```
subroutine update
use data_module
implicit none

!
!.... local variables
integer :: i,j,l
real :: temp
real :: omega_bar,omega_bar_a,p
real, dimension(4) :: effux,s,u,w
!
!.... axisymmetric h-grid case?
omega_bar = 0.
if ( igrid_topology == 1 .and. h_grid_axisymmetric ) then
omega_bar = 1.
endif
!
!.... loop cells and compute soluton update
do j = 1,jmax-1
do i = 1,imax-1

effux(:) = - ( &
+ f_i(:,i,j) * l_i(i,j) * ( ( 1. - omega_bar ) + omega_bar * 0.5 * ( y(i,j) +
y(i,j+1) ) ) &
+ f_j(:,i,j) * l_j(i,j) * ( ( 1. - omega_bar ) + omega_bar * 0.5 * ( y(i,j) +
y(i+1,j) ) ) &
- f_i(:,i+1,j) * l_i(i+1,j) * ( ( 1. - omega_bar ) + omega_bar * 0.5 * ( y(i+1,j)
+ y(i+1,j+1) ) ) &
- f_j(:,i,j+1) * l_j(i,j+1) * ( ( 1. - omega_bar ) + omega_bar * 0.5 * ( y(i,j+1)
+ y(i+1,j+1) ) ) &
)

u(:) = uc(:,i,j)
call primitive_variables(gamma,u,w)
p = w(4)

s(1) = 0.
s(2) = 0.
s(3) = p
s(4) = 0.

effux(:) = effux(:) + omega_bar * s(:) * area(i,j)

omega_bar_a = ( 1. - omega_bar ) + omega_bar * yc(i,j)

duc(:,i,j) = dt_cell(i,j) / area(i,j) * effux(:) / omega_bar_a

uc(:,i,j) = uc(:,i,j) + duc(:,i,j)

enddo
```

```
enddo

!
!.... compute magnitude of solution residual
temp = 0.
do j = 1,jmax-1
do i = 1,imax-1
do l = 1,4
temp = temp + duc(l,i,j)**2
enddo
enddo
enddo
log10_l2_residual = log10( sqrt(temp) / float( 4 * (imax-1) * (jmax-1) ) )

!
!.... compute solution residual

end subroutine update
```