Normal multilevel triangulations for geometric image compression

Ward Van Aerschot, Maarten Jansen and Adhemar Bultheel

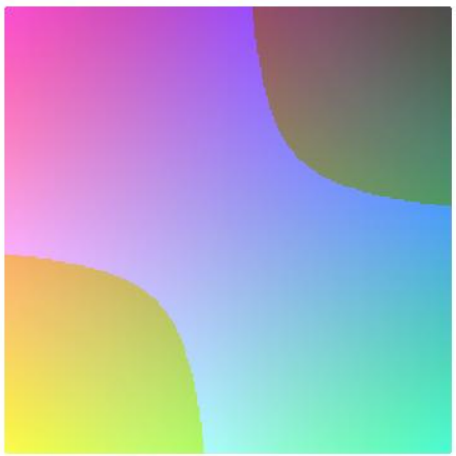
K.U.Leuven
www.cs.kuleuven.be/∼ward

Mathematical Methods for Curves and Surfaces
Tønsberg, June 26 - 1 July



Problem description

Compression of images consisting of:

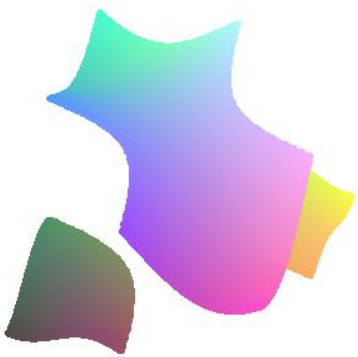


1. Smoothly coloured regions, separated by
2. Smooth contours

- In a 3D presentation the contours show up as discontinuities lying on a smooth curve in the domain.

Problem description

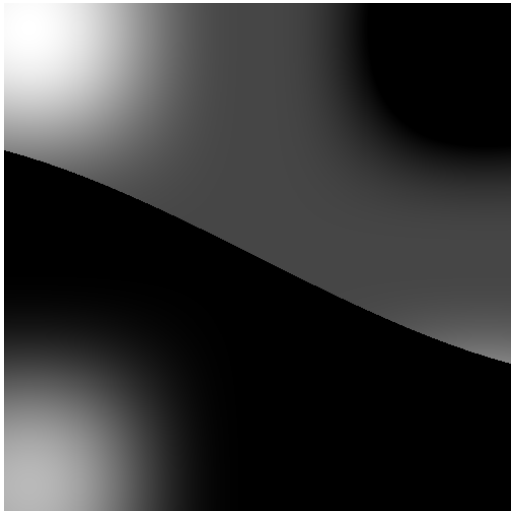
Compression of images consisting of:



1. Smoothly coloured regions, separated by
2. Smooth contours

- In a 3D presentation the contours show up as discontinuities lying on a smooth curve in the domain.

Problem description



Piecewise smooth images

$\mathcal{PS}^{\alpha,\beta}$ is the class of functions f such that there exists a curve $y = c(x)$, $c \in H^{\alpha}$, $\alpha \in (1,2]$, and two functions $f_1, f_2 \in H^{\beta}$, $\beta \in (1,2]$ such that

$$\mathcal{PS}^{\alpha,\beta} := \{f : [0,1]^2 \to \mathbb{R} \to \begin{cases} f_1 & \text{if } \ y \leq c(x) \\ f_2 & \text{otherwise} \end{cases} .$$

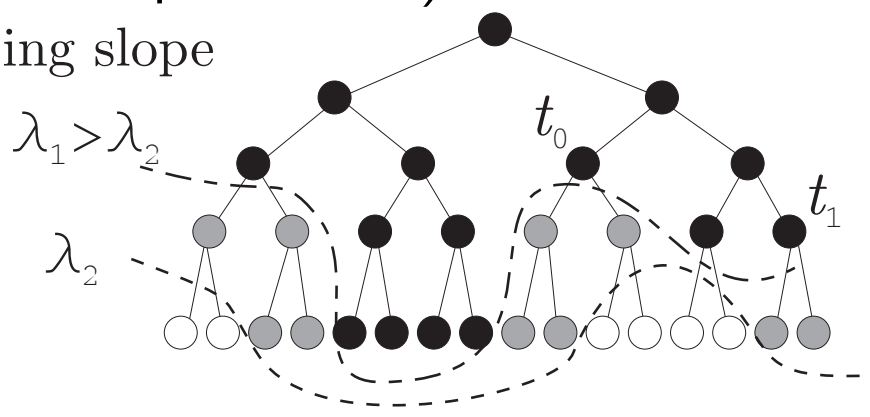
Two approaches

1. nonlinear approximation from fixed representations
   - 1.1 Wavelets
   - 1.2 Curvelets
   - 1.3 Wedgelets
2. nonlinear approximation from nonlinear spaces (highly nonlinear)
   - 2.1 Bandelets
   - 2.2 Binary Space Partitions
   - 2.3 Tin approaches (decimation techniques)
   - 2.4 ...

Adaptive partitions

Nested partitions

- easy do handle (tree structures)
- easy to optimize if the tree functionals are monotonic (CART-like procedures)



How to model $\mathcal{PS}$ images?

- Exist of smooth regions and (smoothly aligned) point singularities
  - Classical Besov spaces
  - Characterized by nonlinear wavelet approximation (Jackson & Berstein estimates)
    Defined over RDP.
  - Approximation results: if $f \in \mathcal{B}^{\alpha}_{2,q}$ $(1/q = 1/\alpha + 1/2)$

$$\sigma_n(f) = \mathcal{O}(n^{-1/2})$$

where $\alpha < 1$ for $f \in \mathcal{PS}$

How to model $\mathcal{PS}$ images?

- For 2D with smooth line singularities:
  - geometric **B**-spaces (Karaivanov, Nonlinear piecewise polynomial approximation beyond Besov spaces, Comput. Harmon. Anal., 2003)
  - characterized by Piecewise polynomial approximations
  - if $f \in \mathbf{B}^{\alpha}$ over triangulation $\mathcal{T}$

$$\tilde{\sigma}_n(f, \mathcal{T}) = \mathcal{O}(n^{-\alpha})$$

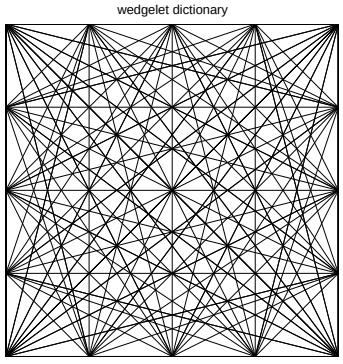
$$\tilde{\sigma}_n(f) = \inf_{\mathcal{T}} \tilde{\sigma}_n(f, \mathcal{T})$$

Finding the optimal triangulation (if it exists) giving the highest α is the hardest part!

How to create these partitions?

Finding the 'best' triangulation is an NP-complete problem.
Most used solutions

1. use Recursive Dyadic Partitions and prune
2. decorate each dyadic square at the highest level with a line split (Wedgelets)



3. use Recursive binary splitting

$\longrightarrow$ computational intensive (BSP, top-down Wedgelet)

Approximations vs. Compression

From approximation, i.e. number of terms, viewpoint:

- Adaptive and nonlinear approximation are great!
- `We only need a small number of atoms to have an accurate approximation, i.e.`

$$\sigma_n(f) = \|f - f_n\| = \mathcal{O}(n^{-\alpha})$$

`with certain regularity conditions on` f`.`

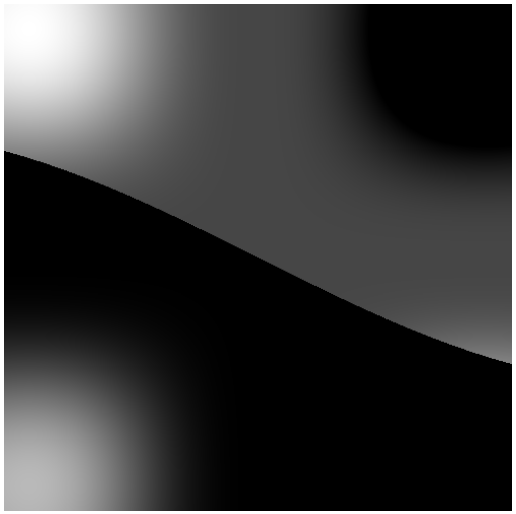
From compression viewpoint:

- `Which atoms do you need?`
  - for fixed partitions $\longrightarrow$ 'selected' atoms have to be stored
- `What does their support looks like?`
  - for adaptive partitions $\longrightarrow$ 'vertices' and 'edges' have to be encoded somehow

Observation

- Since we have to store the domain partition explicitly $\longrightarrow$ why not combine this information to give us also information about the graph of the function?
- Is it possible to reconstruct the surface from partition information?

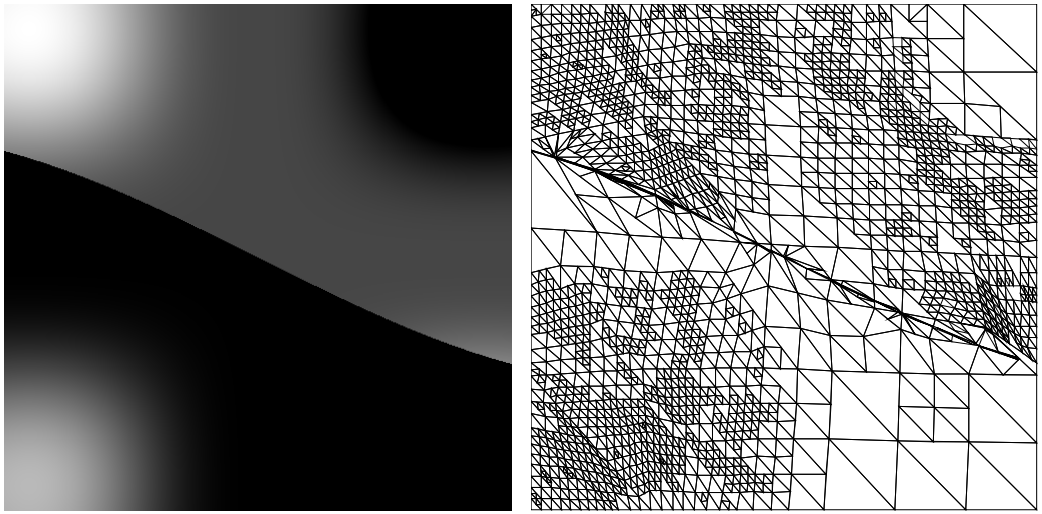
i.e.,



Observation

- Since we have to store the domain partition explicitly $\longrightarrow$ why not combine this information to give us also information about the graph of the function?
- Is it possible to reconstruct the surface from partition information?

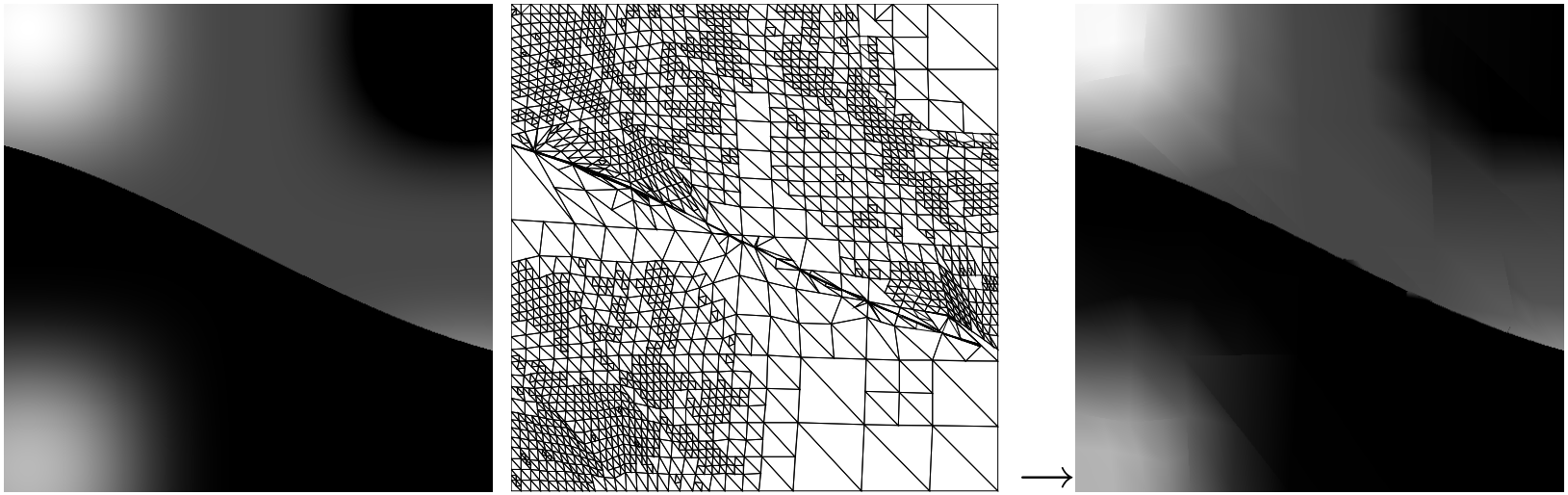
i.e.,



Observation

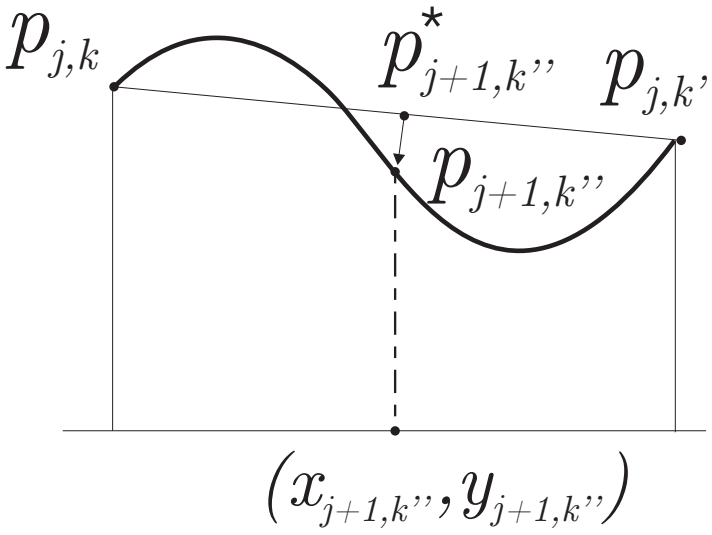
- Since we have to store the domain partition explicitly $\longrightarrow$ why not combine this information to give us also information about the graph of the function?
- Is it possible to reconstruct the surface from partition information?

