Point Lattices in Computer Graphics and Visualization

how signal processing may help computer graphics

Dimitri Van De Ville

Ecole Polytechnique Fédérale de Lausanne
Biomedical Imaging Group

dimitri.vandeville@epfl.ch

VIS 05 MINNEAPOLIS, MN USA

IEEE Visualization 2005 - Tutorial (Image Processing 2D)

Overview

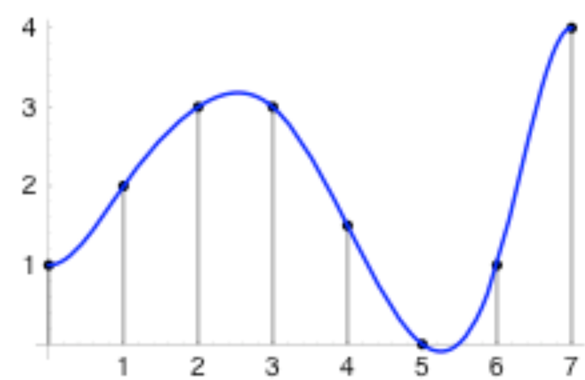
- B-splines: the right tool for interpolation
  - fundamental properties
  - spline fitting
    - interpolation; smoothing; least-squares
  - quantitative approximation quality
- A primer to the wavelet transform
  - multi-resolution, semi-orthogonal wavelets
- 2-D extension: hex-splines on any regular periodic lattice
  - hexagonal versus Cartesian lattice

Notations

- one-sided $$(x)_+ = \max(0, x)$$
- Fourier transform $$\hat{f}(\omega) = \int f(x) e^{-j\omega x} \mathrm{d}x$$ $$f(x) = \frac{1}{2\pi} \int \hat{f}(\omega) e^{j\omega x} \mathrm{d}\omega$$
- Z-transform $$C(z) = \sum_{k \in \mathbb{Z}} c[k] z^{-k}$$

Polynomial splines

- $s(x)$ is a polynomial spline of degree *n* with knots $\ldots < x_k < x_{k+1} < \ldots$ iff
  - Piecewise polynomial (of degree *n*) within each interval $[x_k, x_{k+1}[$
  - Higher-order continuity at the knots of $\frac{d^i s}{dx^i}, i = 0, \ldots, n-1$
- Effective degrees of freedom is 1
- “Cardinal splines”:
  - Unit spacing: $x_k = k$
  - ∞ number of knots



Polynomial B-splines

➤ B-spline of degree *n*

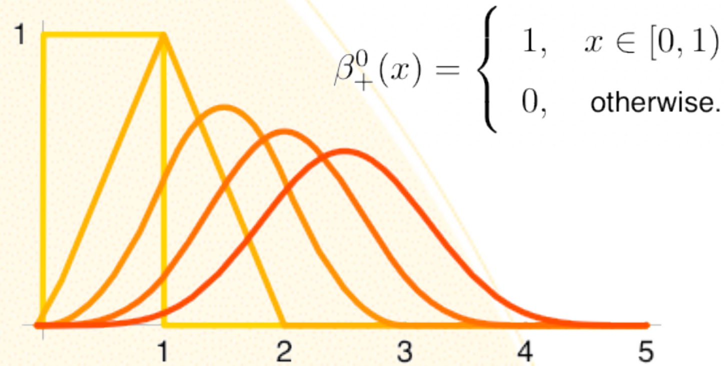
$$\beta_+^n(x) = \underbrace{\beta_+^0 * \beta_+^0 * \cdots * \beta_+^0}_{(n+1)\ \text{times}}(x)$$

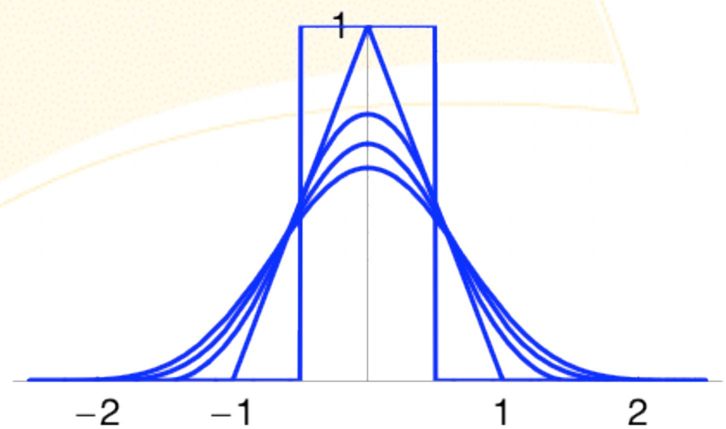
➤ Symmetric B-spline

$$\beta^n(x) = \beta_+^n\left(x + \frac{n+1}{2}\right)$$

➤ Key properties

- compact support
- piecewise polynomial
- positivity
- smoothness (continuity)





