$g = -1$

$g=0$ $\quad g=1$ $\quad g=2$ $\quad g=-1$

the Euler formula applies to a manifold surface

↑
one

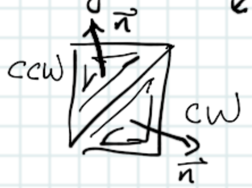
$$v_0 - e_0 + f_0 = 2 - 2g_0$$
$$v_1 - e_1 + f_1 = 2 - 2g_1$$
$$\overline{(v_0+v_1) - (e_0+e_1) + (f_0+f_1) = 4 - 2(g_0+g_1)}$$
$$v - e + f \neq 2 - 2(g_0+g_1)$$

Connected Component Analysis
- → our halfedges, other halves, &c. <u>are</u> a graph
- → so to determine the individual surfaces, we use CCA

```
while (not all vertices processed)
    put a vertex on the queue
    while (queue not empty)
        pop vertex
        add neighbours to queue unless already processed
        mark vertex as processed
then repeat to find the next component
→ BFS
```

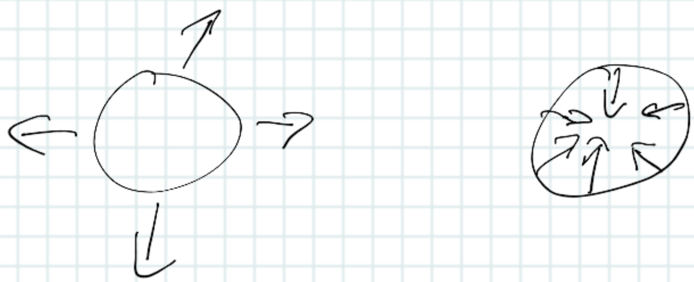
is this manifold?
manifold assumes triangles
we assume CCW triangles

legal?



one normal points out
one normal points in } we <u>don't</u> want this

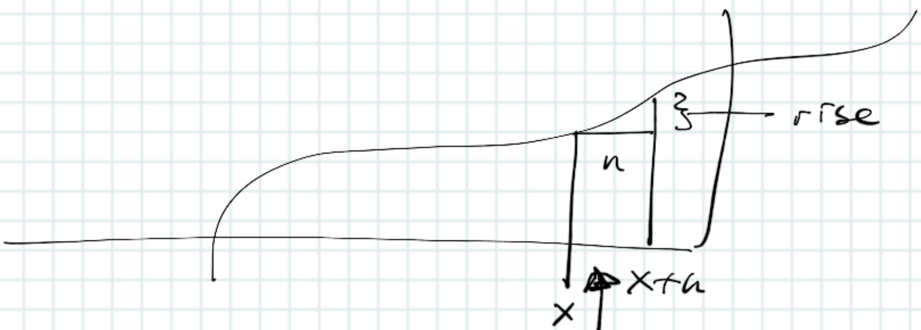
→ in practice, if you detect two edges in the same direction, it's a bad surface



worse - this does occur in downloaded data
why? triangle strips alternate CCW & CW
& a programmer who doesn't know this will output alternating triangles

Discrete Differential Operators

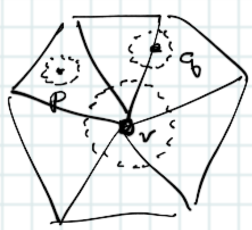
A derivative is $f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{(x+h) - x} = \frac{\text{rise}}{\text{run}}$



calculus is based on small intervals

as you generalise to 2D & higher, your interval becomes one of the following:

preferred - simple, uniform and tidy

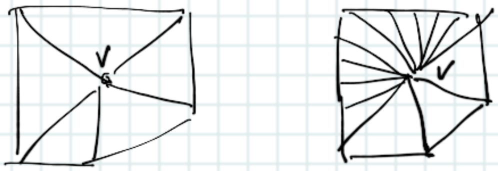


Three possibilities
- point in face p
- point on edge q
- point at vertex v

p's neighbourhood is well defined and smooth, so we won't have problems

q's neighbourhood is 50/50 - half on one △
half on another

v's neighbourhood is the 1-ring



← same surface

suppose my normal at v is the average of the △ normals
the left gives ~~the top left~~ each triangles weight of $\frac{1}{5}$ each
the right gives each △ a weight of $\frac{1}{13}$ each
→ a simple average fails if the triangulation is uneven (poor quality)

→ we substitute a weighted average:

$$\vec{n}_v = \sum_{f \in N_1(v)} w_f \, \vec{n}_f$$

all faces in the 1-ring — weight of f — normal for f

a) $w_f = \frac{1}{\deg(v)}$ — equal weighting

b) $w_f = \frac{\text{area}(f)}{\sum_{g \in N_1(v)} \text{area}(f)}$ ← based on the cross-product of the edge vectors of the triangle

c) $w_f = \frac{\theta_f}{\sum_{f \in N_1(v)} \theta_f}$ ← angle at v on the face f

this is not 2π

← area weighting will give these two all of the weight

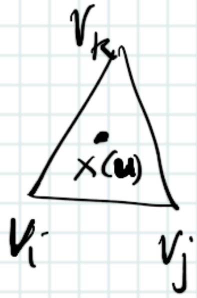
and there are many more options

How to find gradient at a vertex
- a weighted sum of the gradients in the 1-ring
- this simplifies if we know how to find gradient of a function interpolated across a $\triangle$ - i.e. with barycentric interpolation.

