

CS 247 – Scientific Visualization

Lecture 6: Scalar Fields, Pt. 2

Markus Hadwiger, KAUST

Reading Assignment #3 (until Feb 15)



Read (required):

- Data Visualization book, finish Chapter 3 (read starting with 3.6)
- Data Visualization book, Chapter 5 until 5.3 (inclusive)

Quiz #1: Feb 17



Organization

- First 30 min of lecture
- No material (book, notes, ...) allowed

Content of questions

- Lectures (both actual lectures and slides)
- Reading assignments (except optional ones)
- Programming assignments (algorithms, methods)
- Solve short practical examples



Scalar Fields

Contours



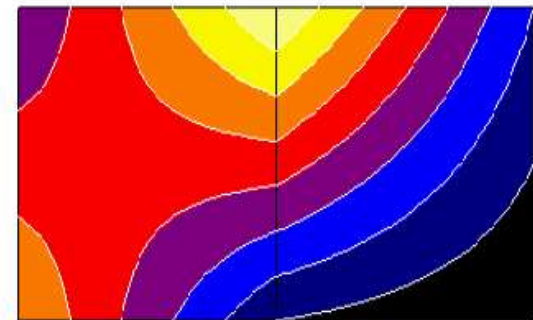
Set of points where the scalar field s has a given value c :

$$S(c) := f^{-1}(c) \quad S(c) := \{x \in \mathbb{R}^n : f(x) = c\}$$

Common contouring algorithms

- 2D: marching squares, marching triangles
- 3D: marching cubes, marching tetrahedra

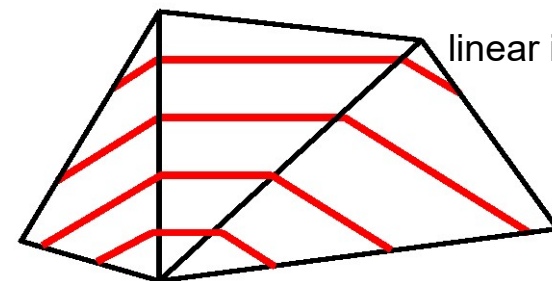
bilinear interpolation



Implicit methods

- Point-on-contour test
- Isosurface ray-casting

linear interpolation



Contours



Set of points where the scalar field s has a given value c :

$$S(c) := f^{-1}(c) \quad S(c) := \{x \in \mathbb{R}^2 : f(x) = c\}$$

Common contouring algorithms

- 2D: marching squares, marching triangles
- 3D: marching cubes, marching tetrahedra

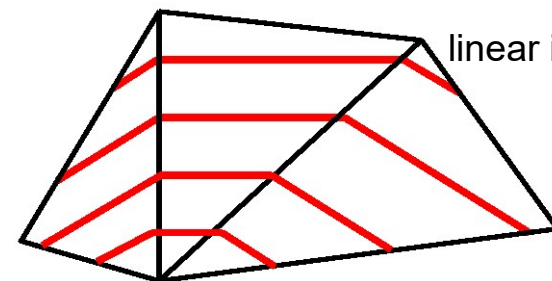
bilinear interpolation



Implicit methods

- Point-on-contour test
- Isosurface ray-casting

linear interpolation



Contours



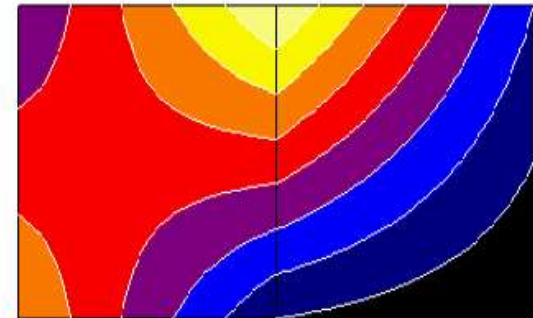
Set of points where the scalar field s has a given value c :

$$S(c) := f^{-1}(c) \quad S(c) := \{x \in \mathbb{R}^3 : f(x) = c\}$$

Common contouring algorithms

- 2D: marching squares, marching triangles
- 3D: marching cubes, marching tetrahedra

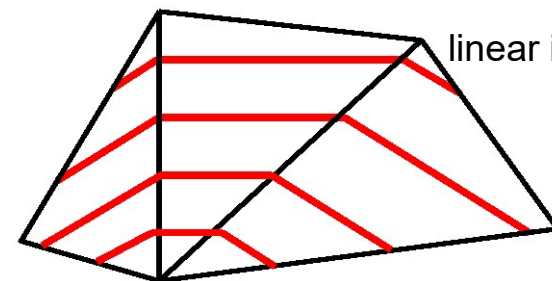
bilinear interpolation



Implicit methods

- Point-on-contour test
- Isosurface ray-casting

linear interpolation



What are contours?