[Schoenberg, 1946]

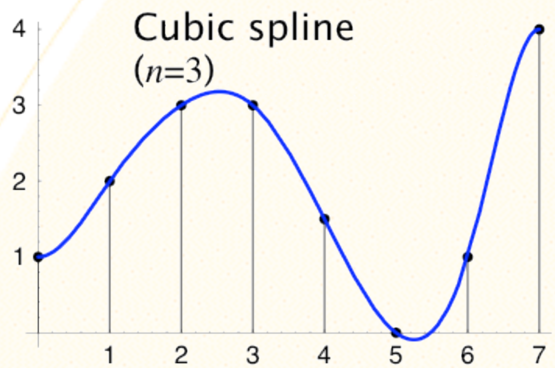
B-spline representation

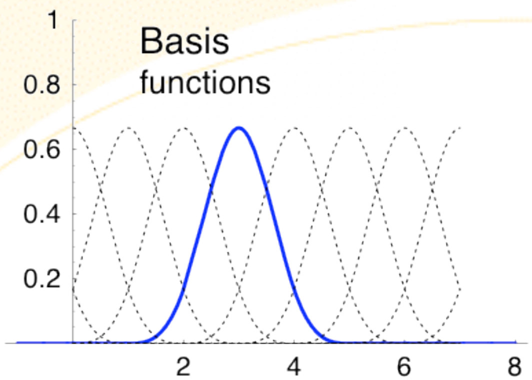
- The link between continuous and discrete

$$s(x) = \sum_{k \in \mathbb{Z}} c[k] \, \beta_+^n(x-k)$$

analog signal in the continuous domain

B-spline coefficients in the discrete domain





Fundamental B-spline properties

- Partition of unity
  - reproduction of the constant

→ and of polynomials up to degree *n*

- Riesz basis
  - *stability*: small perturbation of coefficients results into small change of spline signal
  - *unambiguity*: each representation is unique
- *m*-scale relation (for *m* integer)

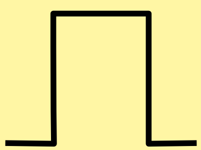
$$\beta_+^n(x/m) = \sum_{k\in\mathbb{Z}} h_m^n[k]\beta^n(x-k) \quad \text{with} \quad H_m^n(z) = \frac{1}{m^n}\left(\sum_{k=0}^{m-1} z^{-k}\right)^{n+1}$$

B-spline Fourier expression

$$\beta_+^n(x) = \beta_+^0 * \beta_+^0 * \cdots * \beta_+^0(x)$$

⇓

Fourier transform of basic element:



$$\beta_+^0 \longleftrightarrow \hat{\beta}_+^0(\omega) = \frac{\sin(\omega/2)}{\omega/2} e^{-j\omega/2} = \frac{1 - e^{-j\omega}}{j\omega}$$

⇓

$$\hat{\beta}_+^n(\omega) = \left(\frac{1 - e^{-j\omega}}{j\omega} \right)^{n+1}$$

B-spline differential property

$$\hat{\beta}^n_+(\omega) = \left(\frac{1 - e^{-j\omega}}{j\omega}\right)^{n+1}$$

"poor man's derivative" (finite difference)

$$\Delta f = f(x) - f(x-1) \leftrightarrow (1 - e^{-j\omega})\hat{f}$$

exact derivative

$$Df \leftrightarrow (j\omega)\hat{f}$$

➤ Link between discrete and exact derivatives

$$D^{m'} s = D^{m'} \left\{c * \beta^m_+\right\} = \Delta^{m'} c * \beta^{m-m'}_+$$

