UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

> Reinventa el mundo <

Computer Graphics

Class 22. Geometry Processing.

Professor: Eric Biagioli

TRANSFORMATEC

Today

- Surface representations.
- Marching squares. Marching triangles. Marching cubes.
- OFF and PLY formats.
- Mesh Data Structures.
- Subdivision surfaces.
- Catmull-Clark algorithm.

References for the class of today: (part of the second partial exam)

- Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Lévy, B. *Polygon Mesh Processing*. A K Peters, 2010. → **CHAPTERS 1 (Surface representations), 2 (Mesh Data Structures) and 3 (Differential Geometry).**

- Hughes, J. F., van Dam, A., McGuire, M., Sklar, D. F., Foley, J. D., Feiner, S., and Akeley, K. *Computer Graphics: Principles and Practice, 3 ed*. Addison-Wesley, Upper Saddle River, NJ, 2013. → **CHAPTER 8 (A Simple Way to Describe Shape in 2D and 3D), Section 23.3 (Catmull-Clark Subdivision Surfaces), CHAPTER 24 (Implicit Representations of Shape), section 24.1 to 24.7 (included).**

Surface representations

- Implicit representations
- Parametric representations
  - Bezier
  - Splines
  - NURBS (Non-Uniform-Rational-B-Splines)
  - …
- Meshes of polygons, cloud of points
- …

Surface representations

- **Implicit representations**

- Parametric representations
  - Bezier
  - Splines
  - NURBS (Non-Uniform-Rational-B-Splines)
  - **Meshes of polygons**, cloud of points

- …

Implicit representations of shape

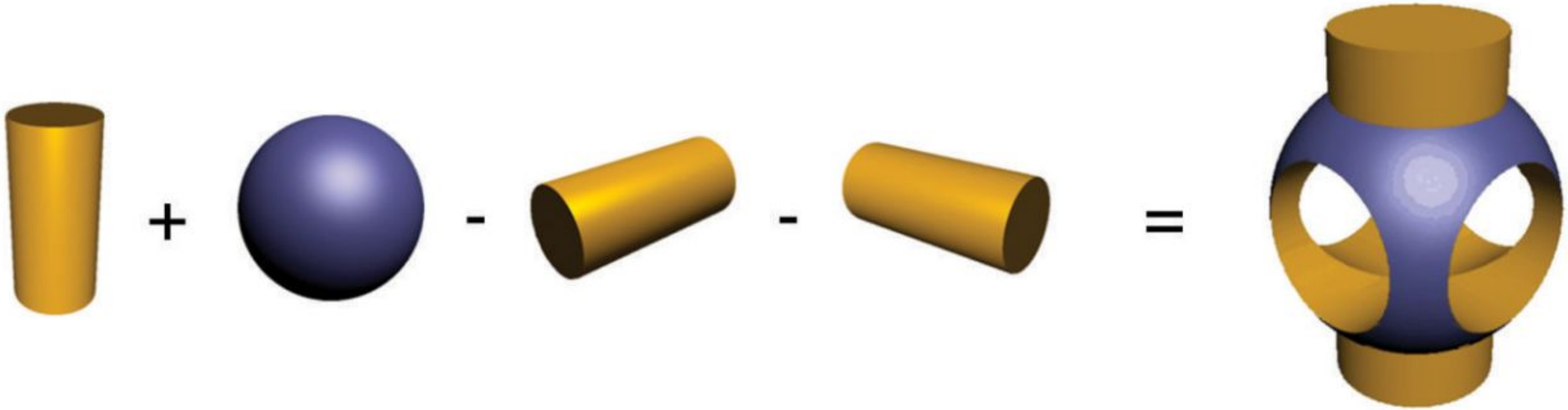
f: *implicit function*

f(x, y, z) = 0 → (x, y, z) on the surface
f(x, y, z) < 0 → (x, y, z) *inside.*
f(x, y, z) > 0 → (x, y, z) *outside.*

Examples:
f(x, y, z) = x^2 + y^2 + z^2 - R^2 → Sphere of radius R centered at (0, 0, 0)

f(x, y) = 0.004 + 0.110 * x - 0.177 * y - 0.174 * x^2 + 0.224 * x * y - 0.303 * y^2
- 0.168 * x^3 + 0.327 * x^2 * y - 0.087 * x * y^2 - 0.013 * y^3
+ 0.235 * x^4 - 0.667 * x^3 * y + 0.745 * x^2 * y^2 - 0.029 * x * y^3 + 0.072 * y^4;

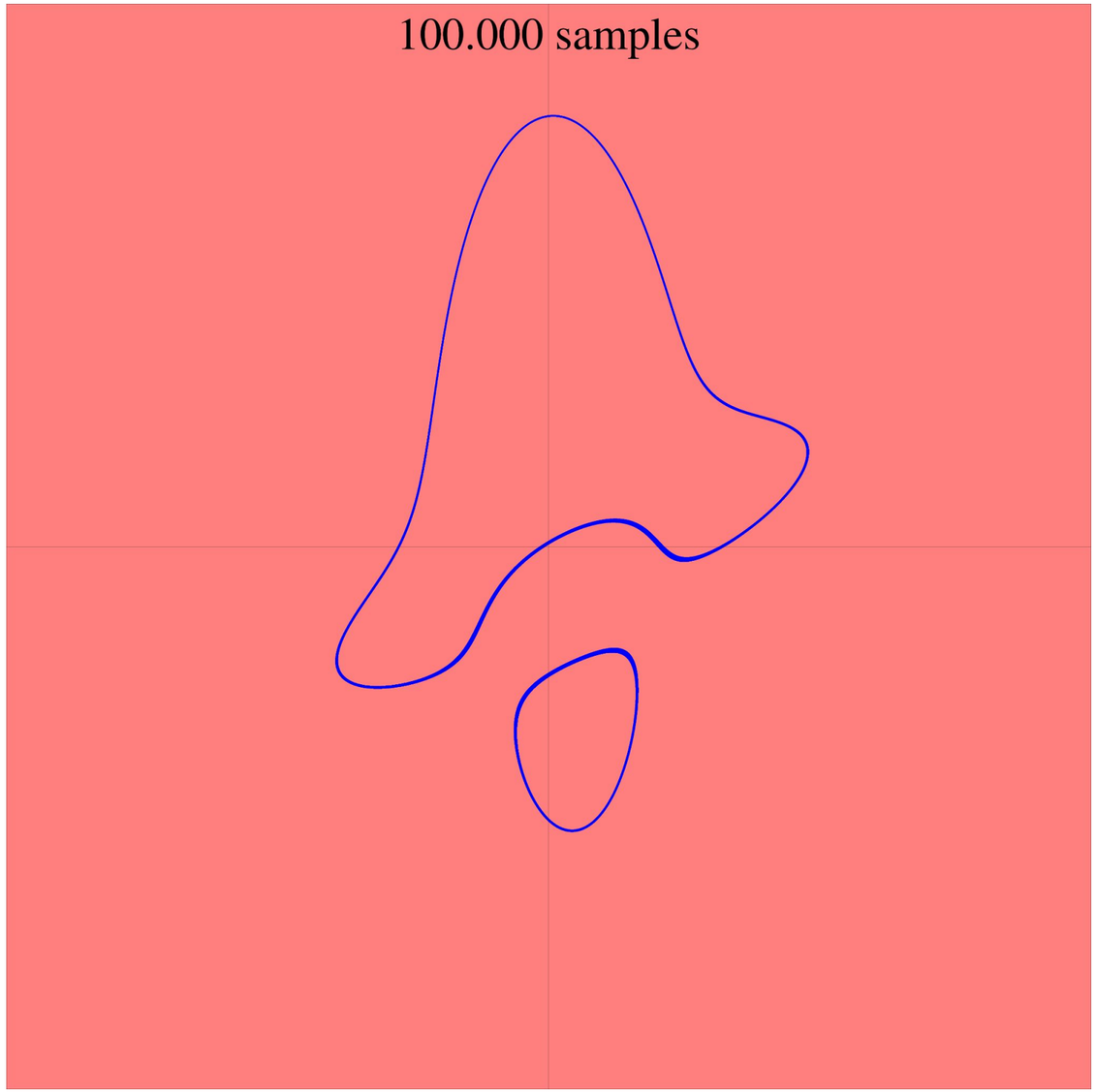
Implicit representations of shape



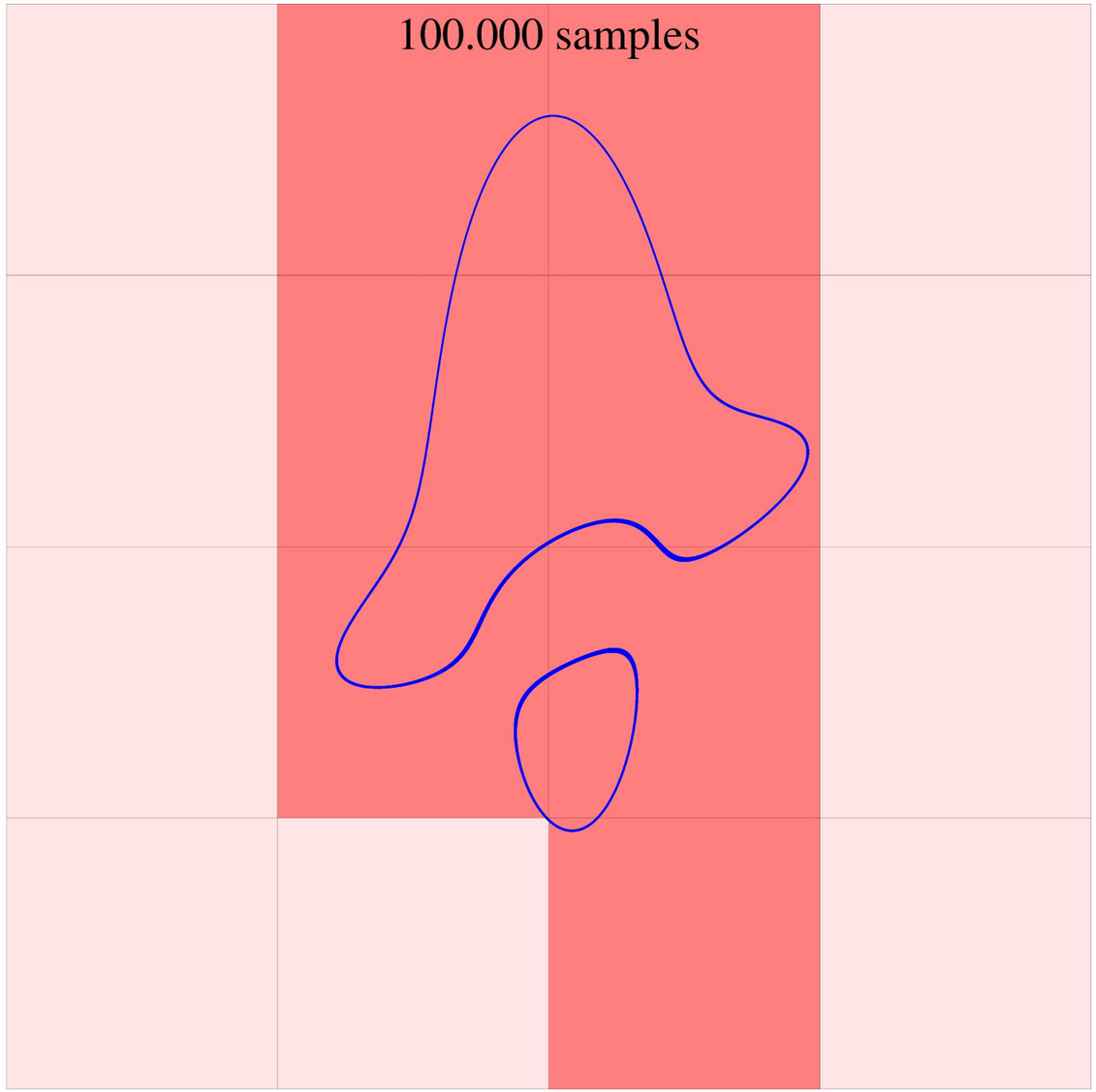
How can we draw an implicit function?

