

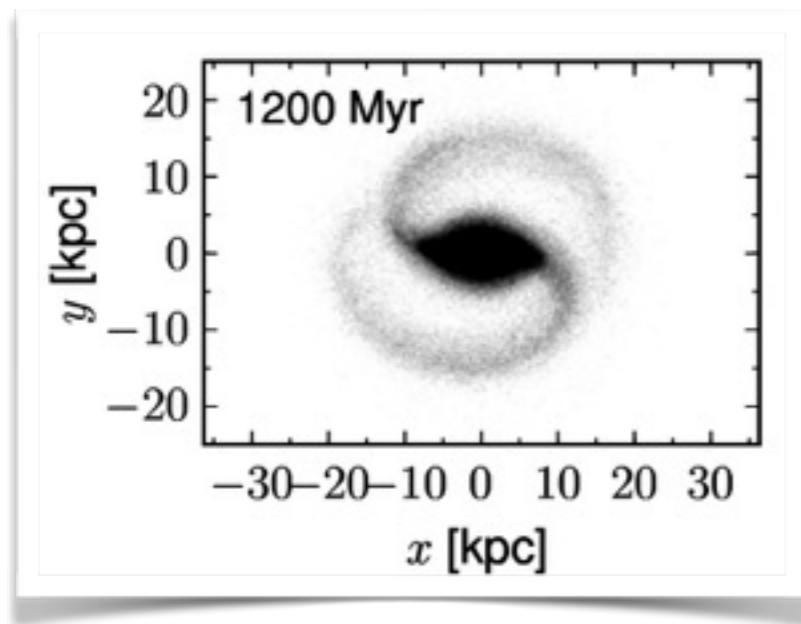
- Who has already worked with RAMSES?
- Who has already used the “phantom” patch?
- Who intends to work with it (personally) in the future?

Could be helpful for most of us to share our experiences, in particular concerning the processing of RAMSES output:

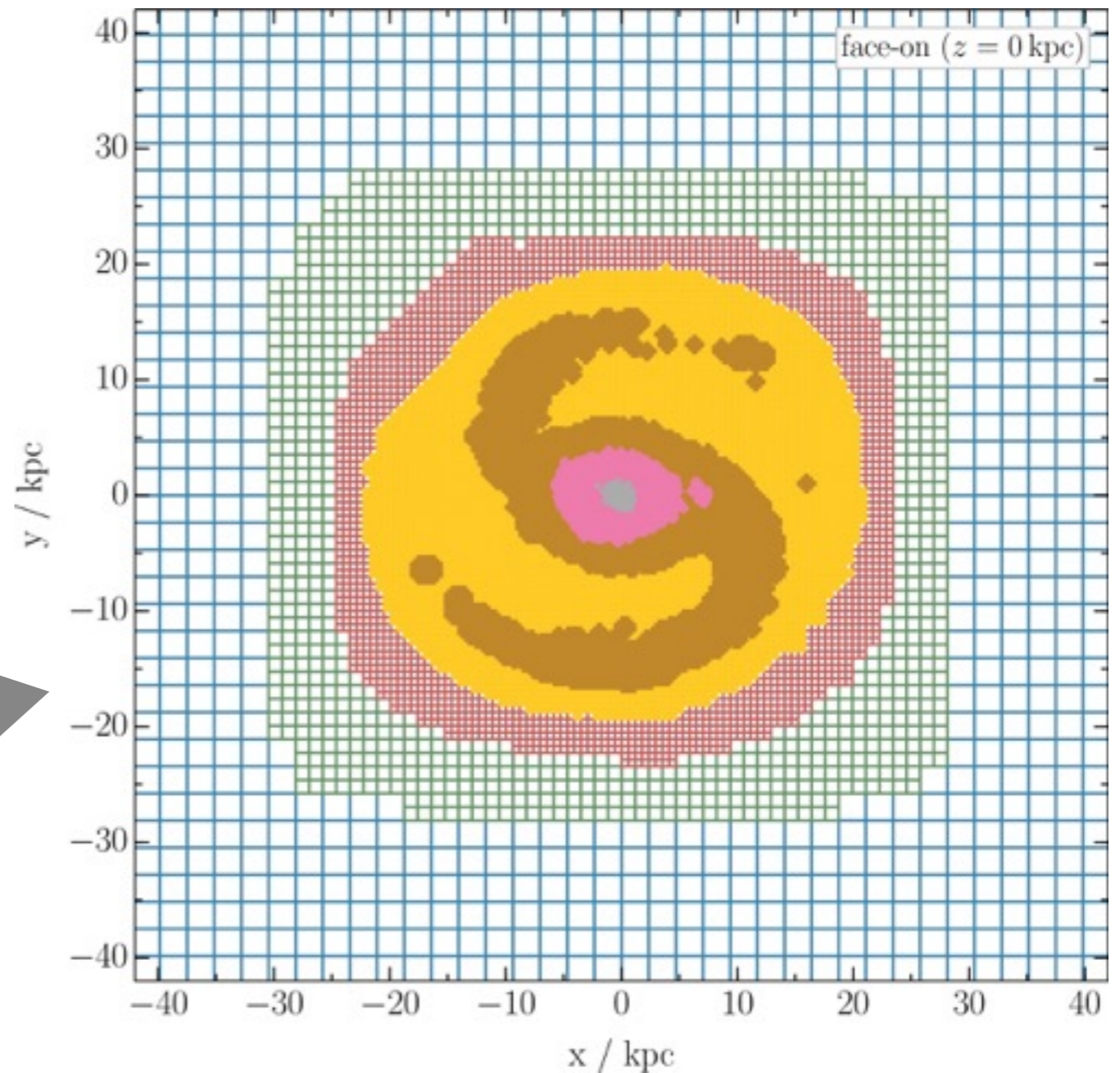
- Snapshots and/or movies
- Already publicly available tools

