Listings Setting for FreeFEM code

Zhang Jiayi

September 25, 2022

```
/*
 * quasi.edp
 */

/*------------------------------------------------------
 * Parameters & Functions
 *------------------------------------------------------*/
real ksquare   = 1;     // coefficient k^2
real alpha     = 0.1;
complex lambda = 1i;    // coefficient of the impedance boundary condition
complex iunit  = 1i;
func f         = x * y; // right hand side for the Helmholtz equation
// the x component of outer normal vector
func nu1       = cos(x) / sqrt(cos(x) * cos(x) + 1);

/*------------------------------------------------------
 * Triangulation
 *------------------------------------------------------*/

// Define mesh boundary
border Gamma1(t=0, 2){x=2*pi; y=t; label=1;}
border Gammab(t=0, 2*pi){x=2*pi - t; y=2; label=2;}
border Gamma2(t=0, 2){x=0; y=2 - t; label=3;}
border Gamma(t=0, 2*pi){x=t; y=sin(t); label=4;}

// show the boundary
plot(Gamma1(10) + Gammab(20) + Gamma2(10) + Gamma(40), wait=true);
mesh Th = buildmesh(Gamma1(10) + Gammab(20) + Gamma2(10) + Gamma(40), fixedborder=true);

// show the triangulation
plot(Th, wait=true);

/*------------------------------------------------------
 * FEspace
 *------------------------------------------------------*/
fespace Vh(Th, P1, periodic=[[1, y], [3, y]]);
Vh<complex> u, v;
// real part and imagnary part of the solution u
Vh realu, imagu;

solve Problem(u, v) = int2d(Th)(dx(u) * dx(v) + dy(u) * dy(v))
                      + int2d(Th)(iunit * alpha * u * dx(v) - iunit * alpha * dx(u) * v)
                      - int2d(Th)((ksquare - square(alpha)) * u * v)
```

```
                      + int1d(Th, Gamma)(lambda * u * v + iunit * nu1 * alpha * u * v)
                      + int2d(Th)(exp(-iunit * alpha * x) * f * v)
                      + on(Gammab, u = 0);

realu = real(u);
imagu = imag(u);

// show the result
plot(realu, fill=true, wait=true);
plot(imagu, fill=true, wait=true);
```