- Randomly collect N samples of a cell, and decide if it is "inside", "outside", or "both".
  - Adaptative: continue recursively where it might be needed
  - Regular: sample all the cells

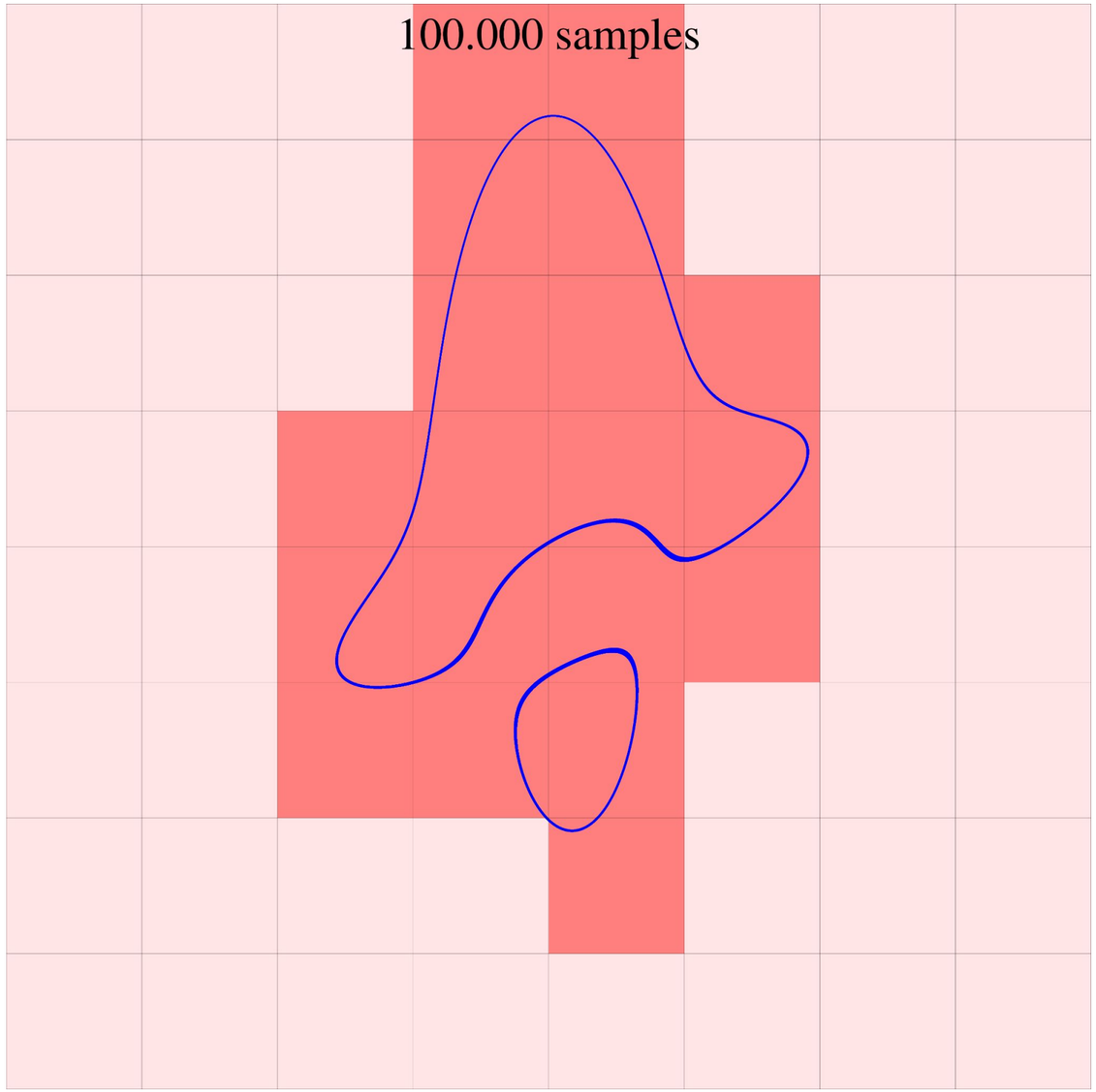
How can we draw an implicit function?



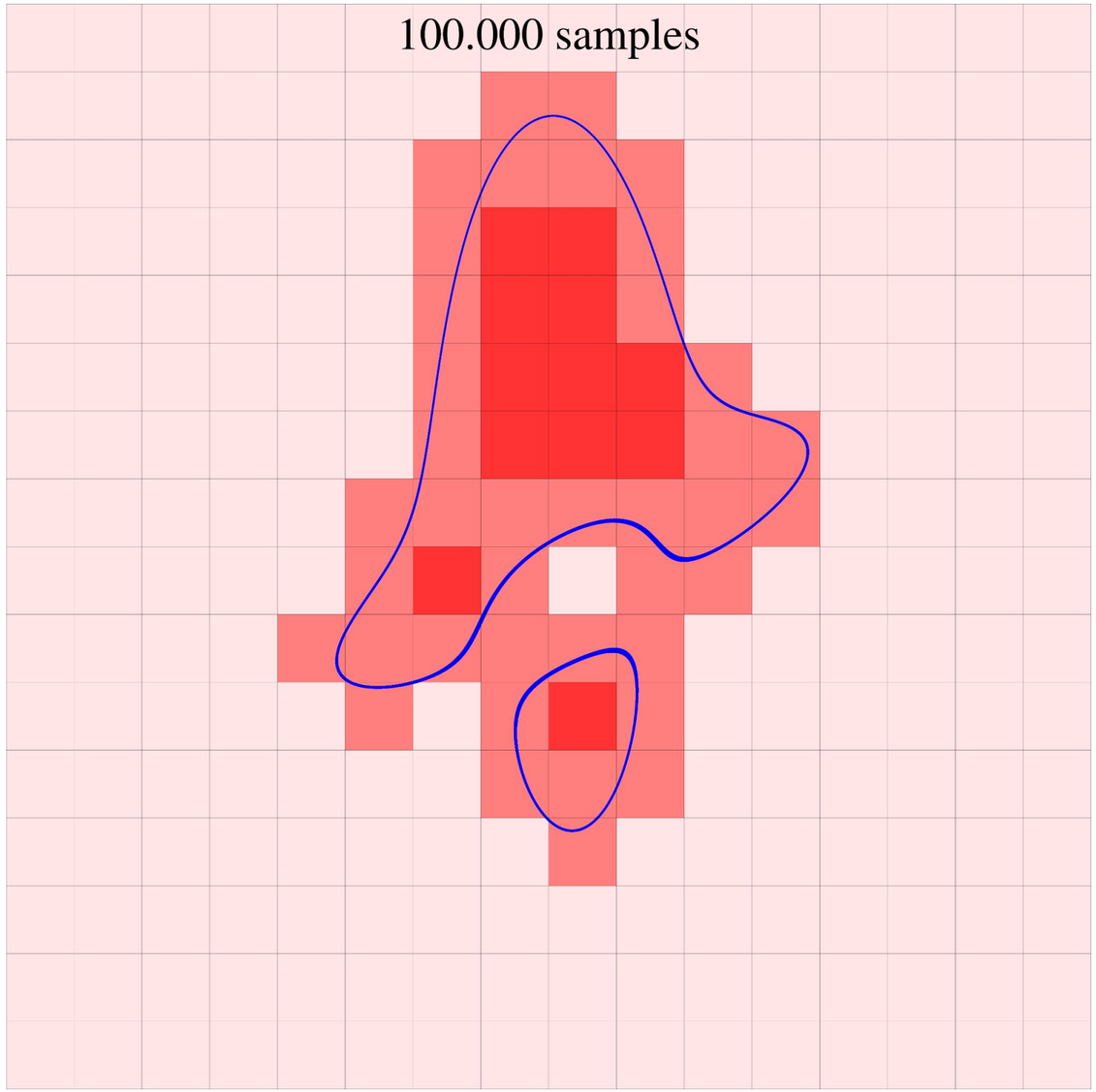
How can we draw an implicit function?



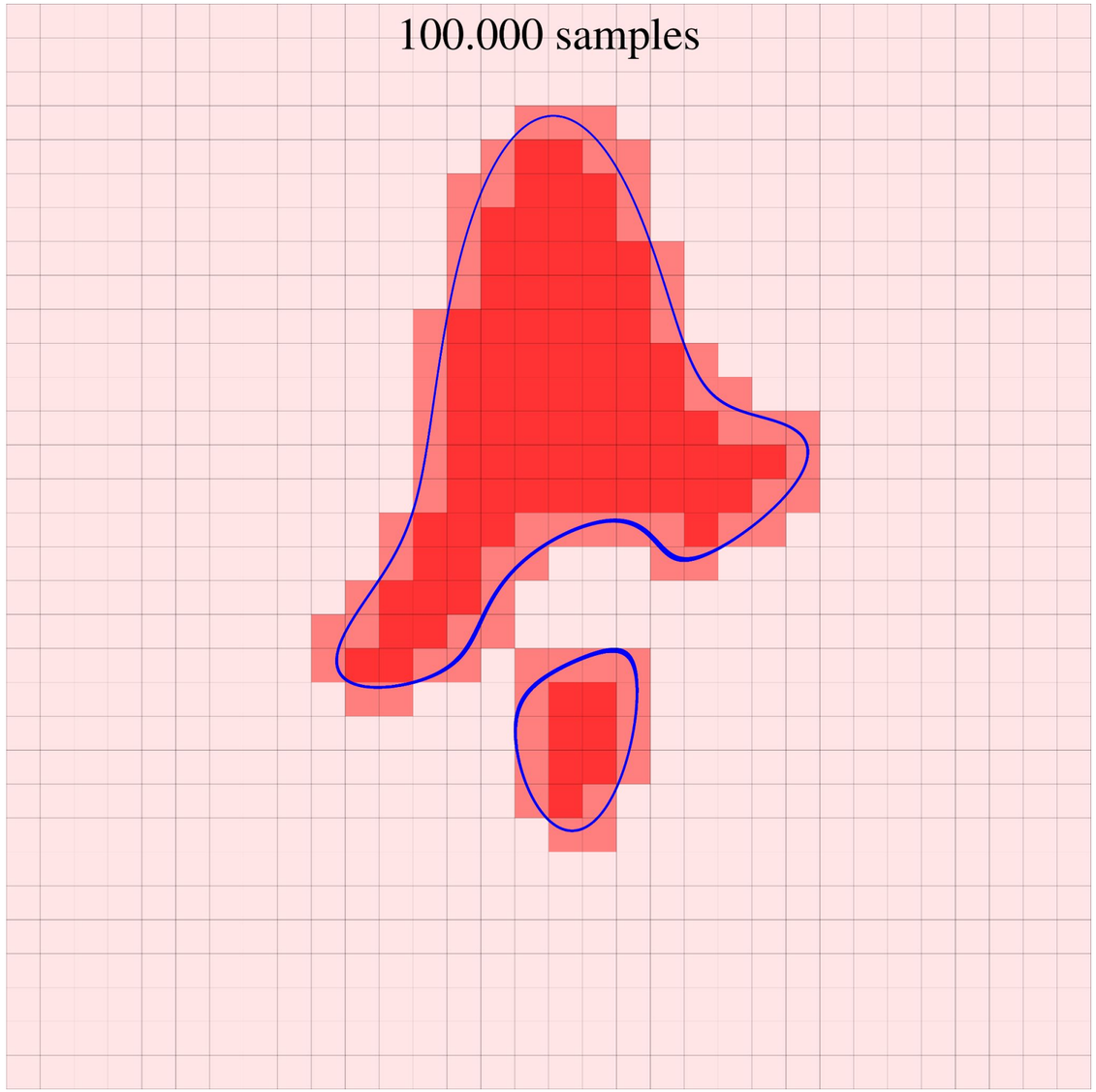
How can we draw an implicit function?



