Mar 21

Review: for a function $f: M \to \mathbb{R}$, we saw in last class

Critical point: all partial derivatives are 0 $\frac{\partial f}{\partial x_i} = 0 \ \forall i$

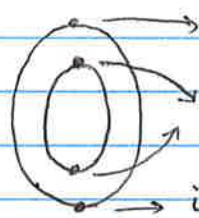
Non-degenerate critical point: $\det(H(x)) \neq 0$
i.e. the Hessian is non-singular (2nd derivative non zero in $\mathbb{R}$)

Morse Lemma: Given $f: M \to \mathbb{R}$. if u is non-degenerate critical point then there exists local co-ordinate chart with $u = (0, 0, \ldots, 0)$ s.t.

$$f(x) = f(u) - x_1^2 - x_2^2 - \cdots - x_q^2 + x_{q+1}^2 + \cdots + x_d^2$$

for every point $x = (x_1, x_2, \ldots, x_d)$ in small neighborhood of u
[follows from Taylor series expansion about u. if $H(x)$ is non-singular then 2nd order terms dominate]

Index of critical point: # of -ve coefficients in the quadratic polynomial.
for d dimensions $\longrightarrow$ $d+1$ possible index values.



local minima : index 0
local maxima : index d
saddles : $0 < \text{index} < d$

Morse functions : $f: M \to \mathbb{R}$ such that
(a) all critical points are non-degenerate (b) critical values are distinct.

Morse Inequality : M : d-dimensional manifold, $f: M \to \mathbb{R}$
and c_q = # critical points with index q
(a) Weak version : $c_q \geq \beta_q(M)$ for all q.
(b) Strong version : $\sum_{q=0}^{j} (-1)^{j-q} c_q \geq \sum_{q=0}^{j} (-1)^{j-q} \beta_q(M)$

when $j = d$, equality holds.

Piecewise Linear (PL) functions

In most cases, we don't know f. We can only sample function values at discrete points
eg. elevation of terrain : measured at grid points
for all intermediate points, function value is approximated using interpolation : piecewise linear functions.

Let K be a simplicial complex with real values specified at all vertices. (Assume distinct values)

$$f : |K| \rightarrow \mathbb{R} \quad , \quad f(x) = \sum_{i=1}^{n} b_i(x) f(u_i)$$

where $b_i(x)$ are barycentric coordinates of x w.r.t. vertices.
i.e. function value at x is linear combination of function values at vertices (known) weighted by barycentric coordinate.

→ Order vertices by increasing value i.e.

$$f(u_1) < f(u_2) < \dots < f(u_n)$$

K_i : subcomplex formed by first i vertices in ordering.

def: Star of vertex u_i : $\mathrm{St}\ u_i$: set of co-faces of u_i in K
⇒ add missing faces to get closed star $\overline{\mathrm{St}}\ u_i$

K $\qquad$ $\mathrm{St}\ u$ $\qquad$ $\overline{\mathrm{St}}\ u$ $\qquad$ $\mathrm{Lk}\ u$

def: Link of vertex u_i : $\mathrm{Lk}\ u_i$ set of simplices that are in the closed star of u_i but not in star of u_i

$$\mathrm{Lk}\ u_i = \{ \sigma \in \overline{\mathrm{St}}\ u_i \mid \sigma \notin \mathrm{St}\ u_i \}$$

def: Lower star of vertex u_i : $St_- u_i$ set of co-faces of u_i such that $f(u_i)$ is the maxima

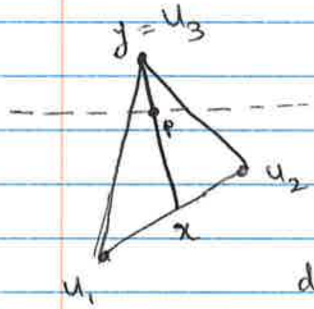
$$St_- u_i = \{ \sigma \in St\ u_i \mid x \in \sigma \Rightarrow f(x) \le f(u_i) \}$$

→ if vertices are ordered by function values then K_i is the union of lower stars of first i vertices.

the filtration $\phi = K_0 < K_1 < \dots < K_n = K$ is called lower star filtration

Sub-level set $|K|_a = f^{-1}(-\infty, a] \simeq K_i$ for

$f(u_i) \le a < f(u_{i+1})$



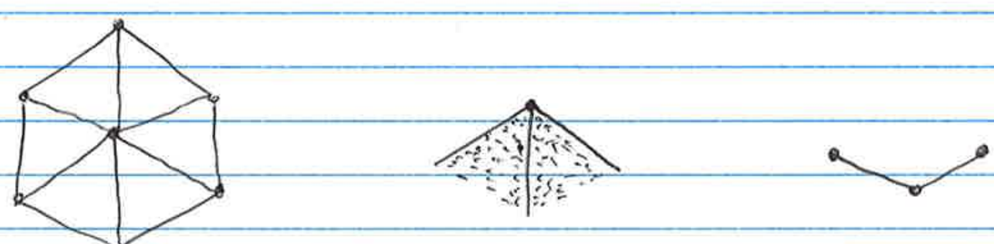
$p = (1-t)y + tx$

t going from 0 to 1 gives a continuous deformation of level set to $(u_1 u_2)$ from u_3 showing that $|K|_a \simeq K_i$