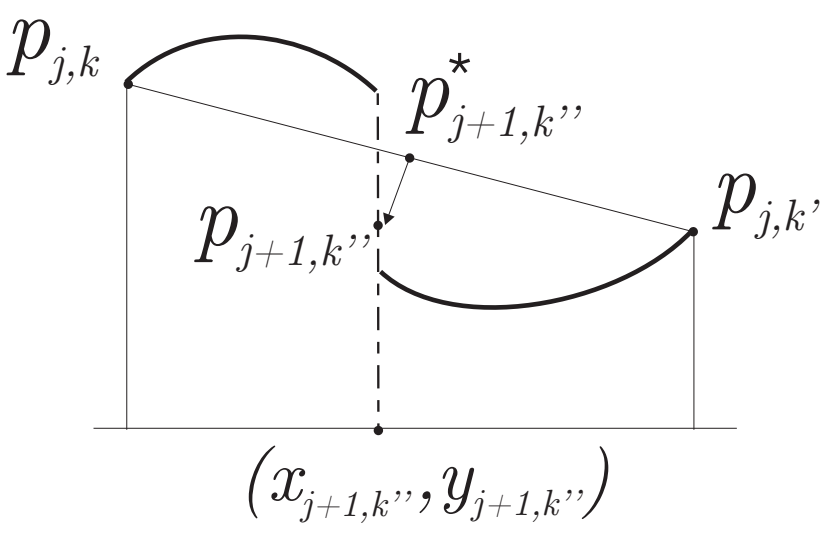
i.e.,



Normal approximations

Normal approximations:
use only scalars to fix the partition and function values

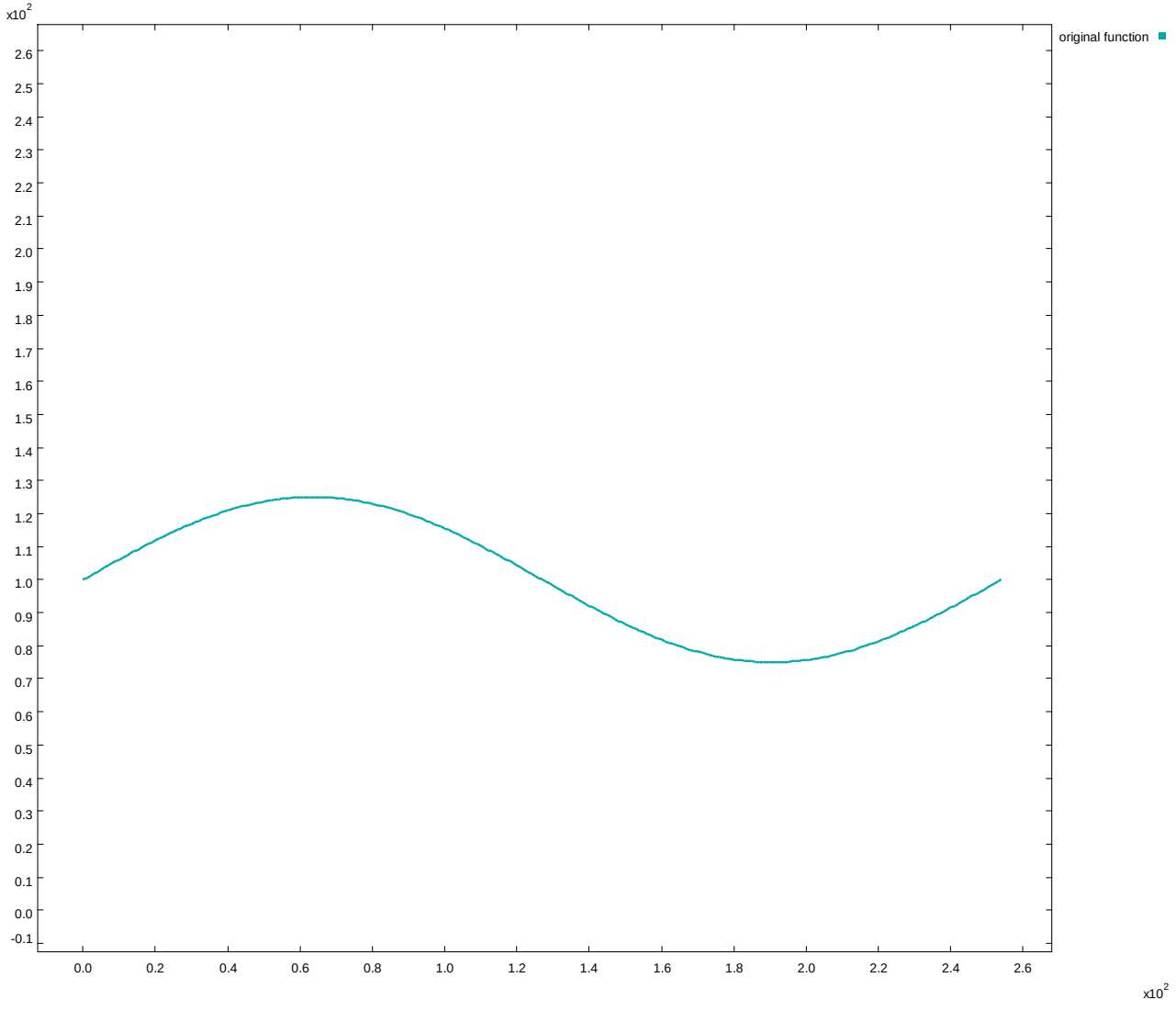




Information about 'what happens' & 'where it happens'. (Jansen et. al, Approximation of Piecewise Smooth Two-Dimensional Functions using Normal Triangulated Meshes, Appl. Comp. Harm. Anal., 2005)

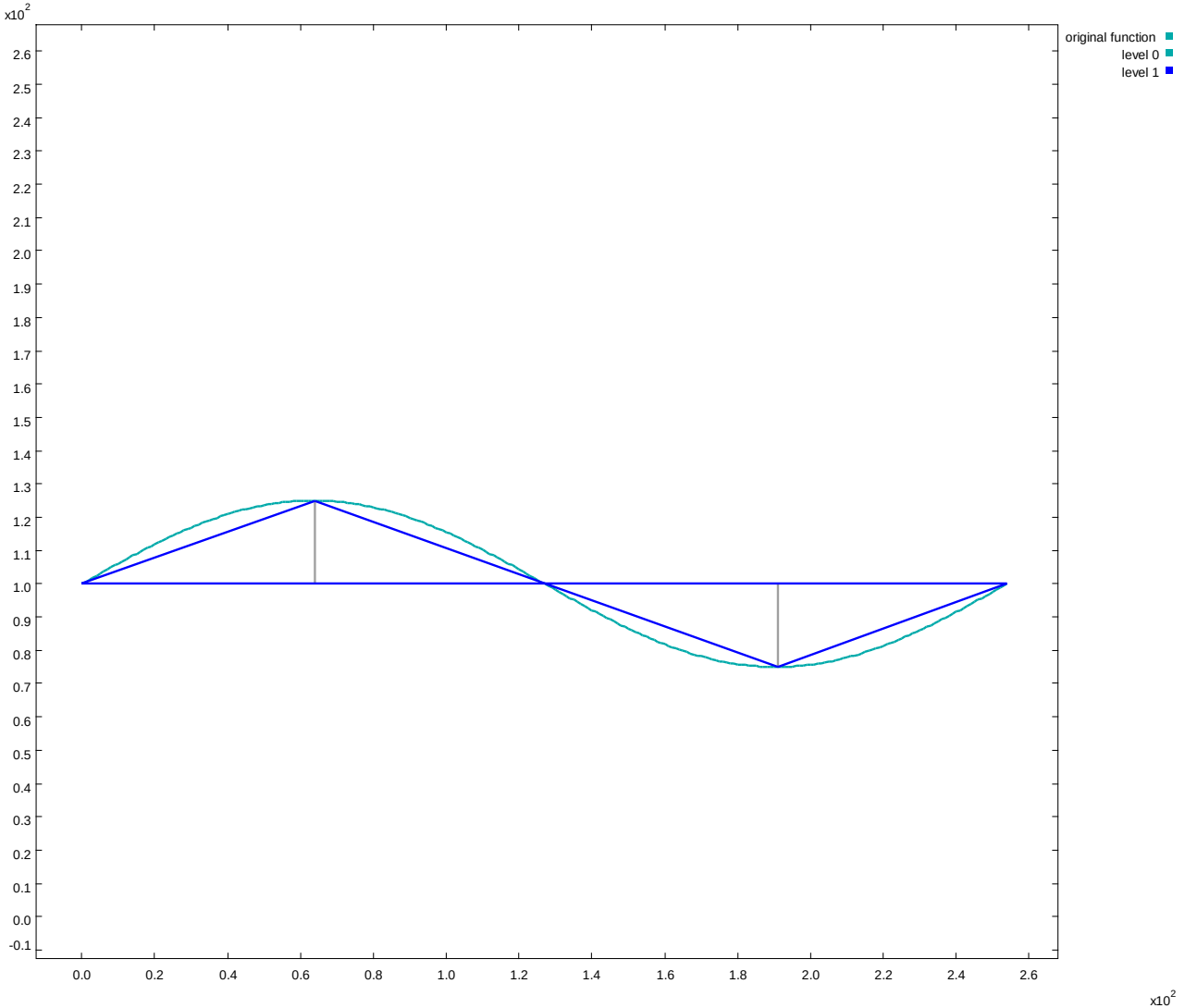
Smooth functions

(Daubechies et. al, Normal Polyline Approximation, Constructive Approximation, 2004, 20, p399-463)



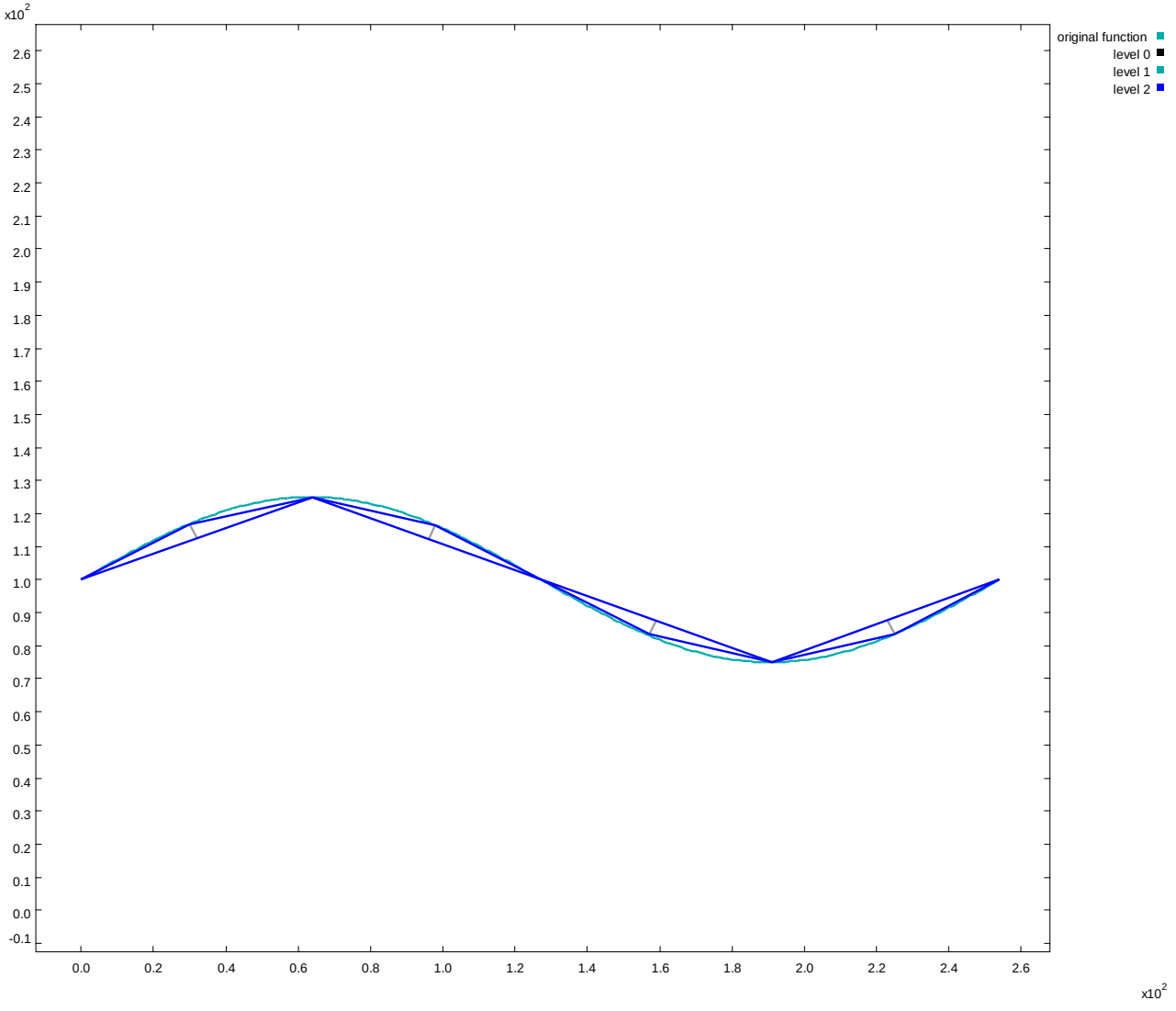
Smooth functions

(Daubechies et. al, Normal Polyline Approximation, Constructive Approximation, 2004, 20, p399-463)



