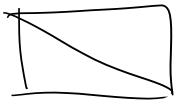
- fem(n, basis, polydeg, plotsln)
  - ↳ Solves $-\Delta u = f$ in $\Omega = (0,1)^2$
    $u = 0$ on $\partial\Omega$

  with $f = 2\pi^2 \sin(\pi x)\sin(\pi y)$

  [Note: known analytical solution: $u = \sin(\pi x)\sin(\pi y)$]

  on a $n \times n$ mesh of squares or triangles

  (triangles are squares split: )

  using one of following basis functions with specified polynomial degree:
  - Lagrange 2-simplex ('tri') $p \geq 1$
  - Lagrange 2-rectangle ('quad') $p \geq 1$
  - Reduced Lagrange 2-simplex ('reduced_tri') $p = 3$
  - Reduced Lagrange 2-rectangle ('reduced_quad') $p = 2, 3$
  - Crouzeix-Raviart ('cr') $p = 1$
  - Rotated bilinear ('rotated_bilinear') $p = 1$

- Returns: u - Vector of values for solution corresponding to degrees of freedom (length = N)

  x - Nx2 matrix - each row contains point of evaluation for each DoF

  mesh - Structure defining the mesh:
  - vertices - Vx2 matrix of vertex coordinates
  - elements - $E \times V_E$ - index of vertex for each element (E = # elements)
  - dof_mapping - ExM - Mapping local DoF to global DoF

l2err, h1err – Error between numerical & analytical solution in L^2-norm & H^1-seminorm.

Run basis examples:
fem(8,'tri',1,1), fem(8,'tri',3,1)
fem(8,'quad',3,1), fem(8,'quad',3,1)
fem(8,'reduced_tri',3,1), fem(8,'reduced_quad',3,1)

Convergence

As know u can compute $\|u-u_h\|_{0,\Omega}$ & $|u-u_h|_{1,\Omega}$

We have that

$$\underbrace{\|u-u_h\|_{1,\Omega}}_{\varepsilon_N} \leq Ch^k|u|_{k,\Omega} = \bar{C}h^k$$

Asymptotically we have that

$$\varepsilon_N \approx \bar{C}h^k$$
$$\log_{10}\varepsilon_N \approx \log_{10}\bar{C} + k\log_{10}h$$
$$\log_{10}\varepsilon_N \approx \tilde{C} + k\log_{10}h$$

So if compute numerical solution for different values of h & plot $\log_{10}\varepsilon_N$ against $\log_{10}h$, then perform linear regression (least squares) we get an estimate for k:

fem_convergence(basis,p,M)

computes for $h = \frac{1}{2}, \frac{1}{4}, \ldots \frac{1}{2^M}$ & estimates k

Basis	p	k(H1)	k(L2)
tri	1, 2, 3, ...	p	p+1
quad	1, 2, 3, ...	p	p+1
reduced_tri	3	p-1	p
reduced_quad	2, 3	p	p+1
cr	1	p	p+1
rotated-bilinear	1	p	p+1

Steps of method

- Create mesh, FE space & quadrature:
  - x - Matrix of coordinates for global DoF (one coordinate per row)
  - bdry - Vector of 1 or 0 denoting if DoF on boundary
  - mesh - As above
  - quadrature - structure containing basis & quadrature:
    - points - Quadrature points on reference element (Lx2)
    - weights - Quadrature weights (L)
    - basis - Matrix of basis functions (column) evaluated at each quadrature point (row)
    - deriv_basis_x, deriv_basis_y
      - Matrix of derivatives of basis functions in x, y at each quadrature point

- Create matrix(A) right hand side vector(b)
  (matrix is sparse so use sparse function which takes three vectors containing non-zero entries (row index, column index, value).
  - Loop over every element
    - Creat affine map $F(x): Bx+c$ for element
    - Compute $J^{-1} = B^{-1}$ & $\det(B)$
    - Loop over test function(φ_i) dofs:
      - Loop over trial functions (φ_j) dofs
        - Compute $\sum_{\ell=1}^{L} \omega_\ell |\det B| B^{-T} \nabla\varphi_i(x_\ell) B^{-T} \nabla\varphi_j(x_\ell)$
      - Compute $\sum_{\ell=1}^{L} \omega_\ell |\det B| f(b_\ell) \varphi_i(x_\ell)$
  - Delete rows from A & b and columns from A corresponding to boundary Dofs.
- Solve linear system $Au=b$
- Compute L^2 & H^1 errors $\left(\begin{array}{l} u = \sin(\pi x)\sin(\pi y) \\ \nabla u = \pi \begin{pmatrix} \cos(\pi x)\sin(\pi y) \\ \sin(\pi x)\cos(\pi y) \end{pmatrix} \end{array} \right)$
  - Set l2error = 0, h1error = 0
  - Loop over every element
    - Create affine map
    - Compute $J^{-1} = B^{-1}$ & $\det(B)$
    - Compute

$$l2error \mathrel{+}= \sum_{\ell=1}^{L} \omega_\ell |\det B| \left| u(x_\ell) - \sum \varphi_i(b_\ell) u_i \right|^2$$

$$h1error \mathrel{+}= \sum_{\ell=1}^{L} \omega_\ell |\det B| \left| \nabla u(x_\ell) - \sum \nabla\varphi_i(b_\ell) u_i \right|^2$$

Construction of mesh, basis & FE space

Utility function for each FE basis type

{name}_fe_space which:

- Creates mesh (construct_tri_mesh or construct_quad_mesh)
- Setup list of nodes for global DOFs & if bdry
- Create DOF mapping (local to global)
- Create quadrature (quadrature_tri or quadrature_rect)
- Evaluate basis functions at quadrature points
  (basis_{type3})
- Evaluate gradient of basis functions at quadrature points:
  (grad_basis_{types})

Discuss {grad_}basis_lagrange_tri & {grad_}basis_lagrange_quad