

Computational Topology with Lab4IQM

Chapters I–VIII: graphs, surfaces, complexes, homology, duality, Morse functions, persistence, stability

Lab4IQM Computational Topology module 0.3 · lab4iqm.com

This companion follows H. Edelsbrunner and J. L. Harer, *Computational Topology: An Introduction* (American Mathematical Society, 2010), chapter by chapter. For each section it gives the Lab4IQM commands that build and draw the objects of the section and run its algorithms. Read the book for the proofs; use Lab4IQM to see the spaces in 3D and to compute their invariants. Type the commands in the Lab4IQM console after

```
import topology
```

`help(homology)` lists the ways to name a space; the Examples panel has one example per section. In a Python script, `import topology` gives the same commands as functions, e.g. `topology.homology("klein", coeff="all")`. This version covers Chapters I–VIII: the foundations, duality, Morse functions, and persistence with its stability.

Naming a space

Put the name in quotes unless it is a single word.

surfaces	<code>sphere</code> , <code>torus</code> , <code>klein</code> (figure-8 immersion), <code>projective</code> (Roman surface), <code>mobius</code> , <code>cylinder</code> , <code>disk</code> , <code>"genus(g)"</code> ($g \leq 6$), <code>torus7</code> (7 vertices), <code>rp2</code> (6 vertices); sizes such as <code>"torus(30, 16)"</code>
polygon words	<code>"word(abAB)"</code> : a polygon whose edges are glued as the word says; a capital letter is an inverse; a letter used once is a boundary edge
complexes	<code>"simplex(n)"</code> , <code>S2</code> , <code>S3</code> , ... (boundaries of simplices), <code>dunce</code> , <code>"circle(n)"</code> , <code>"bouquet(k)"</code> , or maximal simplices with dashes: <code>"a-b-c</code> , <code>b-c-d</code> , <code>d-e</code> "
operations	<code>"cone(K)"</code> , <code>"suspension(K)"</code> , <code>"sd(K)"</code> (barycentric subdivision), <code>"skeleton(K, k)"</code>
point clouds	<code>circle</code> , <code>annulus</code> , <code>figure8</code> , <code>twocircles</code> , <code>disk</code> , <code>square</code> , <code>sphere</code> , <code>torus</code> , <code>random</code> , written <code>name(n, noise, seed)</code> , or a CSV file of points; complexes on them: <code>"rips(C, r)"</code> , <code>"cech(C, r)"</code> , <code>"alpha(C, r)"</code> , <code>"delaunay(C)"</code>

Chapter I: Graphs

I.1 Connected Components

```
unionfind("twocircles(60, 0.03)", r=0.12)
```

The disjoint set system of the section: `unionfind` adds the edges between the points in order of length and keeps only those that join two components, with `FIND` by path compression and `UNION` by size; the kept edges form a minimum spanning forest. The components of the balls of radius r are coloured, and the console lists the radii at which components merge (the 0-dimensional persistence of Chapter VII, in advance).

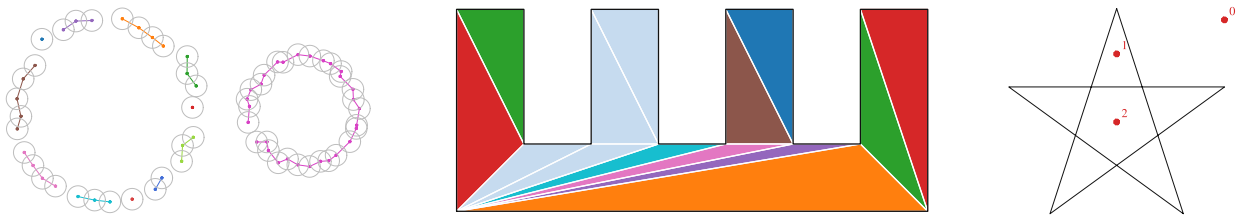
I.2 Curves in the Plane

```

polygon(comb, at=((0.5, 2), (1.5, 2)))      Jordan curve, parity algorithm,
      triangulation
polygon(star, at=((0, 0), (0, 0.6), (0.95, 0.9)))  winding numbers 2, 1, 0
polygon(points=((0,0), (4,0), (4,3), (2,1), (0,3)))

```

`polygon` tests whether a closed polygon is simple. For a simple one, the Jordan Curve Theorem gives an inside and an outside, and the parity algorithm of the section decides which: a ray from the point crosses the curve an odd number of times exactly when the point is inside. The polygon is triangulated into $n - 2$ triangles by cutting off ears. For any closed curve the console also gives the winding number; for the pentagram the parity test fails (the centre has winding number 2 and an even crossing count), because the curve is not simple.



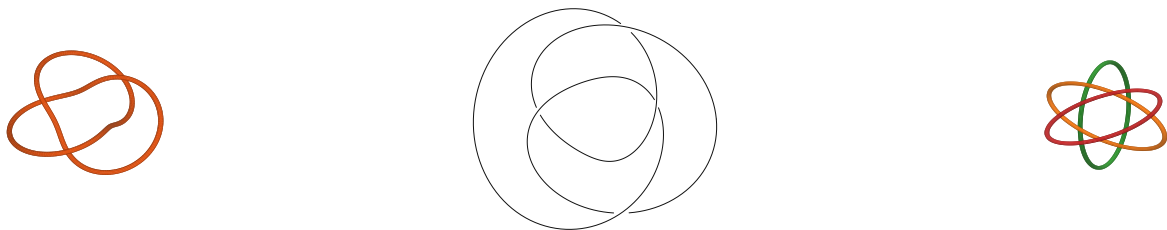
I.3 Knots and Links

```

knot(trefoil)          a tube in 3D
knot(figure8, diagram=true)  the diagram, with gaps at under-crossings
knot(torus, p=3, q=5)    the torus knot T(3,5)
link(hopf)      link(borromean)  link(torus, p=2, q=6)

```

The console counts the crossings of the diagram (the projection seen from above, slightly tilted, so that it is generic) and their signs: the writhe of a knot, and the linking numbers of a link (half the sum of the signs of the crossings between two components), checked against Gauss's linking integral. Reidemeister moves do not change the linking numbers. The Borromean rings, drawn as three ellipses in the coordinate planes, have all linking numbers zero, yet they cannot be split (Exercise I.5).



