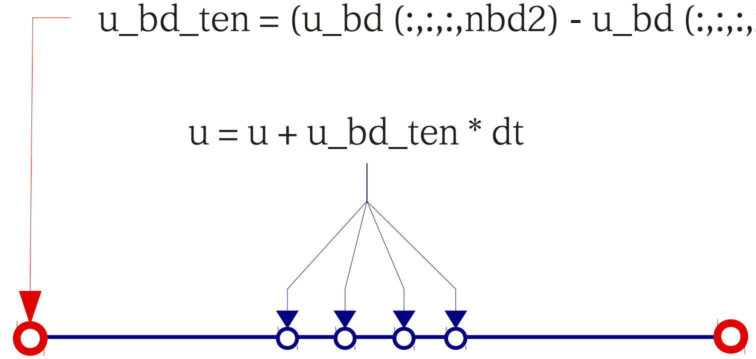
Using two boundary fields (u_bd)

u (i,j,k,nnew) = z1 * u_bd (i,j,k,nbd1) + z2 * u_bd (i,j,k,nbd2)

Using boundary tendency

u_bd_ten = (u_bd (:,:,:,nbd2) - u_bd (:,:,:,nbd1)) / nincbound

u = u + u_bd_ten * dt



Compute Boundary from BD fields or tendencies

We provide two working modes for doing Interpolation of boundaries.

Using two boundary fields (u_bd)

```
CALL lbc_upoint( BCType_Tendency, u(:,:,:,nnew), field_type=BCFieldType_VectorI, &
        nlines=1, bd1=u_bd(:,:,:,nbd1), bd2=u_bd(:,:,:,nbd2), doEW=.TRUE., doNS=.FALSE. )
```

Using boundary tendency

```
CALL lbc_compute_tendency( u_bd_tend, u(:,:,:,nnew), u(:,:,:,nnow), dt, &
        istartu, iendu, jstartu, jendu, nlines=1, doEW=.TRUE., doNS=.FALSE. )

CALL lbc_upoint( BCType_Tendency, u(:,:,:,nnew), field_type=BCFieldType_VectorI, &
        nlines=1, tend=u_bd_tend, dt=dts, doEW=.TRUE., doNS=.FALSE. )
```

Status

- ✔ Implemented subroutines for new boundary conditions in a src_lbc module.
- ✔ Replaced and tested at 3 places in COSMO: fast_waves_rk, advection_pd, initialize_loop
- ✔ Full set of testing functions, testing basic functionality of new BC deployed within src_lbc

DONE

- ✗ Systematically replace all the remaining bc related code in COSMO

NOT DONE

Summary

- ✔ Proposal to clean up the boundary conditions code. Implementated API and full set of boundary condition types.
- ✔ Replaced (and validated) 3 BC blocks in COSMO
- ✔ Testing functions shipped with the module src_lbc.f90
- ✔ Next step: replace all BC code in COSMO with new subroutines.

We are glad to receive feedback, ideas?