CS 7301.003.20F Lecture 23–November 4, 2020

Main topics are #simplicial_homology.

Euler-Poincaré Formula

- Last time, we looked at the (many) basics of simplicial homology. Let's go a bit further today.
- We earlier saw Euler's formula for 2-manifolds. We now have the tools to generalize to arbitrary simplicial complexes.
- Recall,

$$H_k(X) \cong \mathbb{Z}^{\beta_k(X)} \oplus \bigoplus_i (\mathbb{Z}/d_i\mathbb{Z})$$

- where beta_k(X) is known as the kth Betti number.
- Theorem [Euler-Poincaré]: sum_{k ≥ 0} (-1)^k n_k = sum_{k ≥ 0} (-1)^k beta_k.
  - First, some facts from abstract algebra: As stated, every finitely-generated abelian group G can be written as Z^r oplus OPLUS_i (Z / c_i Z) for some integers r and c_i. The integer r is the *rank* of the group.
  - Also, for any subgroup H of any abelian group G, we have rank(G / H) = rank(G) - rank(H).
  - So beta_k(X) = rank(H_k(X)) = rank(Z_k(X)) - rank(B_k(X)).
  - B_k(X) is the row space of partial_{k + 1}, so it has rank equal to the matrix rank of partial_{k + 1}.
  - Z_k(X) is the orthogonal complement of the column space of partial_k, and therefore has rank n_k - rank(partial_k).
  - And now we can compute our sum:

$$\sum_k (-1)^k \beta_k = \sum_k (-1)^k \left(n_k - \mathrm{rank}(\partial_k) - \mathrm{rank}(\partial_{k+1})\right) = \sum_k (-1)^k n_k. \qquad \square$$

- The quantity chi(X) := sum_{k ≥ 0} (-1)^k n_k = sum_{k ≥ 0} (-1)^k is called the *Euler characteristic* or Euler-Poincaré characteristic of X.

Examples: Polygonal Schemata

- Remember these definitions generalize in a natural way to polytopal complexes, and in particular, systems of loops.
- Consider a system of 2g loops ell_1, ell_2, …, ell_{2g} in Sigma(g, 0), the orientable 2-manifold of genus g. Let's compute its homology groups.
- It turns out every cell of this complex has empty boundary:
  - The boundary of a vertex is empty by definition.

    - The boundary of an edge is +1 on one endpoint but -1 on the other, but every edge ℓ_i is a loop.
    - The boundary of a single face f is the sum of its sides, but every edge appears on the boundary of f once in each orientation.
- So the boundary groups B_0, B_1, and B_2 are trivial, and the cycle groups Z_0, Z_1, and Z_2 are equal to their corresponding chain groups.
- Theorem: $H_0(\Sigma(g, 0)) \cong \mathbb{Z}$; $H_1(\Sigma(g, 0)) \cong \mathbb{Z}^{2g}$; and $H_2(\Sigma(g, 0)) \cong \mathbb{Z}$.
- Together with the EulerPoincaré formula, we see $\chi(\Sigma(g, 0)) = 1 - 2g + 1 = 2 - 2g$, which we already knew.
- Now consider a system of g loops $\ell_1, \ldots, \ell_g$ on a non-orientable surface of genus g.
- If we look carefully at the proof of the Surface Classification System, we may assume that exactly one edge, say ℓ_1, appears twice on the boundary of the one face in the same orientation.
    - The boundary of a vertex is again empty by definition.
    - The boundary of any edge is again empty, because every edge is a loop.
    - The boundary of the one face is $2\ell_1$.
- So we have one non-trivial class of 1-boundaries and no non-trivial 2-cycles.
- So just as before, $H_0 \cong \mathbb{Z}$. Also, H_2 is empty, because there are no non-trivial 2-cycles.
- But the first homology group is more interesting. $Z_1 = \langle \ell_1, \ldots, \ell_g \rangle$ and $B_1 = \langle 2\ell_1 \rangle$. Therefore

  $$H_1 = \langle \ell_1, \ldots, \ell_g \rangle / \langle 2\ell_1 \rangle = \mathbb{Z}^{g-1} \oplus (\mathbb{Z} / 2\mathbb{Z})$$
- We have torsion in the first homology group.
- Theorem: $H_0(\Sigma(0, g)) \cong \mathbb{Z}$; $H_1(\Sigma(0, g)) \cong \mathbb{Z}^{g-1} \oplus (\mathbb{Z} / 2\mathbb{Z})$; and $H_2(\Sigma(0, g)) \cong 0$.
- By the Euler-Poincaré formula, $\chi(\Sigma(0, g)) = 1 - (g - 1) + 0 = 2 - g$, as we knew.