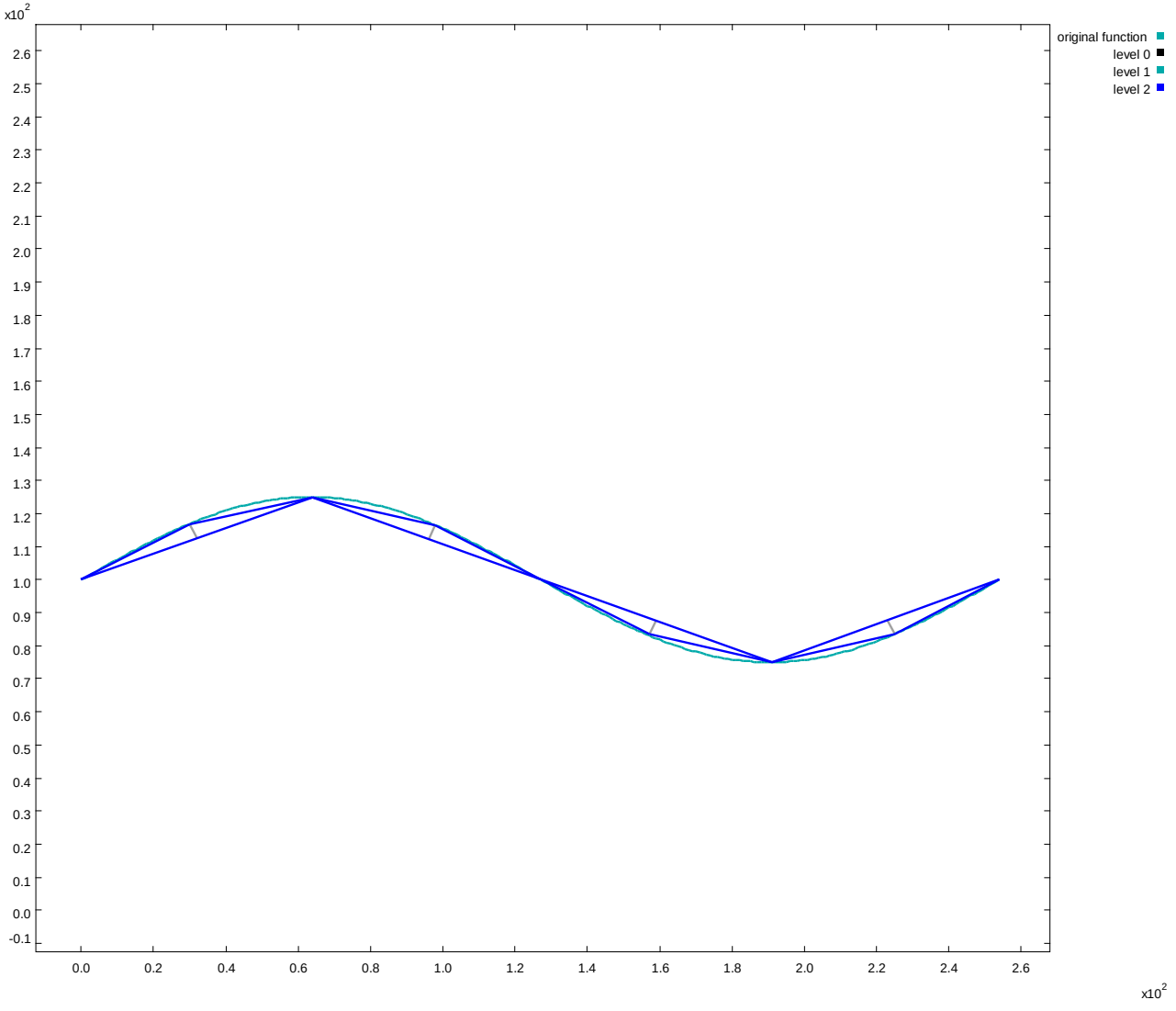
Smooth functions

(Daubechies et. al, Normal Polyline Approximation, Constructive Approximation, 2004, 20, p399-463)



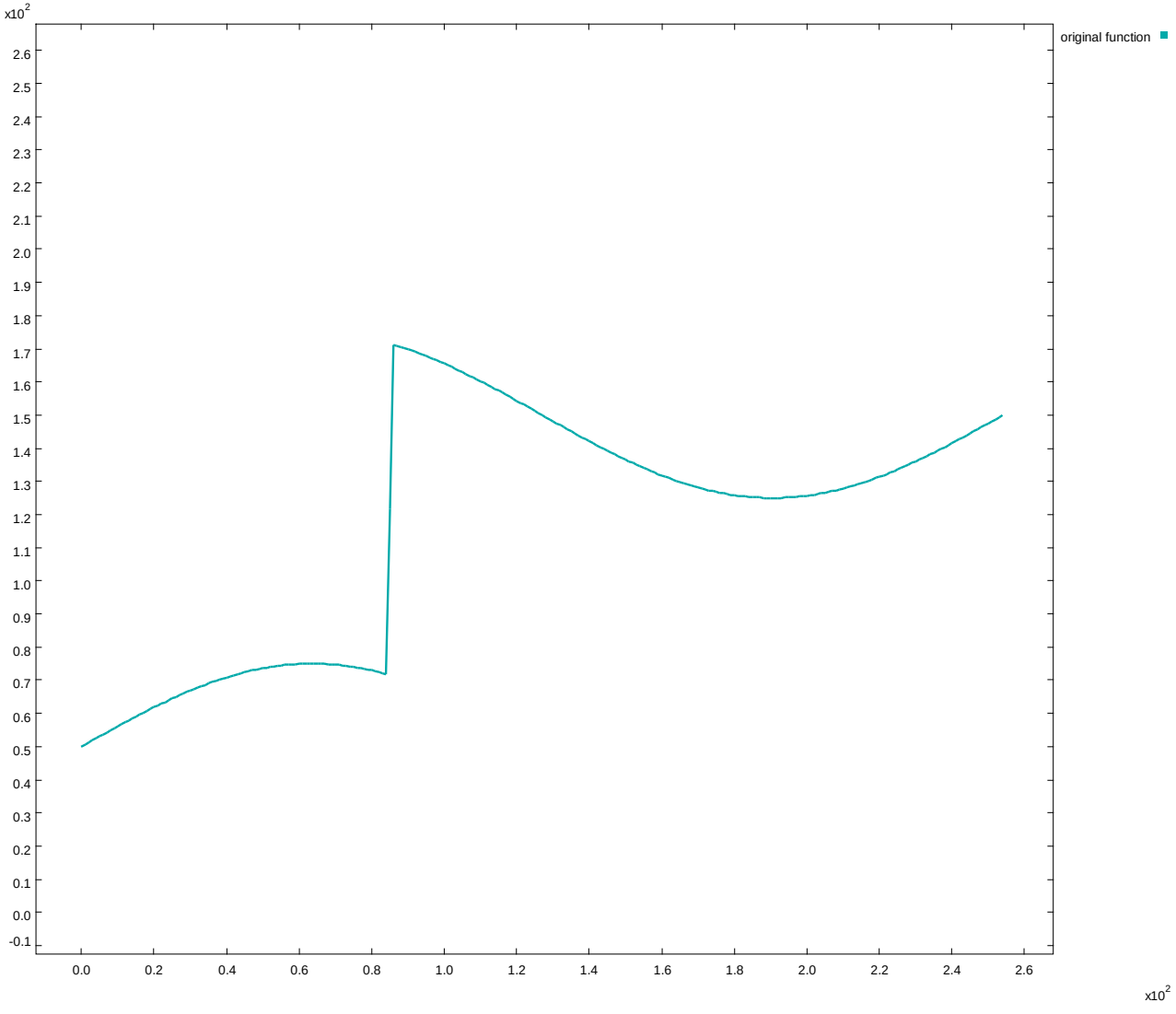
Smooth functions

(Daubechies et. al, Normal Polyline Approximation, Constructive Approximation, 2004, 20, p399-463)



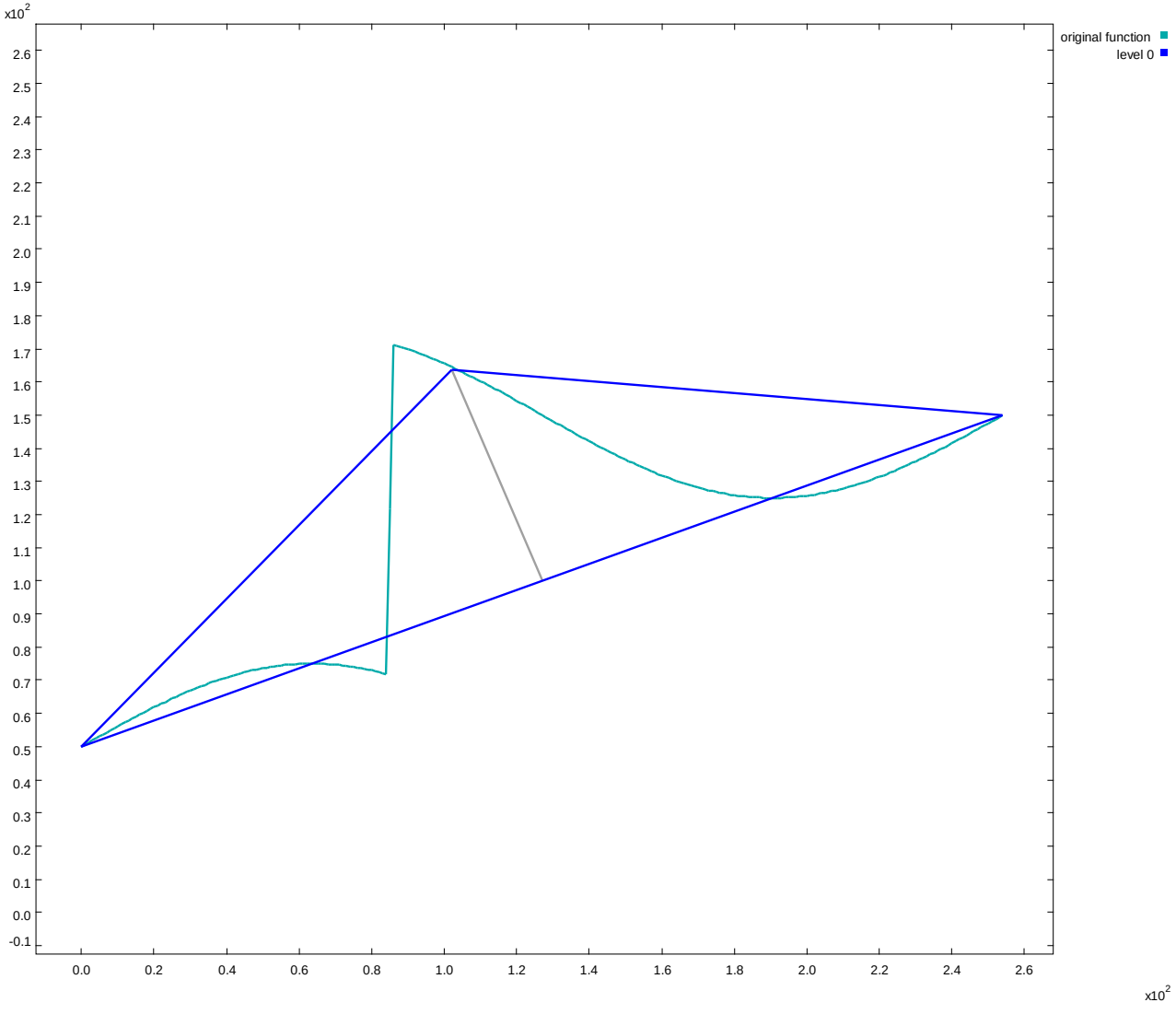
Non-smooth functions

- Partition the interval by shooting normal rays.



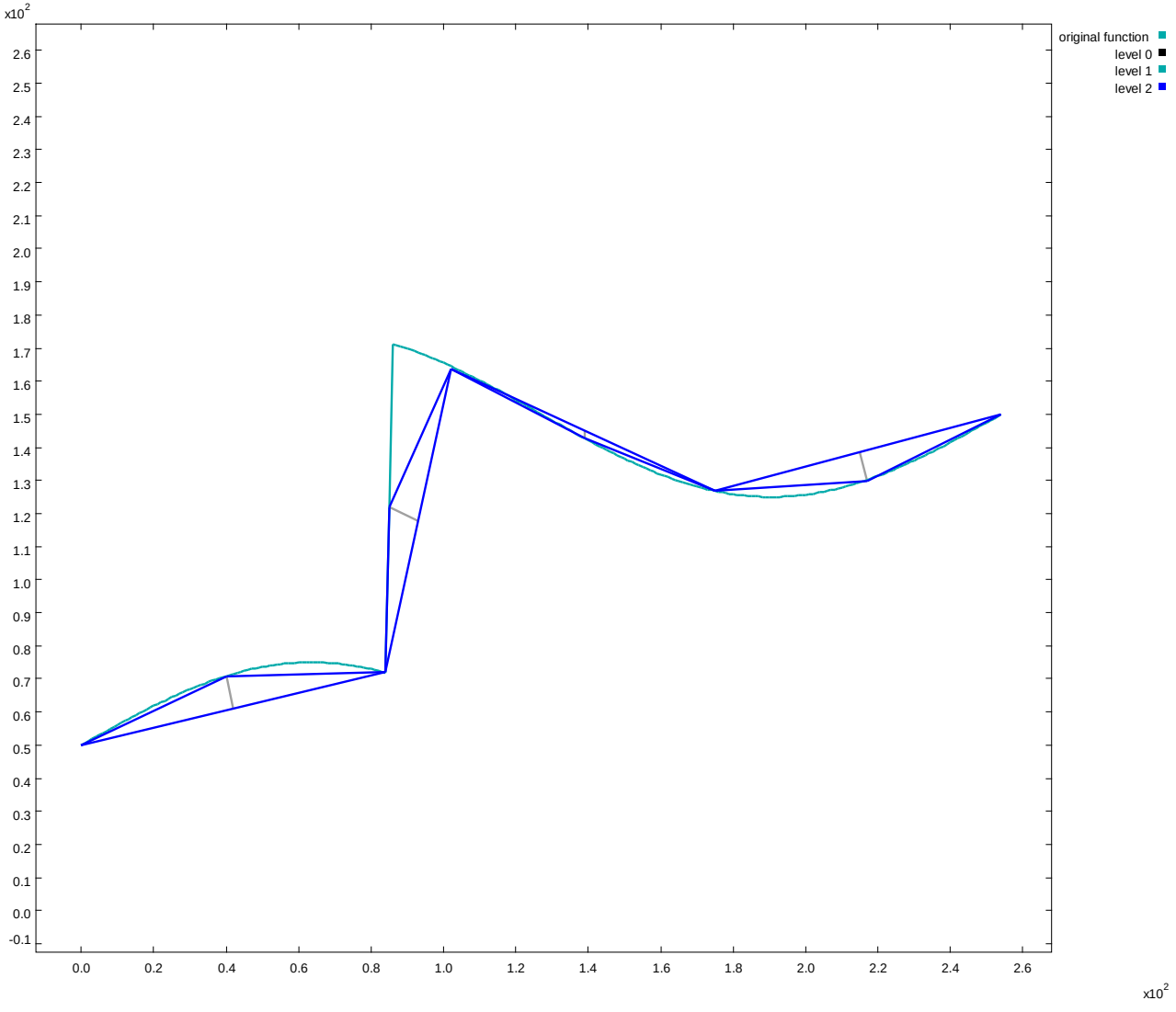
Non-smooth functions

- Partition the interval by shooting normal rays.



