Lab 07 - Manifolds (step-38+)

Numerical Solution of PDEs Using the Finite Element Method

MHPC P2.13_seed

Martin Kronbichler kronbichler@lnm.mw.tum.de and **Luca Heltai** luca.heltai@sissa.it

---

1. The topic of this lab session is a modified version of step-38 made available for you as **lab-7** https://www.dealii.org/8.4.0/doxygen/deal.II/step_38.html
2. Run the program and check the graphical and text output.
3. Modify the code to compute L2 and H1 errors and generate convergence plots for different mapping degrees and different Finite Element orders.
4. Implement `compute_area` that computes the area of our domain Ω_h by integrating $\int_\Omega 1\, dx$. The code is similar to the matrix assembly, except that we are just computing

   $$\int_\Omega 1\, dx \approx \sum_T \sum_q w_q$$

   Check that the area is converging to the correct value and that a higher order mapping helps.
5. Switch the mesh to a torus with R=1.0 and r=0.6 and check that the area is converging correctly. See `GridGenerator::torus()` and the class `TorusBoundary` with constructor

   ```
   TorusBoundary<dim,spacedim>::TorusBoundary(R, r)
   ```

   You need to add the following include at the top of the file:

   ```
   #include <deal.II/grid/tria_boundary_lib.h>
   ```

   Finally: open up the solutions and check that the torus is refined correctly.
6. Without a boundary, the problem is not uniquely solvable, so we add the constraint $\int_\Omega u = 0$. We need to make sure our analytic solution and our discrete solution satisfy this. Compute the mean value using

   ```
   template <int dim, typename VectorType, int spacedim>
   double VectorTools::compute_mean_value (const DoFHandler<dim,spacedim> &dof,
                                           const Quadrature<dim>          &quadrature,
                                           const VectorType               &v,
                                           const unsigned int             component);
   ```

   at the end of `solve()` (`component` is 0, `v` is `solution`) and then subtract it using

   ```
   solution.add(-mean);
   ```

   finally verify that the mean is now close to zero by computing it again.

   Note: how come the linear solver converges even though the linear system is singular?
7. Open `labs/misc/surfacelaplacian_torus.mw` in maple and find the right-hand side f that belongs to the solution

   $$u = \sin(3\phi)\cos(3\theta + \phi)$$

   that you can plug into the code (`Solution<3>::value`) as:

   ```
   double x = p(0);
   double y = p(2);
   double z = p(1);
   ```

```
double r = 0.6;
double R = 1.0;
double phi = atan2(y,x);
double theta = asin(z/r);
if (x*x+y*y < R*R)
  theta = ((z>0)?1.0:-1.0)*numbers::PI - theta;
return sin(3*phi)*cos(3*theta+phi);
```

Note that y and z are swapped between maple and the code.

8. Now make sure L2 norms are converging and you get optimal rates when using a higher order mapping.
9. Bonus: implement the gradient from the spreadsheet and check H1 errors.