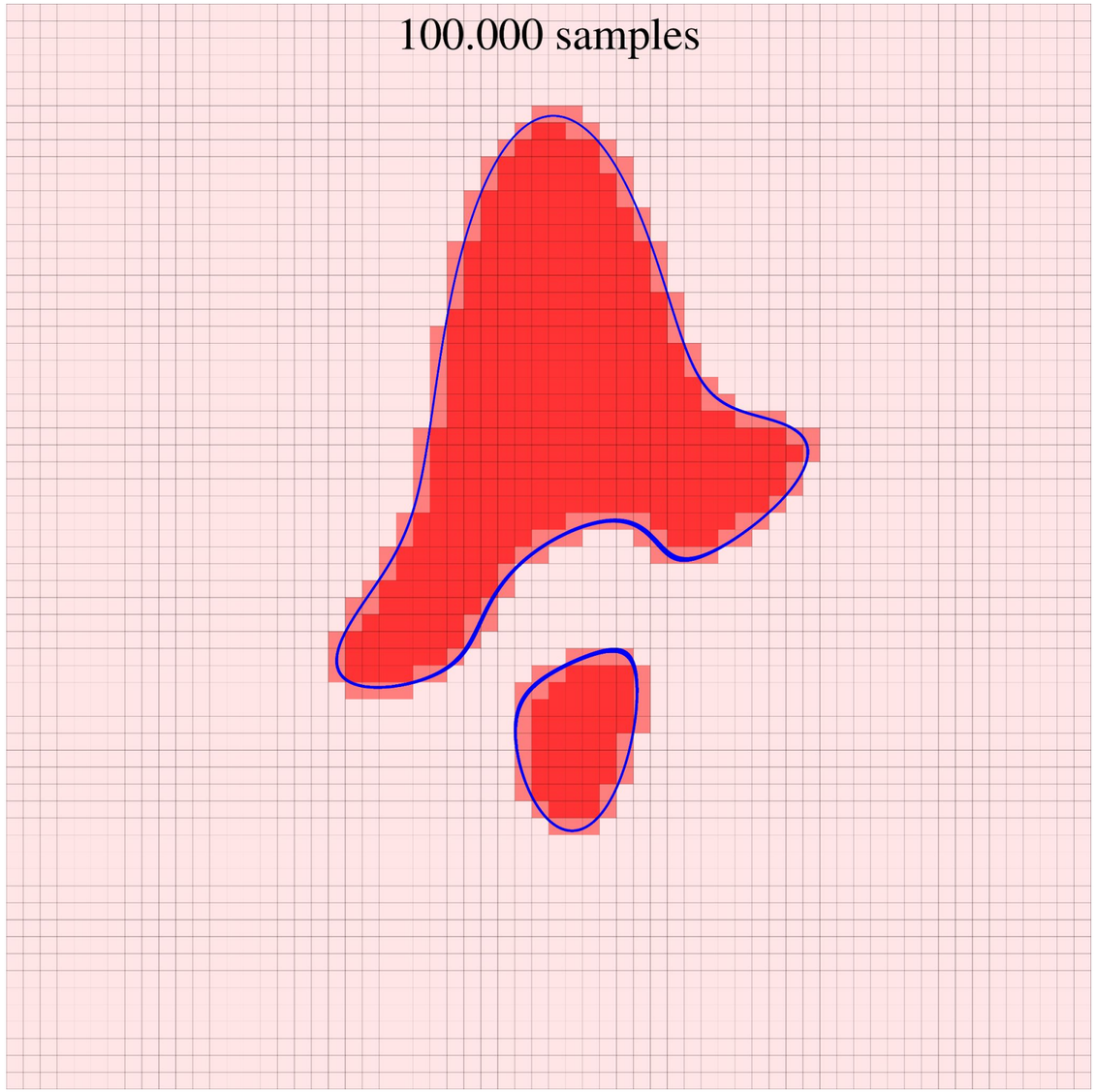
How can we draw an implicit function?



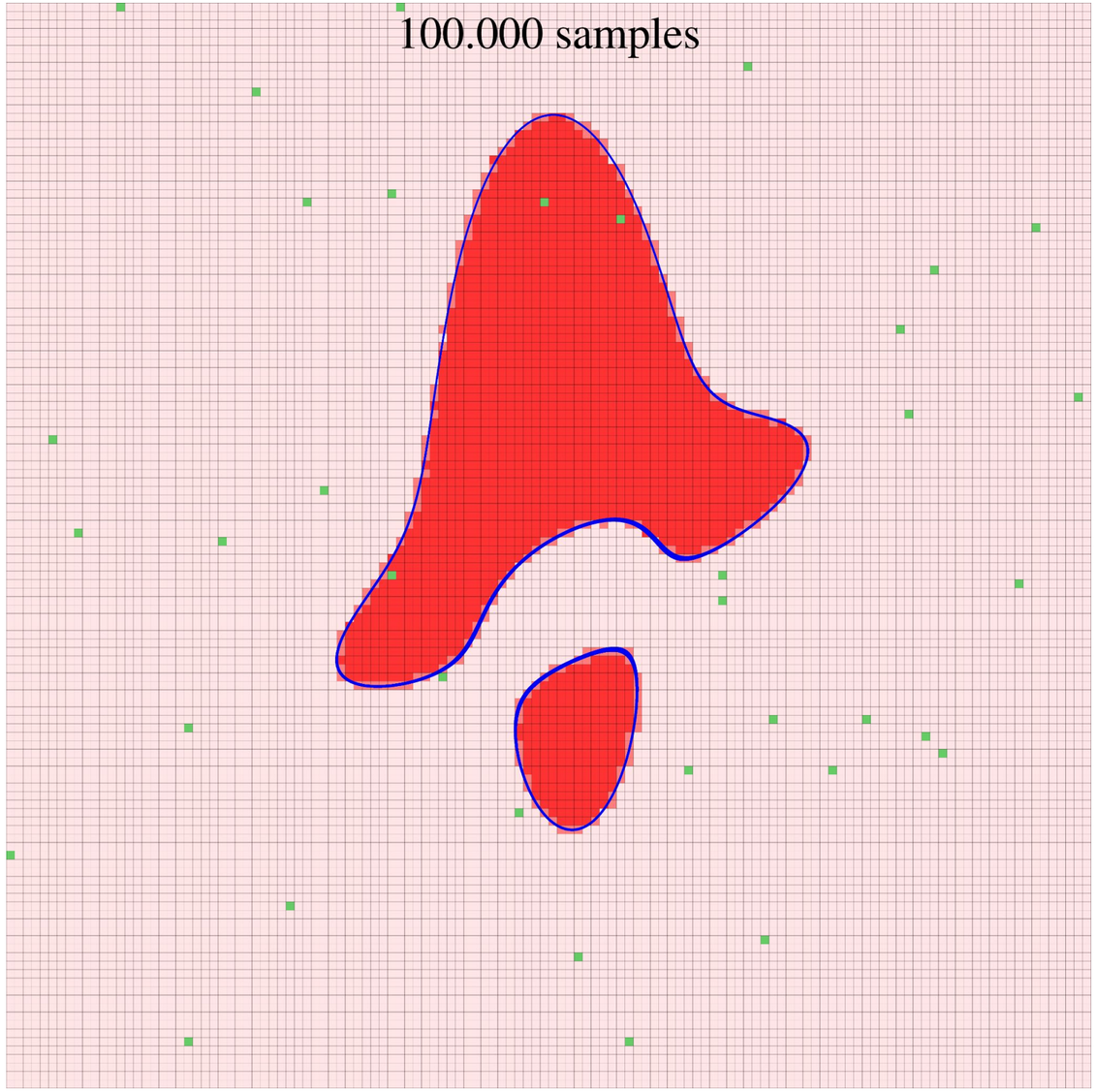
How can we draw an implicit function?



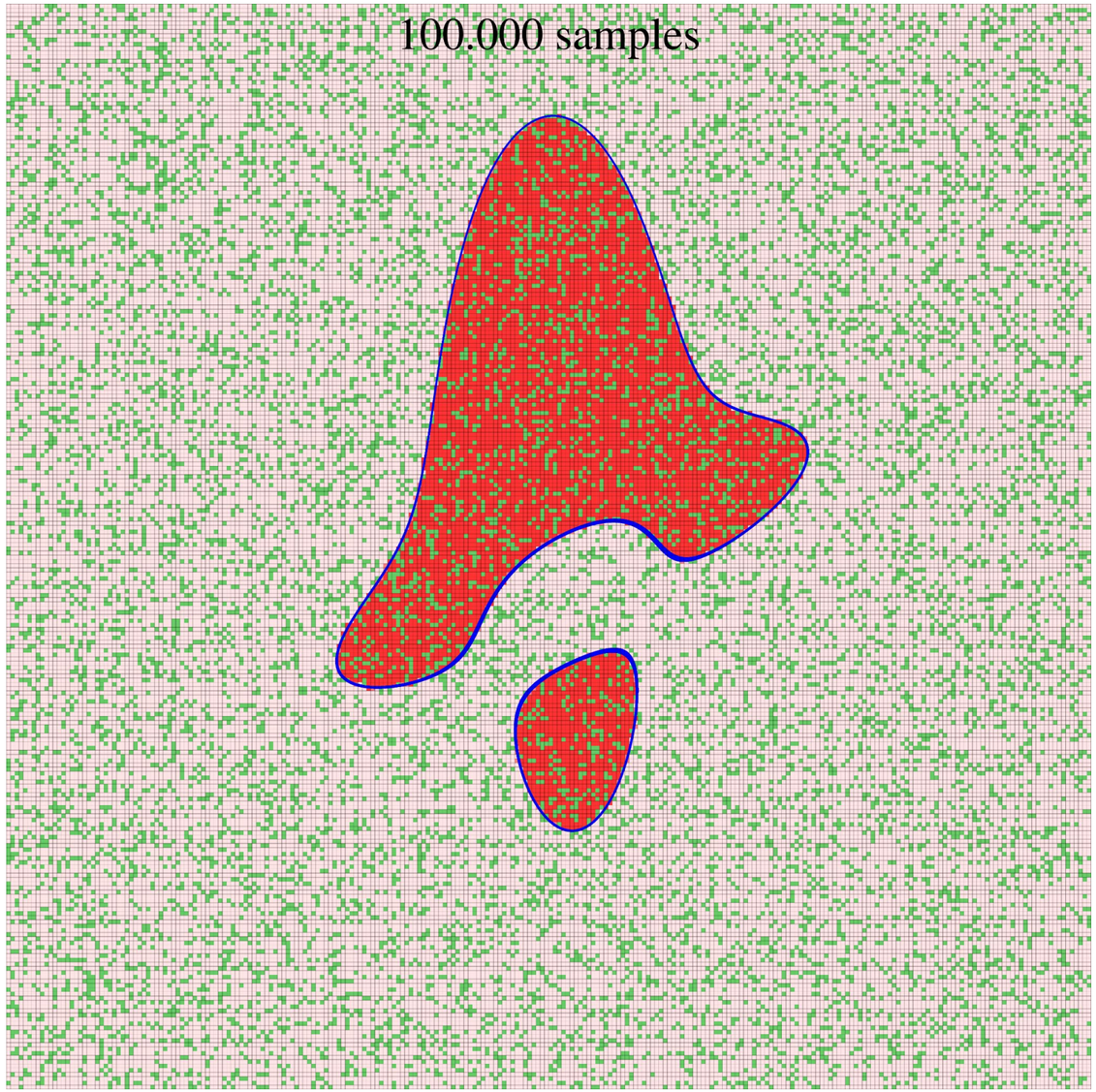
How can we draw an implicit function?