discrete filtering

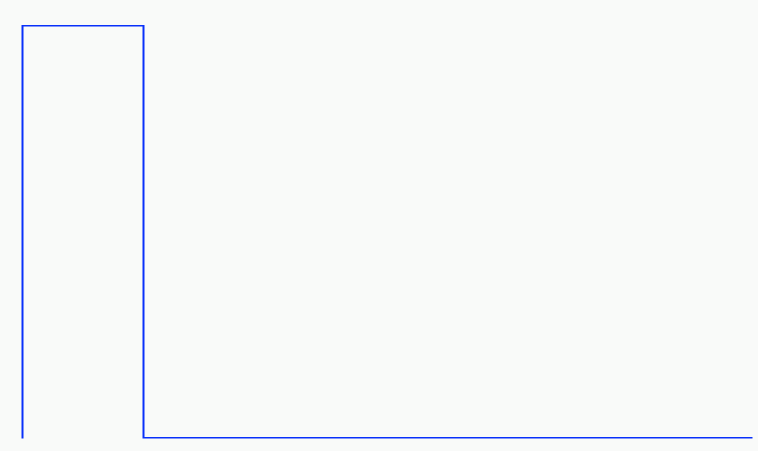
spline degree reduction

Generalized fractional B-splines

- Definition in the Fourier domain

$$\hat{\beta}^{\alpha}_{\tau}(\omega) = \left(\frac{1-e^{j\omega}}{-j\omega}\right)^{\frac{\alpha+1}{2}-\tau}\left(\frac{1-e^{-j\omega}}{j\omega}\right)^{\frac{\alpha+1}{2}+\tau}$$

degree $\alpha \in \mathbb{R}_+$
shift $\tau \in \mathbb{R}$



Fractional B-spline of degree 0

[Unser & Blu 2000, Blu & Unser 2003]

Spline fitting

- How to find the spline coefficients?

$$s(x) = \sum_{k \in \mathbb{Z}} c[k] \, \beta_+^n(x-k)$$

Spline fitting: (1) spline interpolation

- Spline interpolation (exact, reversible)

such that

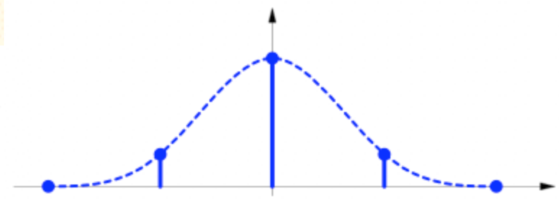
$$s(x)|_{x=k} = f[k]$$

- Smoothing spline
- Least square splines (approximation between spline spaces)

Spline interpolation

- Discrete B-spline kernels

$$b_1^n[k] = \beta^n(x)|_{x=k} \quad \stackrel{z}{\longleftrightarrow} \quad B_1^n(z) = \sum_{k=-\lfloor n/2 \rfloor}^{\lfloor n/2 \rfloor} \beta^n(k) z^{-k}$$



- Satisfying interpolation condition: inverse filter!

$$f[k] = \sum_{l \in \mathbb{Z}} c[l] \beta^n(x-l)|_{x=k} = (b_1^n * c)[k] \Rightarrow c[k] = (b_1^n)^{-1} * f[k]$$

- Efficient recursive implementation:
  - cascade of causal and anti-causal filters
  - e.g., cubic spline interpolation

$$(b_1^n)^{-1}[k] \quad \stackrel{z}{\longleftrightarrow} \quad \frac{6}{z+4+z^{-1}} = \frac{(1-\alpha)^2}{(1-\alpha z)(1-\alpha z^{-1})}$$

anti-causal causal

Spline interpolation

- Generic C-code
  - main recursion

```
void ConvertToInterpolationCoefficients ( double c[ ], long DataLength, double  z[ ], long NbPoles, double  Tolerance)
{
		double Lambda = 1.0; long n, k;
		if (DataLength == 1L) return;
		for (k = 0L; k < NbPoles; k++)
				Lambda = Lambda * (1.0 - z[k]) * (1.0 - 1.0 / z[k]);
		for (n = 0L; n < DataLength; n++) c[n] *= Lambda;
		for (k = 0L; k < NbPoles; k++) {
				c[0] = InitialCausalCoefficient(c, DataLength, z[k], Tolerance);
				for (n = 1L; n < DataLength; n++) c[n] += z[k] * c[n - 1L];
				c[DataLength - 1L] = (z[k] / (z[k]*z[k] - 1.0)) * (z[k]*c[DataLength - 2L] + c[DataLength - 1L]);
				for (n = DataLength - 2L; 0 <= n; n--) c[n] = z[k] * (c[n + 1L]- c[n]);
		}
}
```

