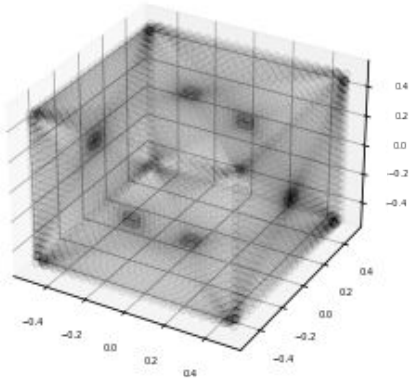
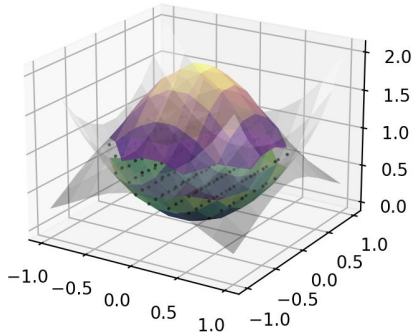
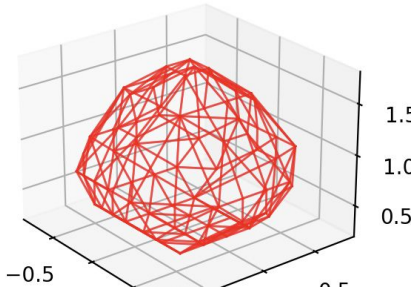
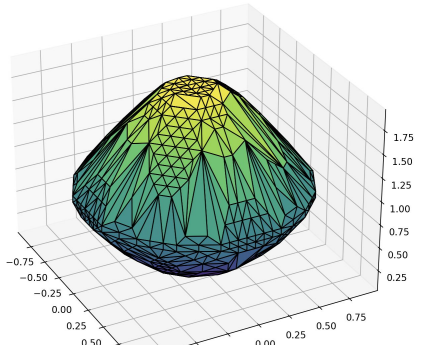


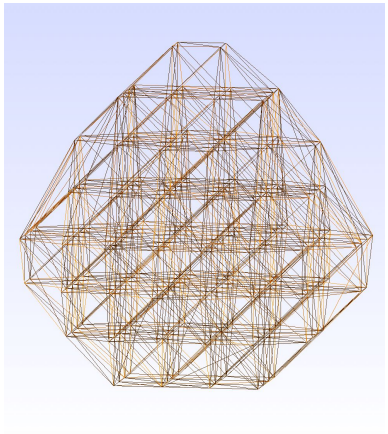


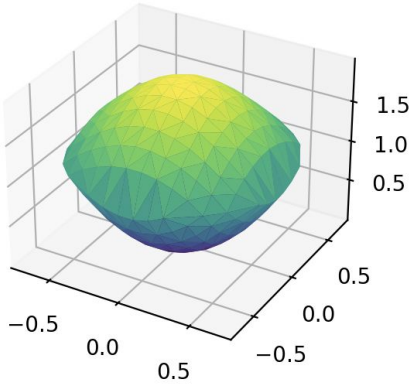


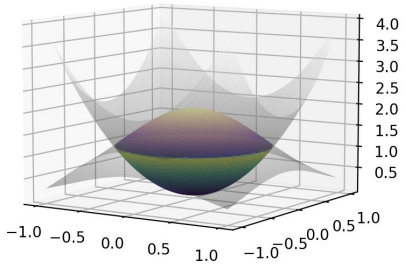


Mesh generation

Physics 129L







Zihang Wang

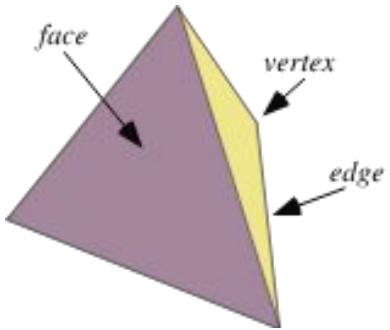
01/14/2025

Geometric definitions

A **point** is the most basic unit in geometry, representing an exact location in space.

A **vertex** is a point where two or more edges, lines, or curves meet.

An **edge** is a straight line segment connecting two vertices.



https://mathworld.wolfram.com/Face.html

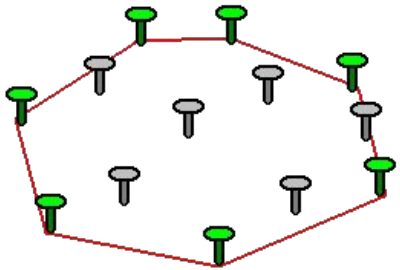
A closed area formed by edges is a **facet**

https://mathworld.wolfram.com/ConvexHull.html

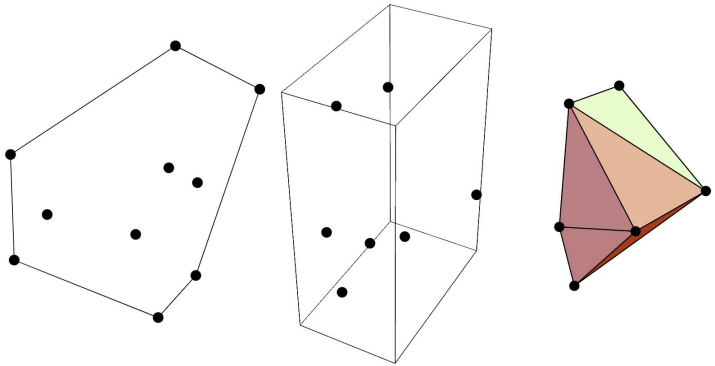
Convex Hull

The convex hull is defined as the smallest convex set that encloses a given set of points. It can be visualized as the shape formed by stretching a rubber band around the outermost points of a dataset in 2D or wrapping plastic around points in 3D space.