I.4 Planar Graphs

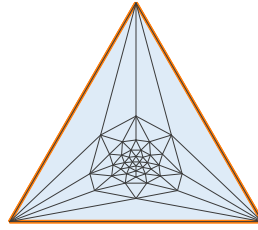
```

tutte("sphere(1)")      tutte(S2)      tutte("suspension(circle(8))")

```

The 1-skeleton of a triangulated sphere is a maximally connected planar graph. `tutte` puts one triangle on a circle and every other vertex at the average of its neighbours: by Tutte's theorem this straight-line drawing has no crossings and convex faces (the console counts crossings: none). It checks the Euler Relation $n - m + f = 2$ and $m = 3n - 6$; the inequalities $m \leq 3n - 6$ and,

without triangles, $m \leq 2n - 4$ show that K_5 and $K_{3,3}$ are not planar, the two graphs of Kuratowski's theorem.



Exercises of Chapter I

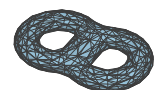
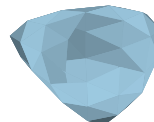
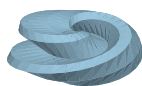
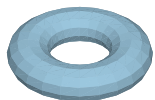
1	<code>unionfind</code> uses the disjoint set system; <code>homology(K)</code> gives b_0 , the number of components, from a matrix reduction.
2	Shelling disks: <code>collapse</code> on a triangulated disk removes triangles one at a time.
3	Jordan curves on the torus: <code>homology(torus, show=1)</code> draws two curves that do not separate.
5	<code>link(borromean)</code> : all linking numbers vanish.
6	<code>link(..., diagram=true)</code> draws diagrams for experimenting with Reidemeister moves.
7–8	Planar colourings: <code>tutte</code> draws the planar triangulations to colour.

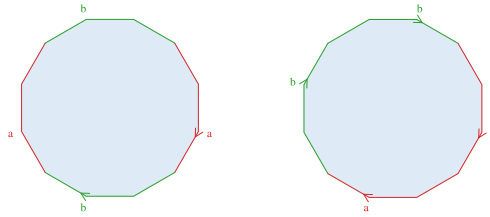
Chapter II: Surfaces

II.1 Two-dimensional Manifolds

<code>surface(torus)</code>	<code>surface(klein)</code>	<code>surface(projective)</code>	<code>surface(mobius)</code>
<code>surface("genus(2)")</code>			
<code>surface("word(abAB)")</code>	<code>surface("word(aabb)")</code>	<code>surface("word(abacbc)")</code>	

`surface` names a surface by the classification theorem: a sphere with g handles or with k cross-caps (connected sums of tori or of projective planes), minus some disks. The non-orientable surfaces are drawn as immersions: the Klein bottle as the figure-8 bottle, the projective plane as the Roman surface. For a polygonal schema, "`word(...)`" glues the edges of a polygon as the word says (capitals are inverses); the polygon is drawn with its edges coloured by letter, and the console reads χ and orientability off the word: the corners fall into V classes, so $\chi = V - E + 1$, and the surface is orientable if and only if every letter appears once in each direction. The genus- g surfaces are smooth unions of g tori.





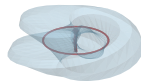
II.2 Searching a Triangulation

`surface` works on the triangles alone, in the spirit of the section: it checks that every edge lies in at most two triangles and that every vertex link is a path or a cycle, computes $\chi = V - E + F$, finds the boundary circles, and decides orientability by a breadth-first search over the triangles, orienting each new triangle compatibly with the one it was reached from; a conflict means the surface is non-orientable.

II.3 Self-intersections

<code>intersections(klein)</code>	<code>intersections(projective)</code>	<code>intersections(torus)</code>
-----------------------------------	--	-----------------------------------

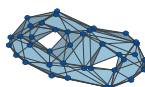
Every pair of triangles that share no vertex and whose bounding boxes overlap is tested; two triangles cross along a segment, found by the edge-triangle tests of the section. The segments are drawn in red: the curves along which the immersed Klein bottle and Roman surface pass through themselves. Embedded surfaces such as the torus have none.



II.4 Surface Simplification

<code>simplify("genus(2)", target=80)</code>	<code>simplify("sphere(2)", target=4)</code>
--	--

Edges are contracted cheapest first by the quadric error (the sum of squared distances to the planes of the triangles around the merged vertex), but only when they satisfy the link condition $\text{Lk } a \cap \text{Lk } b = \text{Lk } ab$, which guarantees that the topology does not change. The console counts the contractions refused by the condition and confirms that χ and the Betti numbers are kept. A sphere simplifies down to the boundary of a tetrahedron, its only irreducible triangulation (Exercise II.6). The genus- g surfaces of the module were made this way, from a fine mesh of a smooth union of tori.