RAMSES

- Hydrodynamics, MHD
- Particle–mesh N-body
- Adaptive mesh refinement



gas, stars



density

RAMSES

Particle mesh technique

Stars are mapped on a density grid $\rho(\boldsymbol{x})$

→ Newtonian potential $\phi_{\text{N}}(\boldsymbol{x})$ via $\nabla^2 \phi_{\text{N}}(\boldsymbol{x}) = 4\pi G \rho(\boldsymbol{x})$

→ acceleration $\boldsymbol{g}_{\text{N}}(\boldsymbol{x}) = -\nabla \phi_{\text{N}}(\boldsymbol{x})$

→ move particles according to eqs. of motion

Quasi-linear Poisson equation

Grav. potential: $\Phi = \phi_b + \phi_{\text{ph}}$

Eq. of motion: $\ddot{\mathbf{x}} = -\nabla\Phi$

$$\nabla^2 \phi_b(\mathbf{x}) = 4\pi G \rho_b(\mathbf{x}) \quad \text{Baryonic matter}$$

$$\nabla^2 \phi_{\text{ph}}(\mathbf{x}) = 4\pi G \rho_{\text{ph}}(\mathbf{x}) \quad \text{"Phantom DM"}$$

Effective potential

bary. matter

"Phantom DM"

Poisson eq.

$$\nabla^2 \Phi(\mathbf{x}) = 4\pi G (\rho_b(\mathbf{x}) + \rho_{\text{ph}}(\mathbf{x}))$$

with the density $\rho_{\text{ph}}(\mathbf{x}) = \frac{\nabla \cdot [\nu (|\nabla \phi_b(\mathbf{x})|/a_0) \nabla \phi_b(\mathbf{x})]}{4\pi G}$

RAMSES

QUMOND style :)

Stars are mapped on a density grid $\rho_b(\mathbf{x})$

→ Newtonian potential via $\nabla^2 \phi_b(\mathbf{x}) = 4\pi G \rho_b(\mathbf{x})$

→ PDM density $\rho_{\text{ph}}(\mathbf{x}) = \frac{\nabla \cdot [\nu (|\nabla \phi_b(\mathbf{x})|/a_0) \nabla \phi_b(\mathbf{x})]}{4\pi G}$

→ QUMOND potential via $\nabla^2 \Phi(\mathbf{x}) = 4\pi G (\rho_b(\mathbf{x}) + \rho_{\text{ph}}(\mathbf{x}))$