It defines the boundary for a given set of points



https://brilliant.org/wiki/convex-hull/



https://mathworld.wolfram.com/ConvexHull.html

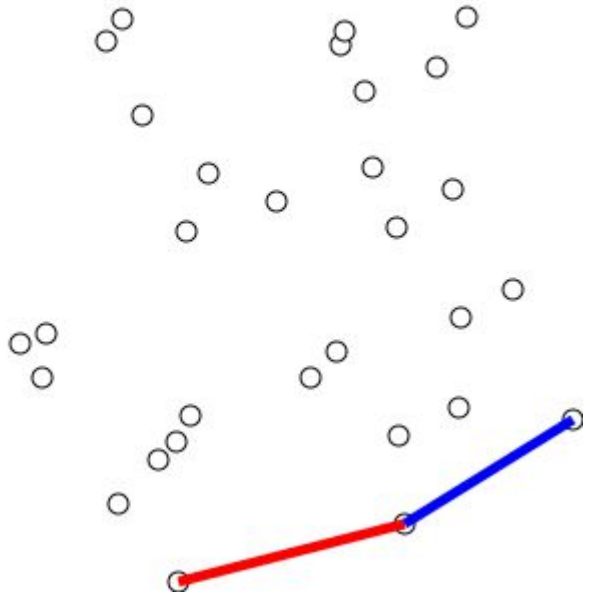
Graham Scan

It is an efficient algorithm used to compute the convex hull of a set of points in a 2D plane.

1) Let's start with a point that has the smallest y coordinates (or lowest left corner).

2) Then, we sort the angle formed from the positive x-axis direction (small to large). Let's call this sorted array (**Do we need to calculate the angle?**)

3) If two points have the same angle, keep only the farthest one from the starting point.



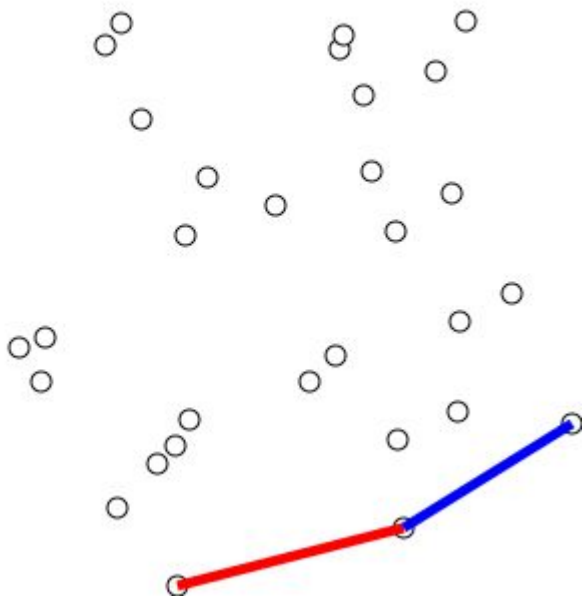
https://en.wikipedia.org/wiki/Graham_scan#/media/File:GrahamScanDemo.gif

Graham Scan

Let’s prepare a new list. The starting point and the first two points will be put into the list.

We go through the sorted array we prepared previously (small to large).

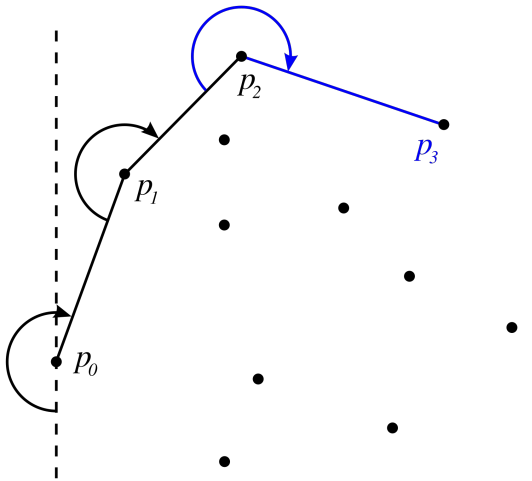
If adding a new point creates a clockwise turn with the last two points on the list, remove the last point (the one prior to the new added point) from the list.



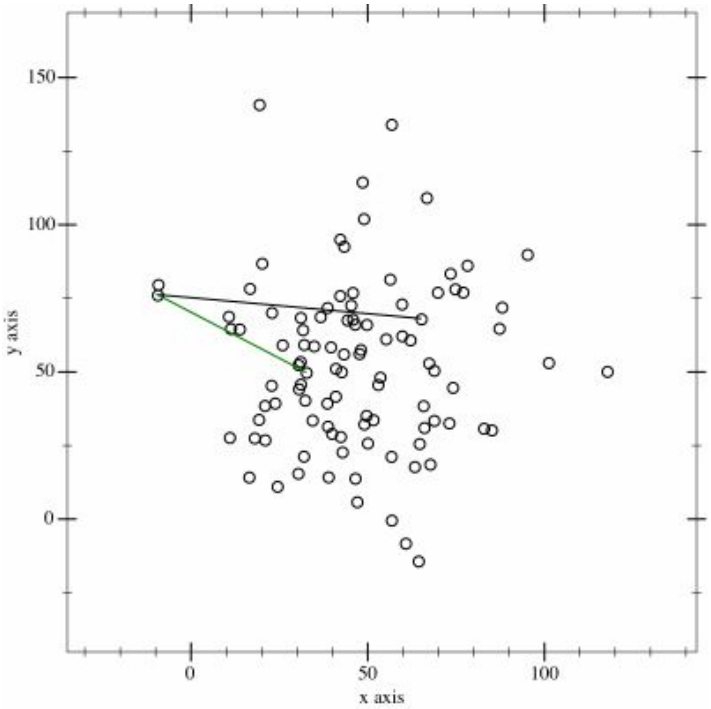
https://en.wikipedia.org/wiki/Graham_scan#/media/File:GrahamScanDemo.gif

Jarvis march

Let’s start with the leftmost point, and selects the next point such that all points are to the right of the line.