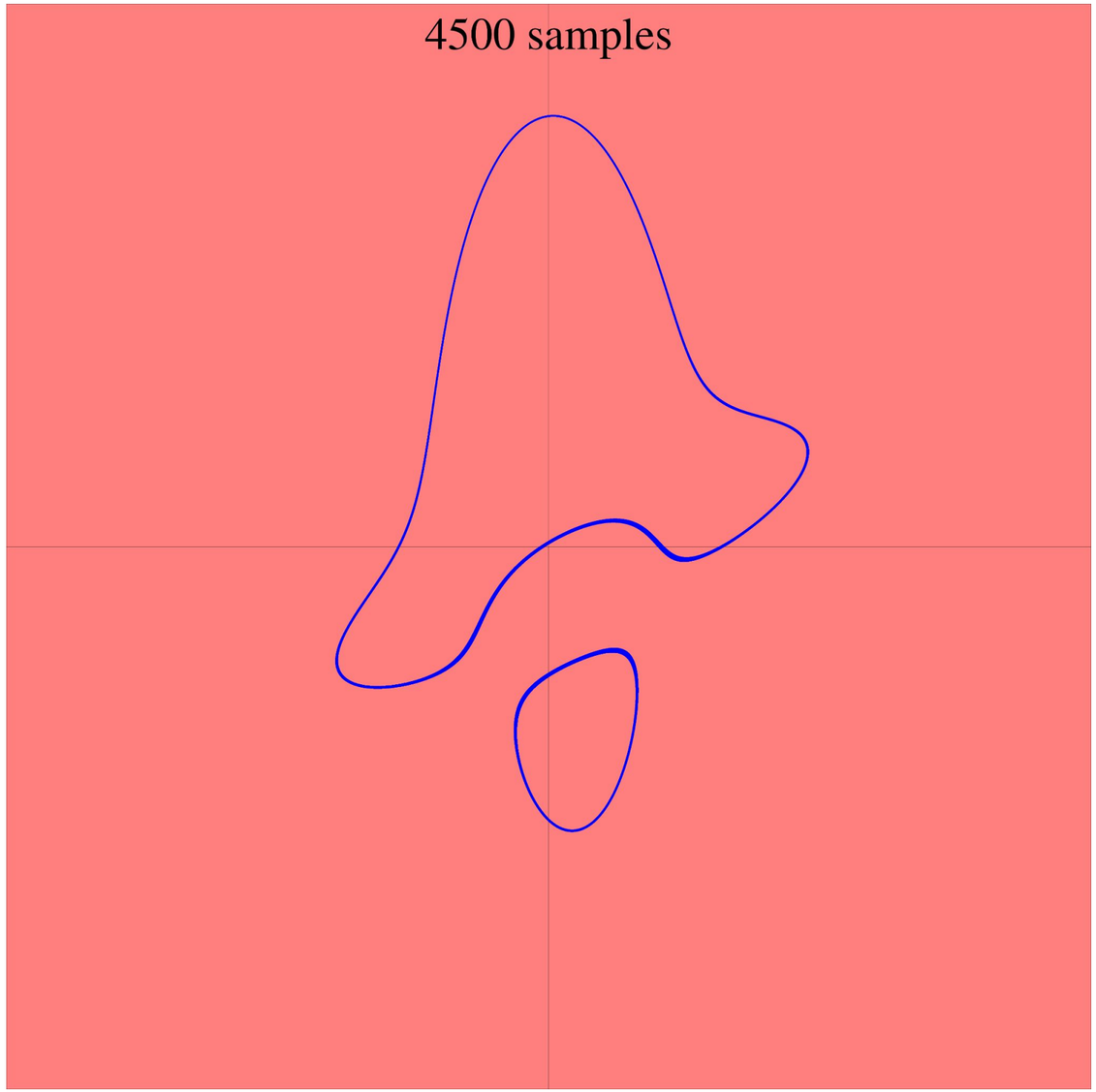
How can we draw an implicit function?



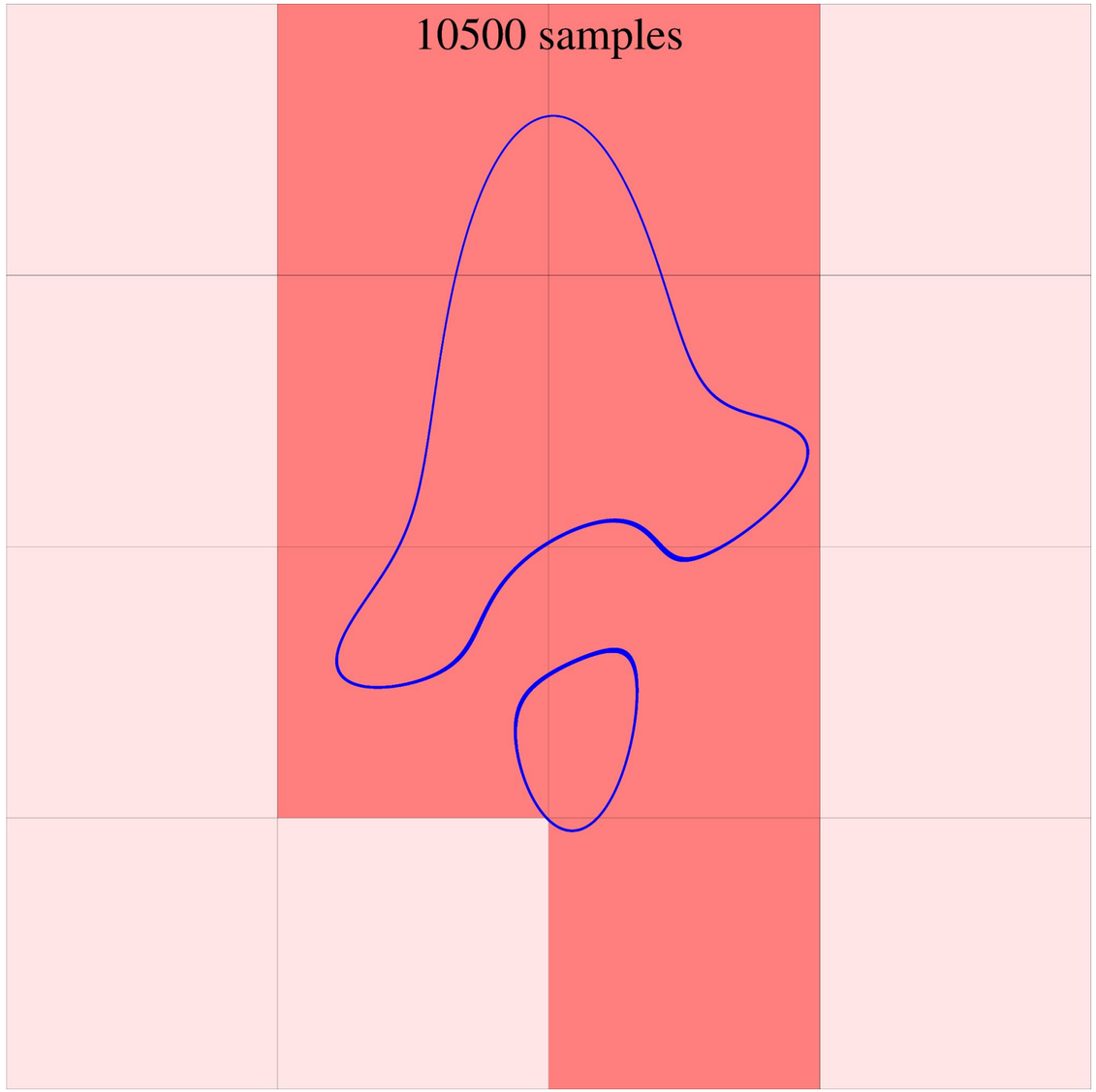
How can we draw an implicit function?



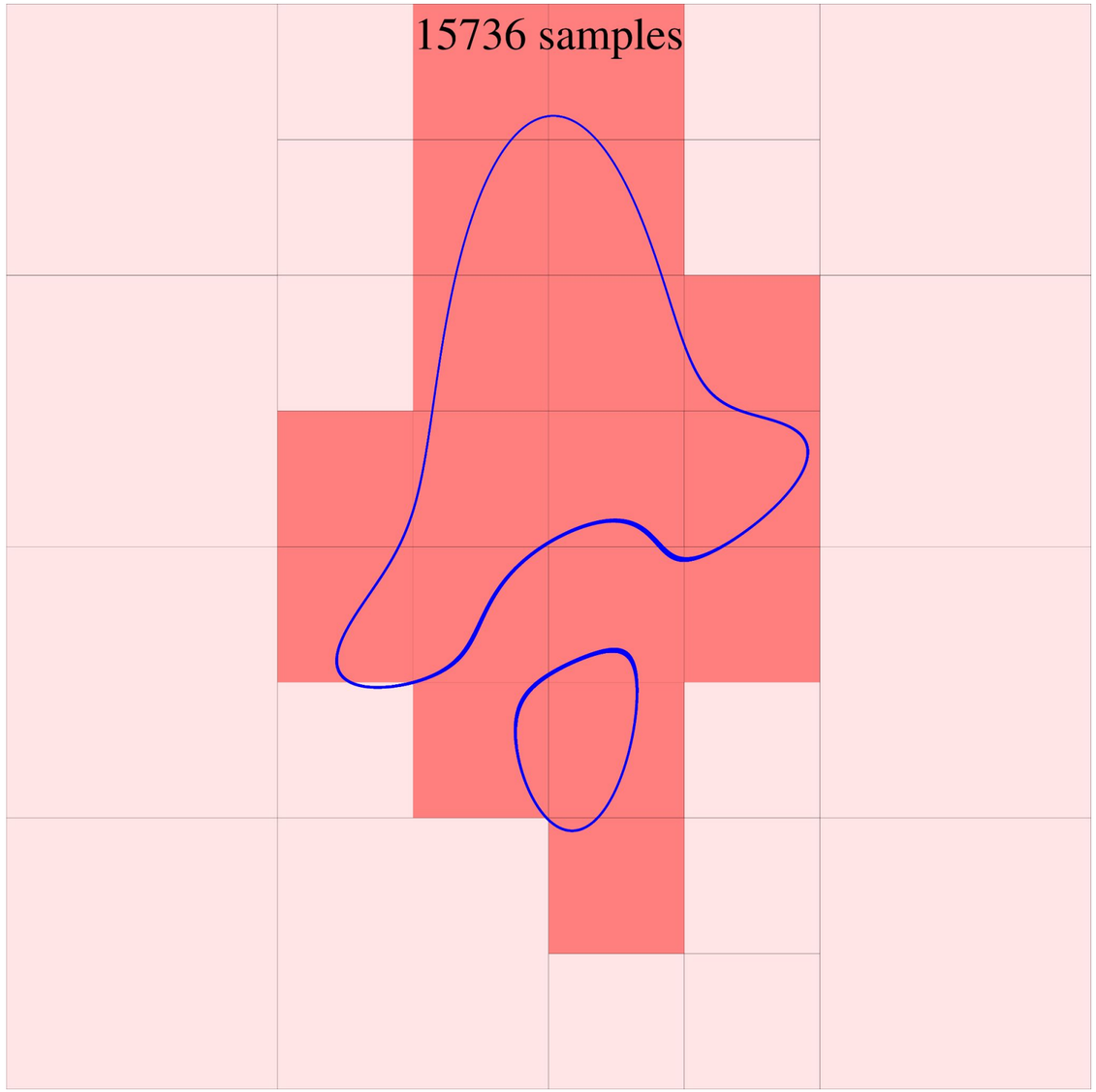
How can we draw an implicit function?