→ Going from K_{i-1} to K_i is same as gluing closed lower star of u_i to K_{i-1} along the lower link

def: lower link of vertex u_i

$$Lk_- u_i = \{ \sigma \in Lk\ u_i \mid x \in \sigma \Rightarrow f(x) \le f(u_i) \}$$

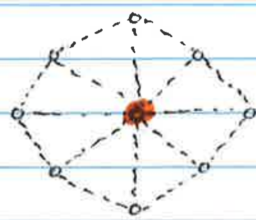


$St_- u$ $Lk_- u_i$

\# We can classify vertices using reduced Betti numbers of the lower link.

$\boxed{\tilde{\beta}_0 = \tilde{\beta}_1 = 0}$

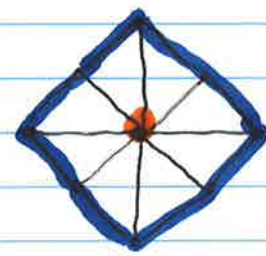
Regular Vertex



if u is $\boxed{\text{local minima}}$

LK_u is empty

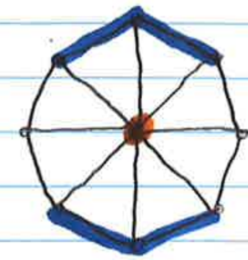
$\boxed{\tilde{\beta}_{-1} = 1}$



if u is $\boxed{\text{local maxima}}$

LK_u forms a loop

$\boxed{\tilde{\beta}_1 = 1}$



if u is $\boxed{\text{simple saddle}}$

LK_u has multiple C.C.

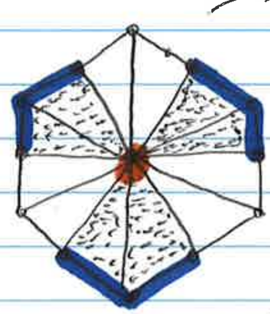
$\boxed{\tilde{\beta}_0 = 1}$

Def: Simple PL critical vertex of index q: if the lower link of vertex u LK_u has reduced homology of (q-1)-sphere then it is called a simple PL vertex of index q.

[$\beta_{q-1} = 1$ is the only non-zero reduced Betti number of LK_u]

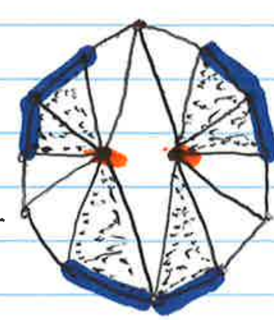
Def: PL Morse function: $f : |K| \to \mathbb{R}$ is PL Morse function if (a) Each vertex of K is either a regular vertex or a simple PL critical vertex

(b) function values at vertices are distinct.



$\to \tilde{\beta}_0 = 2 \therefore$ 2-fold saddle

Can be "unfolded" into 2 simple saddles $\to$



Each vertex is a simple PL critical vertex $\leftarrow$

$\because \tilde{\beta}_0 = 1$ for both.

PL version of Monkey Saddle

$\tilde{\beta}_0 = 2 \; (\neq 1) \therefore$ Not a simple PL Critical Vertex

PL Morse Inequalities: Let K be triangulation of a d-manifold and $f: |K| \to \mathbb{R}$ be a PL Morse function. Let C_q be the number of PL critical vertices of index q.

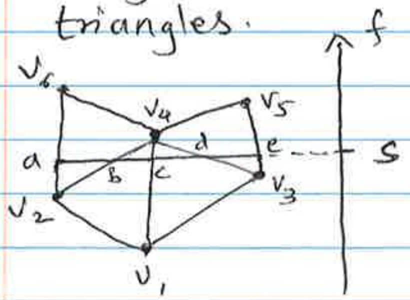
(a) Weak version: $C_q \geq \beta_q(K)$ for all q

(b) Strong version: $\sum_{q=0}^{j} (-1)^{j-q} C_q \leq \sum_{q=0}^{j} (-1)^{j-q} \beta_q(K) \quad \forall j$

Constructing Reeb graph: Consider $f: M \to \mathbb{R}$ where M is a triangulated 2-manifold and f is PL Morse.