Set of points where the scalar field s has a given value c :

$$S(c) := \{x \in \mathbb{R}^n : f(x) = c\}$$

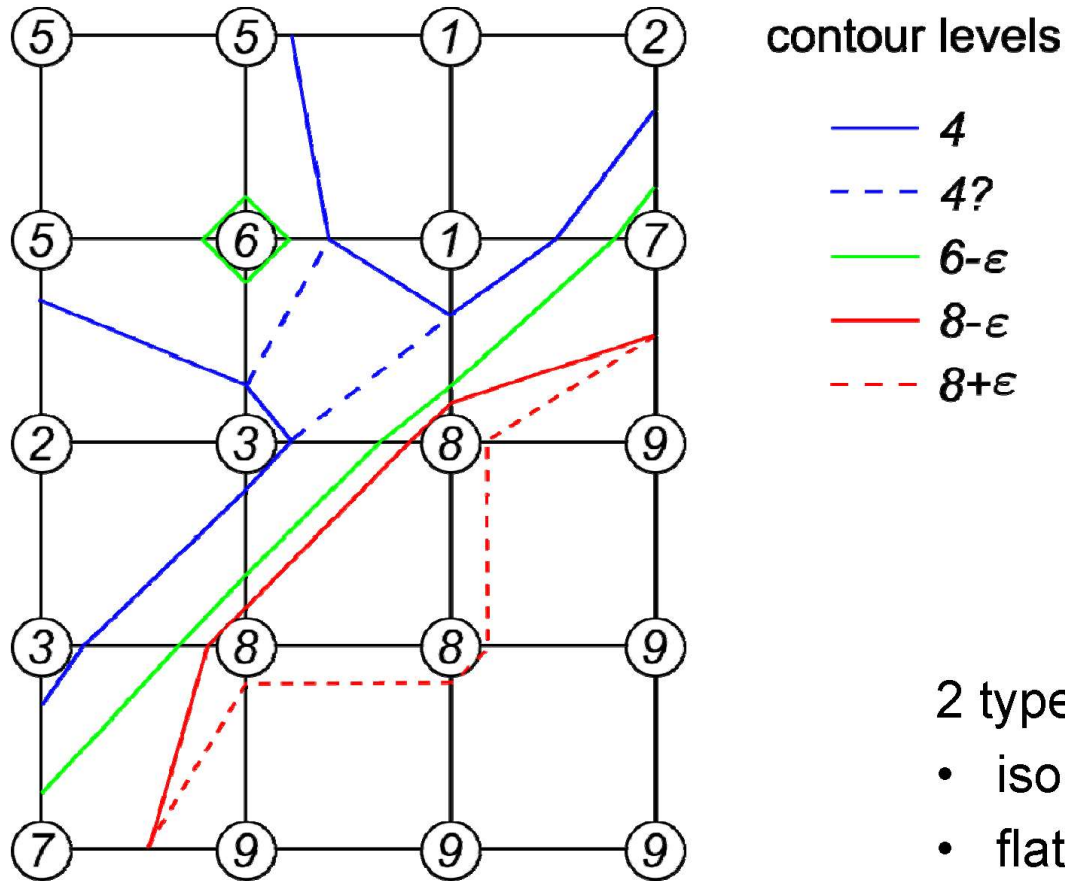
Examples in 2D:

- height contours on maps
- isobars on weather maps

Contouring algorithm:

