```
else
        vk2 = (v/2)**2 * (2/dx)**2 * sin(k*dx/2)**2  / (1 -
                                (4./bi) * sin(k*dx/2)**2 )

do iw = 1, nw {                                  # loop over all ω
        omega = 2*pi * (iw-1.) / nw
        if( omega > pi )    omega = omega - 2*pi
        omega = omega / dt
        cz = cexp( cmplx( 0., omega * dt ) )
        if( omhat == 0 )
                cs = cmplx( 1.e-5 / dt, - omega)
        else
                cs = (2./dt) * (1. - rho * cz) / (1. + rho * cz)
        if ( degree == 90 )
                cikz = vk2 / ( csqrt( cs * cs + vk2 ) + cs )
        if ( degree == 15  |  degree == 45 )
                cikz = vk2 / ( eps + (r0+1.) * cs )
        if (degree == 45 )
                cikz =  vk2 / ( 2.*cs + cikz)
        if( real( cikz) < 0.) call erexit("cikz not positive real")

        if( kzhat == 0 )
                cp(iw) = cexp( - tau0  * cikz )
        else
                cp(iw) = ((1.-cikz * dz/2) / (1.+cikz * dz/2) ) ** (tau0/dz)
        cp(iw) = cp(iw) * cexp(cmplx(0., omega * tau0 ))          # unretard

        if( tfilt >= 1 ) cp(iw) = cp(iw) * (1+cz) / (1 - .8 * cz)
        if( tfilt >= 2 ) cp(iw) = cp(iw) * (1-cz) / (1 - .8 * cz)
        if( xfilt == 1 ) cp(iw) = cp(iw) * (1+cos(k*dx)) / (1+.85*cos(k*dx))
        cp(iw) = cp(iw) * cexp( cmplx(0.,k*x0) )
        }
    call rite( outfd, cp, 8*nw )    # write
    }
stop;   end
# Finally, you must 2-D Fourier Transform (Section 1.7), take real part, and plot.
```