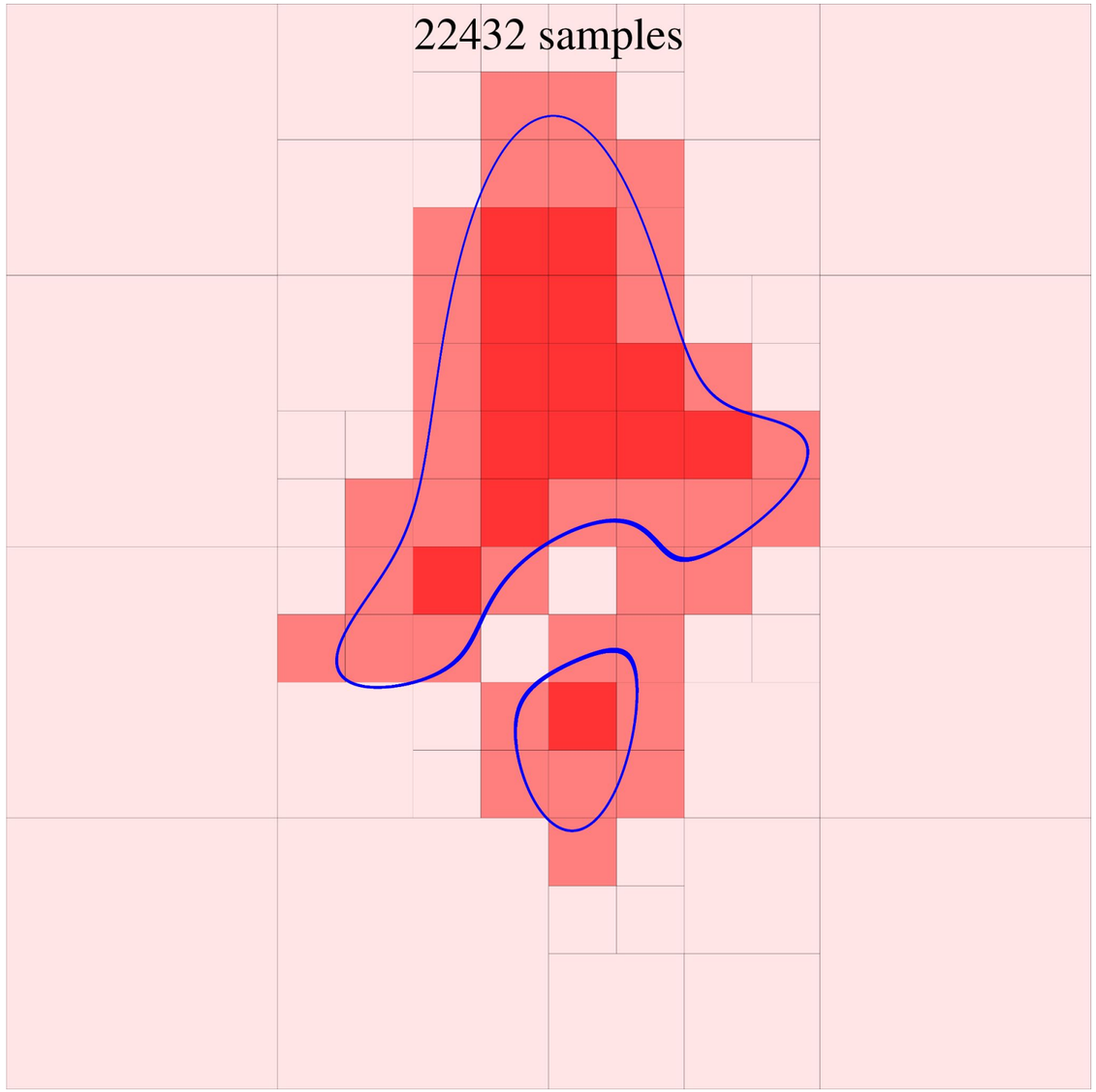
How can we draw an implicit function?



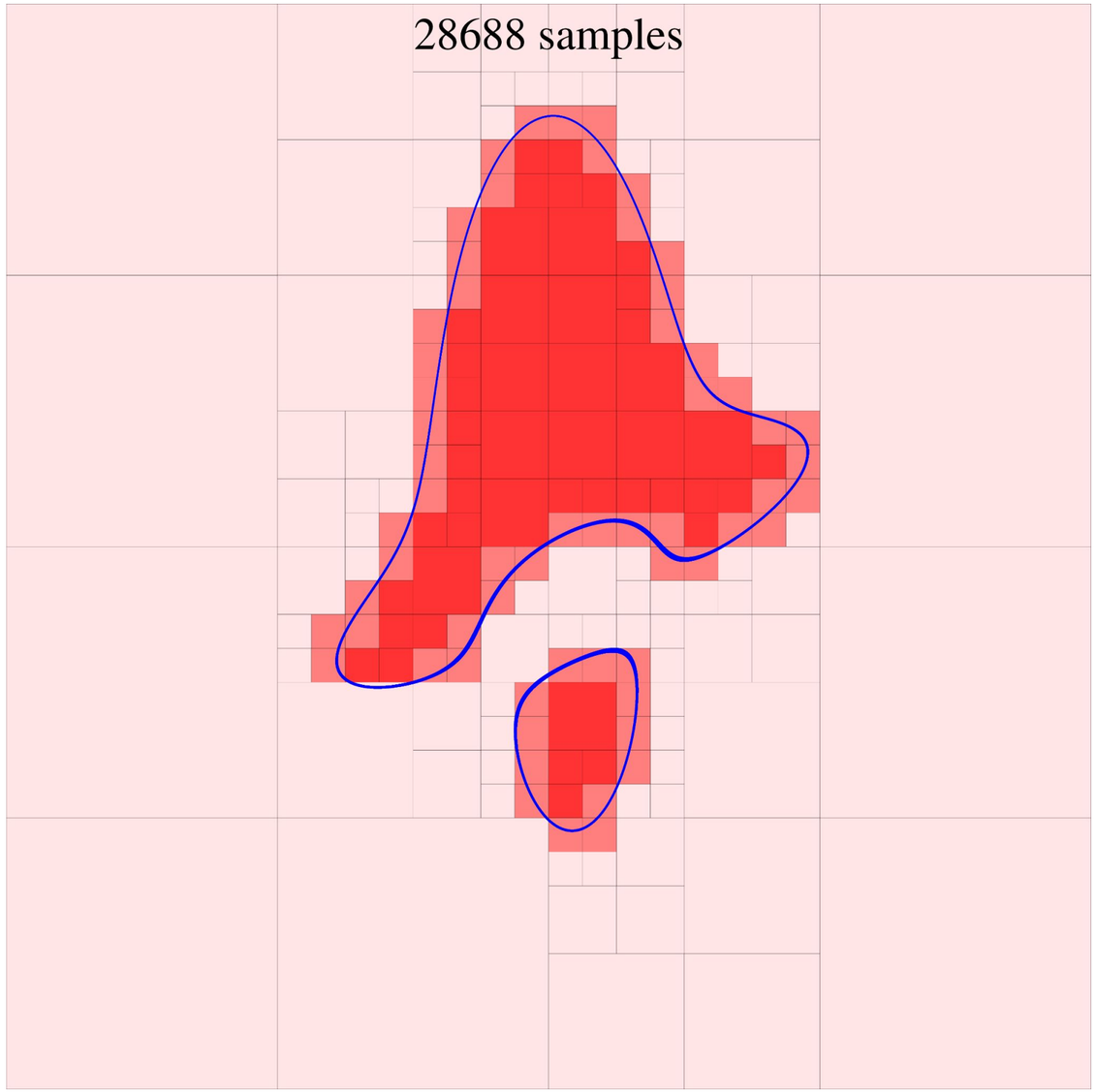
How can we draw an implicit function?



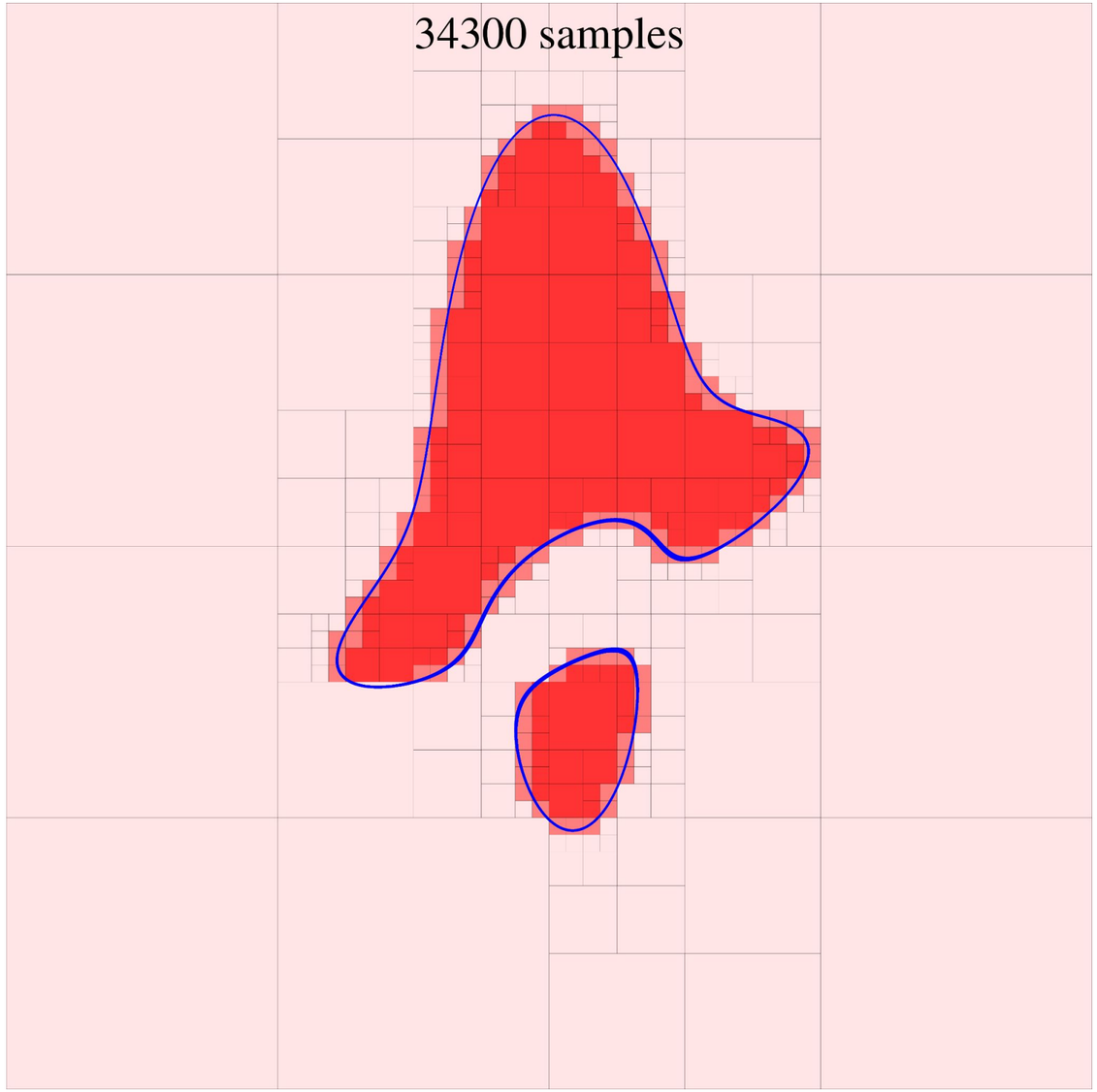
How can we draw an implicit function?



