```
// example 2 : poisson-matrix.edp   [slide page 11]
// finite element solution of Poisson equation with mixed boundary condition
// with P1 element
// for tutorial by Japan SIAM, Tokyo, 11-12 Feb.2016, Atsushi Suzuki

int n = 20;
mesh Th=square(n,n);
fespace Vh(Th,P1);

Vh u,v;
real err, hh;

func f = 5.0/4.0 * pi * pi * sin(pi * x) * sin(pi * y / 2.0);
func h = (-pi)/2.0 * sin(pi * x);
func g = sin(pi * x) * sin(pi * y / 2.0);
// for error estimation
func sol = sin(pi * x) * sin(pi * y / 2.0);
func solx = pi * cos(pi * x) * sin(pi * y / 2.0);
func soly = (pi / 2.0) * sin(pi * x) * cos(pi * y / 2.0);

varf aa(u,v) = int2d(Th)( dx(u)*dx(v)+dy(u)*dy(v) )
               + on(2,3,4,u=g); // u=1 is enough to say which is Dirichlet node
varf external(u,v) = int2d(Th)( f*v ) + int1d(Th,1) (h * v)
                   + on(2,3,4,u=g); // inhomogenoues Dirichlet data are given
real tgv=1.0e+30;
matrix A = aa(Vh,Vh,tgv=tgv,solver=CG);
real[int] ff = external(0,Vh,tgv=tgv); // containing Dirichlet data with tgv

u[] = A^-1 * ff;

hh = 1.0 / real(n) * sqrt(2.0);

// int2d uses qf5pT : 5th order integration quadrature
err = int2d(Th)( (dx(u) - solx) * (dx(u) - solx) +
                 (dy(u) - soly) * (dy(u) - soly) +
                 (u - sol) * (u - sol));
err = sqrt(err);

cout << "DOF=" << u[].n << "\t h=" << hh << " err-H1=" << err << endl;
```