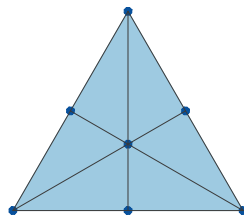
Exercises of Chapter II

-
- | | |
|---|---|
| 1 | Build your surfaces with <code>"word(...)"</code> and classify them with <code>surface</code> . |
| 2 | <code>complex("sd(torus7)")</code> : the barycentric subdivision of a surface. |
| 3 | <code>surface("word(aabb)")</code> and <code>surface("word(abAb)")</code> : both Klein bottles. |
| 4 | The manifold tests of <code>surface</code> (edges in two triangles; vertex links). |
| 5 | <code>intersections(K)</code> uses such tests. |
| 6 | <code>simplify("sphere(2)", target=4)</code> ends at the tetrahedron. |
| 7 | Graphs on the Möbius strip: <code>surface(mobius)</code> . |
| 8 | The quadric error of <code>simplify</code> is such a squared-distance function. |
-

Chapter III: Complexes

III.1 Simplicial Complexes

<code>complex("0-1-2, 1-2-3, 2-3-4, 4-5, 5-6, 6-4")</code>	f-vector, chi, maximal simplices
<code>complex(torus7, link=0)</code>	the link of a vertex: a 6-cycle
<code>complex(torus7, star=0)</code>	its star
<code>complex("sd(simplex(2))")</code>	barycentric subdivision



III.2 Convex Set Systems

<code>cech("circle(30, 0.05)", r=0.3)</code>	the nerve of the balls (drawn)
<code>rips("circle(40, 0.05)", r=0.25)</code>	Vietoris-Rips: diameter $\leq 2r$

The Čech complex is the nerve of the balls of radius r (a simplex for every set of balls with a common point, tested with the smallest enclosing ball); by the Nerve Theorem it has the homotopy type of their union. It is contained in the Vietoris–Rips complex of the same radius, which only needs the pairwise distances, and by the Vietoris–Rips Lemma that is contained in Čech($\sqrt{2}r$). For Rips and Čech only simplices up to dimension `dim=2` are built, so only b_0 and b_1 are reported.

III.3 Delaunay Complexes

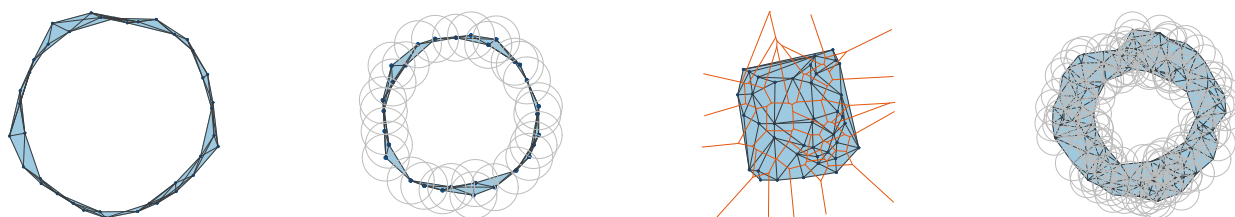
<code>delaunay("square(40)")</code>	Delaunay triangulation and Voronoi diagram
-------------------------------------	--

The Delaunay triangles have empty circumcircles (Bowyer–Watson: insert the points one at a time); the Voronoi diagram is drawn in orange, each Voronoi edge joining the circumcentres of the two triangles that share a Delaunay edge.

III.4 Alpha Complexes

```
alpha("annulus(150)", r=0.2)
collapse("alpha(disk(80), 0.3)")      collapses to a point
collapse(dunce)                       no free face at all
```

The alpha complex keeps the Delaunay simplices whose Voronoi cells meet inside the union of the balls; it has the homotopy type of the union, with far fewer simplices than the Čech complex. The console also compares it with $\check{C}ech(r) \cap \text{Delaunay}$ (Exercise III.7). `collapse` removes free faces with their cofaces as long as it can: elementary collapses do not change the homotopy type, and the alpha complex of points in a disk collapses to a point.



Exercises of Chapter III

2–3	<code>complex("sd(simplex(d))")</code> : $(d+1)!$ top simplices; the order complex.
5	<code>complex(K, star=v)</code> shows the vertex stars whose nerve is K .
7	<code>alpha(C, r)</code> reports whether $\text{Alpha}(r) \subseteq \check{C}ech(r) \cap \text{Delaunay}$ and counts the simplices of the intersection that are not in $\text{Alpha}(r)$.
8	<code>collapse(dunce)</code> : contractible but not collapsible.

Chapter IV: Homology

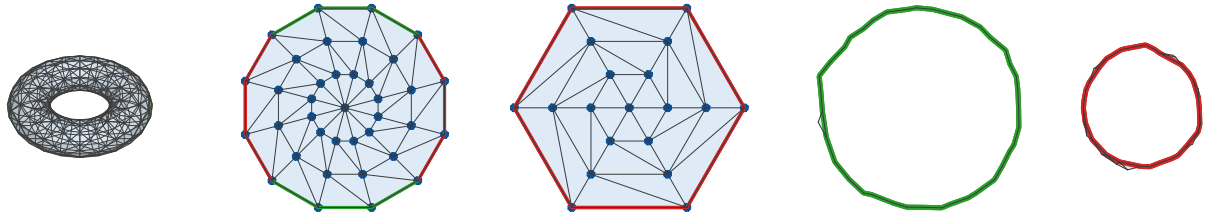
IV.1 Homology Groups

```
homology(torus, show=1)      b = (1, 2, 1) and two generating cycles
homology("alpha(twocircles(80, 0.02), 0.2)", show=1)  two holes in a point cloud
homology(S3)                 homology(torus3)           homology(dunce)
```

IV.2 Matrix Reduction

```
homology(projective, coeff=all, show=1)      H1 = Z/2: torsion
homology(klein, coeff=all)                   Z/2: (1, 2, 1); Q: (1, 1, 0); H1 = Z + Z
/2
homology("word(aaa)", coeff=all)              H1 = Z/3
```

Over $\mathbb{Z}/2$ (the book's coefficients) the boundary matrices are reduced column by column, left to right: $b_k = \text{rank } Z_k - \text{rank } B_k$, and the reduction also records which chains became cycles, so `show=k` draws a basis of H_k , each generator in its own colour. `coeff=all` adds the Betti numbers over $\mathbb{Q}$ and the integral groups, from the Smith normal form of the integer boundary matrices; the torsion explains why the Betti numbers depend on the coefficients. The console checks $\sum (-1)^k b_k = \chi$.