- find intersection with grid edges
- connect points in each cell

Example



Contours in a quadrangle cell

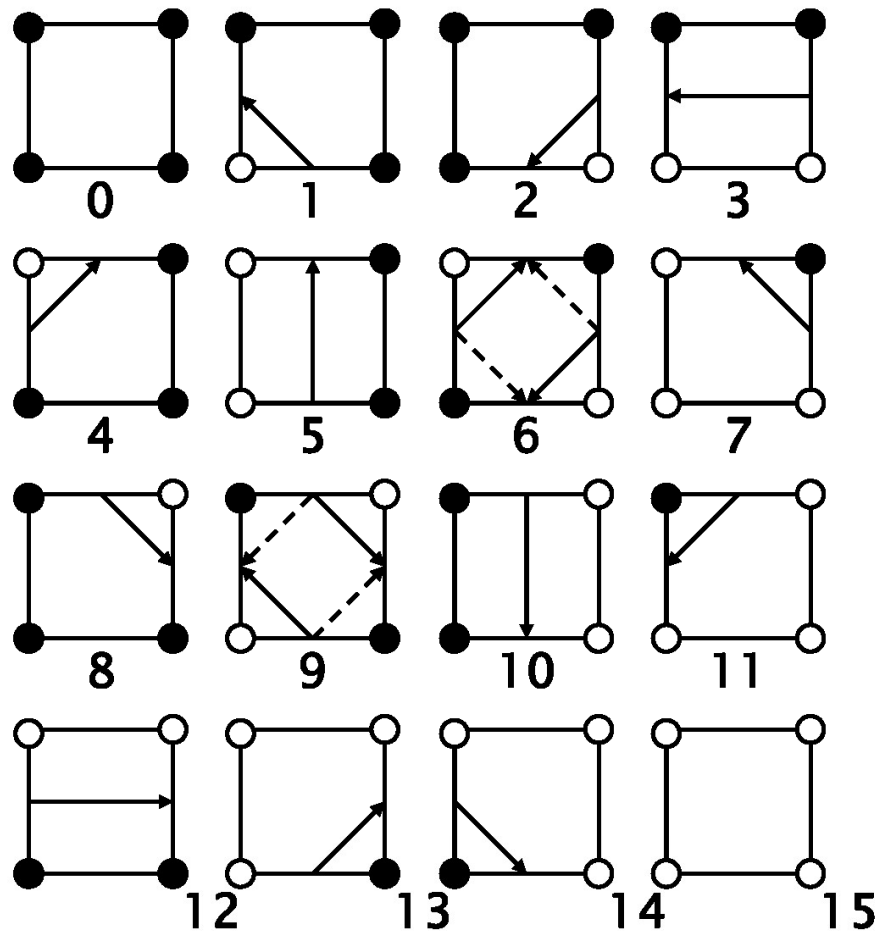
Basic contouring algorithms:

- **cell-by-cell** algorithms: simple structure, but generate disconnected segments, require post-processing
- **contour propagation** methods: more complicated, but generate connected contours

"Marching squares" algorithm (systematic cell-by-cell):

- process nodes in ccw order, denoted here as x_0, x_1, x_2, x_3
- compute at each node \mathbf{x}_i the reduced field $\tilde{f}(x_i) = f(x_i) - (c - \epsilon)$ (which is forced to be nonzero)
- take its sign as the i^{th} bit of a 4-bit integer
- use this as an index for lookup table containing the connectivity information:

