

CSE 270 (FA 2026) Homework 2

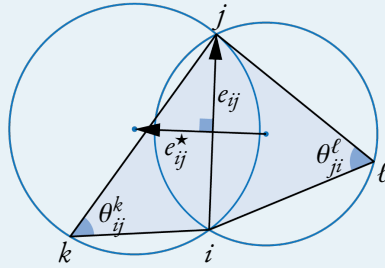
Discrete Cotan Hodge Star

Exercise 2.1 — 10 pts. Consider a triangle mesh. Place the dual vertex of a triangle at the center of its circumcircle. Note that the circumcenter of a triangle is the intersection of all three edge perpendicular bisectors, so the dual edges are already orthogonal to the primal edges ($P_{e_{ij}^\perp}(e_{ij}^\star) = e_{ij}^\star$).

(a)—5 pts Show that in that case the ratio of the dual and primal edge lengths is

$$\frac{|e_{ij}^\star|}{|e_{ij}|} = \frac{1}{2} \left(\cot \theta_{ij}^k + \cot \theta_{ji}^\ell \right) \quad (1)$$

for edge e_{ij} shared by facets f_{ijk} and $f_{ji\ell}$. Here, the angle $\theta_{ij}^k = \angle_i^j$ is the interior angle across from e_{ij} in triangle ijk , and similarly for θ_{ji}^ℓ .



Hint You can use the inscribed angle (circumference angle) theorem

https://en.wikipedia.org/wiki/Inscribed_angle

(b)—5 pts Show that the cotangent weight (the right-hand side of (1)) is negative if and only if $\theta_{ij}^k + \theta_{ji}^\ell > \pi$. Illustrate a picture including both triangles, their circumcircles, e_{ij} and $e_{ij}^\star$ when this condition $\theta_{ij}^k + \theta_{ji}^\ell > \pi$ occurs. (The condition that $\theta_{ij}^k + \theta_{ji}^\ell \leq \pi$, or that the cotan weight is non-negative, is called the **Delaunay condition**.)

Hint You may use softwares/apps like geogebra to experiment on how the illustration should look like. ■

Change of Coordinates

Exercise 2.2 — 15 pts. Consider the 2D Euclidean plane with a standard xy Cartesian coordinate. The polar coordinate r, θ for the plane is related to x, y by

$$x = r \cos \theta, \quad y = r \sin \theta.$$

At each point, let $\vec{e}_x, \vec{e}_y$ be the basis dual to dx, dy and let $\vec{e}_r, \vec{e}_\theta$ be the basis dual to $dr, d\theta$. (That is, the directional derivatives along these basis vectors are $df[\vec{e}_r] = \frac{\partial f}{\partial r}, df[\vec{e}_\theta] = \frac{\partial f}{\partial \theta}$ for any function f on the plane.) Suppose the plane has an inner product structure everywhere so that $\vec{e}_x, \vec{e}_y$ are orthonormal.

(a)—5 pts In the relationship between dx, dy and $dr, d\theta$, and the relationship between two

coordinate bases

$$\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} dr \\ d\theta \end{bmatrix}, \quad [\vec{e}_r \quad \vec{e}_\theta] = [\vec{e}_x \quad \vec{e}_y] \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

what are the expressions $a_{11}, \dots, a_{22}, b_{11}, \dots, b_{22}$ all written in r, θ ?

(b)—5 pts What is the inner product matrix for the polar coordinate basis (written in expressions of r, θ)

$$\mathbf{M} = \begin{bmatrix} \langle \vec{e}_r, \vec{e}_r \rangle & \langle \vec{e}_r, \vec{e}_\theta \rangle \\ \langle \vec{e}_\theta, \vec{e}_r \rangle & \langle \vec{e}_\theta, \vec{e}_\theta \rangle \end{bmatrix} = ?$$

(c)—5 pts Let $u(r, \theta) := r\theta^2$. What is the gradient vector $\text{grad } u = c_1 \vec{e}_r + c_2 \vec{e}_\theta$ written in the polar coordinate basis? ■

Exterior Product

Exercise 2.3 — 5 pts. Let dx, dy, dz, dt denote a basis for $(\mathbb{R}^4)^*$. Define the 2-form $\alpha = u_{12} dx \wedge dy + u_{24} dy \wedge dt + u_{34} dz \wedge dt$ and the 1-form $\beta = w_2 dy + w_3 dz$. Compute $\alpha \wedge \beta$ and $\alpha \wedge \alpha$. ■

Implementation (25 pts): Surface Laplacian

In this implementation exercise you will build a Laplacian (a.k.a. Laplace–Beltrami operator) for a triangle surface. Using the Laplacian you will solve a Poisson equation.

In the following we let $\mathcal{P}$ denote the set of points, $\mathcal{H}$ denote the set half-edges (which is the same as the set of Houdini-vertices) and $\mathcal{F}$ denote the set faces (primitives).