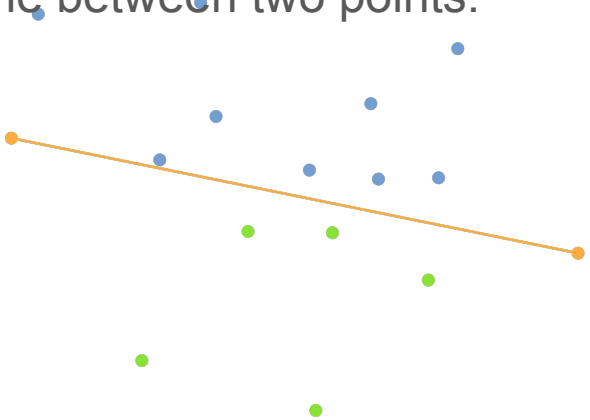
https://en.wikipedia.org/wiki/Gift_wrapping_algorithm

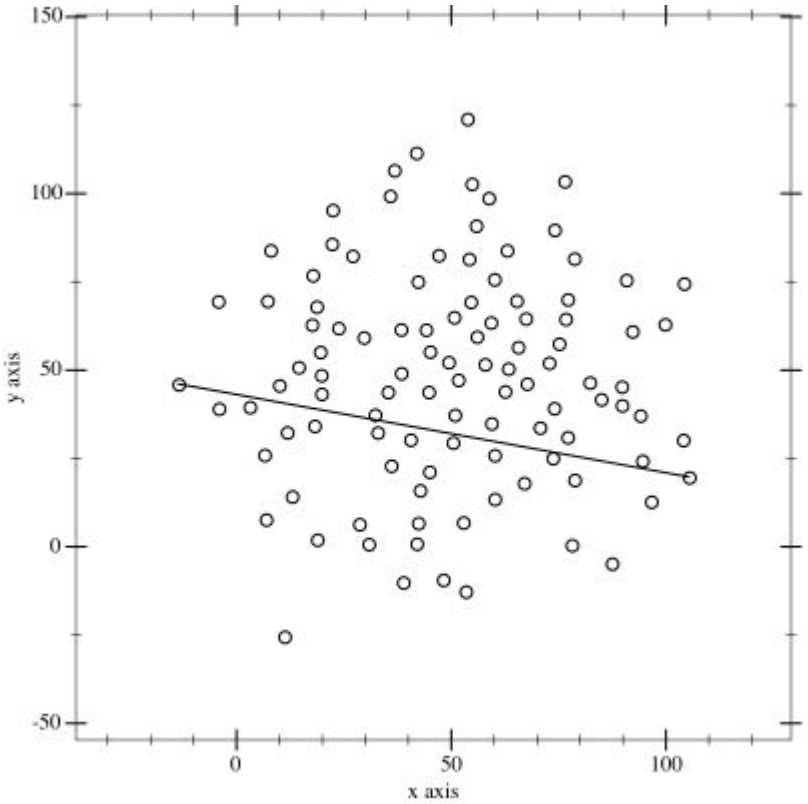


Quickhull (Divide & conquer)

The concept of divide and conquer is very important in computation, where a large system is divided into smaller pieces.

Let’s divide a point cloud by drawing a line between two points.