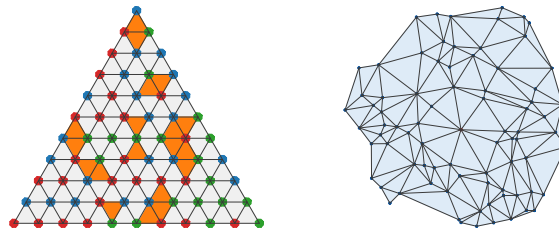
IV.3–IV.4 Relative Homology; Exact Sequences

```
homology(disk, rel=boundary)      H(D, S1) = H(S2): (0, 0, 1)
homology(mobius, rel=boundary, coeff=all)
homology(torus7, rel=(0, 1, 3))  relative to the subcomplex on those vertices
```

The relative groups $H_k(K, L)$ come from the chains of K modulo those of L . The long exact sequence of the pair implies $\chi(K) = \chi(L) + \chi(K, L)$, which the console checks.

Exercises of Chapter IV

-
- | | |
|---|---|
| 1 | <code>sperner(n=10)</code> : Sperner colourings; the fully coloured triangles (an odd number). |
| 2 | <code>homology("wedge(circle(5), circle(5), S2)")</code> and <code>homology(torus)</code> : the same homology, but not homotopy equivalent. |
| 4 | <code>homology(klein, coeff=all)</code> and <code>homology(torus)</code> . |
| 5 | <code>homology(dunce, coeff=all)</code> (reduced Betti numbers vanish) and <code>collapse(dunce)</code> . |
| 6 | <code>homology(torus3)</code> : $\beta = (1, 3, 3, 1)$. |
| 8 | <code>homology(K, rel=...)</code> for pairs; the console checks the Euler characteristics. |
-



V.1 Cohomology

```
cohomology(torus, show=1)      two 1-cocycles and their dual curves
cohomology(cylinder, show=1)   the picket fence from the outer to the inner boundary
cohomology(disk, rel=boundary) relative cohomology
```

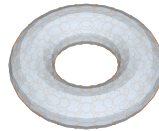
`cohomology` computes $H^p = Z^p/B^p$ over $\mathbb{Z}/2$ from the coboundary matrices, the boundary matrices transposed: the coboundary of a simplex is the sum of its cofaces of one dimension higher (Exercise 1). A 1-cocycle is drawn as its “picket fence” of edges together with the closed dual curve that crosses exactly those edges. The ranks equal the Betti numbers, and the console evaluates every representative cocycle on every representative cycle, $\langle \varphi, c \rangle =$ the parity of the shared simplices: the matrix is invertible, the isomorphism $H^p \rightarrow \text{Hom}(H_p, \mathbb{Z}/2)$ of the Universal Coefficient Theorem (Exercise 2).



V.2 Poincaré Duality

```
poincare(torus)      poincare(torus3)    poincare(mobius)    poincare(dunce)
```

`poincare` builds the dual blocks (for a surface: a point in each triangle, a dual edge across each edge, drawn in orange, a polygon around each vertex) and the block chain complex, in which the boundary of the block of σ is the sum of the blocks of its cofaces. The console checks that K is a combinatorial manifold (every link of an i -simplex is a $(d-i-1)$ -sphere), the Block Complex Lemma (the block complex has the homology of K) and Poincaré duality $\beta_p = \beta^{d-p}$. With a boundary (the Möbius band) Poincaré duality fails and Lefschetz duality $\beta_p(M) = \beta_{d-p}(M, \partial M)$ holds; the dunce hat is not a manifold.



V.3 Intersection Theory

```
pairing(torus)      [0 1; 1 0]
pairing(klein)      a curve with a Moebius neighbourhood meets its push-off once
pairing(mobius)     Lefschetz: H_1(M) x H_1(M, boundary)
pairing(torus3)     H_1 x H_2 of the 3-torus
```

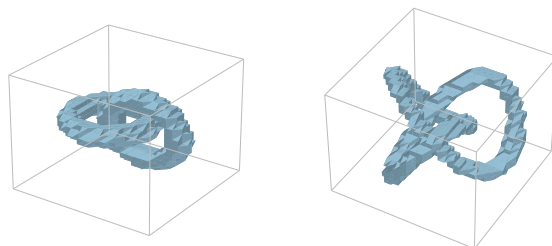
The intersection numbers of a basis of H_p with a basis of H_q , $p + q = d$, computed as the cup product of the Poincaré dual cocycles on the fundamental class (the sum of all top simplices). The matrix has determinant one: the pairing is perfect (the second form of Poincaré duality). Over $\mathbb{Z}/2$ the form of a surface has zero diagonal exactly when the surface is orientable. For $d = 2k$ the Euler characteristic is even iff β_k is.



V.4 Alexander Duality

```
alexander(shell)      alexander(hopf)      alexander(trefoil)
alexander(f=abs(x)+abs(y)+abs(z)-0.7, n=20)
```

S^3 is triangulated as a cube cut into $6n^3$ tetrahedra plus a vertex at infinity coned over its boundary; X is the full subcomplex where a function is ≤ 0 , and $S^3 - X$ retracts to the full subcomplex on the other vertices. The console computes the Betti numbers of X twice: by matrix reduction, and by the algorithm of the section (β_0 by union-find, $\beta_2 =$ the number of components of the complement minus one by Alexander duality, β_1 from the Euler characteristic), and checks the predicted homology of the complement, $\tilde{H}_p(X) \cong \tilde{H}_{2-p}(S^3 - X)$, against its Euler characteristic. A knotted tube and an unknotted one have the same homology.