To find ∇f at a point $\mathbf{u}$ in a triangle

$$f(\mathbf{u}) = B_i f_i + B_j f_j + B_k f_k$$

f_i - function value at vertex v_i.



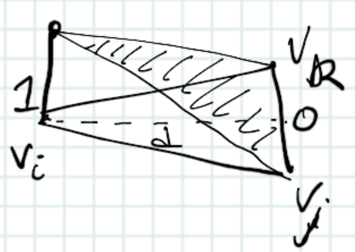
α β γ

constants

$$\nabla f(\mathbf{u}) = \nabla B_i(\mathbf{u}) f_i + \nabla B_j(\mathbf{u}) f_j + \nabla B_k(\mathbf{u}) f_k$$

- a weighted sum
- so we need to work out ∇B_i

what does B_i do?



$$B_i(u) = \begin{cases} 0 & \text{on } v_j v_k \\ 1 & \text{at } v_i \end{cases}$$

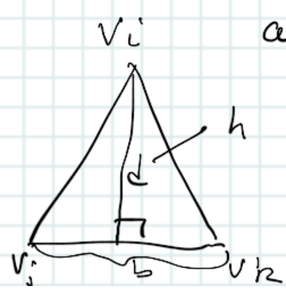
linear in between

→ a flat slope across the triangle i.e. constant

d - distance from v_i to $v_j v_k$

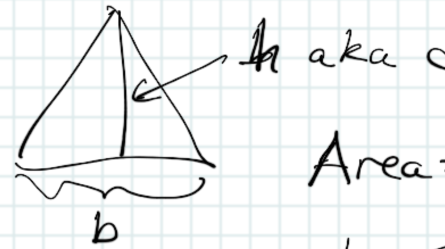
So the slope is $\frac{1}{d}$

and the gradient is $\perp$ to $v_j v_k$ → ie. in the normal direction, so $\nabla B_i(\mathbf{u}) = \frac{1}{d} \frac{\vec{n}_{jk}}{\|\vec{n}_{jk}\|}$



$b = \|\vec{n}_{jk}\|$

$$\nabla B_i(\mathbf{u}) = \frac{1}{d} \; \frac{\vec{n}_{jk}}{\|\vec{n}_{jk}\|}$$



$$= \frac{\cancel{b}}{2\text{Area}} \cdot \frac{\vec{n}_{jk}}{\cancel{b}}$$

Area $= \frac{1}{2} bh$

$d = \frac{2\text{Area}}{b}$

$\frac{1}{d} = \frac{b}{2\text{Area}}$

$$= \frac{1}{2 \cdot \text{Area}} \cdot \vec{n}_{jk}$$

$$= \frac{1}{2\text{Area}} (v_k - v_j)^{\perp}$$

these terms are **all** constant

$\nabla B_i(u)$ is constant on the triangle, so we just compute it once

$$\nabla f(\mathbf{u}) = \frac{(v_k - v_j)^{\perp}}{2A_T} f_i + \frac{(v_i - v_k)^{\perp}}{2A_T} f_j + \frac{(v_j - v_i)^{\perp}}{2A_T} f_k$$

everywhere in T

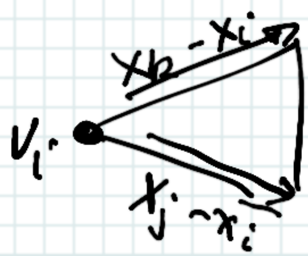
I can now sum all of these in the 1-ring & end up with terms involving the edge vectors.

e.g. Let $v_i = \mathbf{X}(\mathbf{u}_i)$

position function, mapping, vertex position in Ω

$v_i = \mathbf{X}_i$

$$\nabla f(u) = \frac{(f_j - f_i)(\mathbf{X}_i - \mathbf{X}_k)^{\perp}}{2A_T} + \frac{(f_k - f_i)(\mathbf{X}_j - \mathbf{X}_i)}{2A_T}$$



$$\Delta f(v_i) = \underbrace{\frac{1}{|N_1(v_i)|}}_{\substack{\text{size of the}\\ \text{1-ring - i.e.}\\ \text{degree of}\\ \text{the vertex}}} \cdot \underbrace{\sum_{v_j \in N_1(v_i)}}_{\substack{\text{sum over}\\ \text{all incident}\\ \text{edges}}} \underbrace{(f_j - f_i)}_{\substack{\text{difference in values}\\ \text{between the vertices}}}$$

So to approximate the LBO (Laplace-Beltrami Operator) on a mesh, we take the weighted sum of the differences between the vertex' value and it's neighbours and we call this the Discrete LBO or DLBO for short