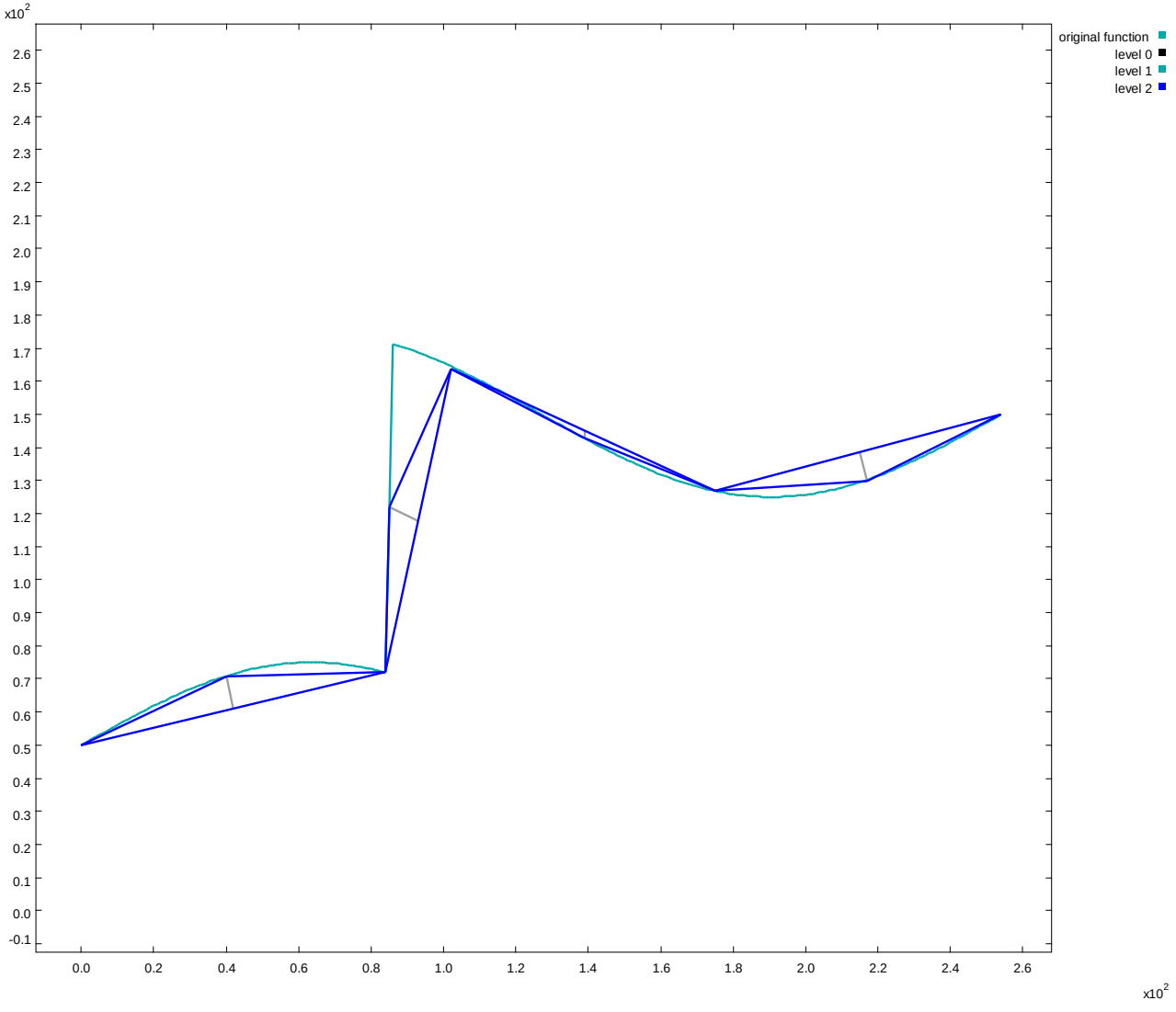
Non-smooth functions

- Partition the interval by shooting normal rays.



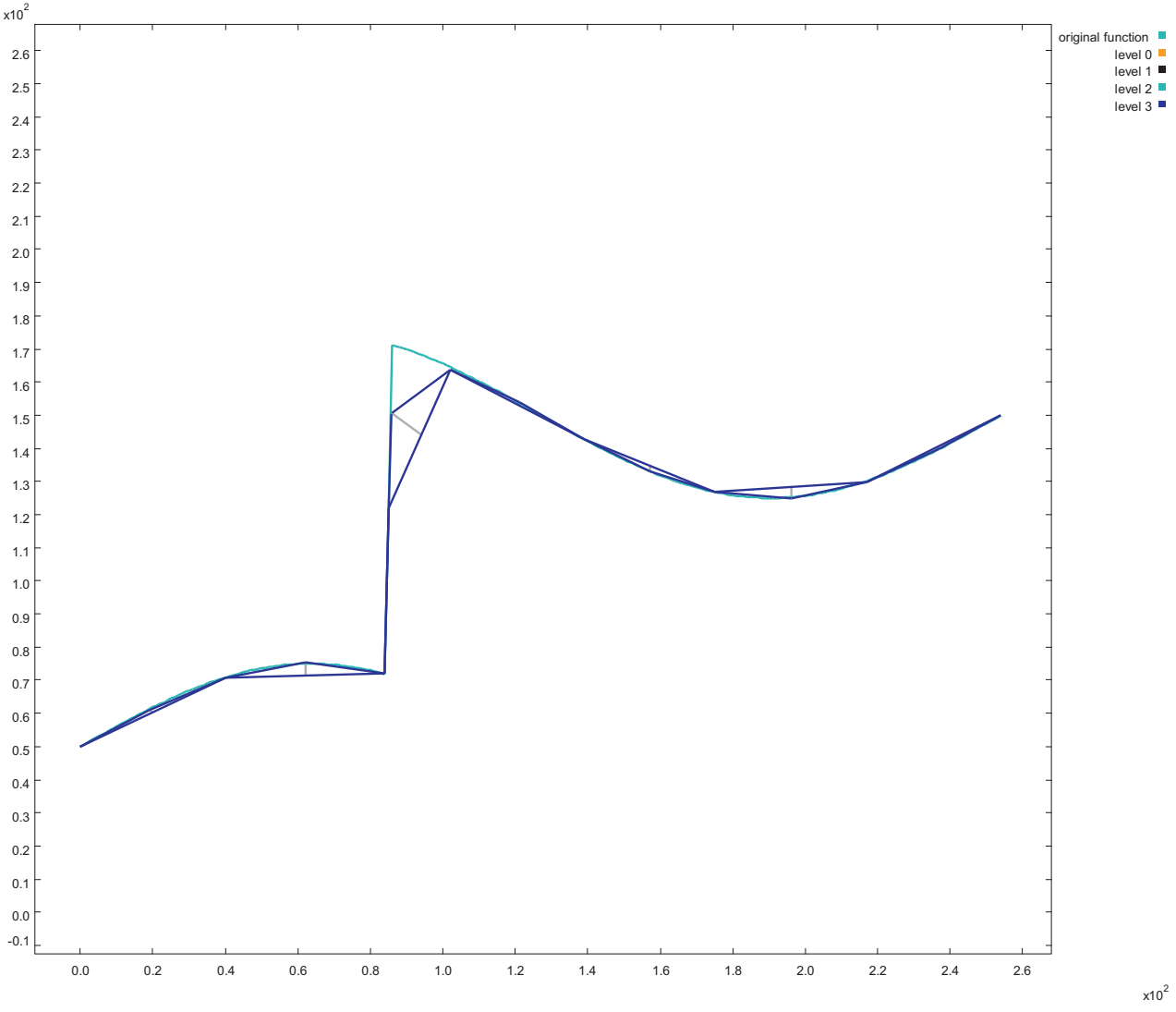
Non-smooth functions

- Partition the interval by shooting normal rays.



Non-smooth functions

- Partition the interval by shooting normal rays.



1D conclusions

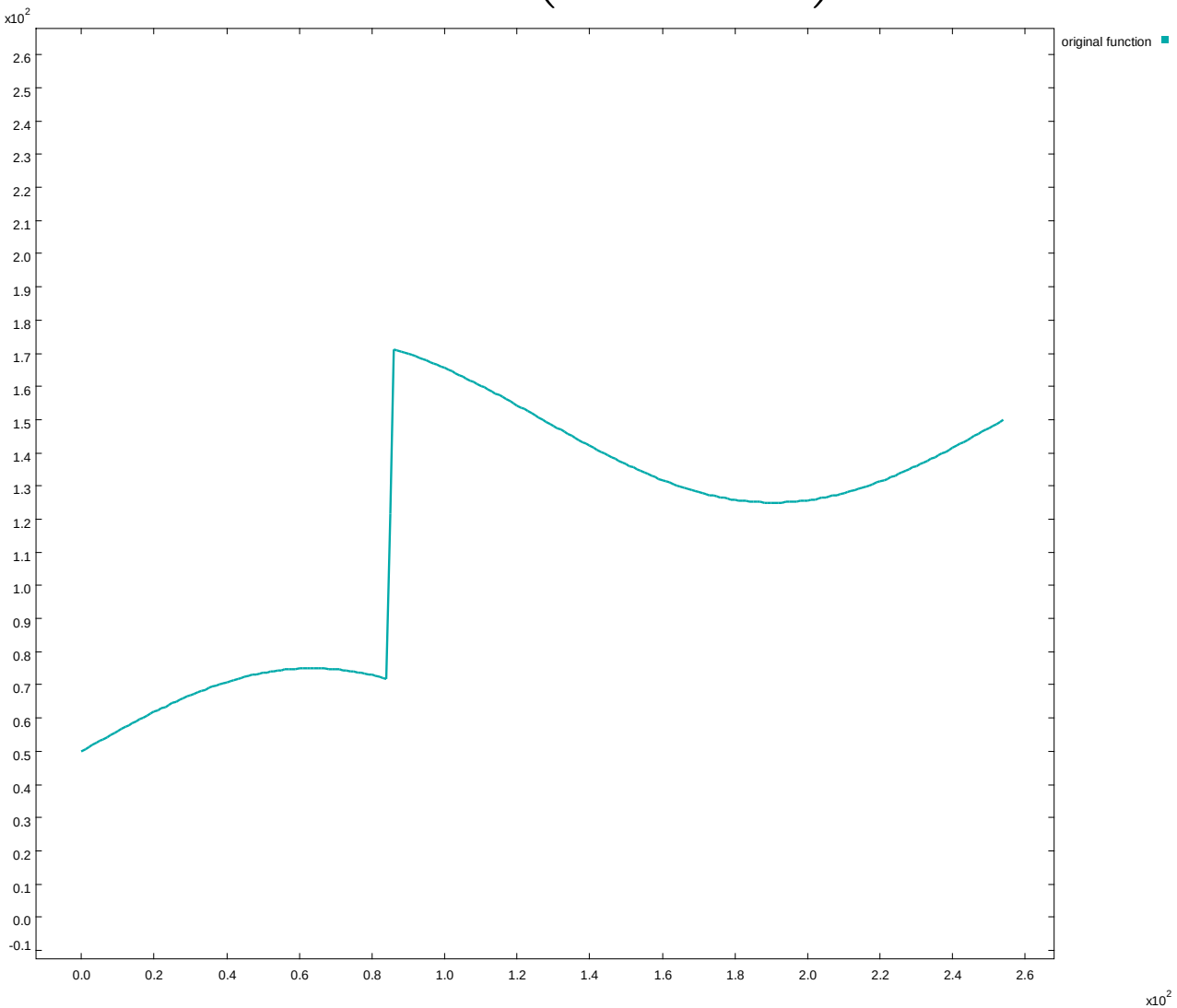
Normal approximations:

1. use only scalars to fix the partition and function values
   At smooth regions $\longrightarrow$ normal mesh is good approximation
2. Locates jumps rapidly. (Fast compared to binary space partitioning)
3. At low resolution levels $\longrightarrow$ strong aliasing effects.
   Closer to discontinuity $\longrightarrow$ introduce a 'polynomial wavelet'.