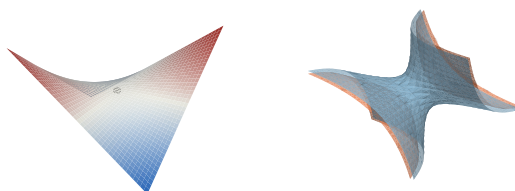
Exercises of Chapter V

1	<code>cohomology(K, show=1)</code> : every triangle has an even number of cocycle edges.
2	<code>cohomology(K)</code> checks that the evaluation matrix on representative bases is invertible.
3, 4	<code>pairing(K)</code> : a perfect pairing gives the isomorphism $V \rightarrow \text{Hom}(U, G)$.
5	<code>poincare(torus), pairing(torus), cohomology(torus, show=1)</code> .

VI.1 Generic Smooth Functions

<code>hessian(f=x^2+y^2)</code>	index 0	<code>hessian(f=x*y)</code>	index 1
<code>hessian(f=(x+y)^2)</code>	degenerate	<code>hessian(f=x*y+x*z+y*z)</code>	index 2
<code>hessian(f=x^3-3*x*y^2)</code>	the monkey saddle (degenerate)		

`hessian` checks that the gradient vanishes at the point `at=` (default the origin), computes the Hessian and its eigenvalues, and gives the index of a non-degenerate critical point with the normal form of the Morse Lemma. It draws the graph of f (two variables) or two level sets (three variables). This is Exercise 1, item by item.

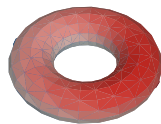


VI.2 Transversality

```
morsesmale(torus, f=z+0.15*x)      morsesmale("genus(2)", f=z+0.2*x)
```

From each saddle of a PL function on a surface, `morsesmale` follows the paths of steepest descent (blue) to minima and of steepest ascent (red) to maxima, the PL analogues of the stable and

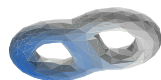
unstable 1-manifolds. The console gives the Morse chain complex of the section (the boundary of a critical point is the sum of the critical points one index lower that its paths reach, mod 2) and its homology, which for the slightly tilted torus is $(1, 2, 1)$ with all boundaries zero (Exercise 3). It counts the cells and checks critical points – arcs + cells = χ ; when every cell has four sides the complex is made of quadrangles (Exercise 4).



VI.3 Piecewise Linear Functions

```
critical(torus, f=z+0.1*x)      critical("genus(2)", f=z+0.2*x, at=0)
critical(torus3)                critical(S3, f=x)
```

Vertices are classified by the reduced Betti numbers of their lower links: regular, a minimum, a simple critical point of index q ($\tilde{\beta}_{q-1} = 1$), or a multiple one such as a k -fold saddle ($\tilde{\beta}_0 = k$). The console checks $\chi = \sum (-1)^{\text{index}}$ and the weak and strong PL Morse inequalities, for f and for $-f$. `at=` colours the sublevel set. Ties are broken by the vertex index, so the perfectly lying torus `critical(torus, f=z)` becomes a PL Morse function arbitrarily close to f (Exercise 2).



VI.4 Reeb Graphs

```
reeb(torus, f=z+0.1*x)    reeb("genus(2)", f=x)    reeb(klein, f=z+0.2*x)    reeb(disk, f=
=x*y)
```

`reeb` sweeps the critical values; between consecutive ones the surface splits into slabs, the slab with the critical vertex becomes a node and the others carry contours through. The graph is drawn to the right of the surface, its arcs through the centroids of the contours. The console checks the Loop Lemma: g loops for an orientable surface of genus g , at most $g/2$ for a non-orientable one, with the number of degree-two nodes (orange) of the parity of g . On a disk the Reeb graph is a contour tree.