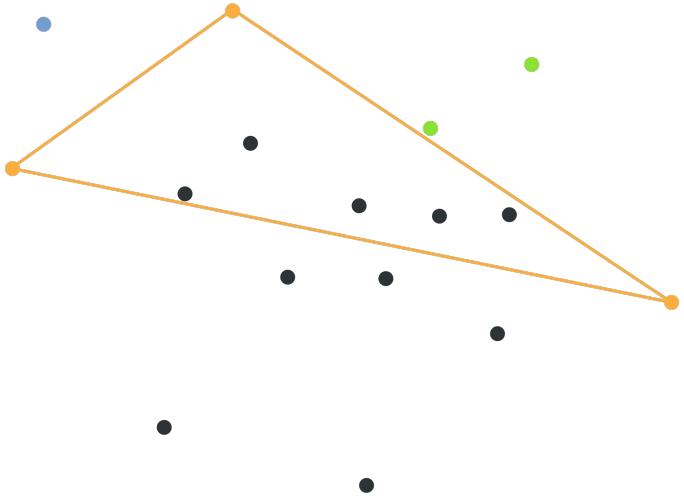


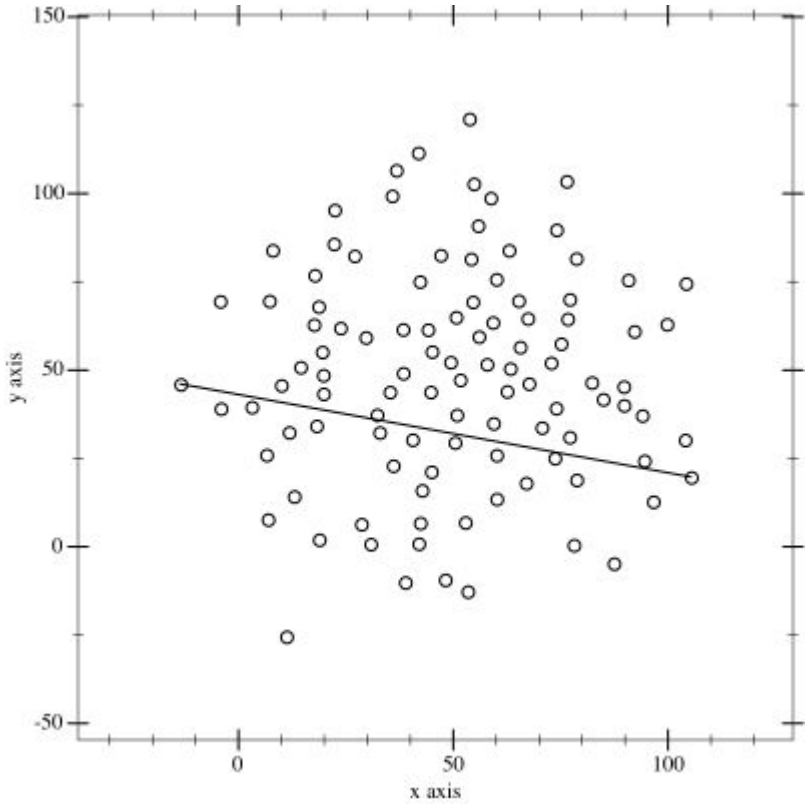


https://en.wikipedia.org/wiki/Quickhull

Quickhull (Divide & conquer)

Then, we look for a new point such that the formed triangle maximizes its area.





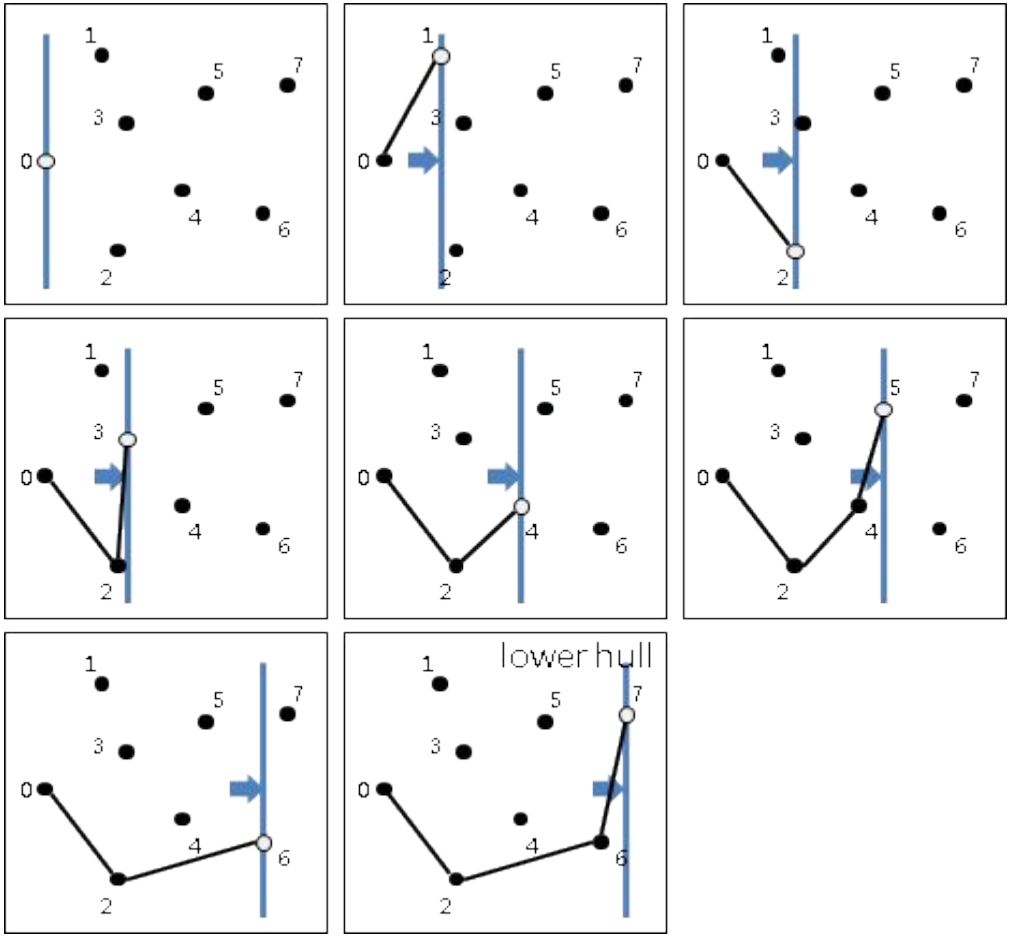
https://en.wikipedia.org/wiki/Quickhull

Monotone chain

We first **sort the points** by their x coordinates, and scan from left to right.

While the last three points in the list make a **negative angle** with respect to the **positive axis**, remove the second-to-last point (middle).