Contours in a quadrangle cell



- $\tilde{f}(x_i) < 0$
- $\tilde{f}(x_i) > 0$

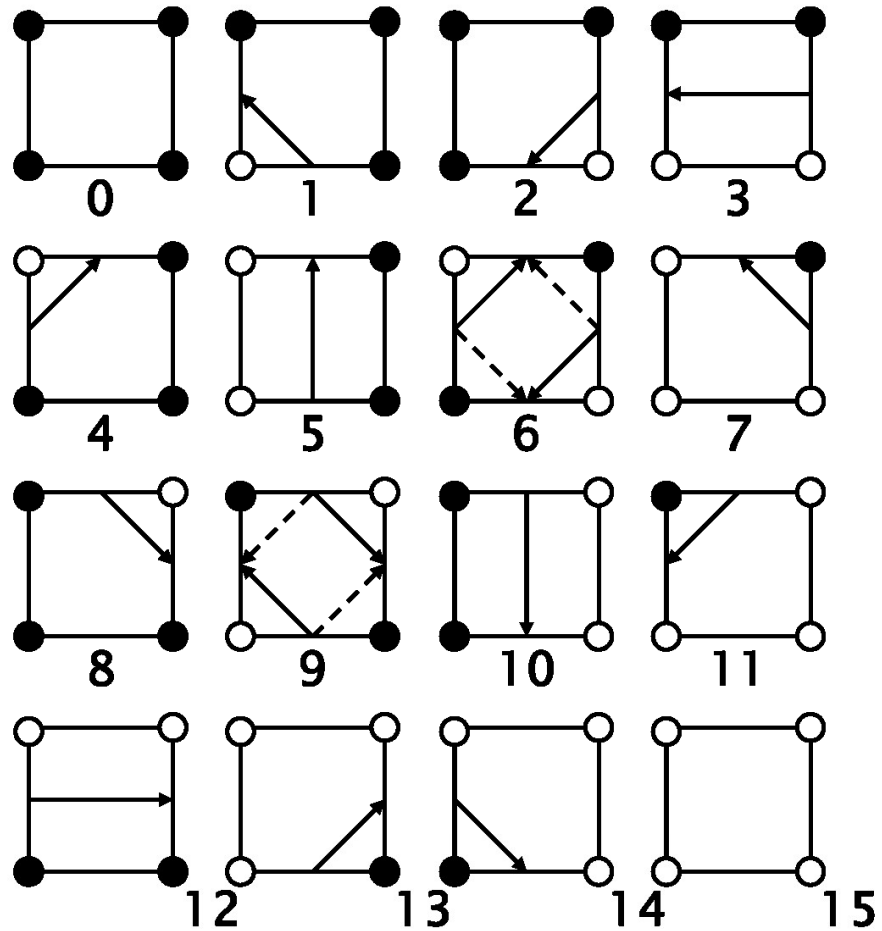
Alternating signs exist in cases 6 and 9.

Choose the solid or dashed line?

Both are possible for topological consistency.

This allows to have a fixed table of 16 cases.

Contours in a quadrangle cell



- $f(x_i) < c$
- $f(x_i) \geq c$

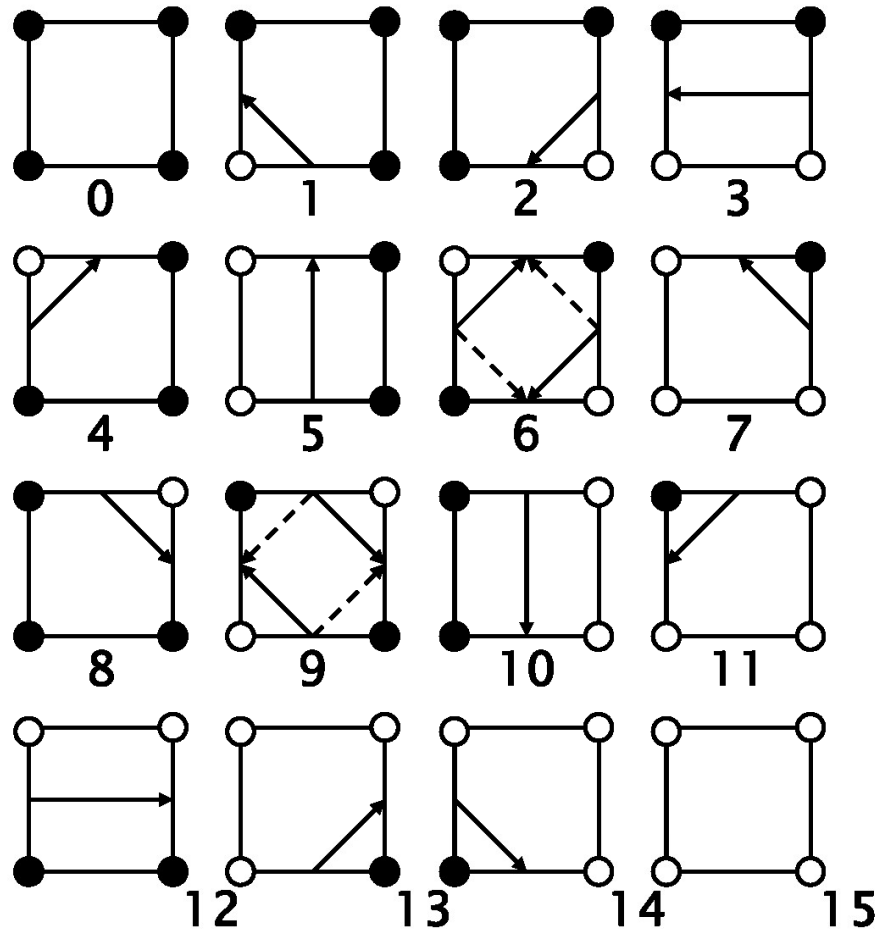
Alternating signs exist in cases 6 and 9.