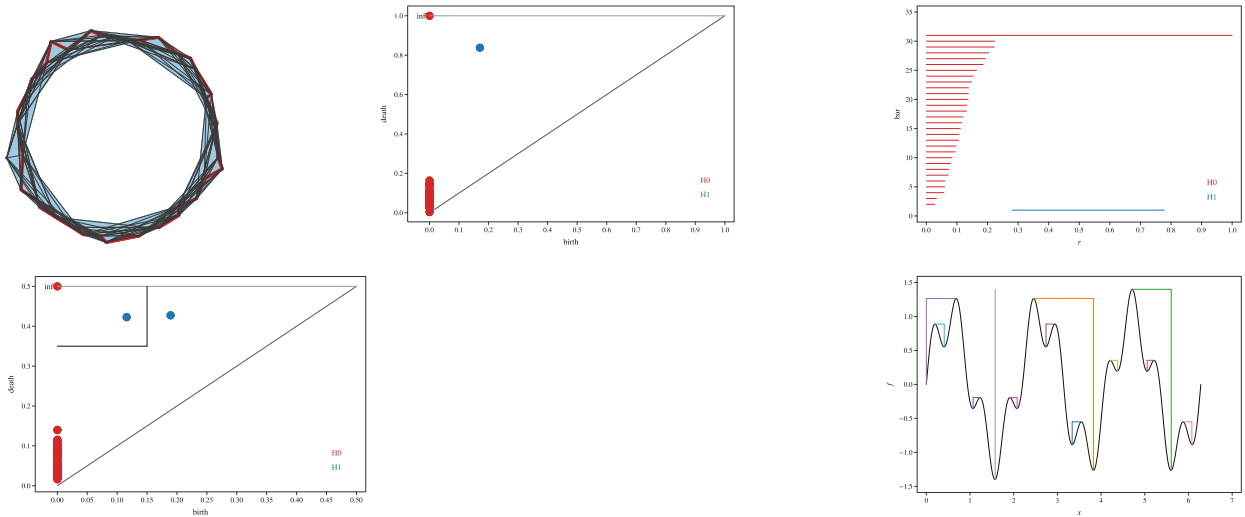
Exercises of Chapter VI

1	<code>hessian(f=...)</code> for the six functions of the exercise.
2	<code>critical(torus, f=z)</code> and <code>reeb(torus, f=z)</code> : the symbolic perturbation gives a nearby Morse function; <code>reeb(torus, f=z+0.1*x)</code> .
3, 4	<code>morsesmale(torus, f=z+0.15*x)</code> : four critical points with zero boundaries; cells and χ .
5	<code>critical(torus, f=sqrt((x-a)^2+(y-b)^2+(z-c)^2), a=0.3, b=0, c=0.2)</code> ; try points on the axis ($a = b = 0$) and on the centre circle.
6	<code>critical(K, f=...)</code> prints the inequalities for $-f$; <code>critical(torus3)</code> : $\chi = 0$ in odd dimension.
7	<code>reeb(torus, f=cos(a)*z+sin(a)*x, a=...)</code> from $a = 1.57$ (upright) to $a = 0$ (on its side, where the height is no longer Morse).

VII.1 Persistent Homology

<code>persistence("circle(40, 0.05)")</code>	Rips filtration of a noisy circle
<code>persistence("circle(30, 0.1)", show=barcode)</code>	the same classes as bars
<code>persistence("figure8(60, 0.02)", a=0.15, b=0.35)</code>	persistent Betti numbers $\beta^{\{a,b\}}$
<code>elder(f=sin(3*x)+0.4*sin(11*x))</code>	the elder rule on an interval

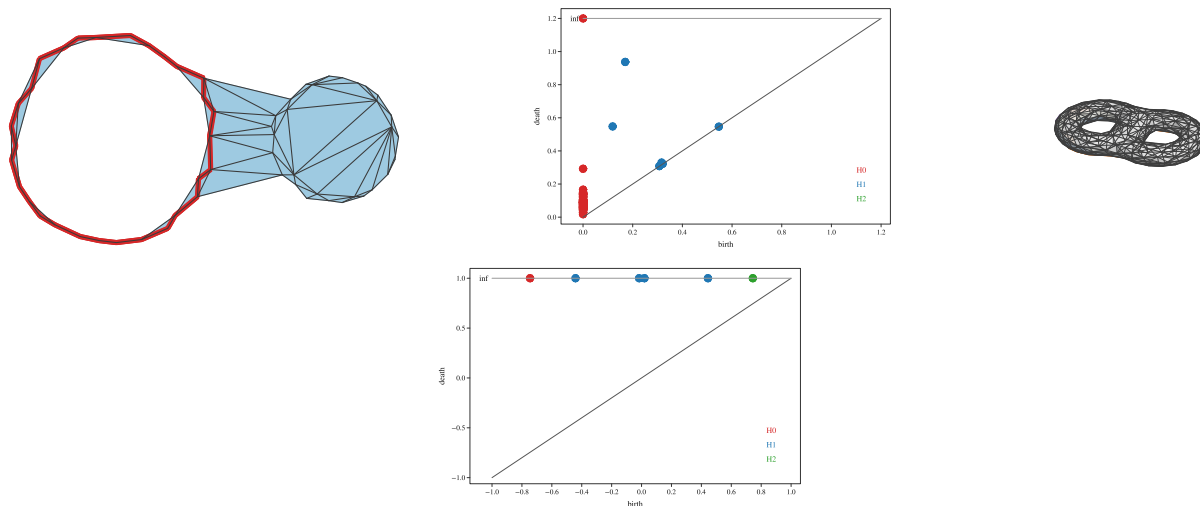
`persistence` builds a filtration, reduces the boundary matrix over $\mathbb{Z}/2$ from left to right (the Persistence Algorithm of the section; the Pairing Lemma reads the pairs from the lowest ones) and draws the diagrams: one point (birth, death) per class, red for H_0 , blue for H_1 , green for H_2 ; classes that never die sit on the line “inf”. On the left it draws the complex $K(a)$ halfway through the longest-lived class, with representative cycles of the classes alive there (columns of V in $R = \partial V$). With `a=` and `b=` it prints $\beta_p^{a,b}$ and draws the quadrant $\{\text{birth} \leq a, \text{death} > b\}$ whose points it counts: the Fundamental Lemma of Persistent Homology. `elder` sweeps the sublevel sets of a function of x : each minimum starts a component and, when two meet, the younger dies.



VII.2 Efficient Implementations

<code>persistence("twocircles(60, 0.03)", filter=alpha)</code>	alpha filtration: Delaunay simplices
<code>persistence("genus(2)", f=z+0.2*x)</code>	lower star filtration of a height
<code>function</code>	
<code>persistence("circle(30, 0.05)", variant=sparse)</code>	the sparse implementation
<code>persistence("sphere(60, 0.02)", maxdim=3)</code>	a 2-sphere hidden in a point cloud

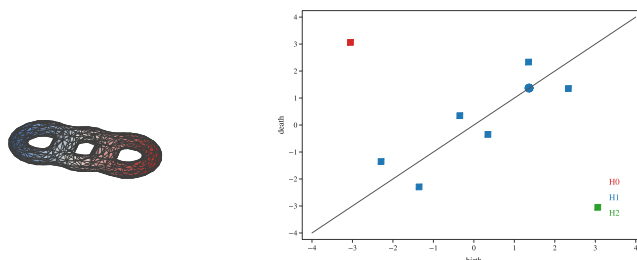
Filtrations (`filter=`): `rips` (default for point clouds; a simplex enters at half its diameter, up to `r=`), `cech` (at the radius of its smallest enclosing ball), `alpha` (points in the plane, at their alpha values) and `lowerstar`, used when `f=` gives a function of x, y, z on the vertices of a space: each simplex enters at the largest value of its vertices. By the Persistence Equivalence Theorem its diagrams are those of the piecewise linear function. `variant=sparse` is the linked-list version of the section; every run also computes the zeroth diagram by union-find with the elder rule and checks that it agrees, and prints the number of column additions.