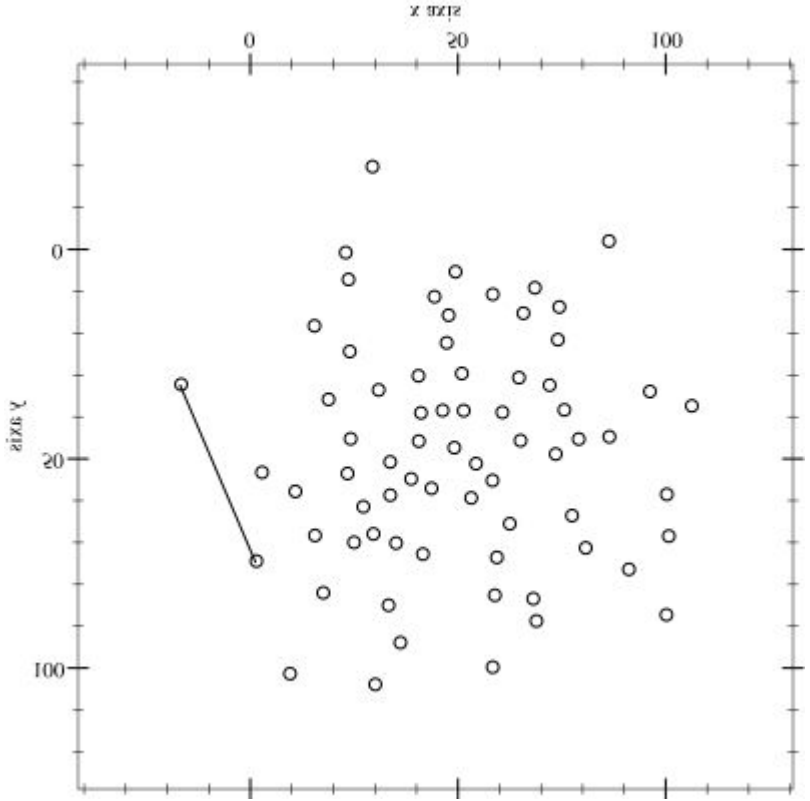
This step ensures the hull remains convex, and we stop until it gets to the far right point.



https://web.ntnu.edu.tw/~algo/ConvexHull.html

Monotone chain

We then going backwards by scanning from right to left. Now when the last three points in the list make a **positive angle** with respect to the **negative axis**, remove the second-to-last point (middle).



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

Chan’s Algorithm

Chan’s algorithm contains two phases.

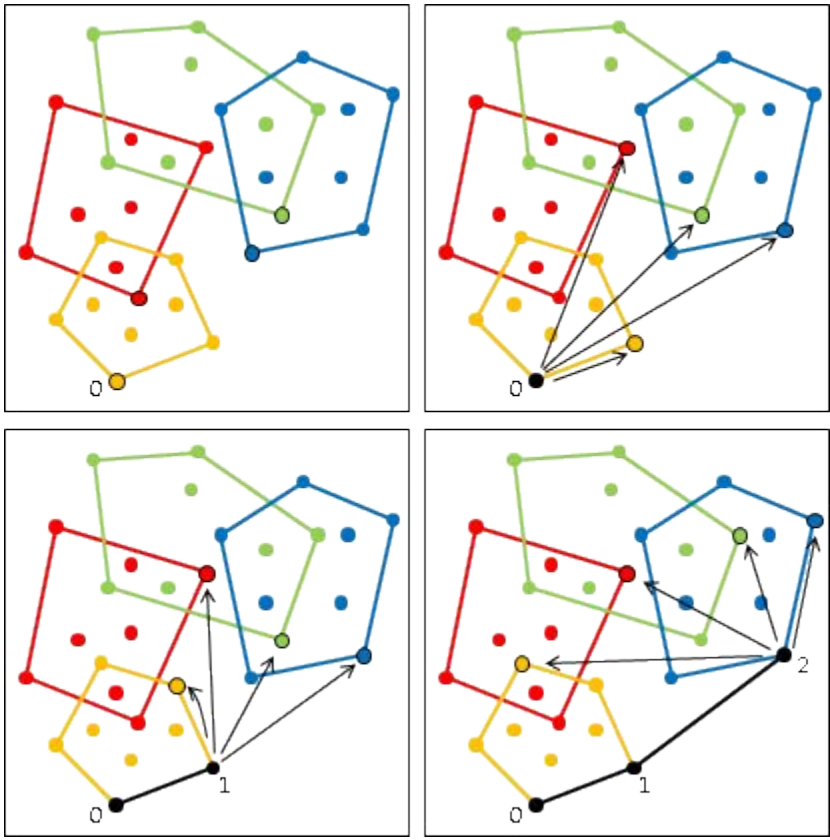
Phase 1: Partition with a guess m.

It divides the point cloud with total n points into R point clouds, and each cloud contains at most m points, such that n~mR.

Phase 2: Graham's scan

Compute the convex hull for each subset using Graham's scan.