Choose the solid or dashed line?

Both are possible for topological consistency.

This allows to have a fixed table of 16 cases.

Contours in a quadrangle cell



- $f(x_i) \leq c$
- $f(x_i) > c$

Alternating signs exist in cases 6 and 9.

Choose the solid or dashed line?

Both are possible for topological consistency.

This allows to have a fixed table of 16 cases.

Topological consistency

To avoid degeneracies, use **symbolic perturbations**:

If level c is found as a node value, set the level to $c - \varepsilon$ where ε is a symbolic infinitesimal.

Then:



- contours intersect edges at some (possibly infinitesimal) distance from end points
- flat regions can be visualized by pair of contours at $c - \varepsilon$ and $c + \varepsilon$
- contours are **topologically consistent**, meaning:

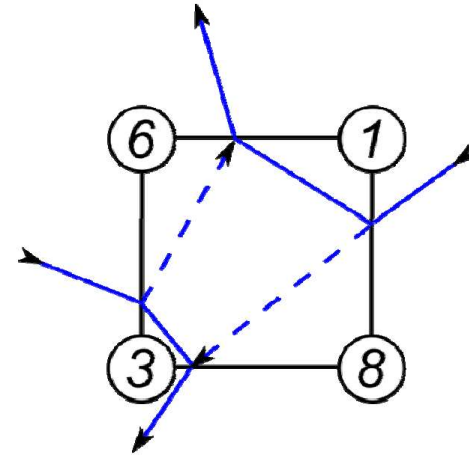
Contours are **closed, orientable, nonintersecting lines**.

Ambiguities of contours

What is the **correct** contour of $c=4$?

Two possibilities, both are orientable:

- connect high values 
- connect low values 



Answer: correctness depends on interior values of $f(x)$.

But: different interpolation schemes are possible.

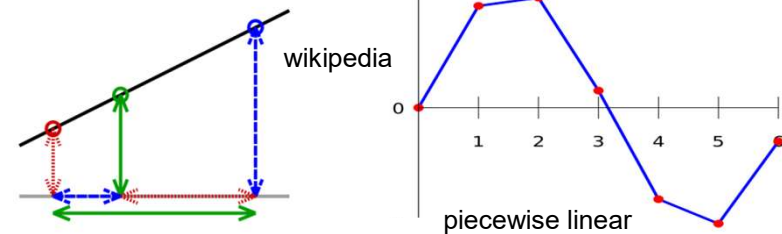
Better question: What is the correct contour with respect to bilinear interpolation?

Linear Interpolation / Convex Combinations



Linear interpolation in 1D:

$$f(\alpha) = (1 - \alpha)v_1 + \alpha v_2$$



Line embedded in 2D (linear interpolation of vertex coordinates/attributes):

$$f(\alpha_1, \alpha_2) = \alpha_1 v_1 + \alpha_2 v_2$$
$$\alpha_1 + \alpha_2 = 1$$

$$f(\alpha) = v_1 + \alpha(v_2 - v_1)$$
$$\alpha = \alpha_2$$

Line segment: $\alpha_1, \alpha_2 \geq 0$ (\rightarrow convex combination)

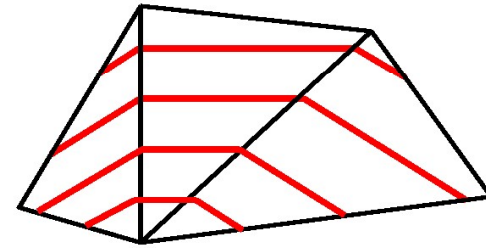
Compare to line parameterization
with parameter t:

$$v(t) = v_1 + t(v_2 - v_1)$$

Contours in triangle/tetrahedral cells

Linear interpolation of cells implies piece-wise linear contours.