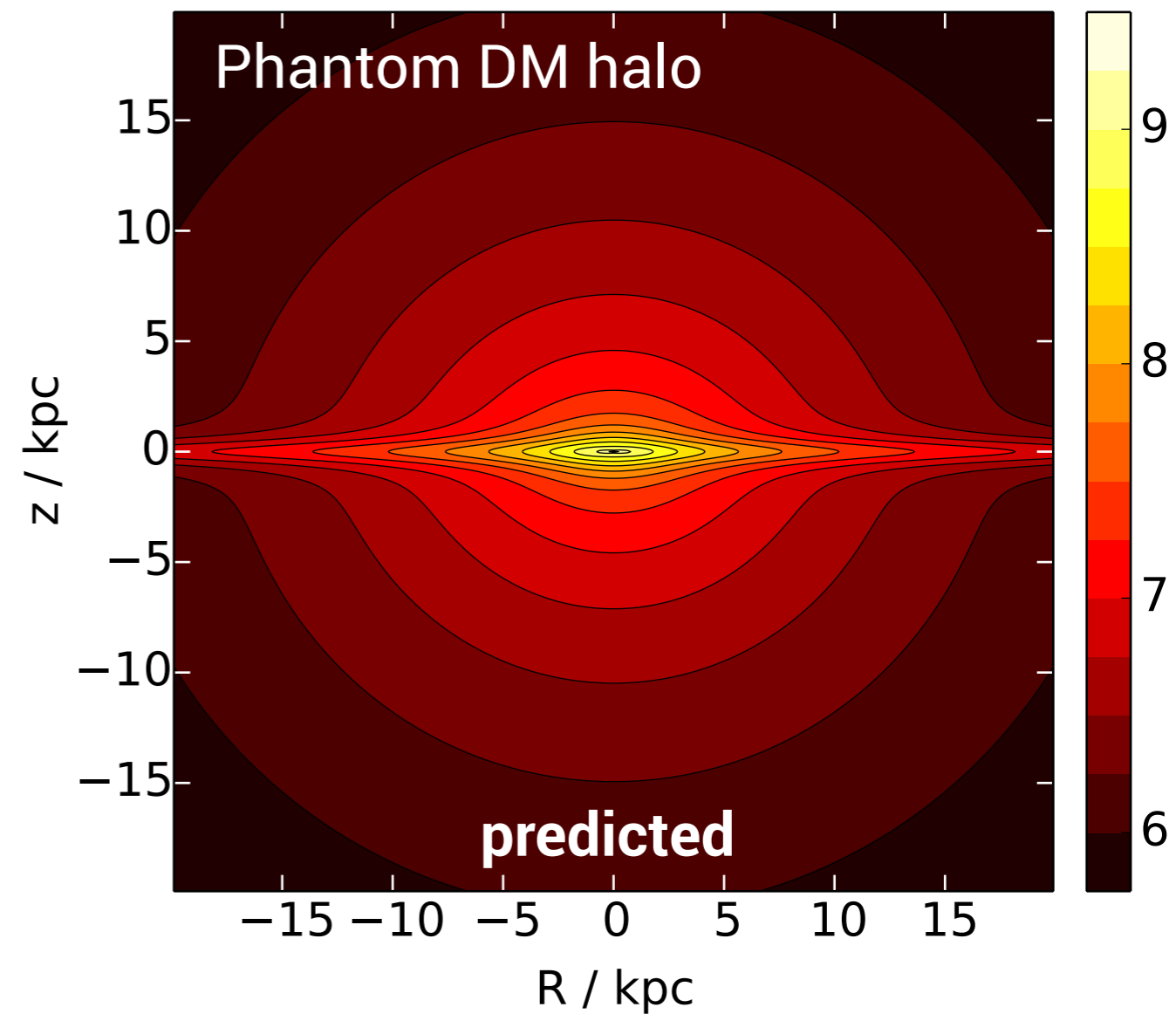
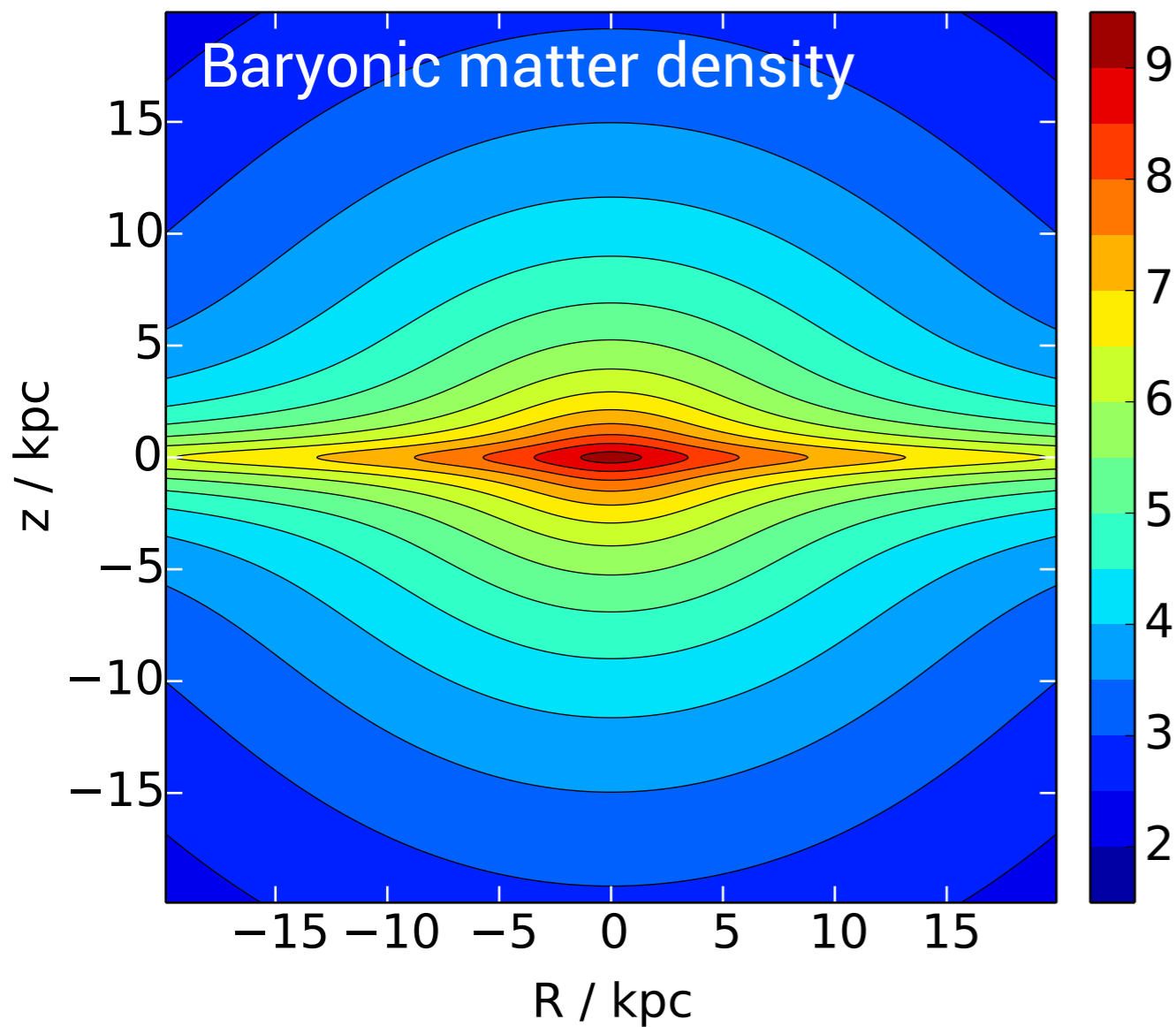
→ acceleration $\mathbf{g}(\mathbf{x}) = -\nabla \Phi(\mathbf{x})$