Those are called **sub hulls**



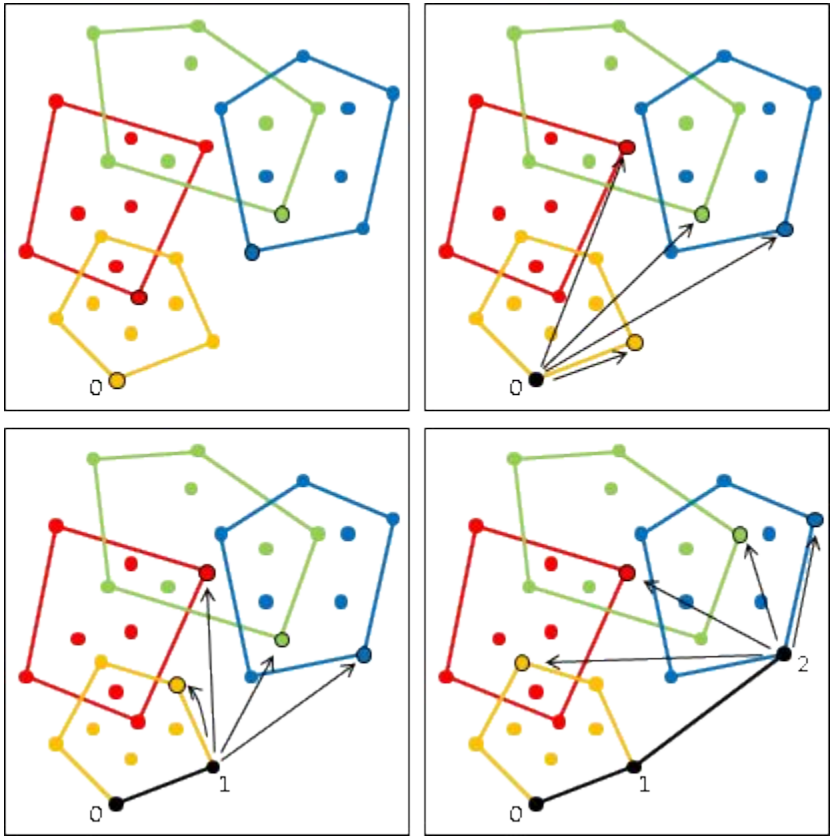
https://en.wikipedia.org/wiki/Quickhull

Chan’s Algorithm

Chan’s algorithm contains two phases.

Phase 3: Jarvis march

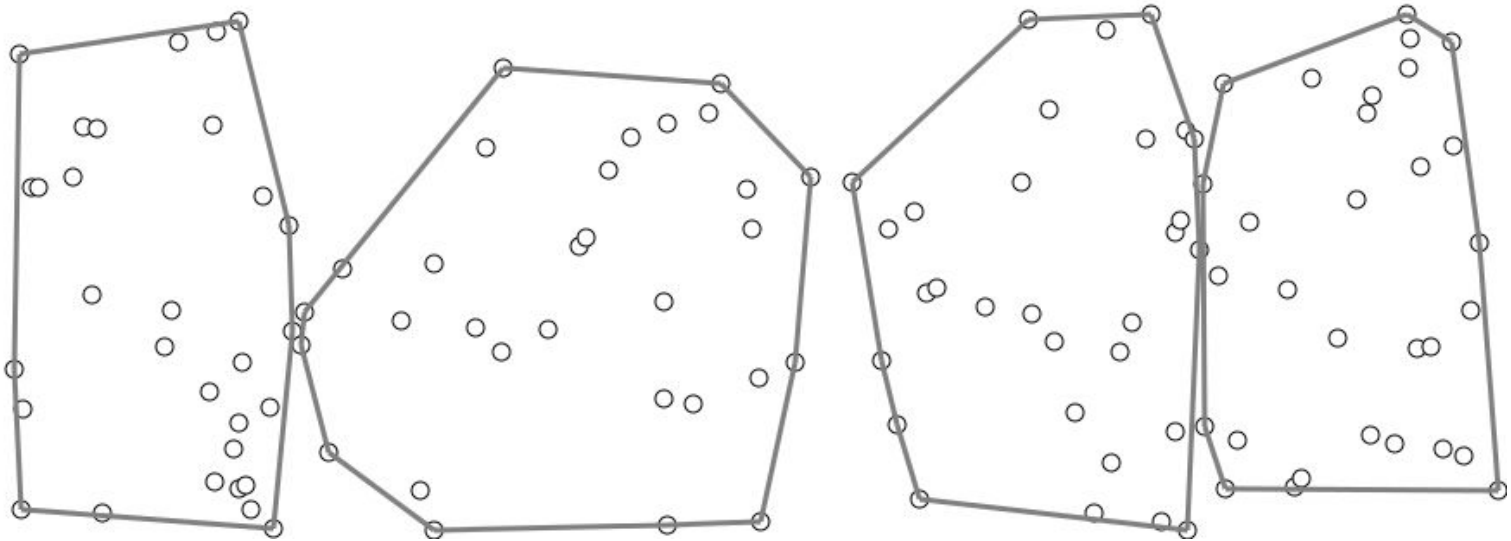
Let’s start with a point on one of the sub hulls. The next point is determined by Jarvis march algorithm, where the point clouds are all on all sub hulls.



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

Chan’s Algorithm

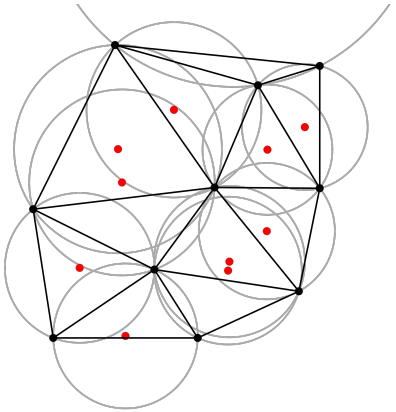
Phase 4. Repeat: m is initially guessed is doubled and the same process starts again.



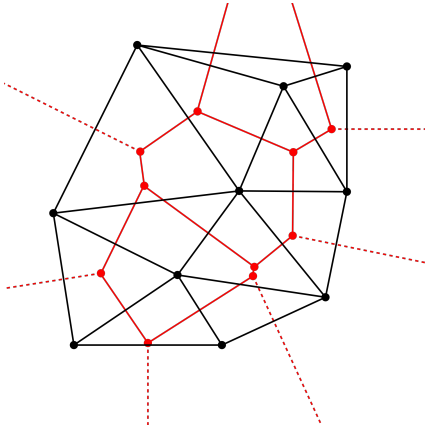
https://en.wikipedia.org/wiki/Quickhull

Delaunay triangulation

Set of points in the plane subdivides **convex hull** into triangles circumcircles that do not contain any vertex.