Polynomial wavelet

piecewise polynomial approximation

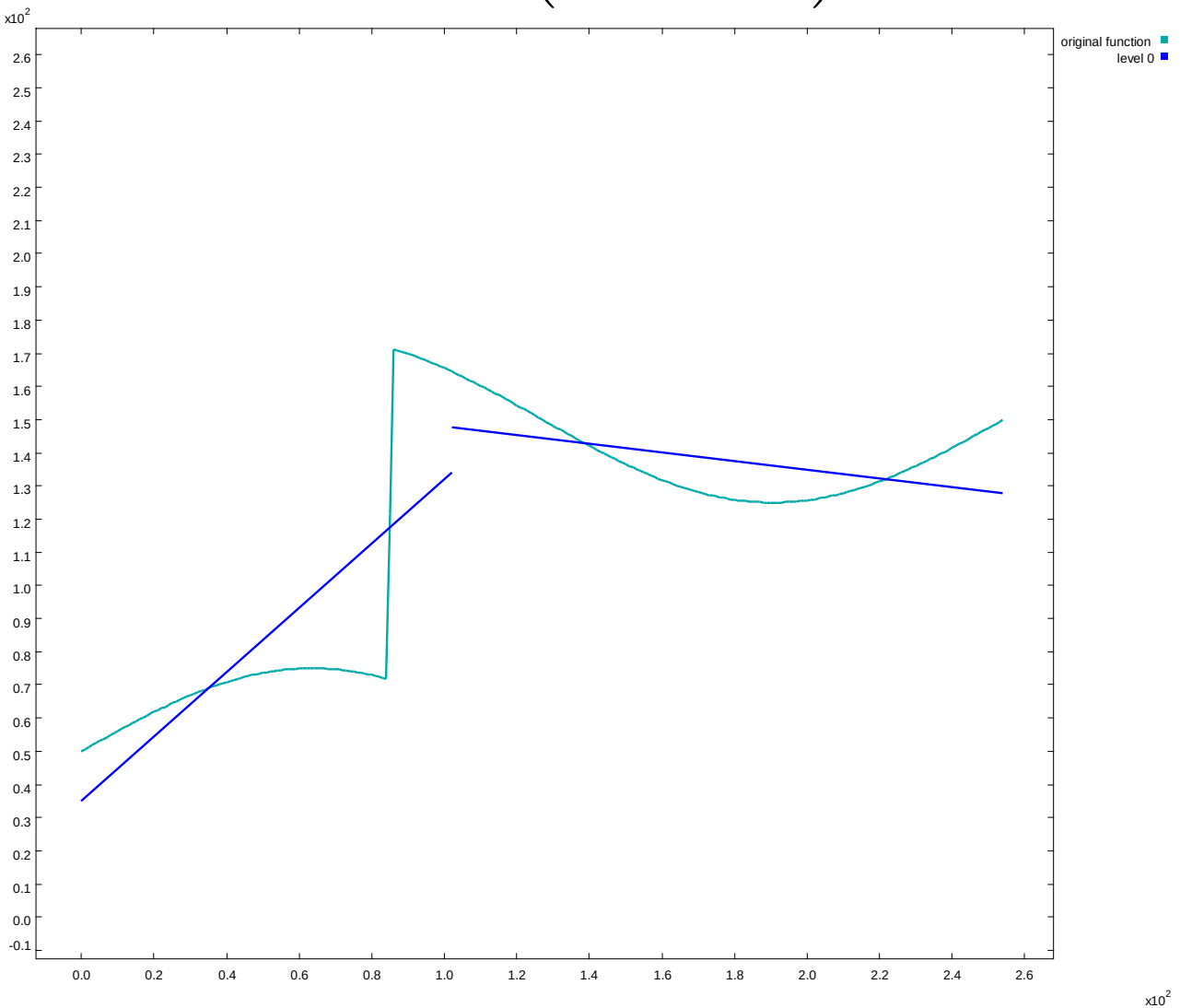
$$\psi_{I_{j,k}} := \left(P^*_{I_{j,k}} - P_{I_{j,k}} \right) \tag{1}$$



Polynomial wavelet

piecewise polynomial approximation

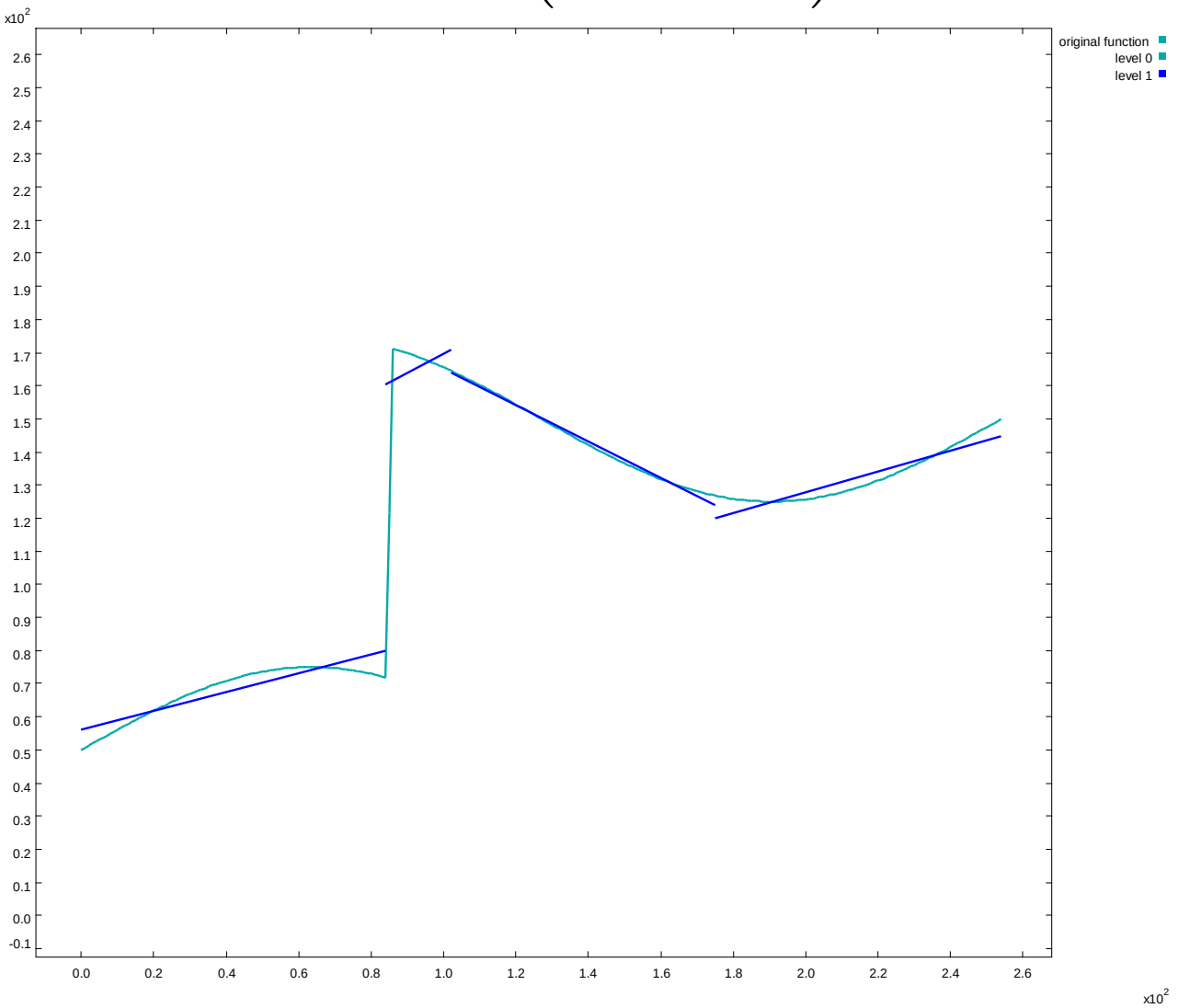
$$\psi_{I_{j,k}} := \left(P^*_{I_{j,k}} - P_{I_{j,k}} \right) \tag{1}$$



Polynomial wavelet

piecewise polynomial approximation

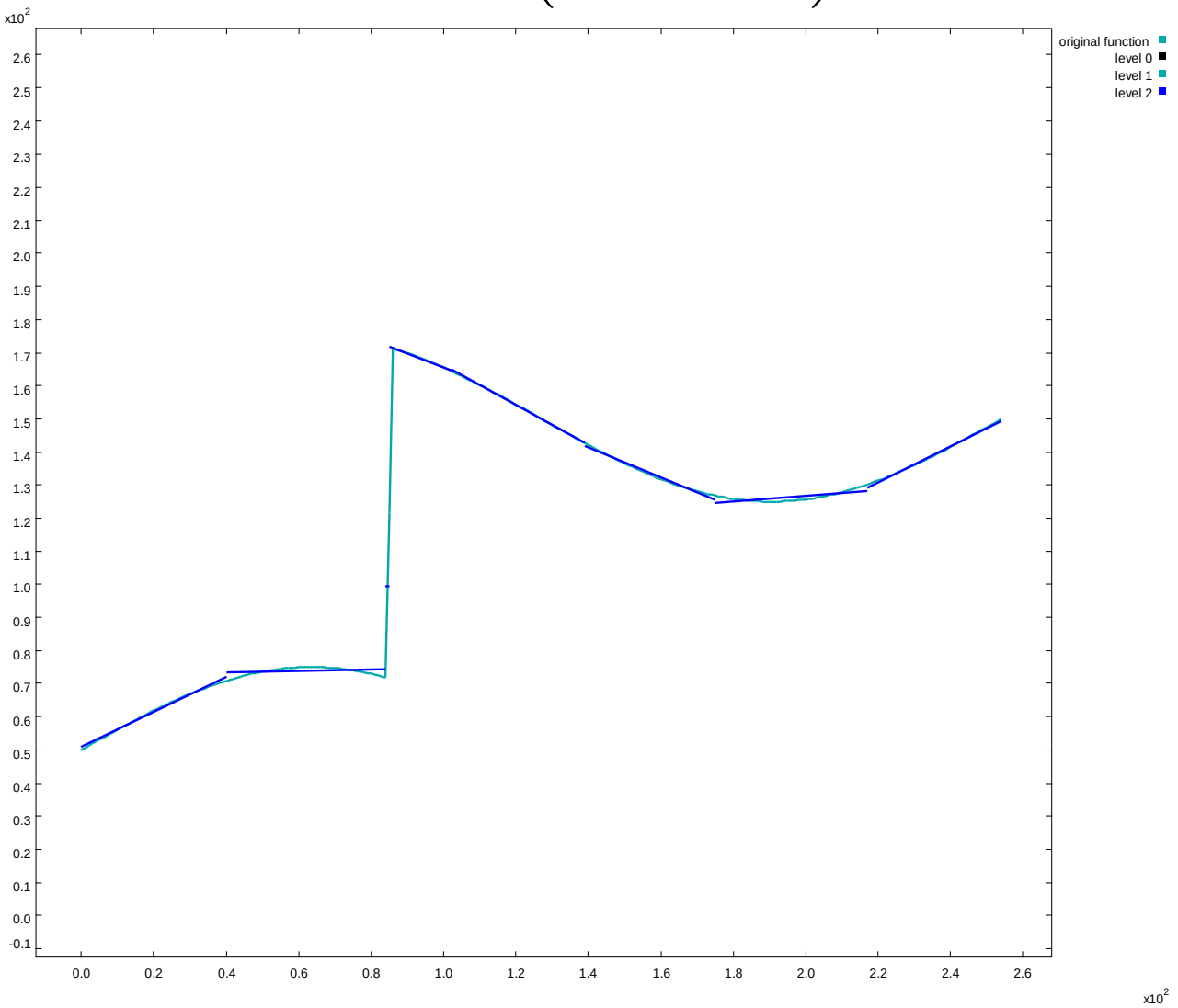
$$\psi_{I_{j,k}} := \left(P^*_{I_{j,k}} - P_{I_{j,k}} \right) \tag{1}$$