→ move particles according to eqs. of motion

Phantom DM – example

Miyamoto–Nagai disk galaxy model

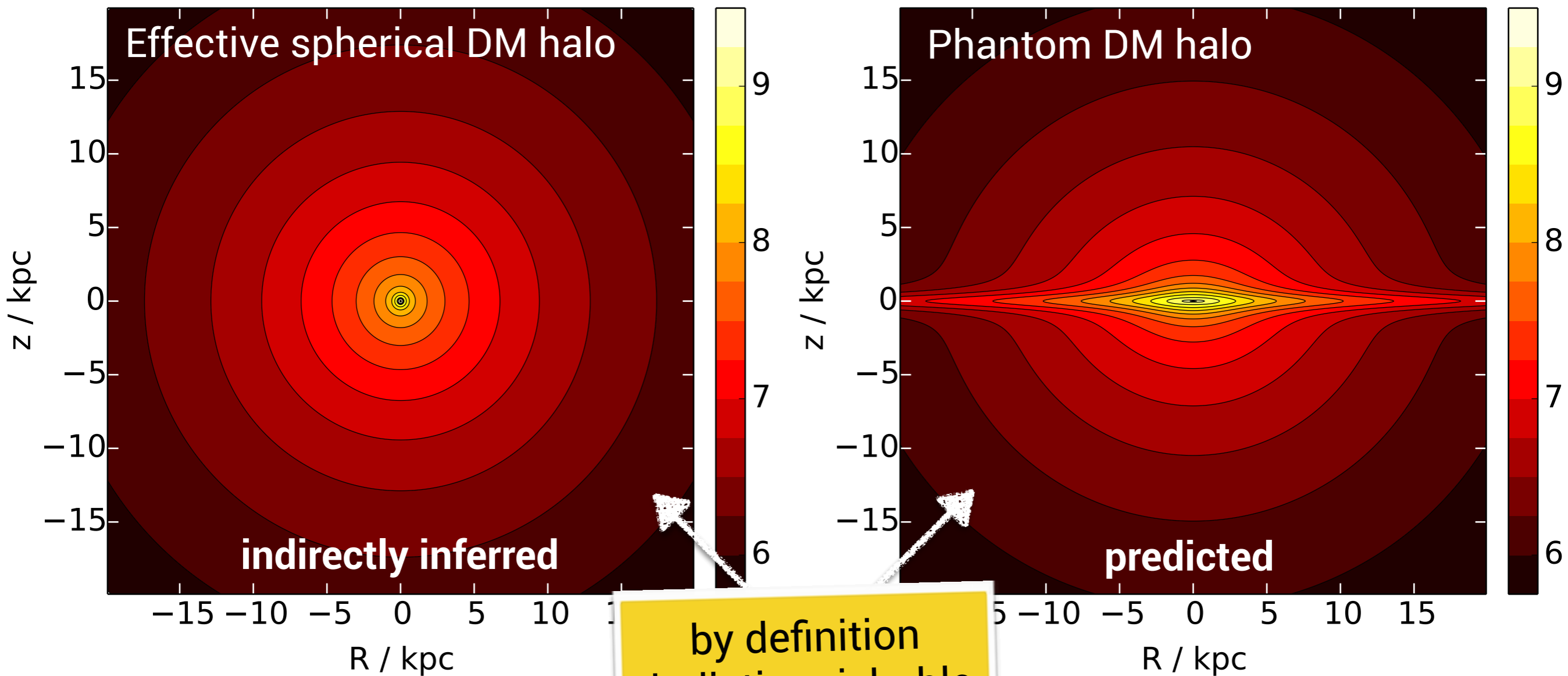
$$\log_{10} (\rho / (M_{\odot} \text{ kpc}^{-3}))$$