Delaunay triangulation



Voronoi diagram

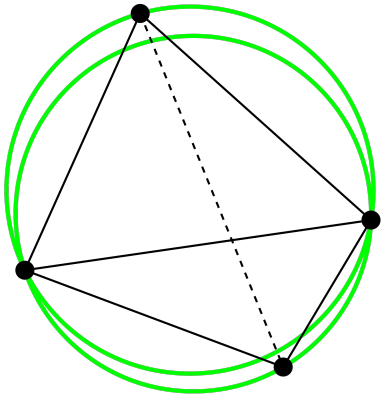
It is dual to the voronoi diagram (geo center of the circumcircles). This duality has great impact in error corrections (look at surface code, dual lattice, if you are interested in quantum error correction)

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

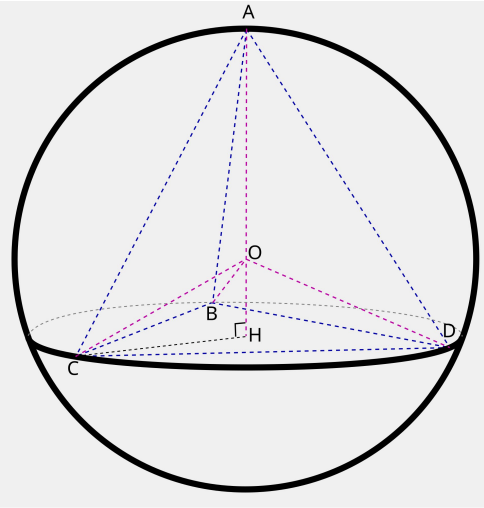
Delaunay triangulation

You might notice the triangle, it has a different name called **Simplex.** It is a generalization of the notion of a triangle to arbitrary dimensions.

For example, in 3D, the simplex is a Tetrahedron.



Delaunay triangulation



Voronoi diagram

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

Delaunay triangulation

The parabolic lifting map sends each point $p = (x, y) \in \mathbb{R}^2$ to a point $p^+ = (x, y, x^2 + y^2) \in \mathbb{R}^3$. Call p^+ the *lifted companion* of p.

In other words, we are allow to perform coordinate transformations such that the triangulation is preserved. (**What are the conditions of this mapping?**)

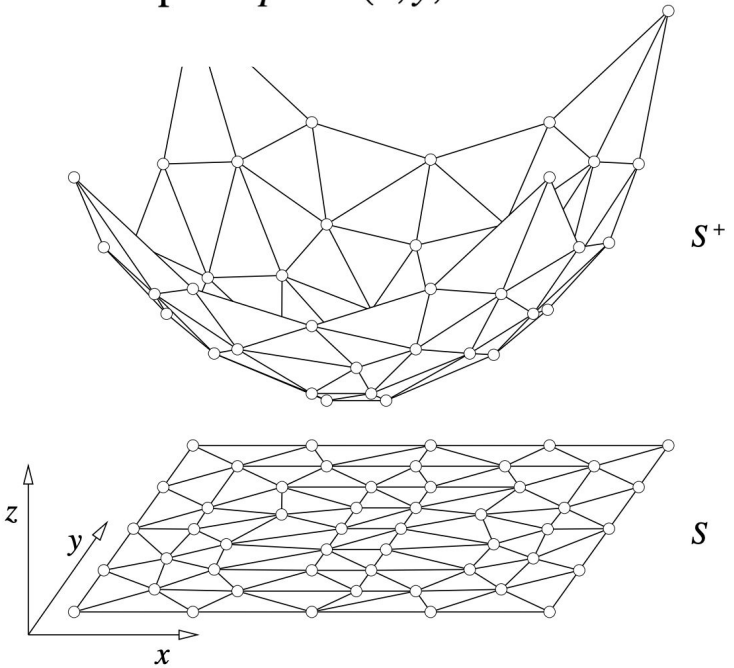


Figure 2.5: The parabolic lifting map.

https://people.eecs.berkeley.edu/~jrs/papers/meshbook/chapter2.pdf

Gradient and covariant derivatives

The derivative of a vector in a given coordinate system has the following form,

$$\partial_j \mathbf{n} = (\partial_j n^i)\hat{\mathbf{e}}_i + n^i \partial_j \hat{\mathbf{e}}_i,$$

The term $\partial_j \hat{\mathbf{e}}_i$ can be expressed via Christoffel symbols $\partial_j \hat{\mathbf{e}}_k = \Gamma^i_{jk}\hat{\mathbf{e}}_i$,

$$\partial_j \mathbf{n} = \partial_j n^i \hat{\mathbf{e}}_i + n^i \partial_j \hat{\mathbf{e}}_i = \partial_j n^i \hat{\mathbf{e}}_i + n^k \Gamma^i_{jk}\hat{\mathbf{e}}_i = (\partial_j n^i + n^k \Gamma^i_{jk})\hat{\mathbf{e}}_i = (\nabla_j n^i)\hat{\mathbf{e}}_i.$$

Where we have the Christoffel symbol,

$$\Gamma^m_{ij} = \frac{1}{2} g^{km} \left(\partial_i g_{jk} + \partial_j g_{ki} - \partial_k g_{ij}\right).$$

Gradient and covariant derivatives

Covariant derivatives can be written as the derivative in the ambient space projected onto the surface,

One can define the surface derivative ∇_s on an embedding in $\mathbb{R}^3$,

$$g(x, y, z) = f(x, y) - z = 0 \tag{42}$$

The unit outward normal is given by the gradient,

$$\mathbf{N} = -\frac{\nabla g}{|\nabla g|} = -\frac{[\partial_x f \hat{\mathbf{x}} + \partial_y f \hat{\mathbf{y}} - \hat{\mathbf{z}}]}{\sqrt{\partial_x f^2 + \partial_y f^2 + 1}}. \tag{43}$$