Polynomial wavelet

piecewise polynomial approximation

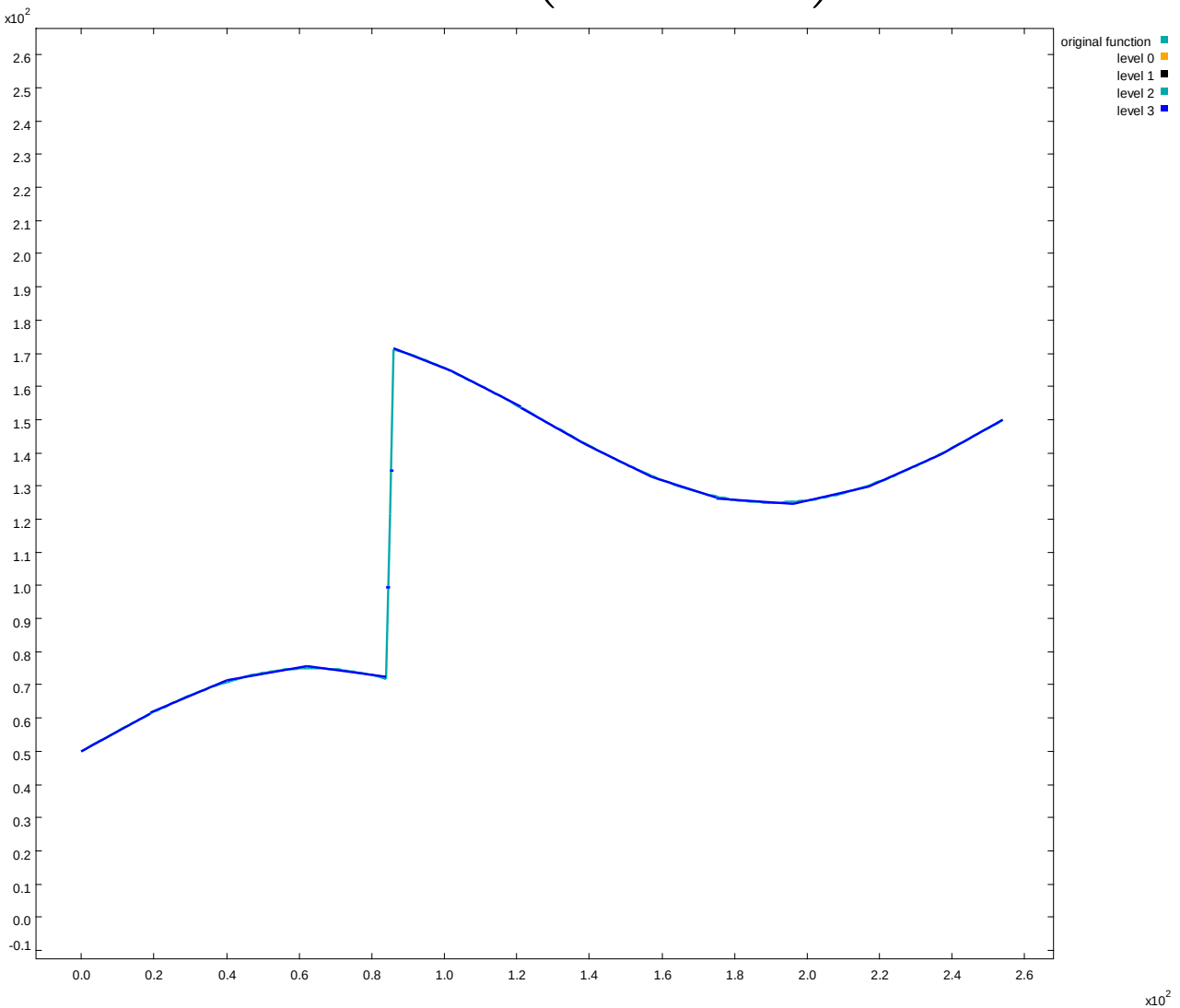
$$\psi_{I_{j,k}} := \left(P^*_{I_{j,k}} - P_{I_{j,k}} \right) \tag{1}$$



Polynomial wavelet

piecewise polynomial approximation

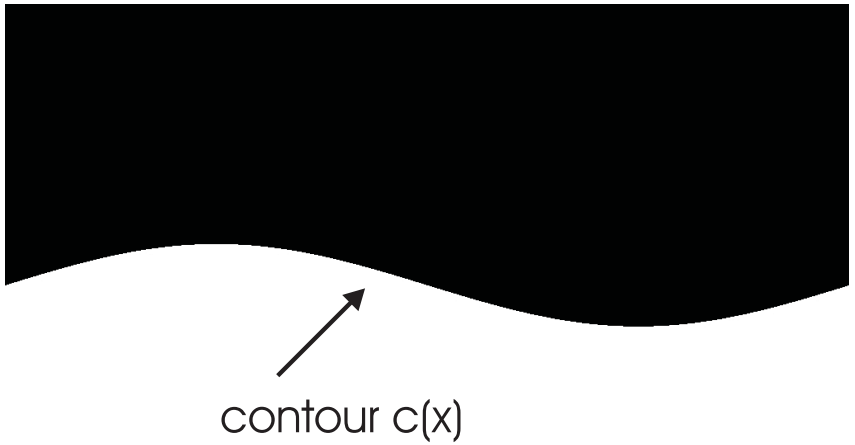
$$\psi_{I_{j,k}} := \left(P^*_{I_{j,k}} - P_{I_{j,k}} \right) \tag{1}$$



From 1D to 2D

1D Nothing to gain from approximation point of view since wavelets perform optimal wrt. point singularities.

2D In 2D not only point singularities, but also line discontinuities (contours)

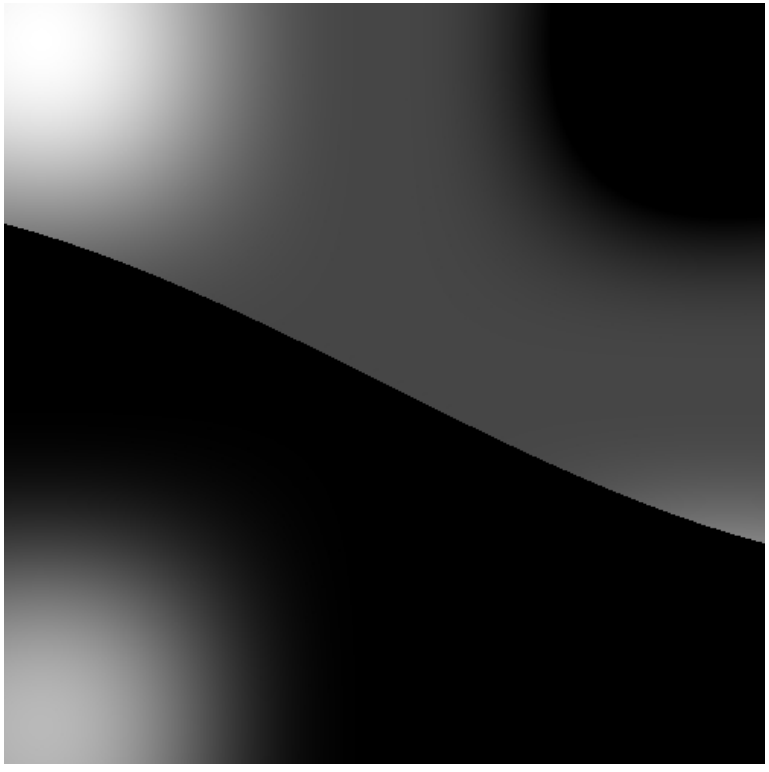


2D normal offsets

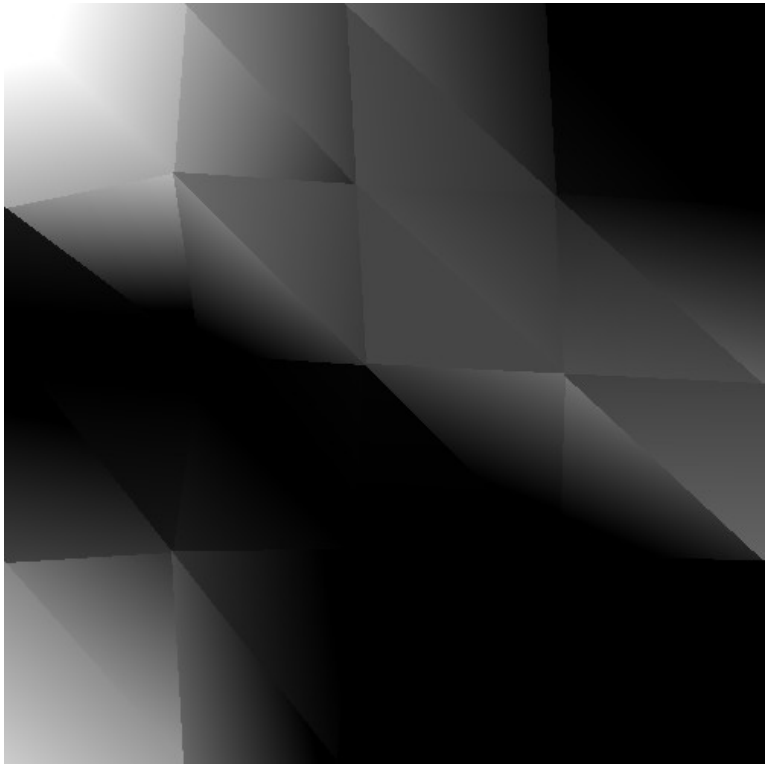
Ingredients:

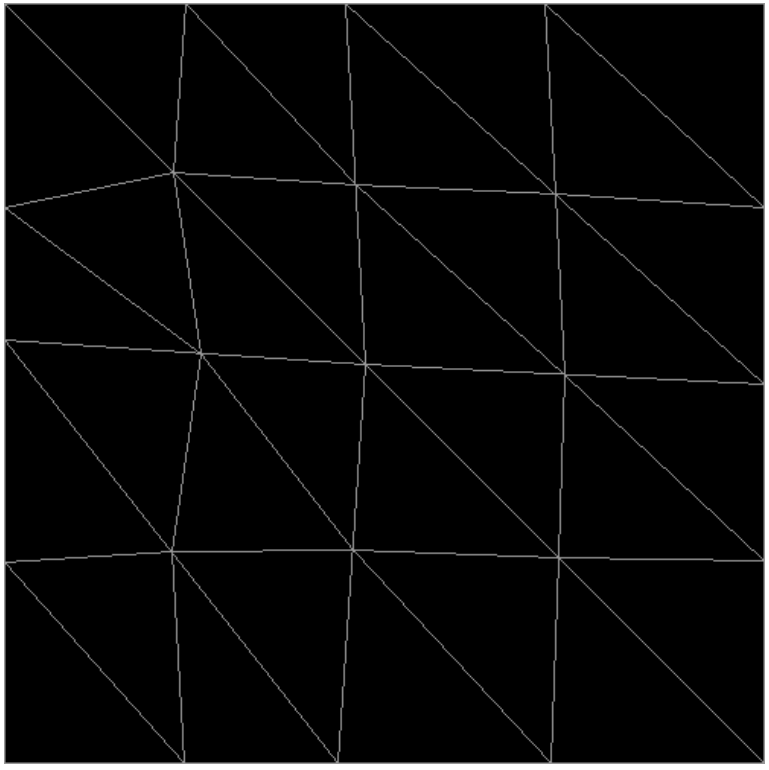
- top down refinement strategy.
- Edge refinement.
- Adaptive tesselation.
  $\longrightarrow$ irregular nested triangulations
- Tree pruning $\longrightarrow$ creating non-conformal triangulations