  - initialization

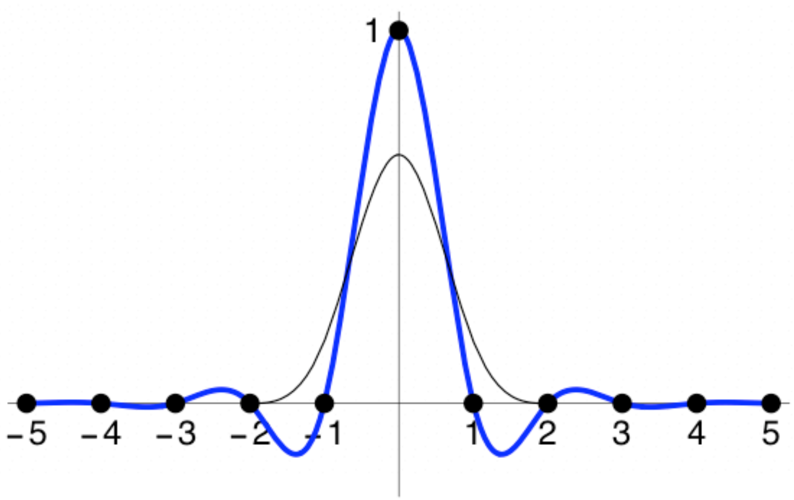
```
double  InitialCausalCoefficient ( double  c[ ], long DataLength, double  z, double  Tolerance)
{
		double Sum, zn, z2n, iz; long n, Horizon;
		Horizon = (long)ceil(log(Tolerance) / log(fabs(z)));
		if (DataLength < Horizon) Horizon = DataLength;
		zn = z; Sum = c[0];
		for (n = 1L; n < Horizon; n++) {Sum += zn * c[n]; zn *= z;}
		return(Sum);
}
```

Spline interpolation

- Interpolating or fundamental B-spline

$$s(x) = \sum_{k \in \mathbb{Z}} c[k]\beta^n(x-k) = \sum_{k \in \mathbb{Z}} \left(s(k) * (b_1^n)^{-1}[k]\right) \beta^n(x-k)$$

$$= \sum_{k \in \mathbb{Z}} s(k)\varphi_{\text{int}}^n(x-k)$$



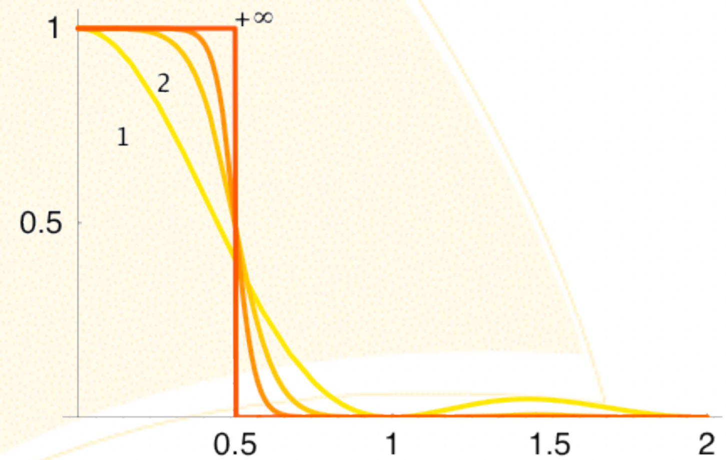
$$\varphi_{\text{int}}^n(x) = \sum_{k \in \mathbb{Z}} (b_1^n)^{-1}[k]\ \beta^n(x-k)$$

Spline interpolation

- The fundamental spline converges to sinc as the degree goes to infinity

$$\lim_{n\to\infty} \varphi^n_{\mathrm{int}}(x) = \mathrm{sinc}(x)$$

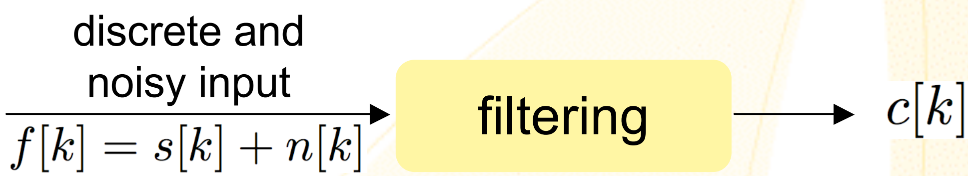
$$\lim_{n\to\infty} \left(\frac{\sin(\omega/2)}{\omega/2}\right)^{n+1} \frac{1}{B_1^n(e^{j\omega})} = \mathrm{rect}\left(\frac{\omega}{2\pi}\right)$$