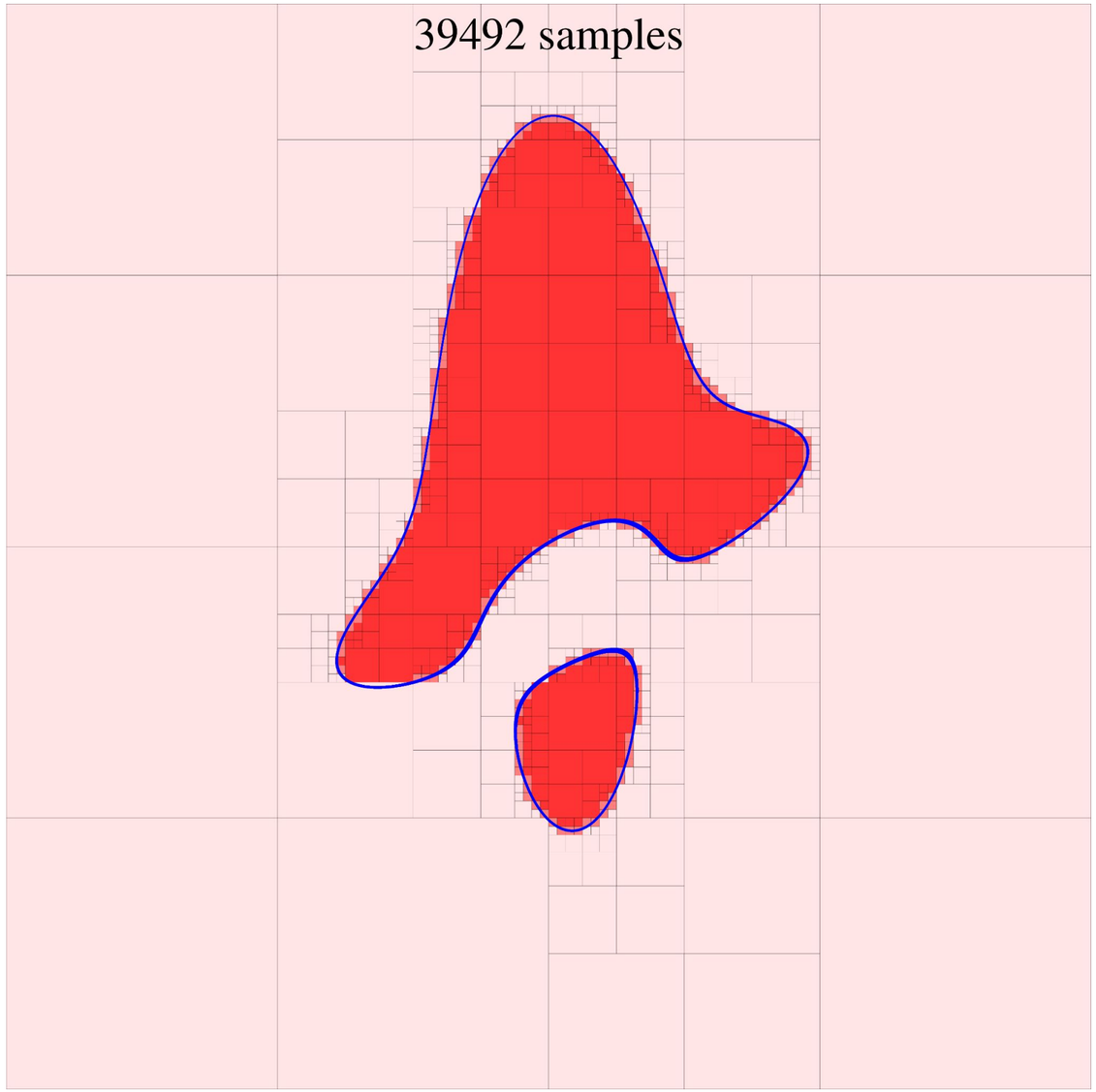
How can we draw an implicit function?



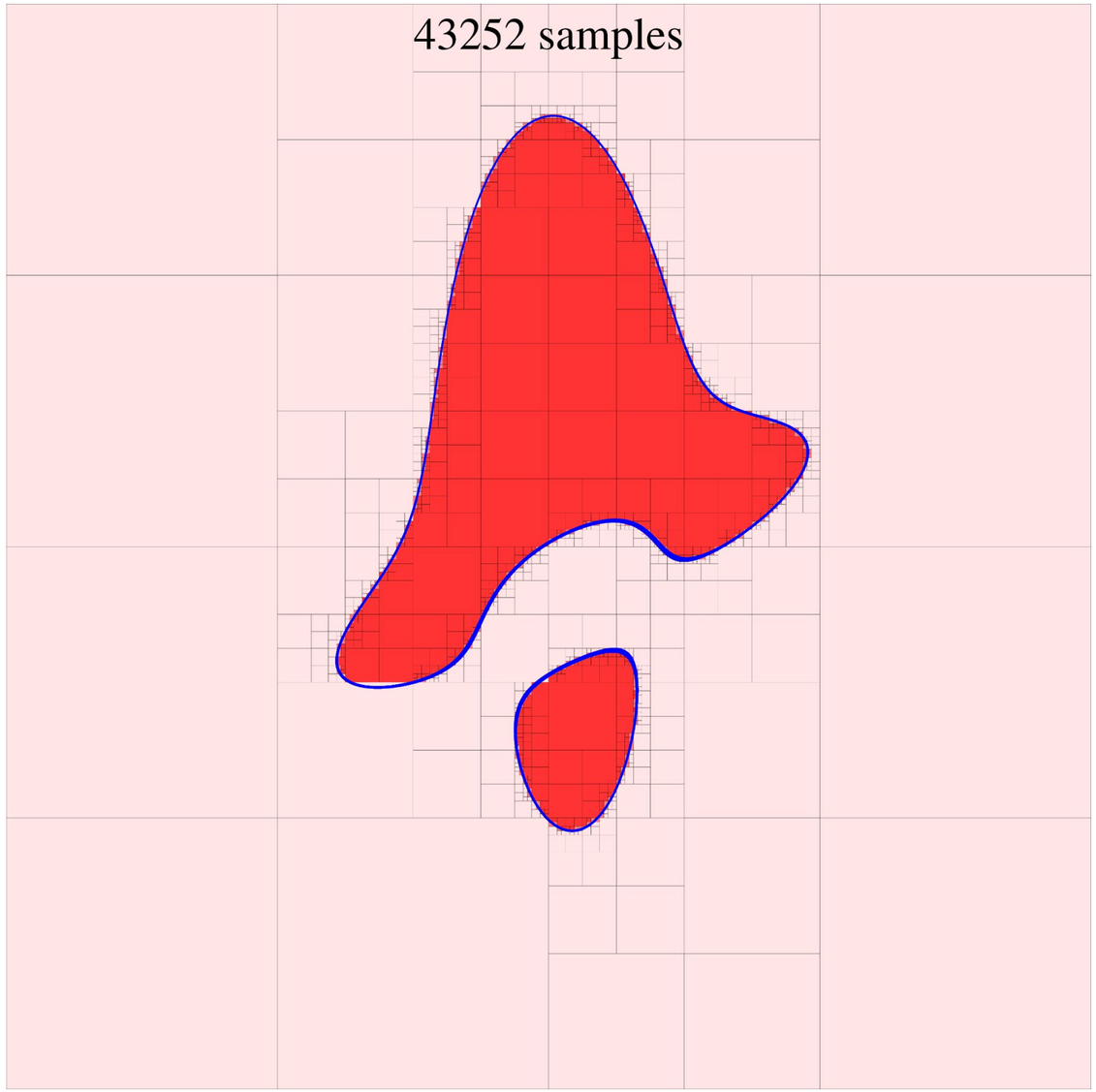
How can we draw an implicit function?



How can we draw an implicit function?



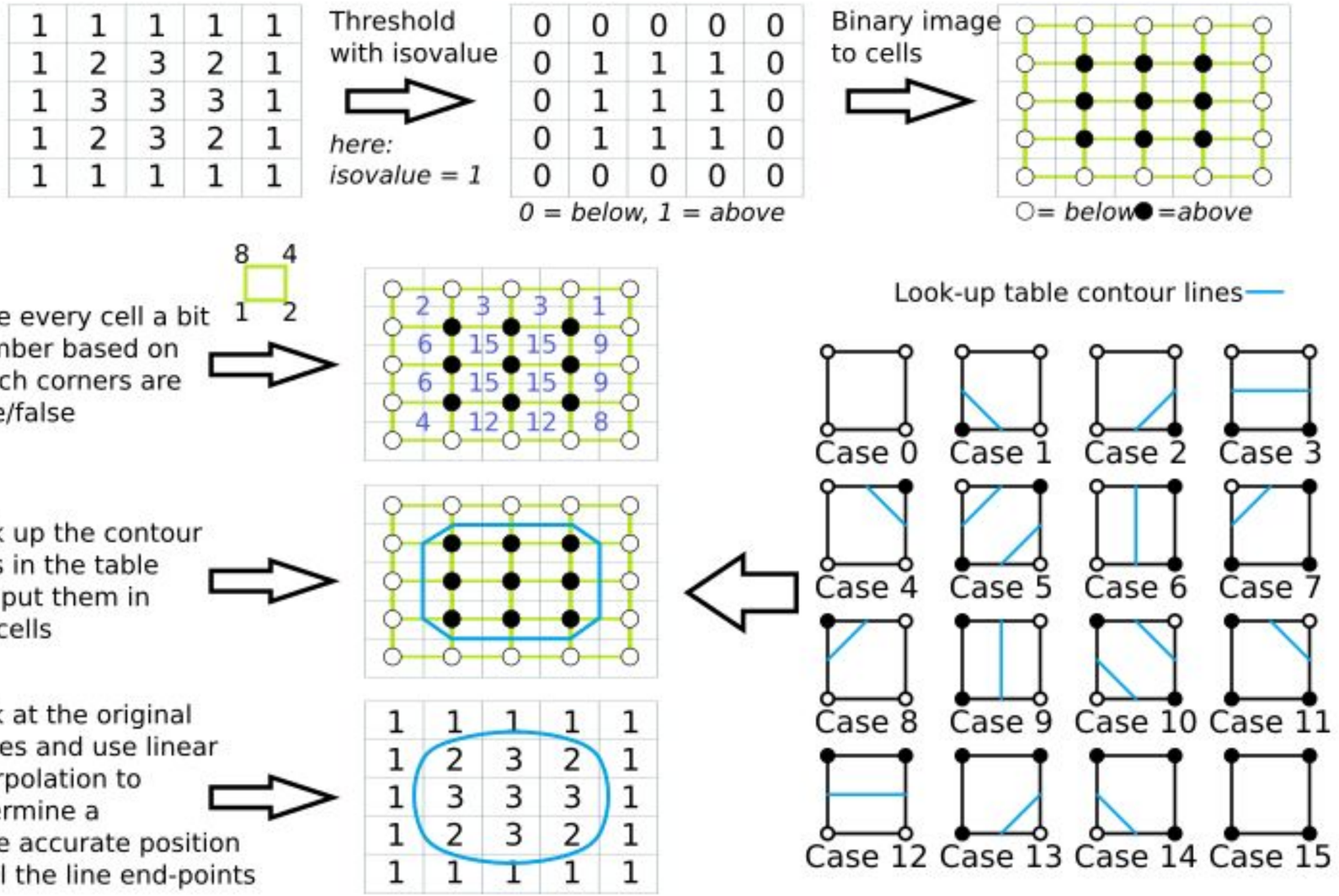
How can we draw an implicit function?



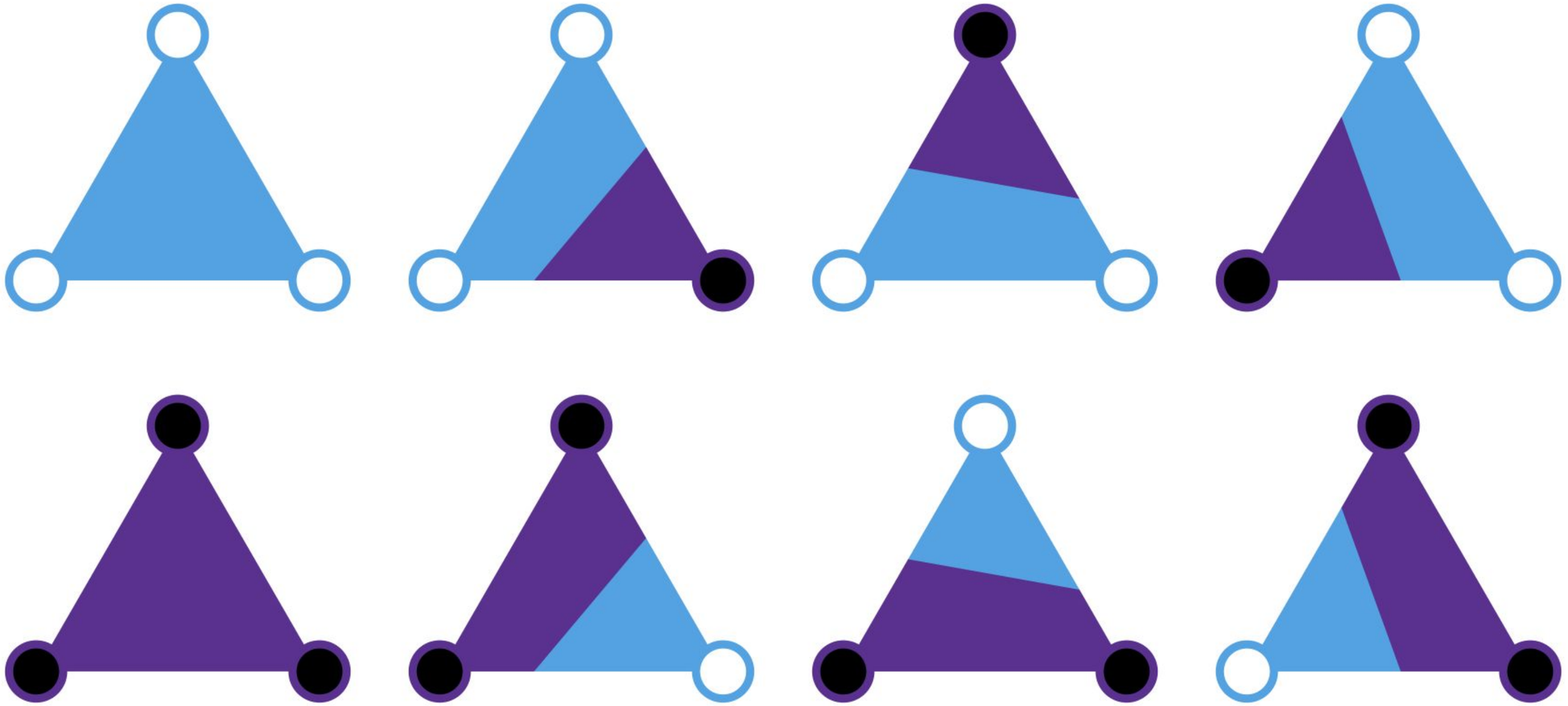
How can we draw an implicit function?