Contours are unambiguous, making "marching triangles" even simpler than "marching squares".

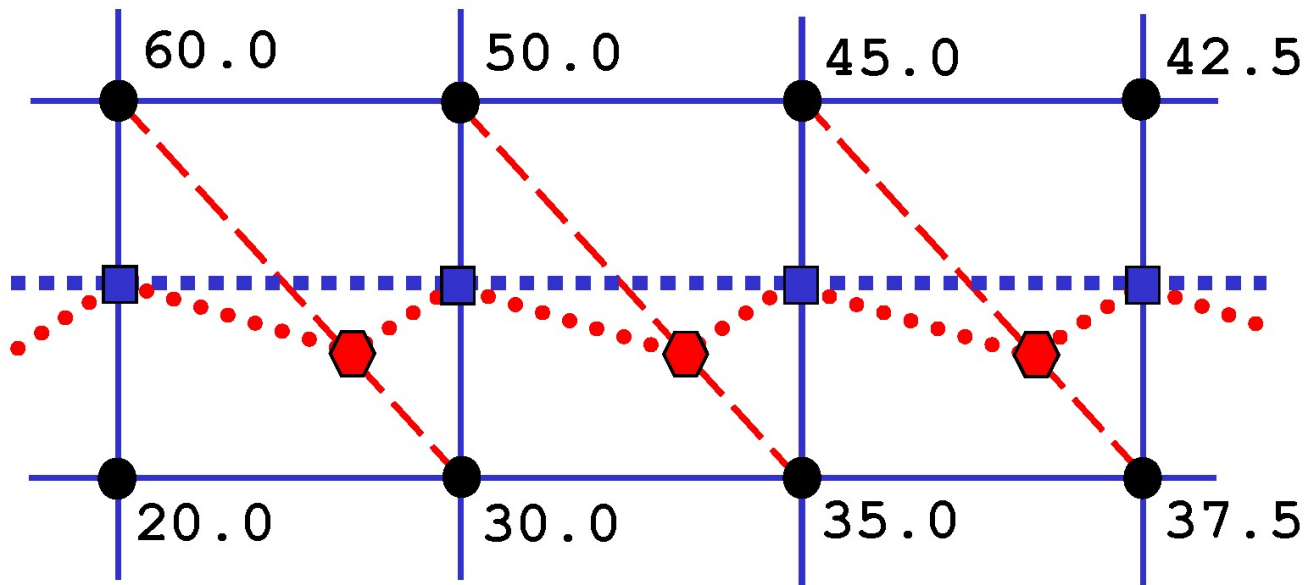


Question: Why not split quadrangles into two triangles (and hexahedra into five or six tetrahedra) and use marching triangles (tetrahedra)?

Answer: This can introduce periodic artifacts!

Contours in triangle/tetrahedral cells

Illustrative example: Find contour at level $c=40.0$!