⇒ Sort vertices s.t. $f(v_i) < f(v_{i+1})$ $1 \leq i \leq n$

→ Each contour in the triangulation is a sequence of line segments. Each line segment falls on a triangle ⇒ we can represent contours as list of triangles.



$f^{-1}(s)$ is made up of line segments ab, bc, cd, de. Each segment falls on one triangle in the triangulation: $\{v_2 v_6 v_4,\ v_2 v_4 v_1,\ v_1 v_3 v_4,\ v_3 v_4 v_5\}$

→ also note that the list of triangles remains the same for all $f(v_3) < s < f(v_4)$

- Important to note: contours represented by list of triangles. The list only changes when we pass through a vertex.

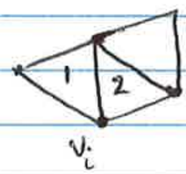
⇒ sub-level set filtration can be achieved by only looking at how the triangle list changes at vertices.

Depending on the vertex type we have following:

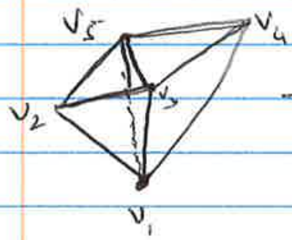
Case 1: vertex v_i is a minimum: Begins an arc in Reeb graph

→ Add a degree-1 node to graph.

→ The arc is associated with cyclic list of triangles initialize it with triangles in [star of v_i]



→ initialize cyclic list of triangles with triangles 1, 2

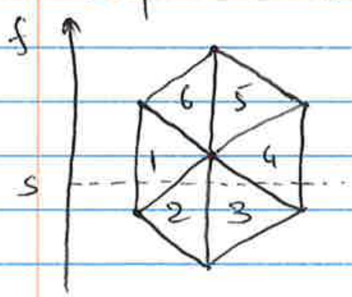


→ The list will be initialized with the four vertical faces: $\{v_1v_2v_3,\ v_1v_3v_4,\ v_1v_4v_5,\ v_1v_5v_2\}$

→ list is cyclic.

Case 2: v_i is regular vertex. We already have one or more cyclic lists representing contours of the triangulation below $f(v_i)$.

→ Two or more triangles in St v_i form a contiguous sequence in one of the existing cyclic lists



Suppose v_i is the vertex at center.

We have already passed through contour corresponding to s before reaching v_i so the cyclic list of triangles we already have is $\{1, 2, 3, 4\}$

As we pass through v_i the contours will no longer fall on triangles in lower star of v_i $\{2, 3\}$

⇒ Replace triangles in lower star of v_i by triangles in upper star of v_i i.e. $\{1, 2, 3, 4\} \rightarrow \{1, 6, 5, 4\}$

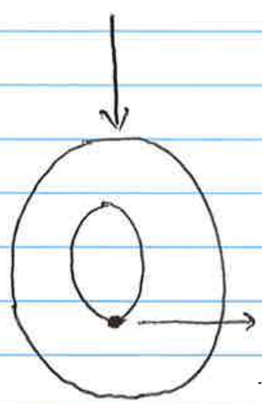
Case 3: v_i is a saddle: its lower star has two or more disconnected parts ∴ The triangles in star of v_i form two or more distinct sequences in the existing cyclic lists of triangles. Same as case 2, we keep the first and last of the contiguous sequence and replace the other triangles (these are in lower star of v_i) by triangles in upper star of v_i

→ Depending on type of saddle, this will result in either splitting a list, merging two or more lists or simply re-ordering the existing list.

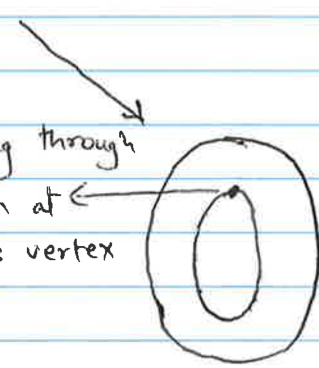
first two cases ⇒ degree-3 nodes in Reeb graph
third case ⇒ degree-2 node.

split list in two

~~split~~ merge two lists in one



passing through triangulation at this saddle vertex



when passing through triangulation at ← this vertex

<u>Case 4</u>: v_i is a maximum : Remove the cyclic list in its star (since v_i is maximum, one of the cyclic lists is entirely made up of triangles in star of v_i)

→ in the Reeb graph → add new degree-1 node.

⇒ For implementation, we need data structure with following operations:

(a) CUT : Cut cyclic list open
(b) GLUE : Attach two ends of a list
} for merging, splitting lists.

(c) DROP : Drop triangle from open list
(d) APPEND : Add new triangle to open list
} only at the ends.

(e) FIND : find the list that contains a triangle.

→ Balanced Search Tree
→ Run time ~ $O(m^2)$ where m = # edges in triangulation.