Computing Homology

- We never really discussed it before, but computing the fundamental homology groups cannot be done in general. However, there is a well-defined algorithm for computing homology groups.
- It's called the *reduction algorithm*, discovered by Poincaré but closely resembling an algorithm by Smith for computing a certain normal form of an integer matrix.
- The algorithm not only computes a description of the homology groups, but with enough care, we can find bases for the cycle, boundary, and homology groups.
- I'll give a high level description of the algorithm including intuition for why it works. Time permitting, we'll go through a simple example.
- Let $\mathrm{diag}(d_1, d_2, \ldots, d_m)$ denote the $m \times m$ square matrix with integers $d_1, d_2, \ldots, d_m$ along the diagonal and 0's elsewhere.

- The *Smith normal form* of an r x c matrix M is a product of integer matrices S ~M T = M, where S is an invertable r x r matrix, T is an invertable c x c matrix, and ~M is an r x c matrix of the form

$$\tilde{M} = \left[\begin{array}{c|c} Diag(d_1, d_2, \ldots, d_m) & 0 \\ \hline 0 & 0 \end{array}\right]$$

  where each integer d_i divides d_{i + 1}. Both M and ~M have the same rank m.
- The reduction algorithm to compute the Smith normal form iteratively modifies M using O(n^2) elementary row and column operations, similar to Gaussian elimination. These operations are:
  1. exchange two rows or columns
  2. multiply any row or column by -1
  3. add an integer multiple of one row or column to another
- As we do these operations, we can update S or T with each row or column operation, respectively, but we don't actually need to do so to compute homology.
- The algorithm for homology reduces each boundary matrix partial_k to its Smith normal form S_k ~partial_k T_k. Here's what's going on behind the scenes:
- The oriented k-simplices in X define a basis for the chain group C_k(X), and the boundary matrix partial_k expresses the linear boundary homomorphism wrt the standard bases of C_{k - 1} and C_k.
- Each row or column operation can be thought of as a change of basis in C_{k - 1} or C_k, respectively.
  1. Exchanging two rows or columns exchanges the indices of two basis elements.
  2. Multiply a row or column by -1 reverses the orientation of a boundary element.
  3. If e_i and e_j are the ith and jth basis elements of C_k, then adding column i to column j replaces e_j with a new basis element e_i + e_j.

     If eps_i and eps_j are the ith and jth basis element of C_{k - 1}, then adding row i to row j replaces eps_i with a new basis element eps_i - eps_j.
- Yes, the last one is unintuitive, but if you take the matrix product after the row operation, you will end up with the right multiples of the new basis elements for the correct boundary chain.
- Let {e_1, …, e__{n_k}} be the final basis of C_k and {eps_1, …, eps_{n_{k - 1}} be the final basis for C_{k - 1}. These are bases of the column and row spaces, respectively, of ~partial_k.
- But we're in Smith normal form. Column basis elements e_{m_k + 1}, …, e_{n_k} comprise a basis for the cycle group Z_k.
  - Z_k ~= Z^{n_k - m_k}
- Scaled row basis elements d_1 eps_1, …, d_{m_k} eps_{m_k} comprise a basis for the boundary group B_{k - 1}.
  - B_k ~= Z^{m_{k + 1}}

- Finally, it's possible to carefully perform the same row reductions on each partial_{k} as we perform column reductions on partial_{k - 1} so that the basis of each boundary group B_k is a (scaled) subset of the basis for the corresponding cycle group Z_k.
- The complement of that subset forms a basis for the free component of H_k.
- And multiplying each eps_{(k+1}i} by an integer between 1 to d_{(k+1)i} - 1 lets us fill in the torsion components. (So if d_{(k+1}i} = 1, that torsion component is trivial and can be ignored.)
    - H_k ~= Z^{n_k - m_k - m_{k + 1}} oplus OPLUS_{i = 1}^{m_{k + 1}} (Z / d_{(k+1)i) Z)

Matrix Reduction Example

- Let's go through an example. Consider the boundary of the standard 3-simplex Delta_3 with vertices w, x, y, and z. This is a triangulation of the sphere S^2. We'll arbitrarily orient the cells as follows:

$$\begin{aligned} \text{Vertices: } & w, x, y, z \\ \text{Edges: } & wx, wy, wz, xy, xz, yz \\ \text{Facets: } & wxy, wxz, wyz.xyz \end{aligned}$$