The projection operator,

$$\mathbf{P} = \mathbf{I} - \mathbf{N}\mathbf{N}^T = \mathbf{I} - \frac{1}{\partial_x f^2 + \partial_y f^2 + 1} \begin{bmatrix} \partial_x f^2 & \partial_x f \partial_y f & -\partial_x f \\ \partial_x f \partial_y f & \partial_y f^2 & -\partial_y f \\ -\partial_x f & -\partial_y f & 1 \end{bmatrix}, \tag{44}$$

Gradient and covariant derivatives

When it acts on a vector field, it becomes a rank-2 tensor,

$$\begin{aligned}\nabla_s\left(\boldsymbol{\phi}_{\boldsymbol{x}}\hat{\mathbf{x}}\right) = P_{ij}\partial_j\boldsymbol{\phi}_{\boldsymbol{x}} = &\left[\partial_x\boldsymbol{\phi}_{\boldsymbol{x}} - \frac{\partial_x f^2\partial_x\boldsymbol{\phi}_{\boldsymbol{x}} + \partial_x f\partial_y f\partial_y\boldsymbol{\phi}_{\boldsymbol{x}} - \partial_x f\partial_z\boldsymbol{\phi}_{\boldsymbol{x}}}{\partial_x f^2 + \partial_y f^2 + 1}\right]\hat{\mathbf{x}}\otimes\hat{\mathbf{x}} \\ &+ \left[\partial_y\boldsymbol{\phi}_{\boldsymbol{x}} - \frac{\partial_y f^2\partial_y\boldsymbol{\phi}_{\boldsymbol{x}} + \partial_x f\partial_y f\partial_x\boldsymbol{\phi}_{\boldsymbol{x}} - \partial_y f\partial_z\boldsymbol{\phi}_{\boldsymbol{x}}}{\partial_x f^2 + \partial_y f^2 + 1}\right]\hat{\mathbf{y}}\otimes\hat{\mathbf{x}} \\ &+ \left[\partial_z\boldsymbol{\phi}_{\boldsymbol{x}} - \frac{-\partial_y f\partial_y\boldsymbol{\phi}_{\boldsymbol{x}} - \partial_x f\partial_x\boldsymbol{\phi}_{\boldsymbol{x}} + \partial_z\boldsymbol{\phi}_{\boldsymbol{x}}}{\partial_x f^2 + \partial_y f^2 + 1}\right]\hat{\mathbf{z}}\otimes\hat{\mathbf{x}}.\end{aligned} \tag{46}$$

Ambient space and first and second fundamental form

When studying a curve or surface, the ambient space is a 2D plane or 3D space in which those objects are placed.

Intrinsic: first fundamental form captures the variation with respect to the tangents.

$$\mathrm{I} = ds^2 = [\frac{\partial \mathbf{r}}{\partial u} \cdot \frac{\partial \mathbf{r}}{\partial v}]dudv = g_{ij}dudv,$$

Extrinsic: second fundamental form: captures the variation with respect to local orthogonal deformations

$$\mathrm{II} = [(\partial_u^2 \mathbf{r}) \cdot \mathbf{N}du^2 + (\partial_v^2 \mathbf{r}) \cdot \mathbf{N}dv^2 + 2(\partial_u \partial_v \mathbf{r}) \cdot \mathbf{n}dudv]$$

Shape operator

Shape operator captures the variations in normal vectors when traveling on a surface.

$$\mathcal{S}(u,v) = \begin{pmatrix} (\partial_u \mathbf{r})^2 & (\partial_u \mathbf{r}) \cdot (\partial_v \mathbf{r}) \\ (\partial_u \mathbf{r}) \cdot (\partial_v \mathbf{r}) & (\partial_v \mathbf{r})^2 \end{pmatrix}^{-1} \begin{pmatrix} (\partial_u^2 \mathbf{r}) \cdot \mathbf{N} & (\partial_u \partial_v \mathbf{r}) \cdot \mathbf{N} \\ (\partial_u \partial_v \mathbf{r}) \cdot \mathbf{N} & (\partial_v^2 \mathbf{r}) \cdot \mathbf{N}. \end{pmatrix}$$

The **principle curvatures** at (u, v) are the eigenvalues of the shape operator and the principal directions are the eigenvectors.

The **Gaussian curvature** K is the product of the principal curvatures, or the determinant of the shape operator.

The **mean curvature** H is the average of the principal curvatures, or the trace of the shape operator.

Geodesics

Let’s consider a length functional with the parameterization t,

$$E[\gamma(t)] = \int \sqrt{ds^2} = \int \sqrt{g_{ij} dx^i dx^j} = \int dt \sqrt{g_{ij} \frac{dx^i}{dt} \frac{dx^j}{dt}}$$

Let’s vary the path a bit, and find the stationary condition: the euler lagrange equation

$$\frac{d^2\gamma^i}{dt^2} + \Gamma^i_{uj} \frac{d\gamma^u}{dt} \frac{d\gamma^j}{dt} = 0$$

Parallel Transport

Let’s consider a length functional with the parameterization t,

$$\frac{\partial}{\partial t}\mathbf{n} = \frac{\partial \gamma}{\partial t} \cdot \frac{\partial}{\partial \gamma}\mathbf{n} = \frac{\partial \gamma^j}{\partial t}[\partial_j n^i + n^k \Gamma^i_{jk}]\hat{\mathbf{e}}_\mathbf{i} = [\frac{\partial n^i}{\partial t} + n^k \Gamma^i_{jk}\frac{\partial \gamma^j}{\partial t}]\hat{\mathbf{e}}_\mathbf{i} = 0.$$

The change of the vector with respect to the parameterization is zero