Phantom DM – example

Miyamoto–Nagai disk galaxy model

$$\log_{10} (\rho / (M_{\odot} \text{ kpc}^{-3}))$$

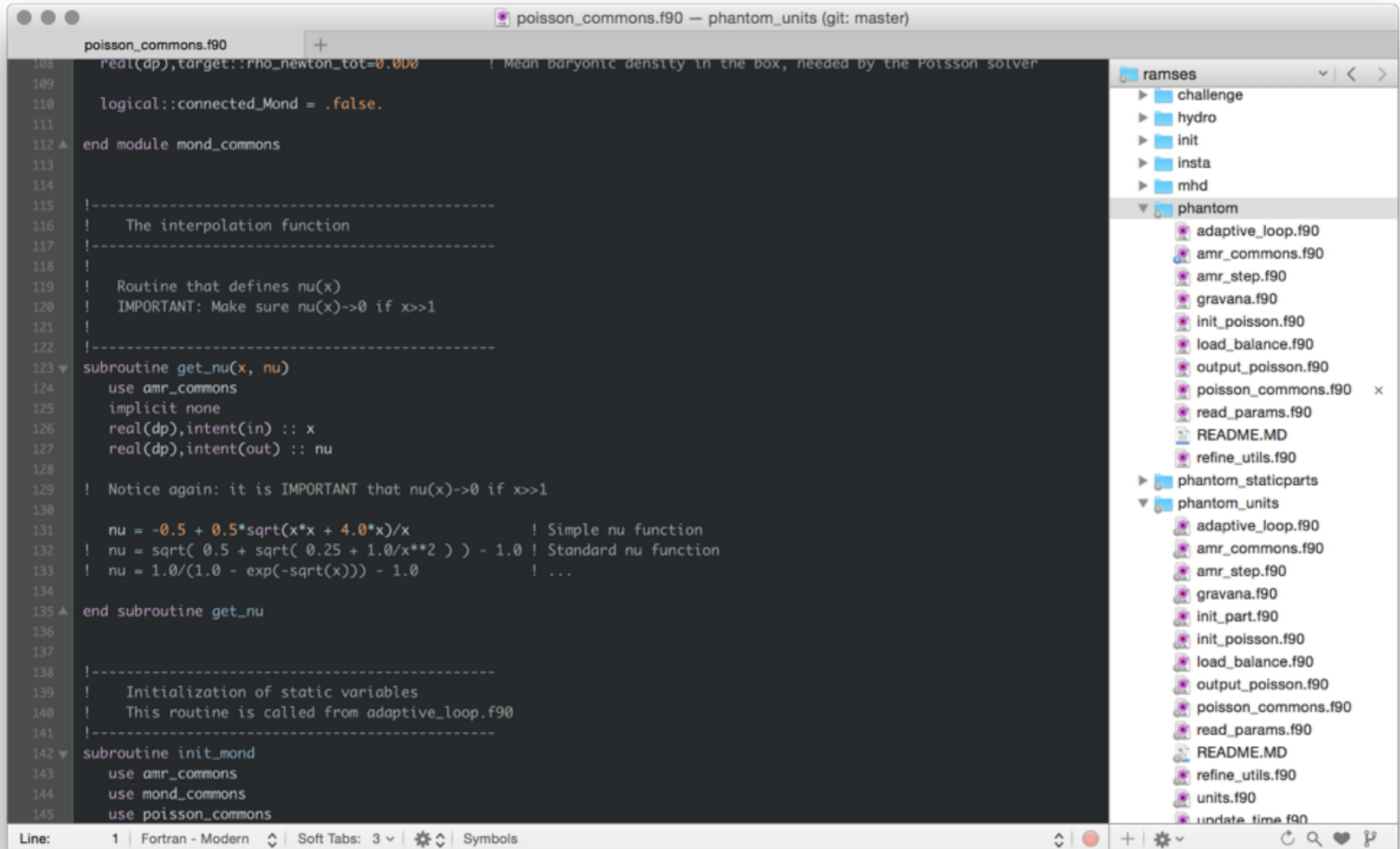


The “phantom” patch

- The code is publicly available through the standard RAMSES package, <https://bitbucket.org/rteyssie/ramses>
- Edit the “Makefile” and
 - Set **patch = ../patch/phantom**
 - Set **NDIM = 3**
 - Also adjust the compiler settings (gfortran, ifort, OpenMPI)
- Finally, run “make” to compile the code
- Beware of *N*-body units (see the RAMSES docs)



See <http://phantomwiki.de>



The image shows a code editor window titled "poisson_commons.f90 — phantom_units (git: master)". The editor displays Fortran code for a module named "mond_commons". The code includes comments about the mean baryonic density and the interpolation function. A subroutine "get_nu" is defined, which calculates the viscosity coefficient ν based on the Mach number x . The code also includes an initialization routine "init_mond".

```
108 real(dp),target::rho_newton_tot=0.0D0 ! Mean baryonic density in the box, needed by the Poisson solver
109
110 logical::connected_Mond = .false.
111
112 end module mond_commons
113
114
115 !-----
116 ! The interpolation function
117 !-----
118 !
119 ! Routine that defines nu(x)
120 ! IMPORTANT: Make sure nu(x)->0 if x>>1
121 !
122 !-----
123 subroutine get_nu(x, nu)
124 use amr_commons
125 implicit none
126 real(dp),intent(in) :: x
127 real(dp),intent(out) :: nu
128
129 ! Notice again: it is IMPORTANT that nu(x)->0 if x>>1
130
131 nu = -0.5 + 0.5*sqrt(x*x + 4.0*x)/x ! Simple nu function
132 ! nu = sqrt( 0.5 + sqrt( 0.25 + 1.0/x**2 ) ) - 1.0 ! Standard nu function
133 ! nu = 1.0/(1.0 - exp(-sqrt(x))) - 1.0 ! ...
134
135 end subroutine get_nu
136
137
138 !-----
139 ! Initialization of static variables
140 ! This routine is called from adaptive_loop.f90
141 !-----
142 subroutine init_mond
143 use amr_commons
144 use mond_commons
145 use poisson_commons
```

The file explorer on the right shows the directory structure of the "phantom" project, including subdirectories like "challenge", "hydro", "init", "insta", "mhd", and "phantom". The "phantom" directory contains files such as "adaptive_loop.f90", "amr_commons.f90", "amr_step.f90", "gravana.f90", "init_poisson.f90", "load_balance.f90", "output_poisson.f90", "poisson_commons.f90", "read_params.f90", "README.MD", and "refine_utils.f90".