2D normal offsets

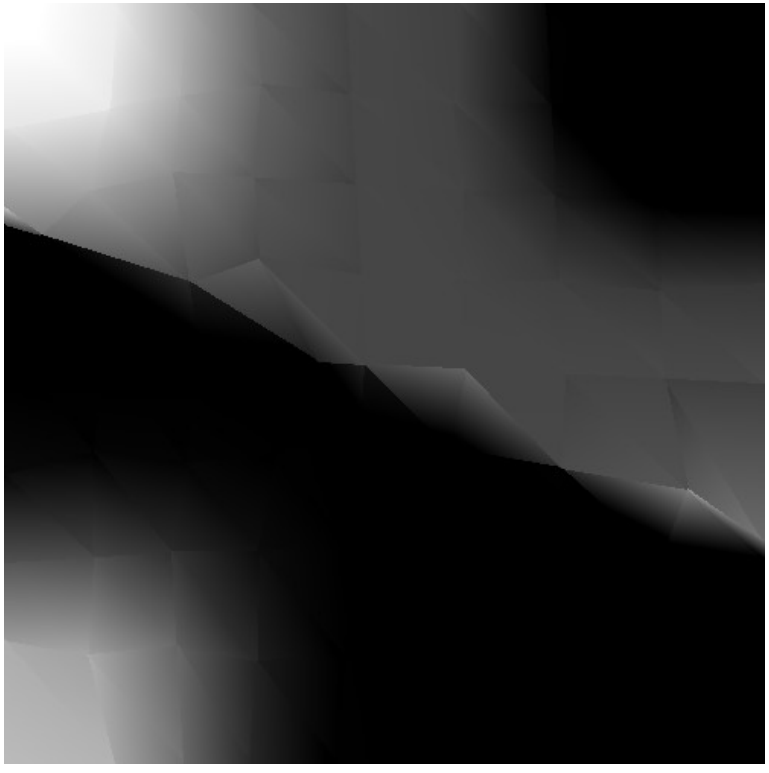


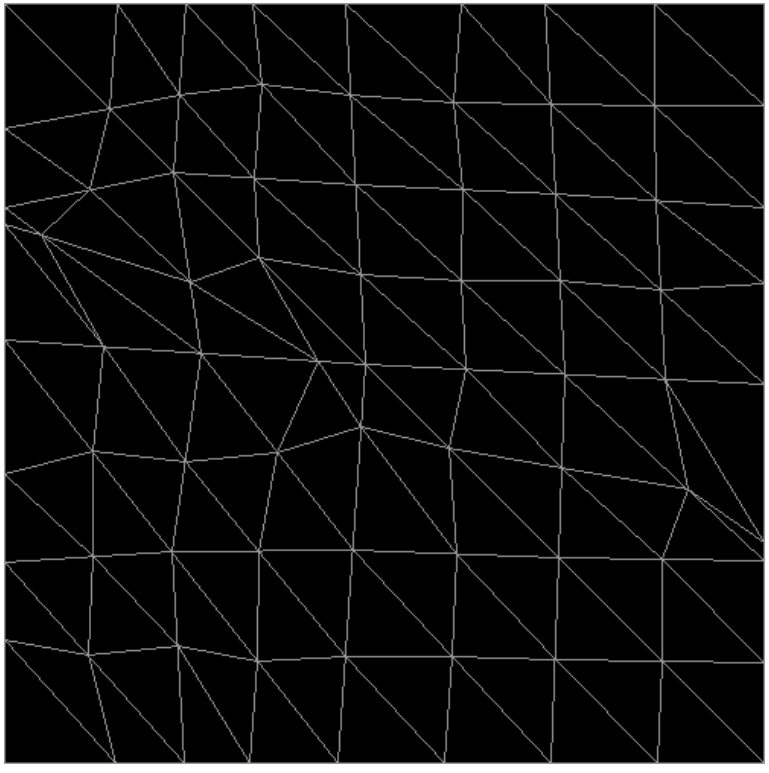
2D normal offsets



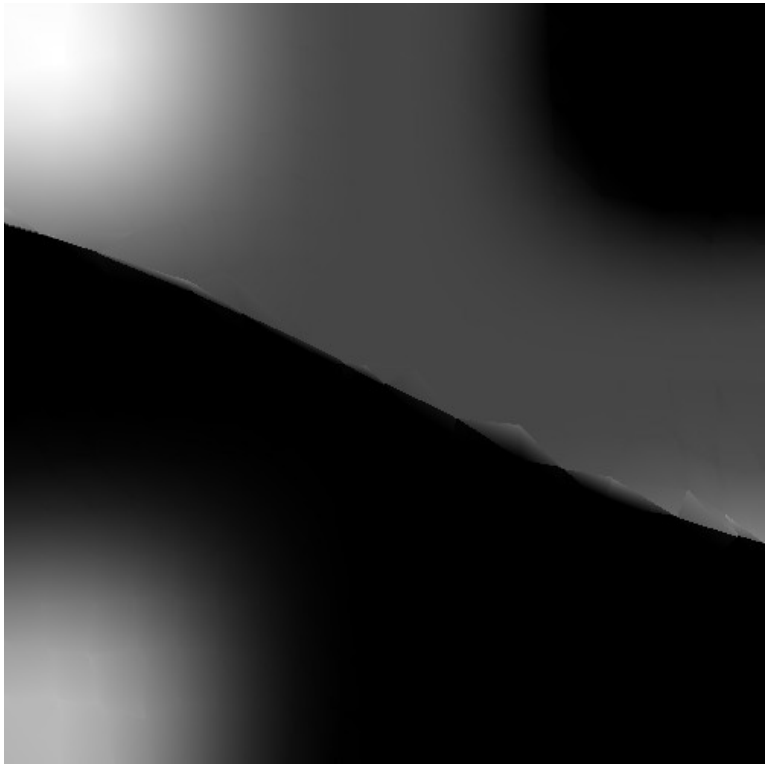


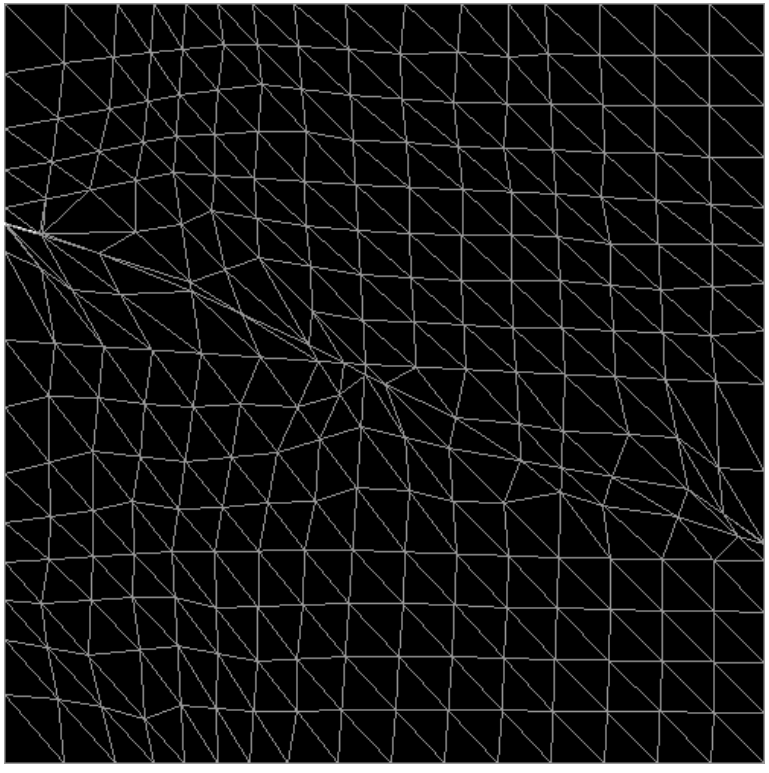
2D normal offsets



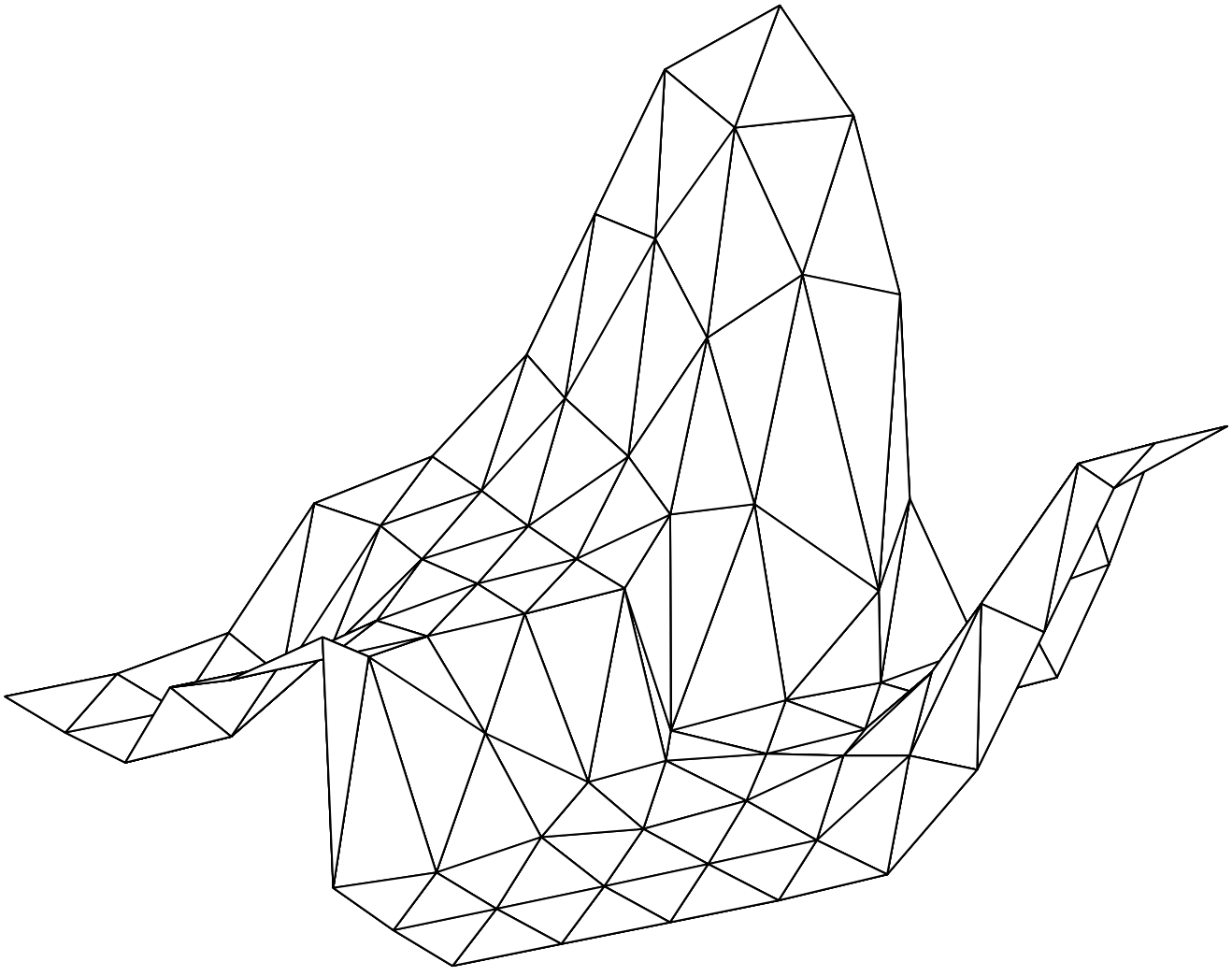


2D normal offsets

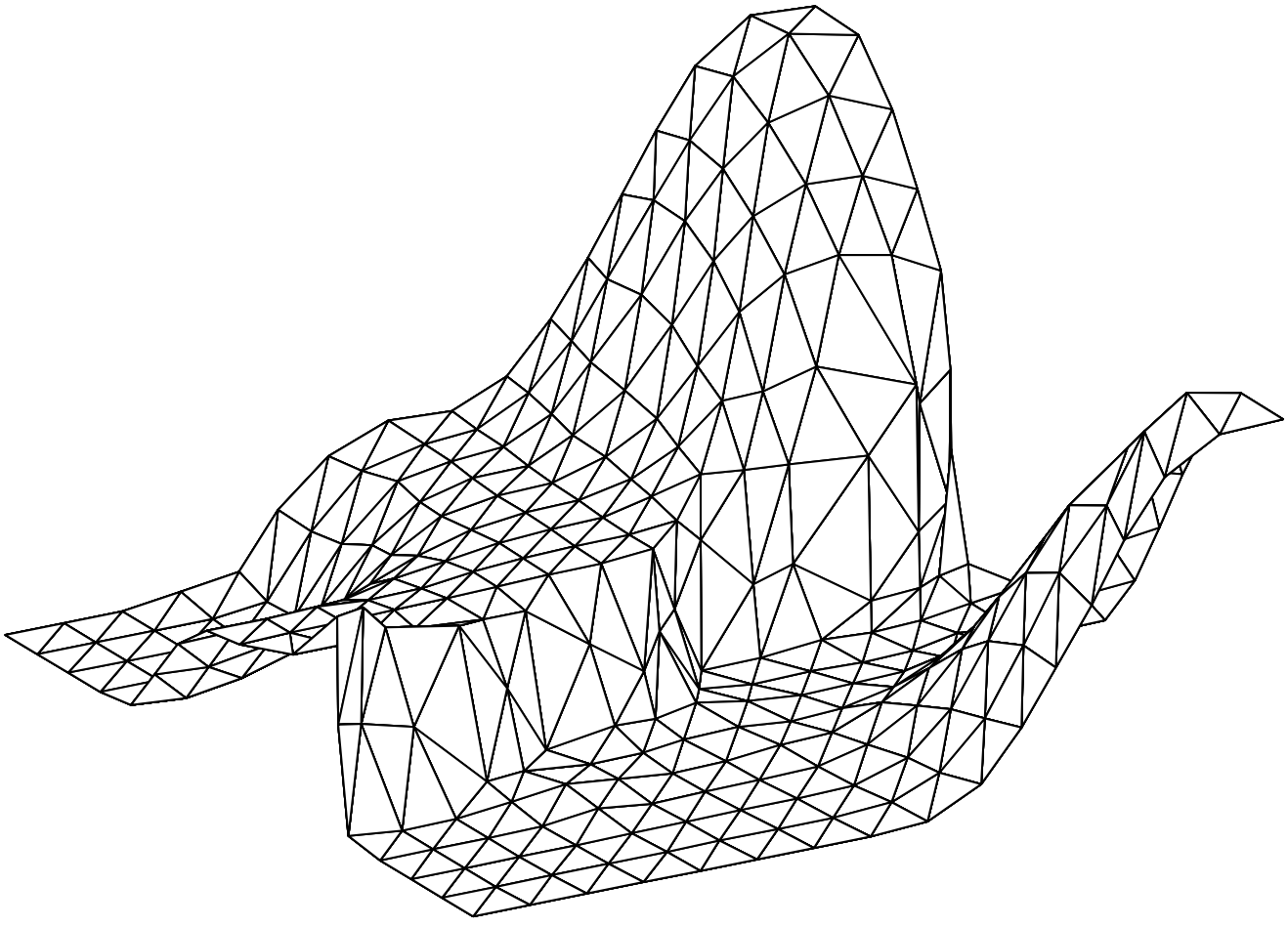




2D normal offsets



2D normal offsets



Pruning

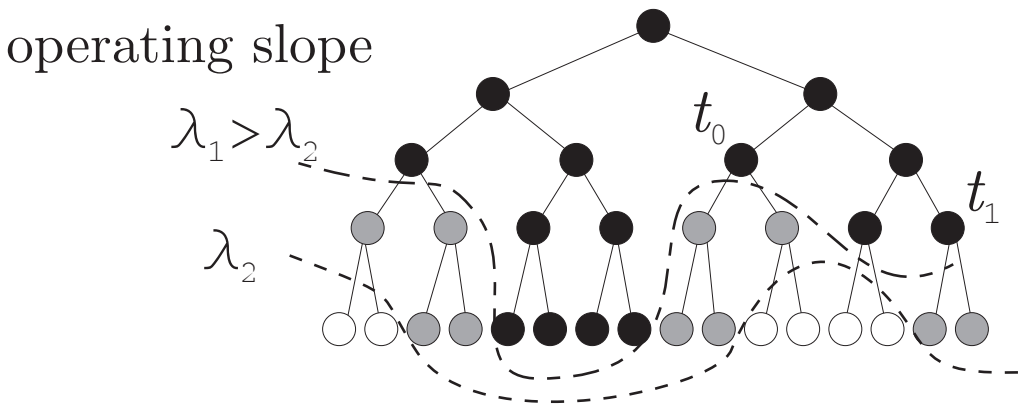
$$\lambda_t := -\frac{\Delta D_a(\mathcal{S}_t) + \Delta D_q(\mathcal{S}_t)}{\Delta R_\theta(\mathcal{S}_t) + \Delta R_q(\mathcal{S}_t)}. \tag{2}$$

where

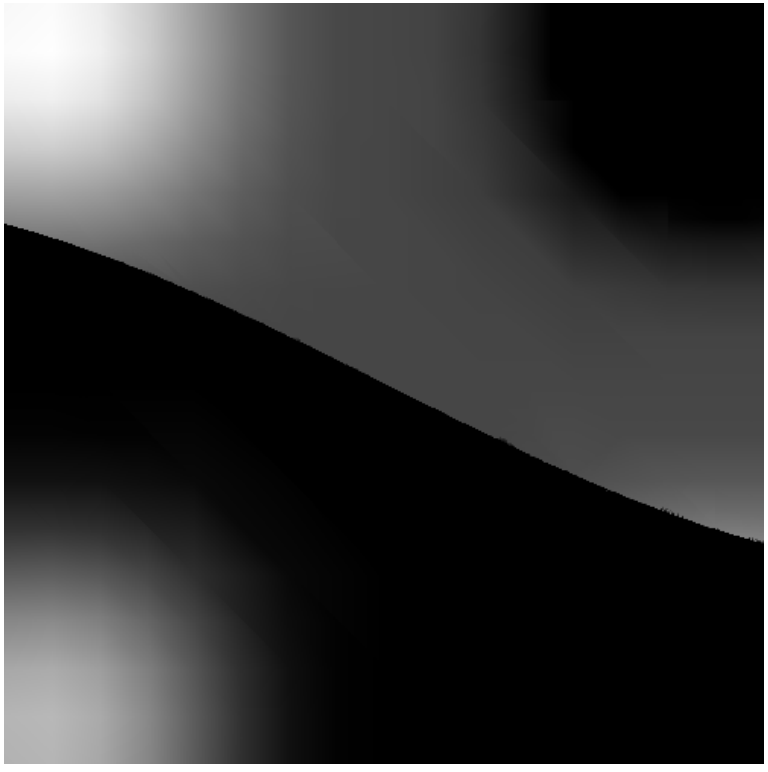
- $D_a(t, \psi) = E_{r-1}(f, \triangle(t))_2$, the approximation error at a node using a best polynomial approximation on $\triangle(t)$.
- $D_a(t, \mathcal{M}) := \|f - Q_\triangle(t)\|_{L_2(\triangle(t))}$, the *approximation* error at a node using the interpolating linear polynomial $Q_\triangle$.
- $\Delta D_a(\mathcal{S}_t, \mathcal{X})$ the difference of the distortion of the leave nodes and node t.
- $R_\theta(t)$ is the bit amount coupled with the subdivision of triangle $\triangle(t)$ (normal offsets + tesselation code).
- $R_q(t, \psi)$ is the bitrate coupled to *quantisation* of the polynomial wavelet at node t.
- $\Delta R_q(\mathcal{S}_t, \mathcal{X})$ is the bitrate coupled to *quantisation* of the polynomial wavelet at the leaves of $\mathcal{S}$.