1. Build Laplacian $L = d^\top \star_1 d$ and the mass matrix $\star_0$.
 - (a) Compute the incident points `i@src`, `i@dst` (which has been computed for you), and the half-edge weight `f@hedge_weight` as attributes defined over $\mathcal{H}$.
 - (b) Compute the point mass `f@point_weight` as attributes defined over $\mathcal{P}$. It can be defined as the barycentric area, *i.e.* the point weight A_p at point p is given by $A_p = \frac{1}{3} \sum_{f \succ p} A_f$ where A_f is the area of the triangle f . Here “ $f \succ p$ ” denotes the face f is incident to the point p .
 - (c) Implement `build_laplacian`. That is, build $d_0, \star_1$, and $\star_0$.
 - d_0 is a $|\mathcal{H}|$ -by- $|\mathcal{P}|$ sparse matrix such that for each half-edge $b \in \mathcal{H}$,

$$(d_0)_{b, \text{src}(b)} = -1, \quad (d_0)_{b, \text{dst}(b)} = 1, \quad (2)$$

in other words,

$$d_0 = \text{SPARSE} \left(\text{ROW} = \begin{bmatrix} 0 \\ \vdots \\ |\mathcal{H}|-1 \\ 0 \\ \vdots \\ |\mathcal{H}|-1 \end{bmatrix}, \text{COL} = \begin{bmatrix} | \\ \text{src} \\ | \\ | \\ \text{dst} \\ | \end{bmatrix}, \text{VAL} = \begin{bmatrix} -1 \\ \vdots \\ -1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \text{SIZE} = (|\mathcal{H}|, |\mathcal{P}|) \right). \quad (3)$$

See SciPy documentation `scipy.sparse.csr_matrix` for constructing sparse matrices.

- $\star_1$ is a $|\mathcal{H}|$ -by- $|\mathcal{H}|$ sparse diagonal matrix such that for each half-edge $b \in \mathcal{H}$,

$$(\star_1)_{b,b} = \text{hedge_weight}(b). \quad (4)$$

- $\star_0$ is a $|\mathcal{P}|$ -by- $|\mathcal{P}|$ sparse diagonal matrix such that for each point $p \in \mathcal{H}$

$$(\star_0)_{p,p} = \text{point_weight}(p). \quad (5)$$

- The values of the (symmetric positive semi-definite) Laplacian $L = d_0^\top \star_1 d_0$ and the mass matrix $\star_0$ are stored as cached data at the node `build_laplacian`. This is accomplished by

```
node = hou.pwd()
cache_node = node # node to store the resulting matrices

...

# laplace matrix
L = d0.transpose().dot(star1.dot(d0))

# cache
cache_node.setCachedUserData("L", L)
cache_node.setCachedUserData("star0", star0)
```

(d) Some useful tips:

- You can open a python shell. A print command in your code will print message in the python shell.
- Middle-mouse-click on the python node will force execute the node.
- On the lower-right corner of the entire Houdini software, there is a dropdown menu defaulted to be “Auto Update.” Any change of your code will cause some recomputation, which could be quite slow sometimes. You can change the update scheme to “Manual” so that the computation is cooked only when you perform middle-mouse-click or resetting the update scheme to auto.

2. Look at a few smallest eigenvalues of the surface Laplacian to make sure your Laplacian is reasonable. This part is for validation that has been implemented for you, but please take time to understand what is going in the network.

The Laplacian from the smooth theory reads more like $(-\Delta) = \star_0^{-1} d^\top \star_1 d$. The associated eigenvalue problem is

$$(-\Delta)u = \lambda u \quad (6)$$

However, $(-\Delta) = \star_0^{-1} d^\top \star_1 d$ is not a symmetric matrix, limiting options for faster numerical eigenvalue solvers. A better arrangement for the eigenvalue problem is the so-called *generalized eigenvalue problem*

$$(d^\top \star_1 d)u = \lambda \star_0 u. \quad (7)$$

Eigenvalue solvers such as `scipy.sparse.linalg.eigsh` can solve generalized eigenvalue problem $Au = \lambda Bu$ for symmetric (Hermitian) A and symmetric positive definite B .

The solution to (6) has the physical interpretation of that λ, u are the frequency and the standing wave on the surface. The smallest few eigenvalues would be the first few harmonic modes.



Figure 1 Bunny harmonics.

If the domain is the unit sphere, then u will be the *spherical harmonics* (please see Wikipedia) and the corresponding frequencies are explicitly given by

$$\begin{aligned} \lambda_0 &= 0, & \lambda_1 &= \lambda_2 = \lambda_3 = 2, & \lambda_4 &= \dots = \lambda_8 = 6, \\ \lambda_9 &= \dots = \lambda_{15} = 12, & \lambda_{16} &= \dots = \lambda_{24} = 20, \dots \end{aligned} \quad (8)$$

The pattern is that there are $2\ell+1$ many modes with frequency $\lambda = \ell(\ell+1)$, $\ell = 0, 1, 2, \dots$. Replacing the unit sphere by the bunny, you get the “bunny harmonics” (Figure 1) with frequencies $\lambda \approx 0, 1.069, 2.847, 3.001, 3.702, 4.231, 6.117, 8.817, \dots$. Note that scaling the bunny will cause the eigenvalues to the inverse square of the scaling factor.

