```
// example 13 : poisson-LinearCC-projection.edp   [slide page 61]
// finite element solution of Poisson equation with full-Neumann condition
// and solved by LinearCG with diagonal preconditioner and orthogonal projection
// for tutorial by Japan SIAM, Tokyo, 11-12 Feb.2016, Atsushi Suzuki

int n;
n = 40;
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 h1 = (-pi)/2.0 * sin(pi * x) * cos(pi * y / 2.0);
func h2 = pi * cos(pi * x) * sin(pi * y / 2.0);
func h3 = pi/2.0 * sin(pi * x) * cos(pi * y / 2.0);
func h4 = (-pi) * cos(pi * x) * sin(pi * y / 2.0);
// for error estimation
func sol = sin(pi * x) * sin(pi * y / 2.0) - 4.0 / (pi * pi);  // int sol = 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) );
varf external(u,v) = int2d(Th)( f*v )
                   + int1d(Th,1)(h1*v) + int1d(Th,2)(h2*v)
                   + int1d(Th,3)(h3*v) + int1d(Th,4)(h4*v);

matrix A = aa(Vh, Vh, solver=sparsesolver);

func real[int] opA(real[int] &pp)
{
  real[int] qq = A * pp;
  pp = qq;
  pp -= pp.sum / pp.n;      // orthogonal projection onto Im A = span[1]^\perp
  return pp;
}

func real[int] opQ(real[int] &pp)
{
  for (int i = 0; i < pp.n; i++) {
    pp(i) = pp(i) / A(i, i);
  }
  pp -= pp.sum / pp.n;      // orthogonal projection onto Im A = span[1]^\perp
  return pp;
}

real[int] ff = external(0, Vh);
ff -= ff.sum / ff.n;        // forcing RHS in image of A

u[] = 0.0;
LinearCG(opA, u[], ff, precon=opQ, nbiter=200, eps=1.0e-10,verbosity=50);

real meanu = int2d(Th)(u) / Th.area;  // mean value of u on FEM mesh
u[] -= meanu;

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;
```