The “phantom” patch

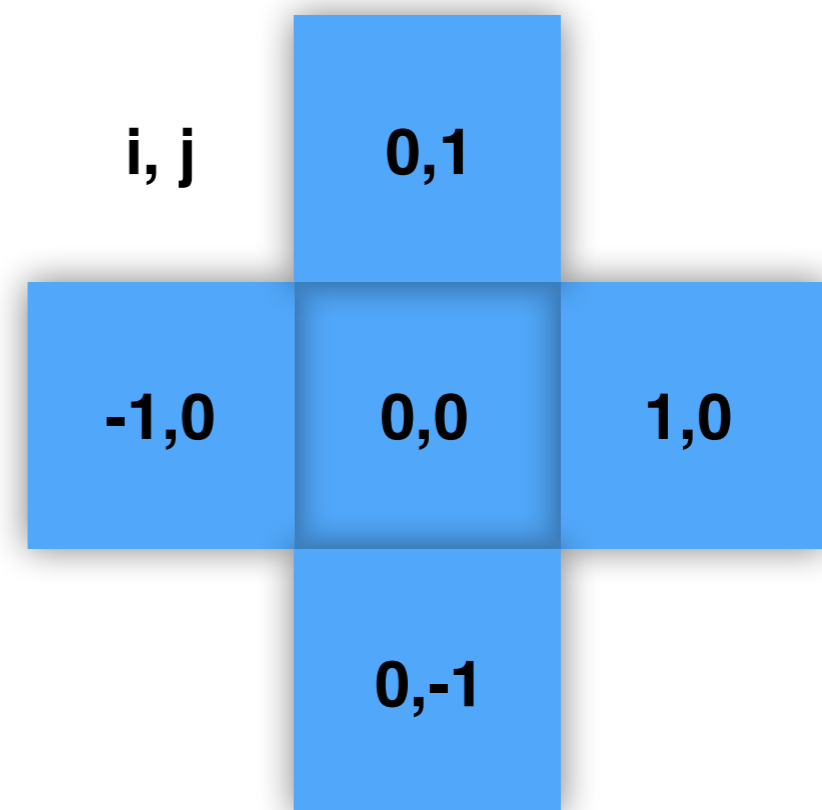
In the run configuration file (*.rml), set

```
&RUN_PARAMS
poisson=.true.
pic=.true.
mond=.true.
a0=(...)
/

&BOUNDARY_PARAMS
nboundary=6
ibound_min=-1, 1, 0, 0, 0, 0,
ibound_max=-1, 1, 0, 0, 0, 0,
jbound_min= 0, 0, -1, 1, 0, 0,
jbound_max= 0, 0, -1, 1, 0, 0,
kbound_min= 0, 0, 0, 0, -1, 1,
kbound_max= 0, 0, 0, 0, -1, 1,
bound_type= 3, 3, 3, 3, 3, 3,
/
```

By the way, the nu-function can be adjusted easily in the code:
See **poisson_commons.f90**,
subroutine **get_nu(x, nu)**

But you will need to recompile it!



The “phantom” patch

Boundary conditions

(applied at the simulation box boundaries)

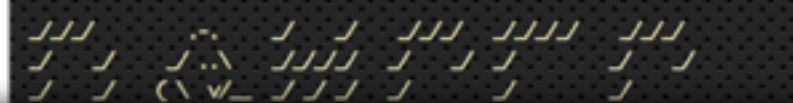
- 1st round: $\phi_b(\mathbf{x}) = GM_b / |\mathbf{x} - \mathbf{x}_0|$
- 2nd round: $\Phi(\mathbf{x}) = (GM_b a_0)^{1/2} \ln(|\mathbf{x} - \mathbf{x}_0|)$

=> The box length must be large enough, if compared to the simulated objects inside, to fulfill these conditions!

Phantom of RAMSES *N*-body code

```
1. ramses3d
gravitygun:MilkyWay fabian$ mpirun -n 8 ramses3d config.mil
```

~ The Phantom of ~



Exponential disk with initial bulge in Milgromian dynamics