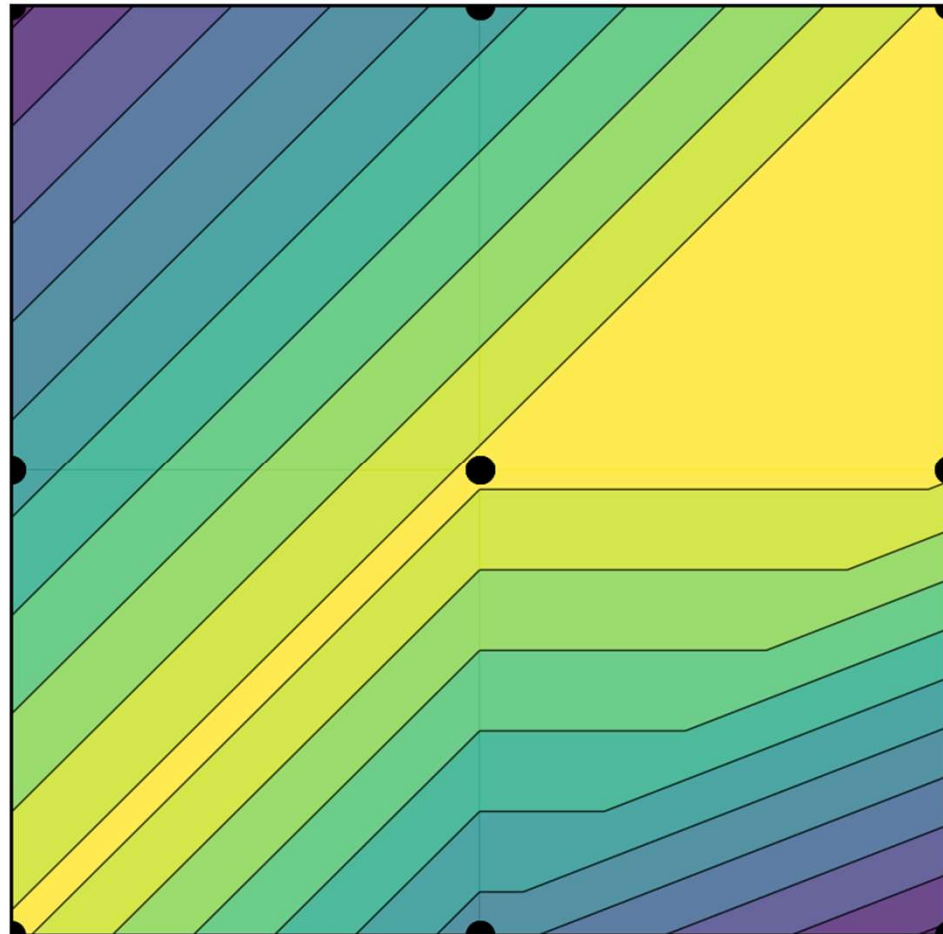
— original quad grid, yielding vertices ■ and contour - - - -
- - - triangulated grid, yielding vertices ⬡ and contour

Bi-Linear Interpolation: Comparisons



linear

(2 triangles per quad;
diagonal:
bottom-left,
top-right)

