Objectives.

1. Construct a 2D FEM approximation
2. Detail assembly using sparse matrices

TODO.

1. Read chapter 6 (Finite element Assembly) in the FEniCS book.
2. Install fenics and run a simple demo.
3. alternative: Use a VM with fenics.

Consider the PDE

\[-\nabla (\kappa(x) \nabla u) = f \quad \text{on } \Omega = [-1,1]^2\]
\[u = g \quad \text{on } \partial \Omega\]

We use

\[g(x,y) = \begin{cases} 20(1-y^2) & \text{for } x = -1 \\ 0 & \text{else} \end{cases}\]

and consider three tests:

<table>
<thead>
<tr>
<th>test</th>
<th>(\kappa(x,y))</th>
<th>(f(x,y))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0.1</td>
<td>(f(x,y) = 0)</td>
</tr>
<tr>
<td>(b)</td>
<td>0.5</td>
<td>(\begin{cases} 25 &amp; \text{for } \sqrt{x^2+y^2} \leq 0.25 \ 0 &amp; \text{else} \end{cases})</td>
</tr>
<tr>
<td>(c)</td>
<td>(\begin{cases} 25 &amp; \text{for } \sqrt{x^2+y^2} \leq 0.25 \ 0.1 &amp; \text{else} \end{cases})</td>
<td>(\begin{cases} 25 &amp; \text{for } \sqrt{x^2+y^2} \leq 0.25 \ 0 &amp; \text{else} \end{cases})</td>
</tr>
</tbody>
</table>

The finite element method in 12 steps:

**Step 0:** Read in a mesh. The mesh provided is an unstructured mesh of the square \([-1,1]^2 \subset \mathbb{R}^2\]. The mesh conforms to a disc of radius 0.25 centered at the origin as shown here:

How do you get a vertex-vertex or element-element neighbor list?

At this point, the list of vertices \((V, \text{nv} \times 2)\) and elements \((E, \text{ne} \times 3)\) are available. Now, for each element we need to do the following

**Step 1:** Define the coordinates of the current element.

**Step 2:** Use the coordinates to define the Jacobian \((J = J)\), the inverse of the transposed Jacobian \((\text{inv} J = J^{-T})\), and the determinant of the Jacobian \((\text{det} J = |J|)\).
Step 3: Define the gradient of the basis functions ($dbasis = \nabla \lambda_r$) on the reference triangle $\tau$, for each basis function.

> What is $\nabla \lambda_r$ for each $r$?

Step 4: Construct the gradient of the basis function on the element ($dphi = \nabla \phi_r = J^{-T} \lambda_R$) for each basis function.

Step 5: Calculate the bilinear form on the element:

$$A_{elem} = \int_E \kappa(x) \nabla \phi_r \cdot \nabla \phi_s \, dx$$

$$= \int_E \kappa(x) (J^{-T} \nabla \lambda_r)^T (J^{-T} \nabla \lambda_s) \, dx$$

$$= (J^{-T} \nabla \lambda_r)^T (J^{-T} \nabla \lambda_s) |J| \int_E \kappa(T(\alpha)) \, d\alpha$$

Step 6: And the linear functional (right-hand side) on each element:

$$b_{elem} = \int_E f(x) \phi_r \, dx$$

$$= |J| \int_S f(T(\alpha)) \lambda_r(\alpha) \, d\alpha$$

Step 7: Add the contributions to to the global matrix ($A$).

Step 8: Construct the COO (non-duplicate form) of the sparse matrix.

> Why construct COO then convert to CSR then convert to COO?

Step 9: Apply boundary conditions: define a function $u_0$ that is equal to $g$ on the $x = 0$ boundary and zero elsewhere by creating two boolean vectors: one that is true for each vertex on the $x = 0$ boundary ($gflag$) and the other that is true for each vertex on the whole boundary ($Dflag$). Then set $u_0$ appropriately.

Next, we need to set

$$b \leftarrow b - Au_0,$$

which is just a matrix-vector product. Finally, we need to set:

$A_{ij} = 0$

$A_{ii} = 1$

$b_i = 0,$

for each boundary node $i$.

Step 10: The matrix problem $Au = b$ is now solved and the solution is formed:

$$u \leftarrow u + u_0$$

Step 11: The solution is plotted.