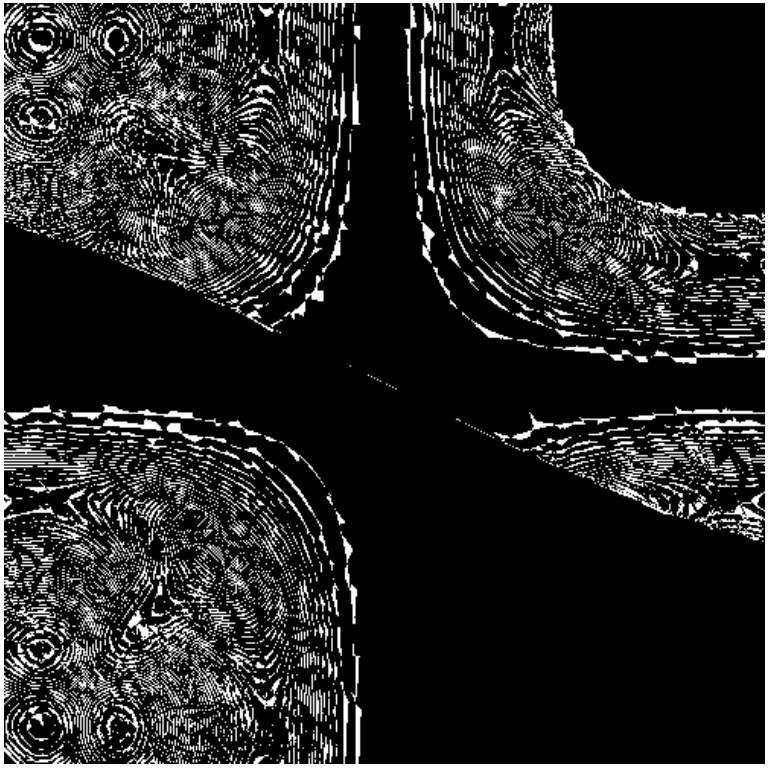
Pruning

$$\lambda_t := -\frac{\Delta D_a(\mathcal{S}_t) + \Delta D_q(\mathcal{S}_t)}{\Delta R_\theta(\mathcal{S}_t) + \Delta R_q(\mathcal{S}_t)}. \tag{2}$$



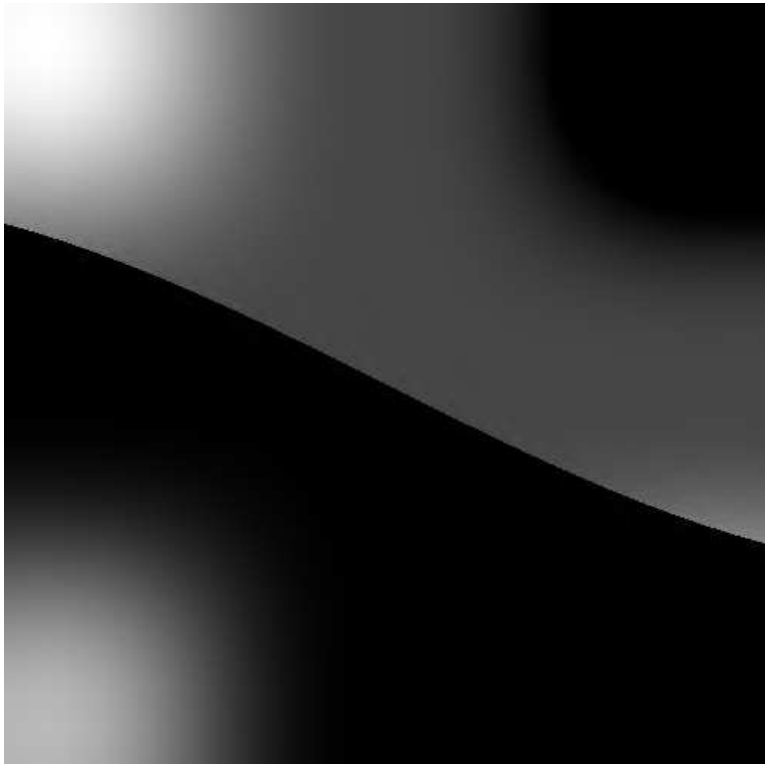
Comparison between JPEG and Normal Triangulation Compression

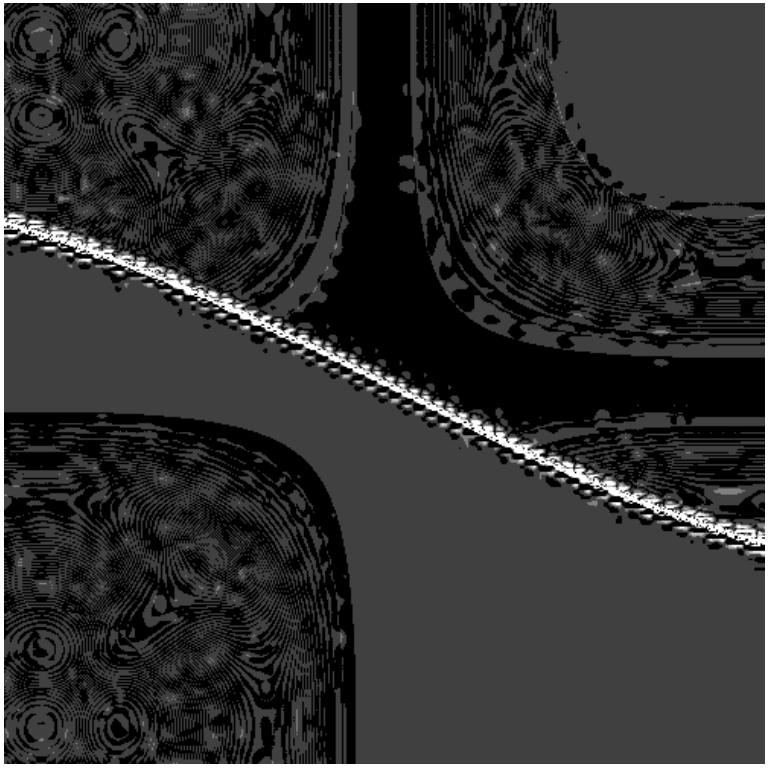




normal (interpolating) approximation (6000 offsets (2kB))

Comparison between JPEG and Normal Triangulation Compression





jpeg2000 (2kB)

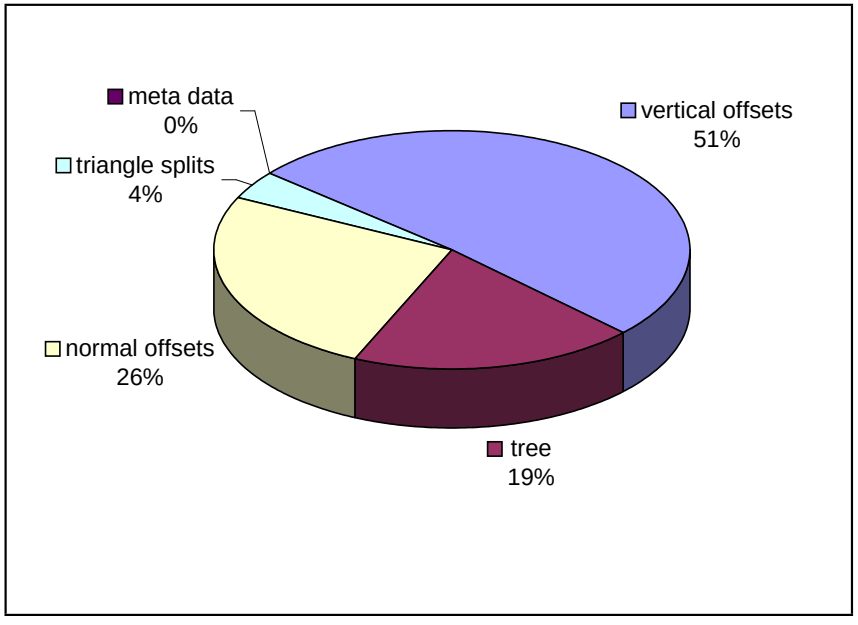
Reallife images



Piecewise smooth + Texture + Noise

Reallife images



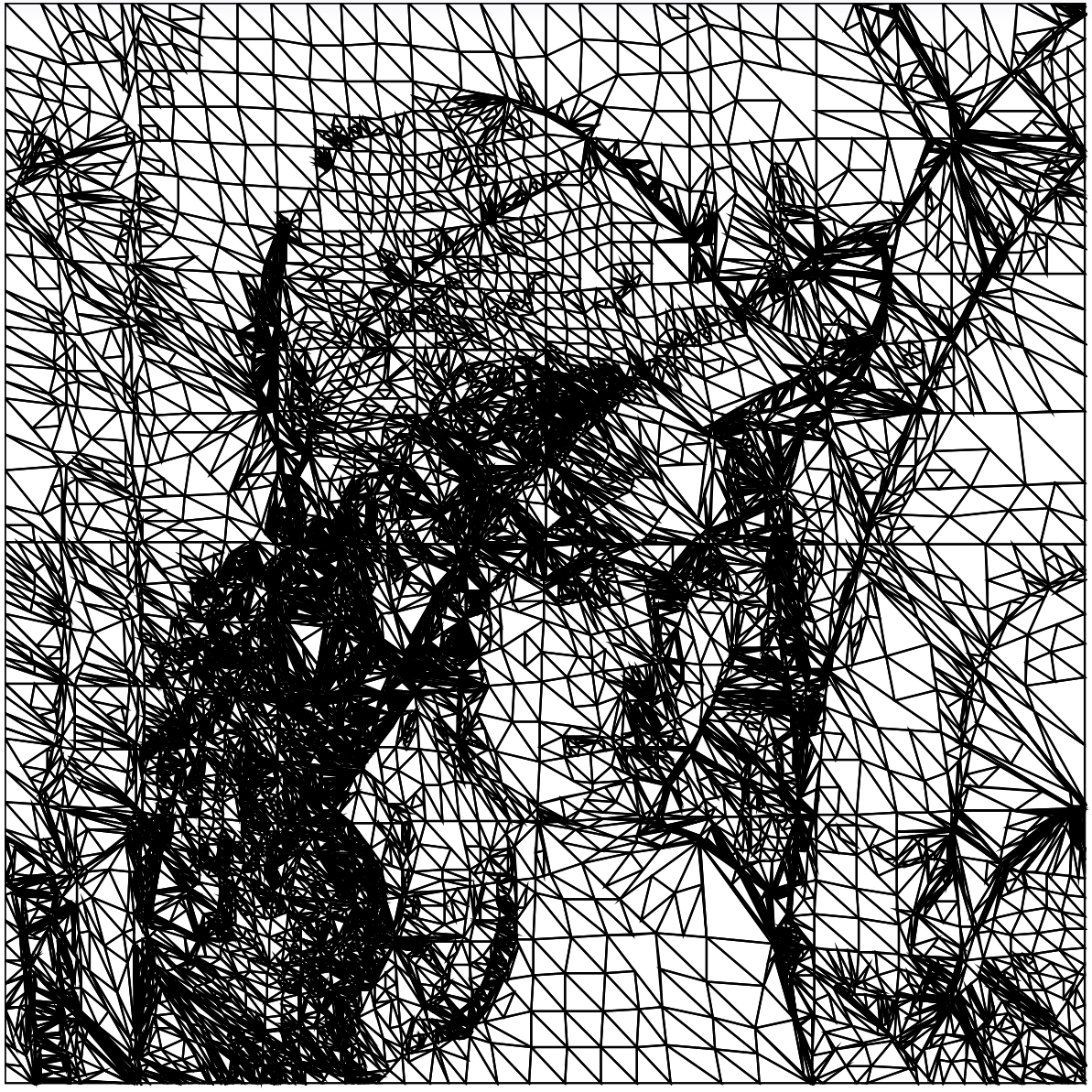


- Interpolating normal meshes $\longrightarrow$ Spend too much bitrate in the vertical offsets (detail coefficients)!!

Reallife images



- Introduce polynomial wavelets for compensating suboptimality of interpolating meshes in nonsmooth areas.



Thanks for your attention!
What are your questions?
www.cs.kuleuven.be/∼ward