VII.3 Extended Persistence

<code>extended(torus, f=z)</code>	Ext only: (min, max), the two essential 1-cycles, (max, min)
<code>extended("genus(3)", f=x)</code>	Exercise 5
<code>extended(cylinder, f=z)</code>	Exercise 6: a boundary breaks duality and symmetry

The ascending pass adds lower stars in increasing order; the descending pass cones off the upper stars in decreasing order, so that every class is born and dies. Points of Ord are discs (above the diagonal), Rel triangles (below), Ext squares (either side). The console checks the Persistence Duality Theorem, $\text{Ord}_p = \text{Rel}_{d-p}^T$, $\text{Ext}_p = \text{Ext}_{d-p}^T$, and the Persistence Symmetry Theorem, comparing with $-f$: $\text{Ord}_p(f) = \text{Ord}_{d-p-1}^R(-f)$, $\text{Ext}_p(f) = \text{Ext}_{d-p}^0(-f)$, $\text{Rel}_p(f) = \text{Rel}_{d-p+1}^R(-f)$. On a closed surface of genus g , Ext_1 has $2g$ points.

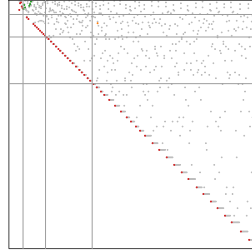


VII.4 Spectral Sequences

<code>spectral("circle(12, 0.1)", blocks=4)</code>	
<code>spectral("circle(6)", blocks=simplices)</code>	Exercise 8: one simplex per block

`spectral` groups the simplices in blocks and reduces the boundary matrix diagonally: phase r uses columns of the blocks $j - r + 1, \dots, j$ whose lowest ones lie in block $j - r + 1$, so the pairs are found

in order of persistence. The matrix is drawn with the block lines, each lowest one coloured by the phase that found it. The console lists $\sum_p \text{rank } E_{p,q}^r$ for every r and checks it against the diagram: a pair of persistence at least r keeps one generator in the dimension of its birth and one in the dimension of its death, an essential class one. It also checks that the result is that of the Persistence Algorithm (Exercise 7).



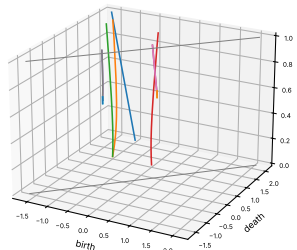
Exercises of Chapter VII

-
- | | |
|------|---|
| 1 | <code>persistence("simplex(3)", filter=dim)</code> , with <code>seed=</code> to shuffle within a dimension: the diagrams do not change. |
| 2 | <code>variant=right</code> adds each column to the columns on its right; the console confirms the same lowest ones. |
| 3 | <code>elder(f=...)</code> and <code>persistence(K, f=..., at=...)</code> draw sublevel sets between and at critical values. |
| 4 | <code>persistence("circle(30)", filter=lowerstar, f=x, variant=sparse)</code> : fewer additions than simplices. |
| 5 | <code>extended("genus(3)", f=x)</code> : the Ord, Ext and Rel sub-diagrams. |
| 6 | <code>extended(cylinder, f=z)</code> , <code>extended(mobius, f=x)</code> : the theorems fail with boundary. |
| 7, 8 | <code>spectral(...)</code> : same reduced lowest ones; <code>blocks=simplices</code> for one simplex per block. |
-

VIII.1 Time Series

<code>vineyard(f=sin(3*x), g=sin(3*x)+0.6*sin(7*x))</code>	two functions of x (time series)
<code>vineyard("genus(2)", f=z, g=y, steps=21)</code>	two height functions on a surface ($p=1$)

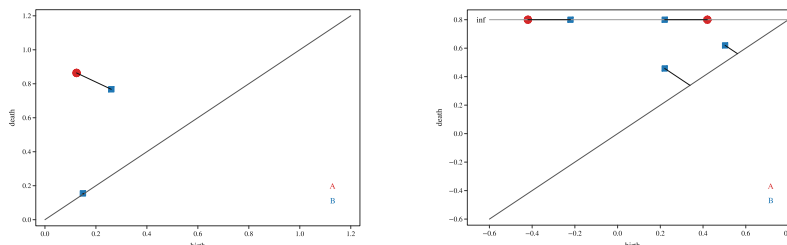
The straight-line homotopy $f_t = (1 - t)f + tg$ gives one diagram for each t ; joining the points gives the vines, drawn in the 3D frame (birth, death, t) above the diagonal plane. A vine that reaches the diagonal ends there. The console counts the transpositions of the simplex order on the way (the Transposition Lemma describes how the pairing changes at each one).



