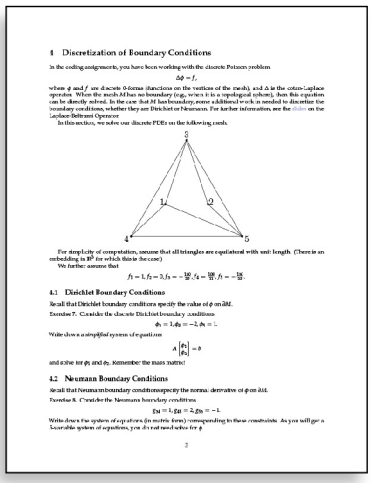
Discrete Differential Geometry
离散差分几何

CARNEGIE MELLON UNIVERSITY 15-458/768 | SPRING 2025 | TUE/THU, TIME TBD | ROOM TBD

Assignment 5: Geodesic Distance
作业 5：测地线距离

Written 写



1 Discretization of Boundary Conditions

In the coding assignments, you have been working with the discrete Poisson problem

$$\Delta\phi = f,$$

where ϕ and f are discrete 0-forms (functions on the vertices of the mesh), and Δ is the cotan-Laplace operator. When the mesh M has no boundary (e.g., when it is a topological sphere), then this equation can be directly solved. In the case that M has boundary, some additional work is needed to discretize the boundary conditions, whether they are Dirichlet or Neumann. For further information, see for instance the Notes on the Laplace-Beltrami Operator.

In this section, we solve our discrete PDEs on the following mesh.

For simplicity of computation, assume that all triangles are equilateral with unit length. (There is an embedding in $\mathbb{R}^2$ for which this is the case.)

We further assume that

$$f_1 = 1, f_2 = 2, f_3 = -\frac{19}{23}, f_4 = \frac{19}{23}, f_5 = -\frac{19}{23}.$$

4.1 Dirichlet Boundary Conditions

Recall that Dirichlet boundary conditions specify the value of ϕ on ∂M.

Exercise 7. Consider the discrete Dirichlet boundary conditions

$$\phi_1 = 1, \phi_2 = -2, \phi_5 = 1.$$

Write down a simplified system of equations

$$A \begin{bmatrix} \phi_3 \\ \phi_4 \end{bmatrix} = b$$

and solve for ϕ_3 and ϕ_4. Remember the mass matrix!

4.2 Neumann Boundary Conditions

Recall that Neumann boundary conditions specify the normal derivative of ϕ on ∂M.

Exercise 8. Consider the Neumann boundary conditions

$$\frac{\partial \phi}{\partial n}\Big|_1 = 1, \frac{\partial \phi}{\partial n}\Big|_2 = 2, \frac{\partial \phi}{\partial n}\Big|_5 = -1.$$

Write down the system of equations (in matrix form) corresponding to these constraints. As you will get a [illegible] system of equations, you do not need to solve for ϕ.

3

The written portion of this assignment can be found here. This time, we're taking off the "training wheels" and having you read a real paper, rather than course notes. Why? Because you're ready for it! At this point you have all the fundamental knowledge you need to go out into the broader literature and start implementing all sorts of algorithms that are built on top of ideas from differential geometry. In fact, this particular algorithm is not much of a departure from things you've done already: solving simple equations involving the Laplacian on triangle meshes. As discussed in our lecture on the Laplacian, you'll find many algorithms in digital geometry processing that have this flavor: compute some basic data (e.g., using a local formula at each vertex), solve a Laplace-like equation, compute some more basic data, and so on.