3. Solve the Poisson equation: given f solve for u

$$(-\Delta)u = f. \quad (9)$$

Again, in order to have a symmetric system matrix, we rearrange the Poisson equation as

$$\underbrace{d_0^\top \star_1 d_0}_{L} u = \underbrace{\star_0 f}_{\rho} \quad (10)$$

In electrostatics, u represents the electric potential and ρ represents the charge. The right-hand side ρ is a (dual) 2-form defined over points, whose values represent integrated quantities. That is $\rho(p)$ is the total charge at p , whereas $f(p)$ is the charge density per unit area. The total charge over the surface, written in terms of ρ , would be a straightforward sum $\sum_{p \in \mathcal{P}} \rho(p)$, contrasting a weighted sum $\sum_{p \in \mathcal{P}} A_p f(p)$ when written in terms of f .

- Pick some positive point charge and negative point charge by selecting some points using the Group node. (The cursor symbol next to the parameter “base group” can be helpful.) Directly assign the value of ρ on those point charges (there is no need to talk about f).

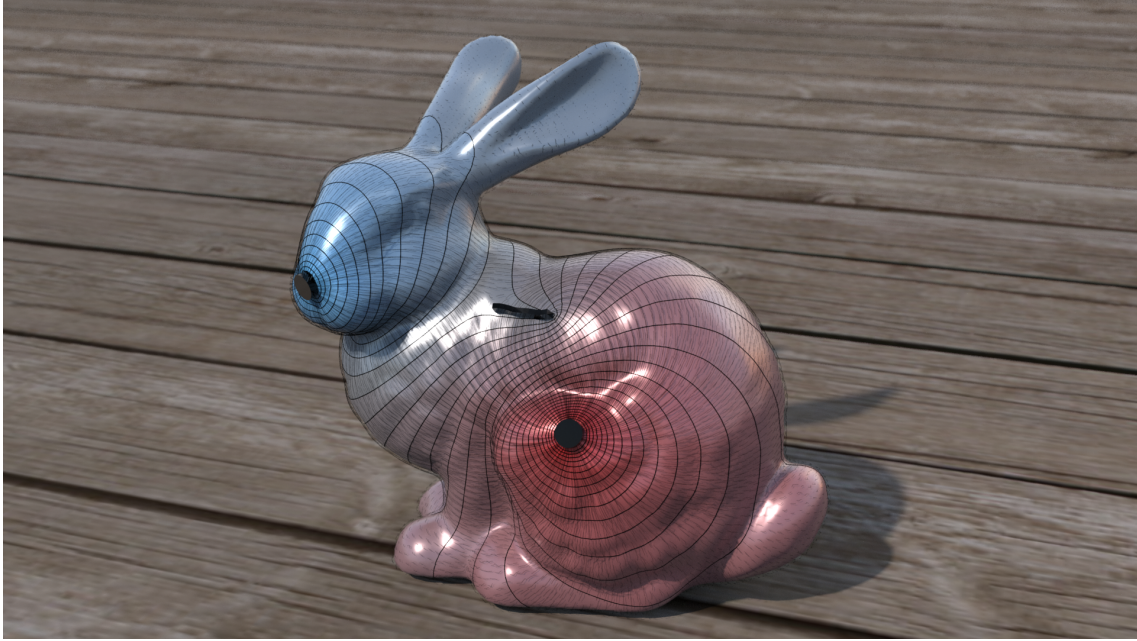


Figure 2 Electrostatics on the bunny.

- The *conjugate gradient (CG) method* is a standard numerical solver for the linear problem $Ax = b$ when A is symmetric positive definite. See `scipy.sparse.linalg.cg`.
- Note that the matrix L in our $Lu = \rho$ is only positive semi-definite, which will cause CG to crash. (There is one eigenvalue that is zero.) To make the matrix strictly positive-definite, you should add a small offset (e.g. $\varepsilon = 10^{-8}$) of the mass the L matrix

$$\tilde{L} = L + \varepsilon \star_0 . \quad (11)$$

- Since L has one null space, you want to make sure ρ is indeed in the image of L . This condition is that the total charge $\sum \rho$ must be zero. To ensure this condition, we ask you to remove the mean from the right-hand side of the equation $\tilde{\rho} = \rho - \bar{\rho}$.
4. Upload a render of the solution of the Poisson problem. You are welcomed to use your own mesh (be sure to include it) and polished rendering setup.