VIII.2 Stability Theorems

<code>bottleneck("circle(40, 0.02)", "circle(40, 0.12)"</code>	two noisy circles (the same points
<code>+ noise)</code>	
<code>bottleneck(torus, f=z, g=z+0.2*sin(3*x))</code>	two functions on a space

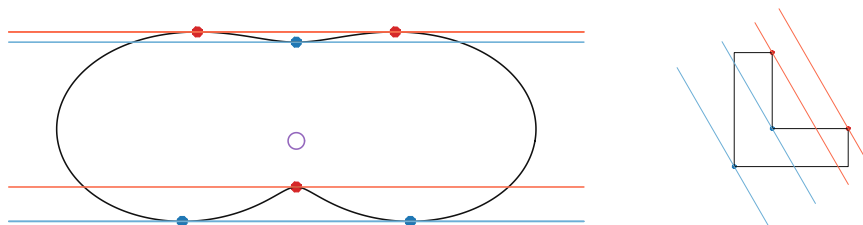
Both diagrams in one frame (A red, B blue) and the optimal bottleneck matching. The console gives W_∞ , W_1 and W_q ($q=$) and checks the Stability Theorem $W_\infty(\text{Dgm}_p(f), \text{Dgm}_p(g)) \leq \|f - g\|_\infty$. For two clouds of the same size the Rips filtrations are taken on one complex, and $\|f - g\|_\infty \leq \max_i |a_i - b_i|$.



VIII.3 Length of a Curve

<code>crofton("flower(5, 0.3)")</code>	length and total curvature from height functions
<code>crofton(bean, vs="circle(0.05)")</code>	the Generalized Fary Theorem
<code>crofton(trefoil)</code> <code>crofton(sphere)</code>	Exercise 5: space curves and surfaces

For a closed curve in the plane: $\text{length} = \frac{1}{2} \int_{S^1} \text{Pers}_0(f_u) du$ (the Cauchy–Crofton Formula, with the global minimum paired with the global maximum) and total curvature $= \frac{1}{2} \int_{S^1} \#\text{crit}(f_u) du$, each next to the direct value. The drawing marks the minima (blue) and maxima (red) of the height in direction $u=$. With `vs=` it checks $|L - L'| \leq (\kappa + \kappa' - 2\pi)F$, with F bounded above by the best matching of evenly spaced samples.



VIII.4 Bipartite Graph Matching

`bottleneck` computes W_∞ as in the section: the candidate lengths are sorted, and a binary search tests each by a maximum matching of the graph $G(\varepsilon)$ (Hopcroft–Karp: shortest augmenting paths found by breadth-first search). Points may be matched to the diagonal (the Reduction Lemma). W_q comes from a minimum-cost perfect matching (the Hungarian algorithm).

Exercises of Chapter VIII

4	<code>vineyard(...)</code> prints the vineyard measure μ_p next to W_1 : <code>vineyard(f=sin(3*x), g=sin(3*x)+0.6*sin(7*x))</code> gives $\mu < W_1$, <code>vineyard(torus, f=z, g=x)</code> gives $\mu > W_1$.
5	<code>crofton(trefoil)</code> : $\text{length} = \frac{1}{\pi} \int_{S^2} \text{Pers}_0(f_u) du$ and $\text{curvature} = \frac{1}{4} \int_{S^2} \#\text{crit}(f_u) du$; <code>crofton(sphere)</code> : $\text{area} = \frac{1}{\pi^2} \int_{S^2} \int_{\mathbb{R}} \text{length}(S \cap \text{plane}) dz du$.
7	The Hopcroft–Karp search uses a single queue for all layers.
8	<code>bottleneck</code> also runs the incremental matching (edges added in order of length, the matching kept) and prints the same W_∞ .

How the results are checked

The module's tests compare the Betti numbers over $\mathbb{Z}/2$ and $\mathbb{Q}$ and the integral homology of 22 spaces with their known values (spheres, tori, the Klein bottle, the projective plane and the pseudo-projective plane of order 3, surfaces of genus 2 and 3, words, the dunce hat, S^3 , S^4); check that cones are contractible, that suspension shifts reduced homology, and that barycentric subdivision keeps it; test the Smith normal form against the gcd of the minors; classify surfaces and random polygon words both ways; check that simplification keeps the topology; check empty circumcircles and the triangle count $2n - 2 - h$ of Delaunay triangulations, that alpha and Čech complexes give the same b_0, b_1 , that $\check{\text{Cech}}(r) \subseteq \text{Rips}(r) \subseteq \check{\text{Cech}}(2r/\sqrt{3})$, that union-find builds a minimum spanning tree, and that linking numbers from crossings equal Gauss's integral. They also check the additions for the exercises: Tutte drawings without crossings, no self-intersections for embedded surfaces, collapses (the dunce hat has none), the 3-torus $(1, 3, 3, 1)$, an odd number of Sperner triangles, the tetrahedron as the end of simplifying a sphere, and $(d+1)!$ top simplices in $\text{Sd } \sigma^d$. For duality and Morse functions they check that cohomology and homology have the same ranks with an invertible evaluation pairing, the Block Complex Lemma and Poincaré duality on closed manifolds (and their failure with boundary), the intersection forms of the torus, Klein bottle, projective plane, genus-2 surface, 3-torus and Möbius band, Alexander duality for balls, shells, tori and two balls (union-find against matrix reduction), the indices of the Hessians of Exercise VI.1, the PL Morse inequalities on surfaces and the 3-torus, a monkey saddle, the Morse homology of the tilted torus, and the loops of Reeb graphs. For persistence they check the triangle example of Section VII.1, that the three reductions and union-find agree, the Fundamental Lemma against the Betti numbers of $K(a)$, that alpha and Čech filtrations have the same diagrams, the duality and symmetry theorems on the sphere, torus and genus-2 surface (and their failure with boundary), $2g$ essential 1-classes, the Spectral Sequence Theorem phase by phase, bottleneck and Wasserstein distances against brute force, the stability theorems, the Cauchy–Crofton and total curvature formulas, Fary–Milnor for the trefoil, the area of the sphere by slicing, and the Generalized Fary inequality.