- Marching squares
- Marching triangles
- Marching cubes

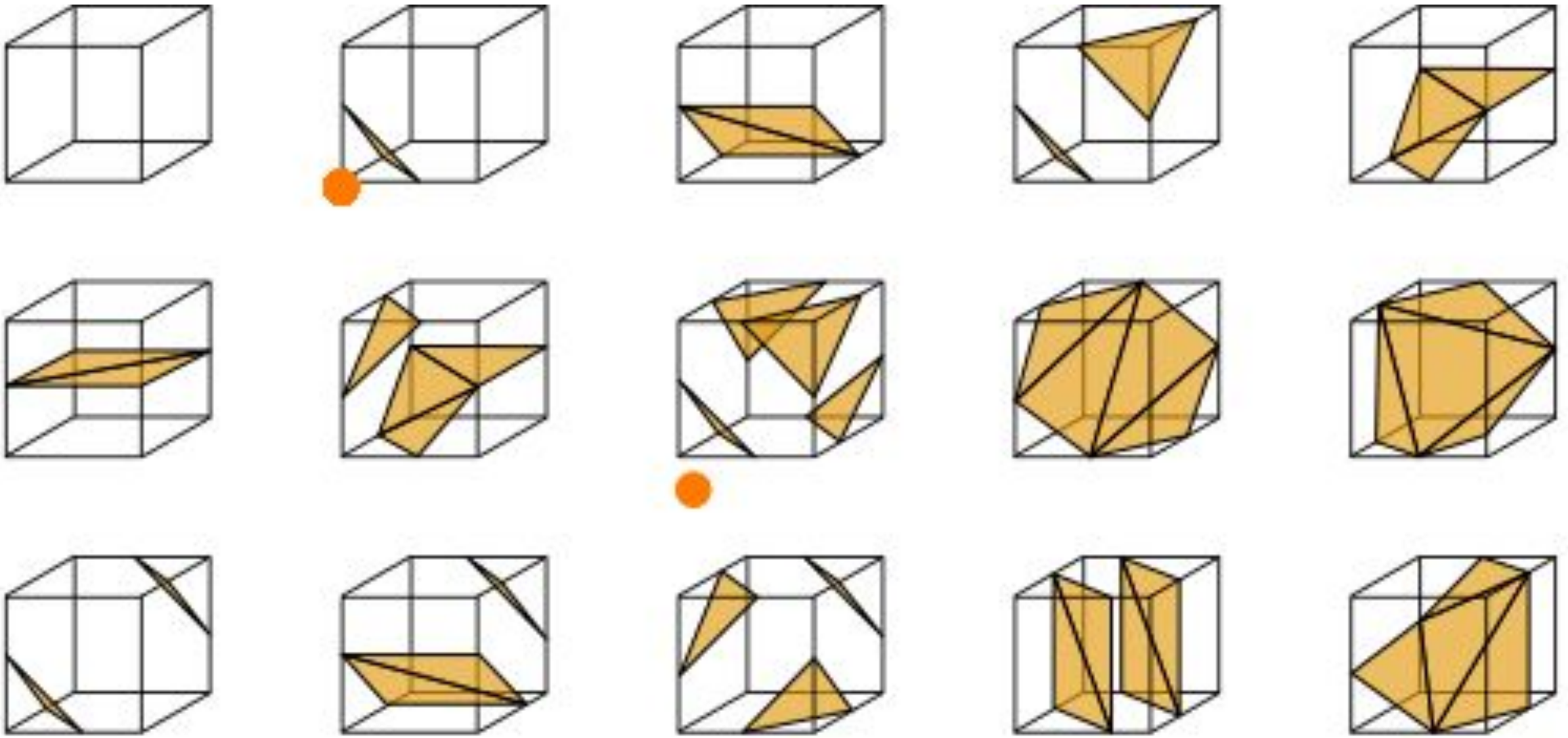
How can we draw an implicit function? – Marching Squares



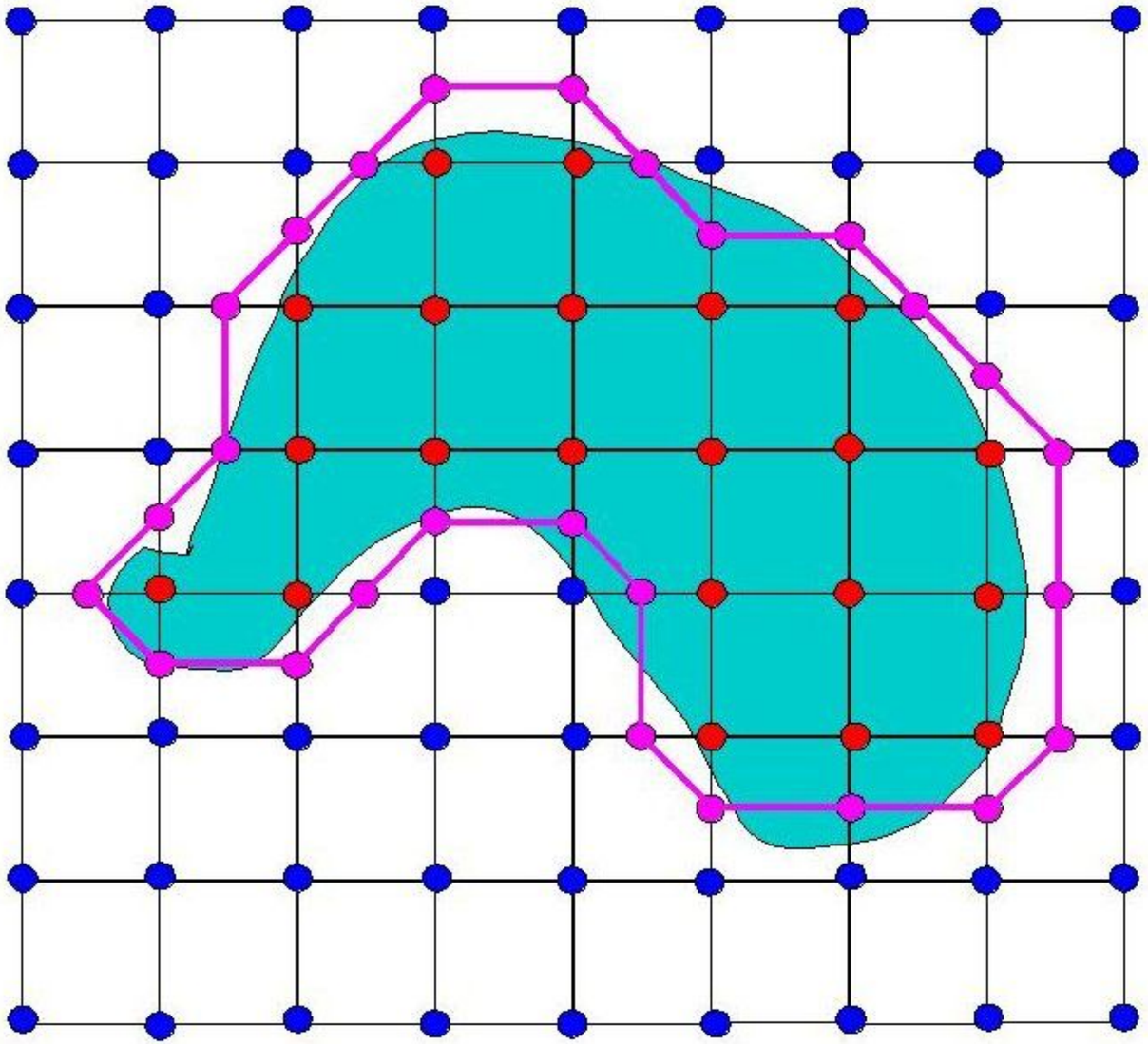
How can we draw an implicit function?



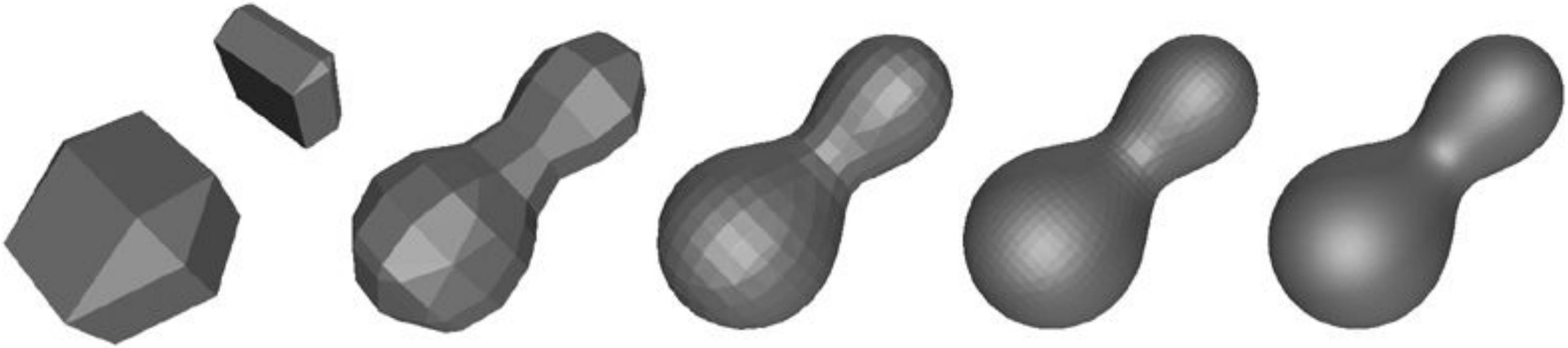
How can we draw an implicit function?



Example



Example



Grid size=10
70 Facets

Grid size=5
220 Facets

Grid size=2
1700 Facets

Grid size=1
6800 Facets

Grid size=0.5
27000 Facets

How can we draw an implicit function?

- Marching squares
- Marching triangles
- Marching cubes

→ Marching quares, marching triangles and marching cubes are parallelizable and simple, but better results can be obtained with much less computational effort using adaptative methods

Meshes of polygons

What is a polygon mesh?

→ A collection of vertices, edges, and faces that defines the shape of a polyhedral object in 3D

→ It is formed by vertices, edges and faces.