- [IN THE FOLLOWING FIGURES, THE ROLES OF ROWS AND COLUMNS HAVE BEEN SWAPPED (maybe Jeff was lying on his side when he wrote his notes?) ALSO, ALL NUMBERS YOU WOULD HAVE EXPECTED HAVE BEEN MULTIPLIED BY -1 (sigh). I WILL TRY TO BE CONSISTENT WITH THE REST OF THE LECTURE IN THE WORDS BELOW AND WHAT I WRITE ON THE BOARD.]
- Here are the boundary maps partial_1 and partial_2:

$$\begin{array}{c|cccc} \partial_1 & w & x & y & z \\ \hline wx & -1 & 1 & 0 & 0 \\ wy & -1 & 0 & 1 & 0 \\ wz & -1 & 0 & 0 & 1 \\ xy & 0 & -1 & 1 & 0 \\ xz & 0 & -1 & 0 & 1 \\ yz & 0 & 0 & -1 & 1 \end{array} \qquad \begin{array}{c|cccccc} \partial_2 & wx & wy & wz & xy & xz & yz \\ \hline wxy & 1 & -1 & 0 & 1 & 0 & 0 \\ wxz & 1 & 0 & -1 & 0 & 1 & 0 \\ wyz & 0 & 1 & -1 & 0 & 0 & 1 \\ xyz & 0 & 0 & 0 & 1 & -1 & 1 \end{array}$$

- To reduce partial_1, we erase the first [ROW] by adding the second, third and forth, and then swap the empty [ROW] to the [BOTTOM].
- Then we clear the last three [COLUMNS] by adding and/or subtracting the first three.

$$\partial_1 \longmapsto \begin{array}{c|ccc:c} & x-w & y-w & z-w & w \\ \hline wx & 1 & 0 & 0 & 0 \\ wy & 0 & 1 & 0 & 0 \\ wz & 0 & 0 & 1 & 0 \\ xy & -1 & 1 & 0 & 0 \\ xz & -1 & 0 & 1 & 0 \\ yz & 0 & -1 & 1 & 0 \end{array} \longmapsto \begin{array}{c|ccc:c} \tilde{\partial}_1 & x-w & y-w & z-w & w \\ \hline wx & 1 & 0 & 0 & 0 \\ wy & 0 & 1 & 0 & 0 \\ wz & 0 & 0 & 1 & 0 \\ \hdashline xy+wx-wy & 0 & 0 & 0 & 0 \\ xz+wx-wz & 0 & 0 & 0 & 0 \\ yz+wy-wz & 0 & 0 & 0 & 0 \end{array}$$

- So m_1 = 3, and d_1 = d_2 = d_3 = 1. And we now have the following bases:

$$\begin{aligned} Z_1 &= \langle xy+wx-wy,\ xz+wx-wz,\ yz+wy-wz \rangle \cong \mathbb{Z}^3 \\ B_0 &= \langle x-w,\ y-w,\ z-w \rangle = \langle \partial wx,\ \partial yw,\ \partial wz \rangle \cong \mathbb{Z}^3. \end{aligned}$$

- To reduce partial_2, we apply the [ROW] operations that are the inverse of the [COLUMN] operations we used for partial_1. Then we do [COLUMN] operations.

$$\partial_2 \longmapsto \begin{array}{c|ccc:ccc} & xy+wx-wy & xz+wx-wz & yz+wy-wz & wx & wy & wz \\ \hline wxy & 1 & 0 & 0 & 0 & 0 & 0 \\ wxz & 0 & 1 & 0 & 0 & 0 & 0 \\ wyz & 0 & 0 & 1 & 0 & 0 & 0 \\ xyz & 1 & -1 & 1 & 0 & 0 & 0 \end{array} \longmapsto$$

$$\begin{array}{c|ccc:ccc} \tilde{\partial}_2 & xy+wx-wy & xz+wx-wz & yz+wy-wz & wx & wy & wz \\ \hline wxy & 1 & 0 & 0 & 0 & 0 & 0 \\ wxz & 0 & 1 & 0 & 0 & 0 & 0 \\ wyz & 0 & 0 & 1 & 0 & 0 & 0 \\ \hdashline xyz-wxy+wxz-wyz & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

- Again, m_2 = 3 and d_1 = d_2 = d_3 = 1. Moreover,

$$Z_2 = \langle xyz - wxy + wxz - wyz \rangle \cong \mathbb{Z}$$
$$B_1 = \langle wx - wy + xy,\ wx - wz + xz,\ wy - wz + yz \rangle = \langle \partial wxy,\ \partial wxz,\ \partial wyz \rangle \cong \mathbb{Z}^3.$$

- Notice that B_1 and Z_1 are generated by the *same* set of 1-cycles, so H_1 is trivial.
- Nw we can compute the homology of partial delta_3.

$$H_0 = C_0/B_0 = \langle w,x,y,z \rangle / \langle x-w,\ y-w,\ z-w \rangle = \langle w \rangle \cong \mathbb{Z}$$
$$H_1 = Z_1/B_1 = Z_1/Z_1 = 0$$
$$H_2 = Z_2 = \langle xyz - wxy + wxz - wyz \rangle \cong \mathbb{Z}$$

- It matches what we would expect for an orientable surface of genus 0!