The Problem

We would like to model the flow of the concentration of a substance through something which has obstacles in its way. The flow obeys the advection-diffusion equation.

The Domain

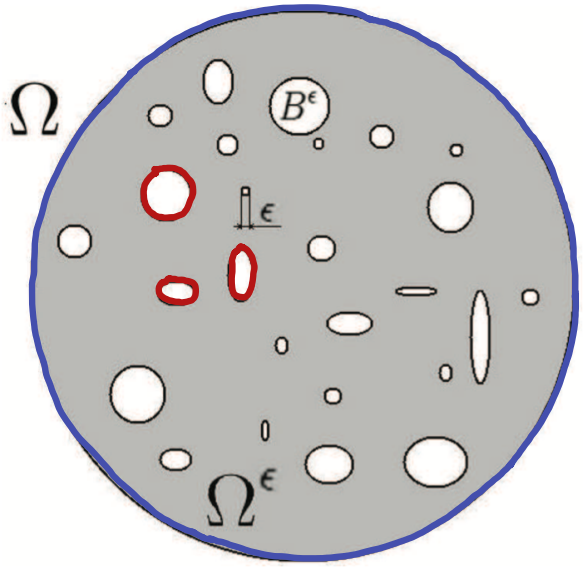


Figure 1: Perforated domain $\Omega \in \mathcal{R}^d$ within which a set B^ϵ of perforations is included. The perforated domain with voids left by perforations is denoted $\Omega^\epsilon = \Omega \setminus B^\epsilon$, where ϵ denotes the minimum width of perforations.

The Governing Equations

ADE

$$-\nabla \cdot (D \nabla v) + \underline{w} \cdot \nabla v = f$$

v = scalar field variable (eg. concentration, Temp)

D = diffusion coefficient

$\underline{w}$ = velocity vector

f = source term

Boundary conditions

$v = 0$ on $\partial B^\epsilon \cap \partial \Omega^\epsilon$,

$v = g$ on $\partial \Omega \cap \partial \Omega^\epsilon$

The Problem

- Second order PDE
- Implies that u is both continuous and very <u>smooth</u>
- As we are simulating using an approximation, dont want it to be smooth everywhere
- Need to convert strong form (PDE) into weak form (integrals)

Via Voodoo witchcraft, and Greens Theorem, the weak form is

$$\int_\Omega \nabla u \cdot \nabla v + \int_\Omega (\bar{w} \cdot u) v = \int_\Omega f v.$$

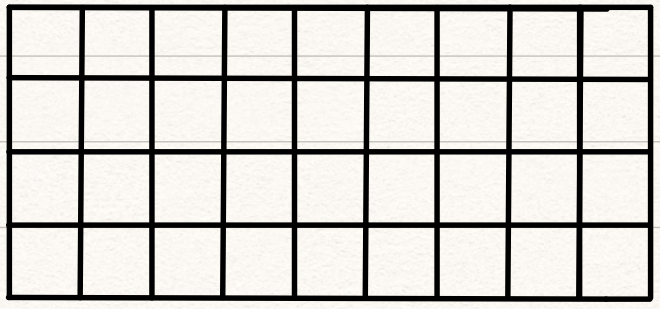
v = test function.

Because its in integrals, function is globally continuous but does not have to be globally smooth. Eg.

But this is still in real life, and we need to apply this in a simulation somehow. So now we need to talk about the domain in a way a computer understands

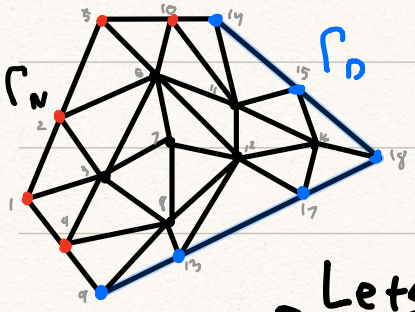
Discretizing the Domain

- Need to split up the domain into grids
- The computer can then calculate the <u>numerical</u> value at each grid
- The entire set of numerical values at each part makes up the solution



Entire Domain split into grids

Note: Grids don't have to be regular, but for my drawing sake they are. Example of non uniform grid is this, but the grid in the code is rectangular. Each box or triangle is called an <u>element</u>, hence finite elements



↳ Lets use this for now

Now lets think about <u>linear</u> functions that occupy a triangle. A linear 2D function, like $f(x,y) = ax + by + c$

A <u>set</u> of those <u>linear</u> functions is a functional space, so here

$$V_h = \{ a_0 + a_1 x + a_2 y, \; a \in \mathbb{R} \}$$

Anytime you see this V_h, think `Its a linear function'

From this, we define the <u>Basis function</u> used as our test function and other stuff too

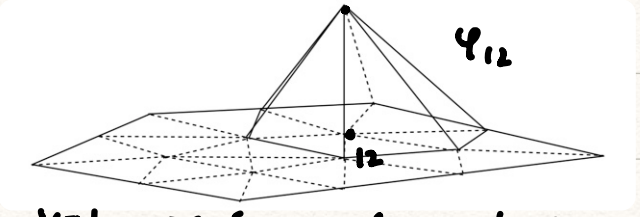
Basis Functions

The vertices of each element is called a node. The nodal Basis function, belonging in the V_h set, is

$$\varphi_i(\underline{P}_j) = \delta_{ij},$$

where $\underline{P}_j$ is the location of the jth vertex.

In english, this function has a value of 1 at that node, and 0 at every other node. In graphical terms, it looks like a tent.