$V - E + F = 2(1 - g)$

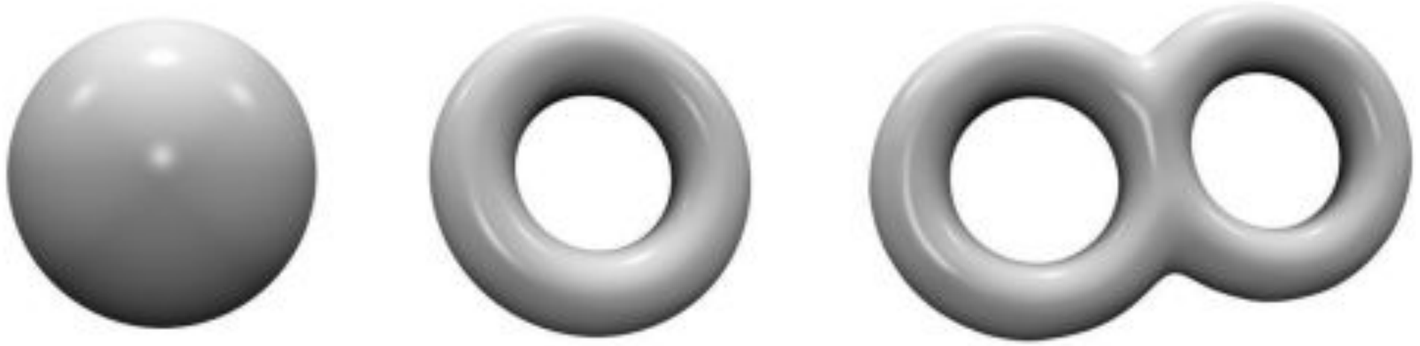
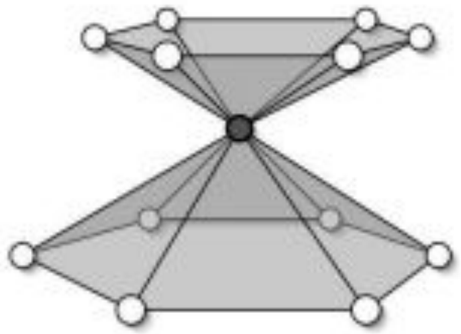
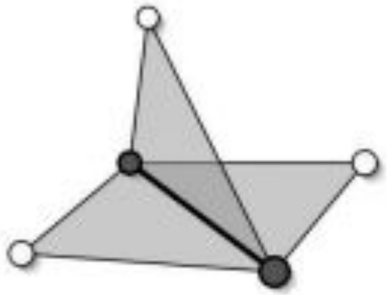


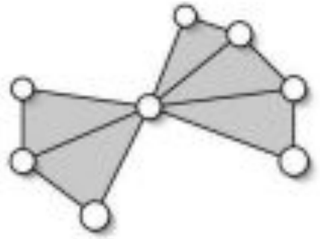
Figure 1.7. A sphere of genus 0 (left), a torus of genus 1 (center), and a double-torus of genus 2 (right). (Image taken from [Botsch et al. 06b].)

Meshes of polygons

→ A polygon mesh is "manifold" iff each edge is shared among exactly 2 faces.







Meshes of polygons - Useful tools

OFF file format

- http://www.geomview.org/docs/html/OFF.html

- Coordinates of the vertices, possibly with normals, colors, and/or texture coordinates.
- Faces

Meshes of polygons - Useful tools

- Meshlab
  https://www.meshlab.net/

Mesh data structures

There are several, and we can invent our own…

- vertex list, in which we have a list of vertices and the faces are make reference to them
- edge list, in which the edges make reference to the vertices and the faces make reference to the edges
- face list, in which the faces explicitly contain the coordinates of the vertices
- (several others)

Mesh data structures

Which operations do we want our data structure MESH to be able to do?

- topological requirements
- algorithmic requirements

Mesh data structures

Operations frequently used:

→ Access to individual vertices, edges, and faces. (including enumeration of all elements)

→ Oriented traversal of the edges of a face.

→ Access to the vertices of a face.

→ Access to the incident faces of an edge. Access to neighboring faces.

→ Given an edge, access to its two endpoint vertices.

→ Given a vertex, at least one incident face or edge must be accessible.

Mesh data structures: the halfedge representation

- split each (unoriented) edge into two oriented halfedges,
- halfedges are oriented consistently in counterclockwise order around each face and along each boundary.
- For each halfedge we store a reference to
  - the vertex it points to,
  - its adjacent face (a zero pointer if it is a boundary halfedge),
  - the next halfedge of the face or boundary (in counterclockwise direction),
  - the previous halfedge in the face, and
  - its opposite (or inverse) halfedge.

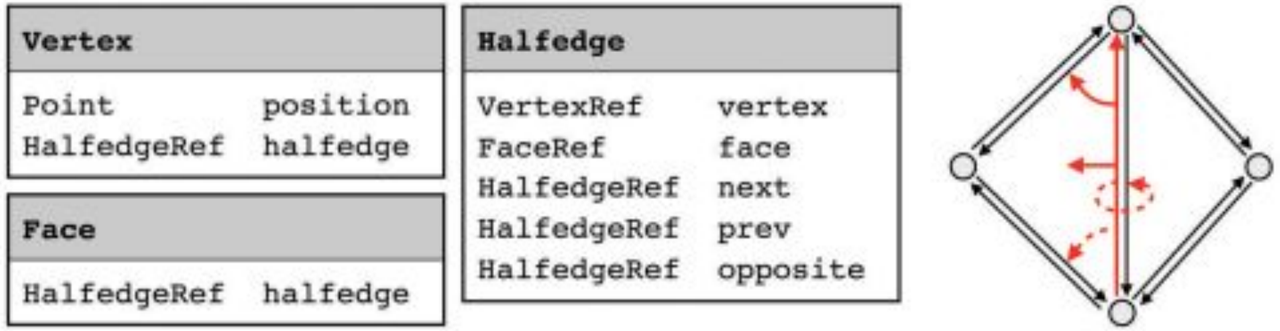


Figure 2.4. Connectivity information stored in an halfedge-based data structure.

Mesh data structures: the halfedge representation

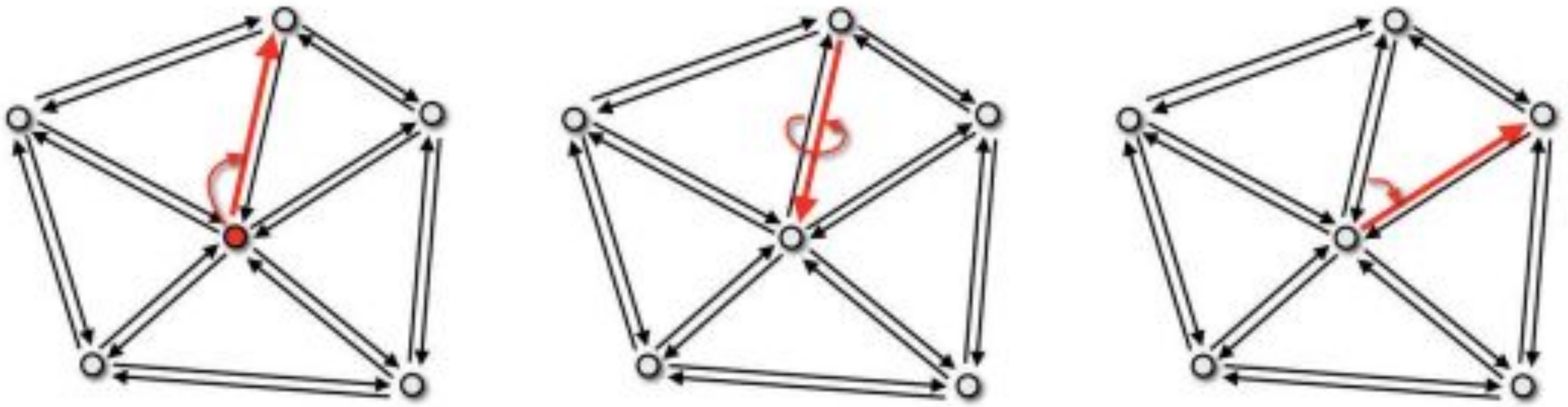
- 

Figure 2.5. The one-ring neighbors of the center vertex can be enumerated by starting with an outgoing halfedge of the center (left), and then repeatedly rotating clockwise by stepping to the opposite halfedge (center) and next halfedge (right) until the first halfedge is reached again.

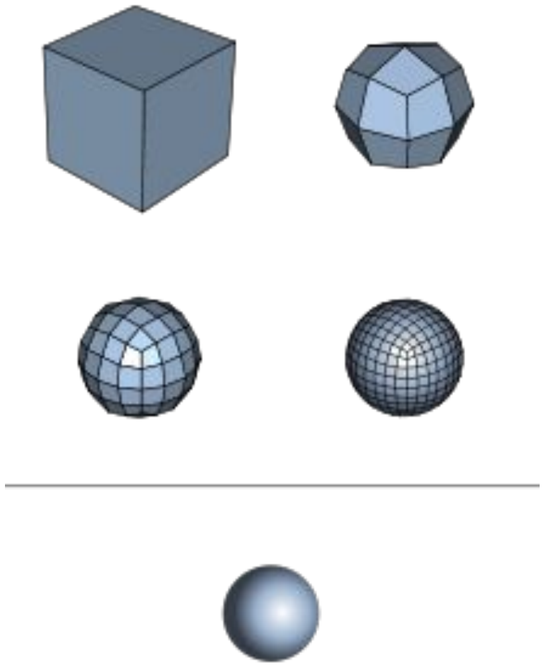
Subdivision surfaces

- creates smooth surfaces through iterative refinement of the mesh
- starts with a low-resolution mesh and refines
- Key aspects:
  - initial mesh
  - subdivision strategy

    Examples:
    - Catmull-Clark https://en.wikipedia.org/wiki/Catmull%E2%80%93Clark_subdivision_surface
    - Loop https://en.wikipedia.org/wiki/Loop_subdivision_surface
    - Doo-Sabin https://en.wikipedia.org/wiki/Doo%E2%80%93Sabin_subdivision_surface

Subdivision surfaces: Catmull-Clark algorithm

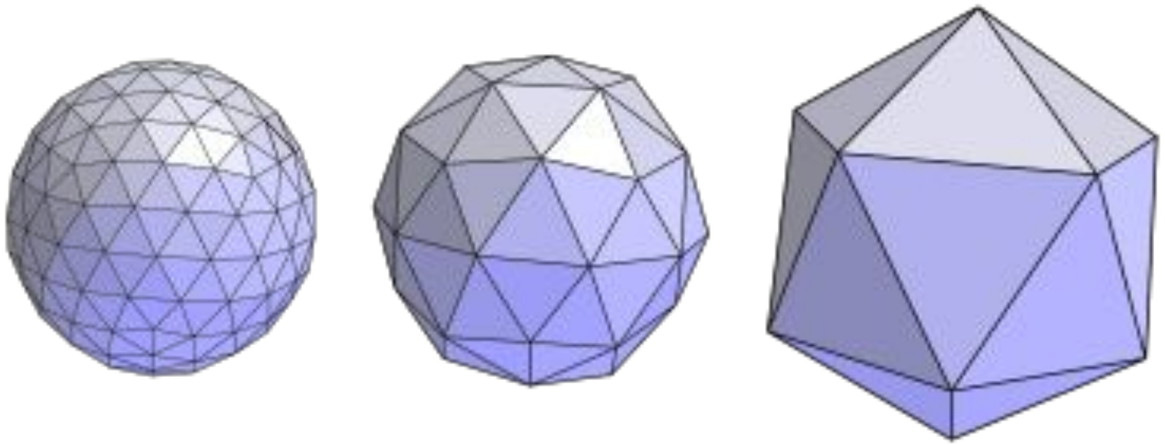
- Particularly good for quadrilaterals
- Subdivision: define *face points* and *edge points.*
  The new position of a vertex is computed as the average of the surrounding face points and edge points, weighted properly. Vertices are added, and faces are subdivided.



- https://en.wikipedia.org/wiki/Catmull%E2%80%93Clark_subdivision_surface
- Foley, section 23.3: Catmull-Clark subdivision surface

Subdivision surfaces: Loop algorithm

- Good for triangles
- Divides each triangle in 4 triangles



Subdivision surfaces: Doo-Sabin algorithm