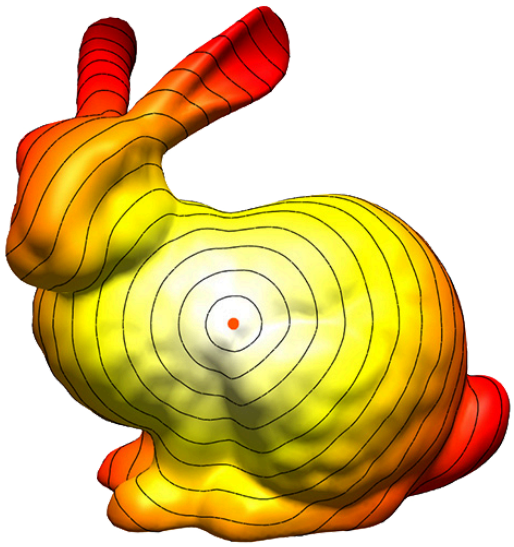
Your main references for this assignment will be:

此作业的书面部分可在此处找到。这一次，我们将取下 "训练轮"，让您阅读真正的论文，而不是课程笔记。为什么？因为您已经准备好了！此时，您拥有了深入研究更广泛的文献并开始实现建立在微分几何思想之上的各种算法所需的所有基础知识。事实上，这种特殊的算法与您已经做过的事情没有太大区别：在三角形网格上求解涉及拉普拉斯算子的简单方程。正如我们在拉普拉斯算子讲座中所讨论的，你会发现数字几何处理中有许多具有这种风格的算法：计算一些基本数据（例如，在每个顶点使用局部公式），求解类似拉普拉斯的方程，计算一些更基本的数据，等等。

您完成此作业的主要参考资料是：

- this video, which gives a brief (18-minute) overview of the algorithm, and
  此视频简要（18 分钟）概述了该算法，以及
- this paper, which explains the algorithm in detail.
  本文详细解释了该算法。

Coding 编码



For the coding portion of this assignment, you will implement the *heat method*, which is an algorithm for computing geodesic distance on curved surfaces. All of the details you need for implementation are described in Section 3 of the paper, up through and including Section 3.2. Note that you need only be concerned with the case of **triangle meshes** (not polygon meshes or point clouds); pay close attention to the paragraph labeled "Choice of Timestep."

Please implement the following routines in:

对于此作业的编码部分，您将实现 *heat 方法* ，这是一种用于计算曲面上的测地线距离的算法 。实现所需的所有细节都在本文的第 3 节中描述，直到第 3.2 节。请注意，您只需关注**三角形网格** （而不是多边形网格或点云）的情况;请密切注意标有 "Choice of Timestep" 的段落。

请在以下 中实现以下例程：

- `projects/geodesic-distances/heat-method.[js/cpp]`:
  - `constructor`
  - `computeVectorField`
  - `computeDivergence`
  - `compute`

Notes
笔记

- Refer to sections 3.2 of the paper for discretizations in Algorithm 1 (page 3).
  请参阅论文的第 3.2 节，了解算法 1 中的离散化（第 3 页）。
- Recall that our Laplace matrix is *positive* semidefinite, which might differ from the sign convention the authors use.
  回想一下，我们的拉普拉斯矩阵是*正*半定矩阵，这可能与作者使用的符号约定不同。
- The tests for `computeVectorField` and `computeDivergence` depend on the `A` and `F` matrices you define in your constructor. So if you fail the tests but

your functions look correct, check whether you have defined the flow and laplace matrices properly.

的测试 `computeVectorField` 和 `computeDivergence` 依赖于您在构造函数中定义的 `A` 和 `F` 矩阵。因此，如果你没有通过测试，但你的函数看起来正确，请检查你是否正确定义了 flow 和 laplace 矩阵。

- Your solution should implement zero Neumann boundary conditions (which are the "default behavior" of the cotan Laplacian) but feel free to tryout other Dirichlet and Neumann boundary conditions on your own.

  您的解决方案应实现零 Neumann 边界条件（这是 cotan Laplacian 的“默认行为”），但您可以自行尝试其他狄利克雷和诺依曼边界条件。

Handin instructions can be found in the Assignments section of the main page.

提交说明可以在主页的 Assignments 部分找到。

Carnegie Mellon University | 15-458/858B | Discrete Differential Geometry