At node 12, value of φ_{12} is 1, and 0 at every other node

Weak form, discretized weak form, Matrix form

weak: $$\int_\Omega \nabla u \cdot \nabla v + \int_\Omega (\bar{w} \cdot u) v = \int_\Omega f v.$$

Disc.W.: $$\int_\Omega \nabla u_h \cdot \nabla v_h + \int_\Omega (\bar{w} \cdot u_h) v_h = \int_\Omega f v_h,$$

where $u_h, v_h \in V_h$, and defined for each element.

Substitute stuff (Convince yourself this is true)

variable $$u_h = \sum_{j \in Node} u_j \varphi_j, \quad \nabla u_h = \sum_{j \in Node} u_j \nabla \varphi_j,$$

test functions $v_h = \varphi_i$, $\nabla v_h = \nabla \varphi_i$, and rearrange

$$\boxed{\sum_{j \in Node} \left(\int_\Omega D \nabla \varphi_j \cdot \nabla \varphi_i + \int_\Omega (w \cdot \nabla \varphi_j) \varphi_i \right) u_j = \int_\Omega f \varphi_i}$$

Matrix Form

$$\underline{\underline{A}}_{ij}\ \underline{v}_j = \underline{f}\ ,\ \Rightarrow \text{computer solvable.}$$

$$\underline{\underline{A}}_{ij} = D_{ij} + W_{ij}$$

$$= \int_\Omega D \nabla \varphi_j \cdot \nabla \varphi_i + \int_\Omega (w \cdot \nabla \varphi_j)\, \varphi_i$$

As $\underline{\underline{A}}_{ij}$ is a matrix with numbers, and f is (?) $\underline{v}_j$ can be easily solved by a computer. So before we can code it we need to find a way to evaluate D_{ij} and W_{ij}

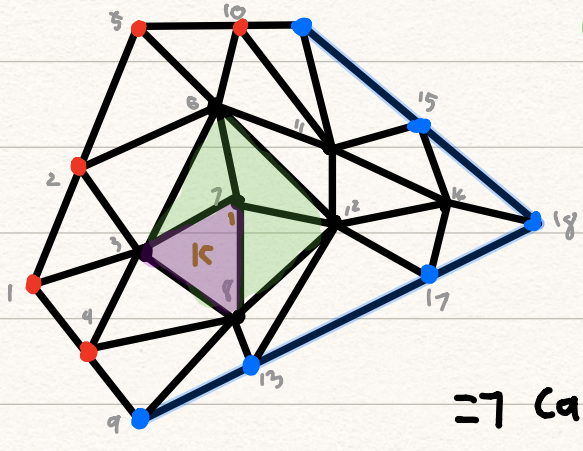
Assembling the matrix

Lets focus on D_{ij}, as W_{ij} is also similar.

we can Decompose the integral over the entire domain, $\int_\Omega$, as the sum of integrals over the different elements k

$$D_{ij} = \int_\Omega D \nabla \varphi_j \cdot \nabla \varphi_i = \sum_k \int_k D \nabla \varphi_j \cdot \nabla \varphi_i = \sum_k D_{ij}^k$$

Each element has a local Nodal basis funcion.



φ_7, Global basis, Picture the camping tent

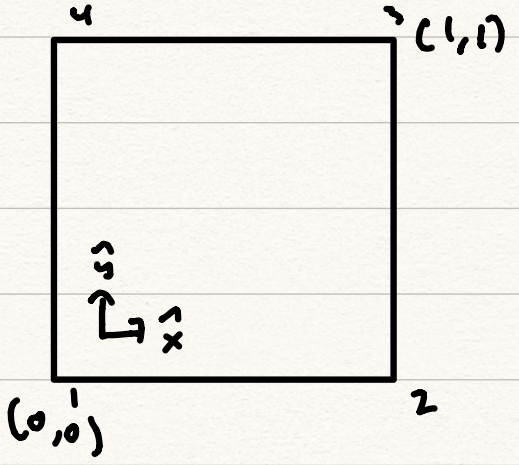
N_1^k, local basis function

Local node 1 of element K = Global node 7

$\Rightarrow$ Calculate $\int_k D \nabla N_\beta^k \cdot \nabla N_\alpha^k = D_{ij}^k$

Reference Element

To calculate the integral, we use a reference element



$$\hat{N}_1 = (1-\hat{x})(1-\hat{y})$$
$$\hat{N}_2 = (\hat{x})(1-\hat{y})$$
$$\hat{N}_3 = (\hat{x})(\hat{y})$$
$$\hat{N}_4 = (1-\hat{x})(\hat{y})$$

Because the elements are rectangular, we don't need to do a lot of fancy stuff and just do a simple conversion of $(x,y) \to (\hat{x},\hat{y})$ and calculate $\hat{N}$ and $\nabla\hat{N}$. So now we have numerical results for $\hat{N}$ and $\nabla\hat{N}$ at any particular point.

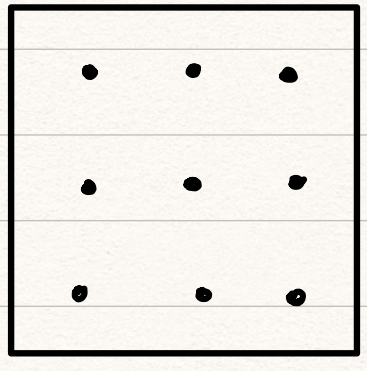
Gaussian Quadrature

How to calculate the integral bit? use Gaussian quad.

$$\int_{-1}^{1} f(x) = \sum_i w_i f(x_i)$$

for 3 points, $x_i = 0, \pm\sqrt{\frac{3}{5}}$, and $w_i = \frac{8}{9}, \frac{5}{9}$.

Calculating the local Matrix elements



Calculate value of ∇N_m and ∇N_n at all quadrature points (x_i, y_j)

And add it all up!