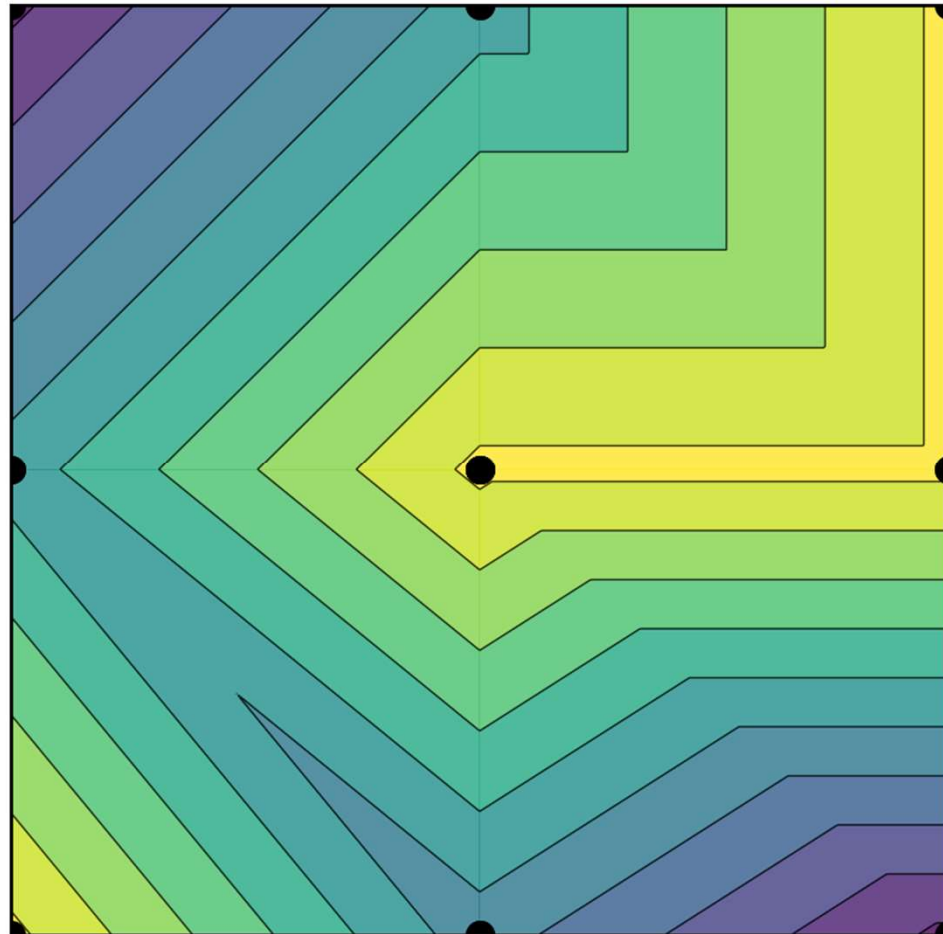


Bi-Linear Interpolation: Comparisons



linear

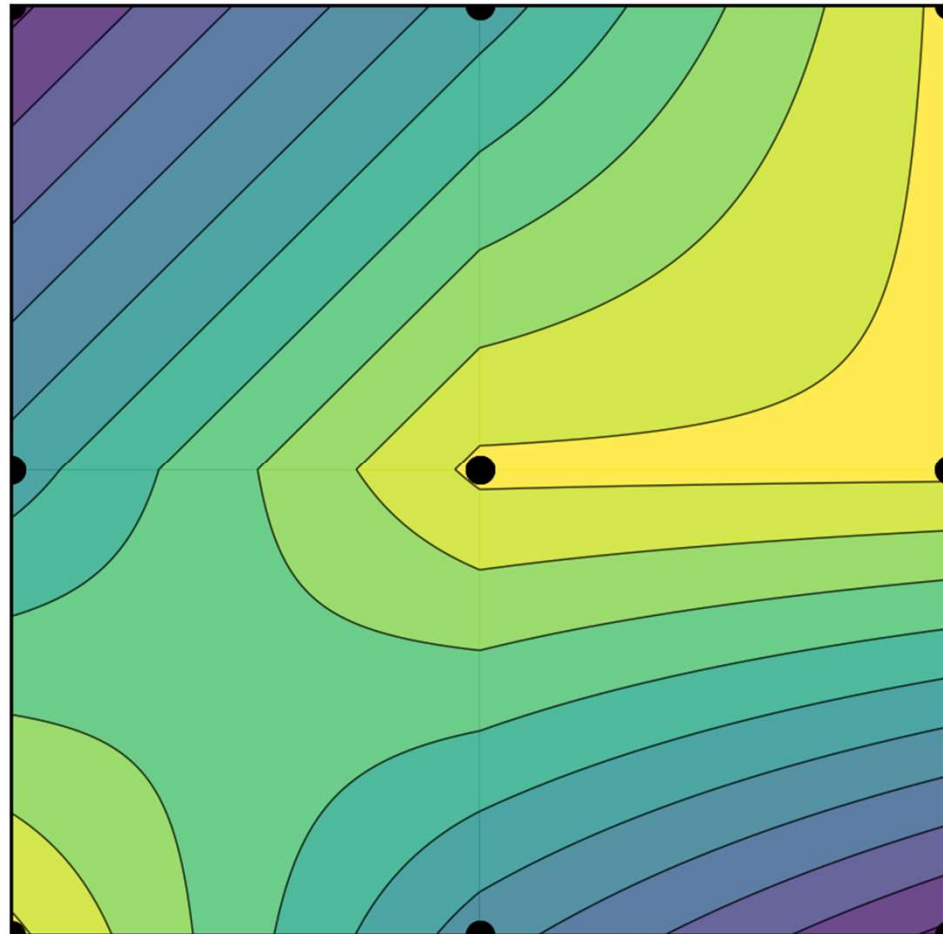
(2 triangles per quad;
diagonal:
top-left,
bottom-right)



Bi-Linear Interpolation: Comparisons



bi-linear



Thank you.

Thanks for material

- Helwig Hauser
- Eduard Gröller
- Daniel Weiskopf
- Torsten Möller
- Ronny Peikert
- Philipp Muigg
- Christof Rezk-Salama