- Shannon's theory appears as a particular case

Spline fitting: (2) smoothing spline

- Spline interpolation
- Smoothing spline



subject to
regularization

- Least square splines (approximation between spline spaces)

Smoothing spline

- The solution (among *all* functions) of the smoothing spline problem

$$\min_{s(x)} \left\{ \sum_{k\in\mathbb{Z}} |f[k] - s(k)|^2 + \lambda \int_{-\infty}^{+\infty} |D^m s(x)|^2 \, \mathrm{d}x \right\}$$

is a cardinal spline of degree *2m-1*. Its coefficients can be obtained by suitable digital filtering of the input samples:

$$c[k] = h_\lambda * f[k]$$

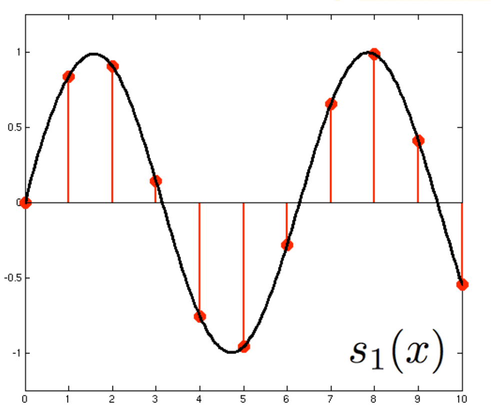
- Related to: MMSE (Wiener filtering); splines form optimal space!!!
- Special case: the draftman's spline
Minimum curvature interpolant is obtained for $m = 2, \lambda \to 0$
= cubic spline!

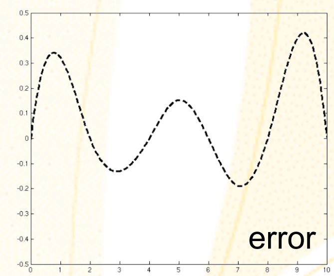
[Unser & Blu 2005; Ramani et al. 2005]

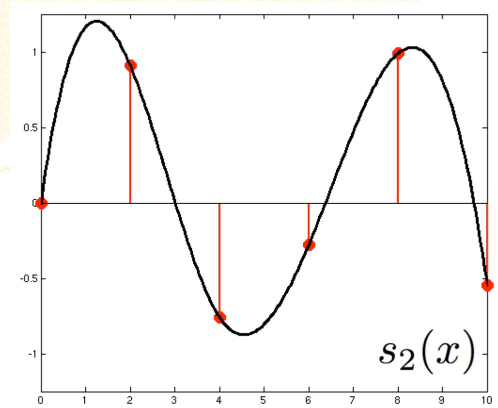
Spline fitting: (3) least-square spline

- Spline interpolation
- Smoothing spline
- Least-square spline (approximation between spline spaces)

spline model $s_1(x)$ → $c_1[k]$ → **resampling & filtering** → $c_2[k]$ → $s_2(x)$







such that $\min_{s_2} ||s_1 - s_2||^2_{L_2}$

Least-square spline

- Minimize quadratic error between splines

$$\{c_\kappa[k]\} = \arg \min_{\{c_\kappa[k]\}} ||s_1 - s_\kappa||_{L_2}$$

with $s_1(x) = \sum_{k\in\mathbb{Z}} c_1[k]\beta^n(x-k)$
$s_\kappa(x) = \sum_{k\in\mathbb{Z}} c_\kappa[k]\beta^n(x/\kappa-k)$

1. determine $c_1[k]$; e.g., by spline interpolation $(b_1^n)^{-1}$
2. resample using

$$d_\kappa[k] = \sum_{l\in\mathbb{Z}} c_1[l]\xi_\kappa^n(k\kappa - l)$$

with $\xi_\kappa^n(x) = \frac{1}{\kappa}\left(\beta^n(\cdot) * \beta^n(\cdot/\kappa)\right)(x)$

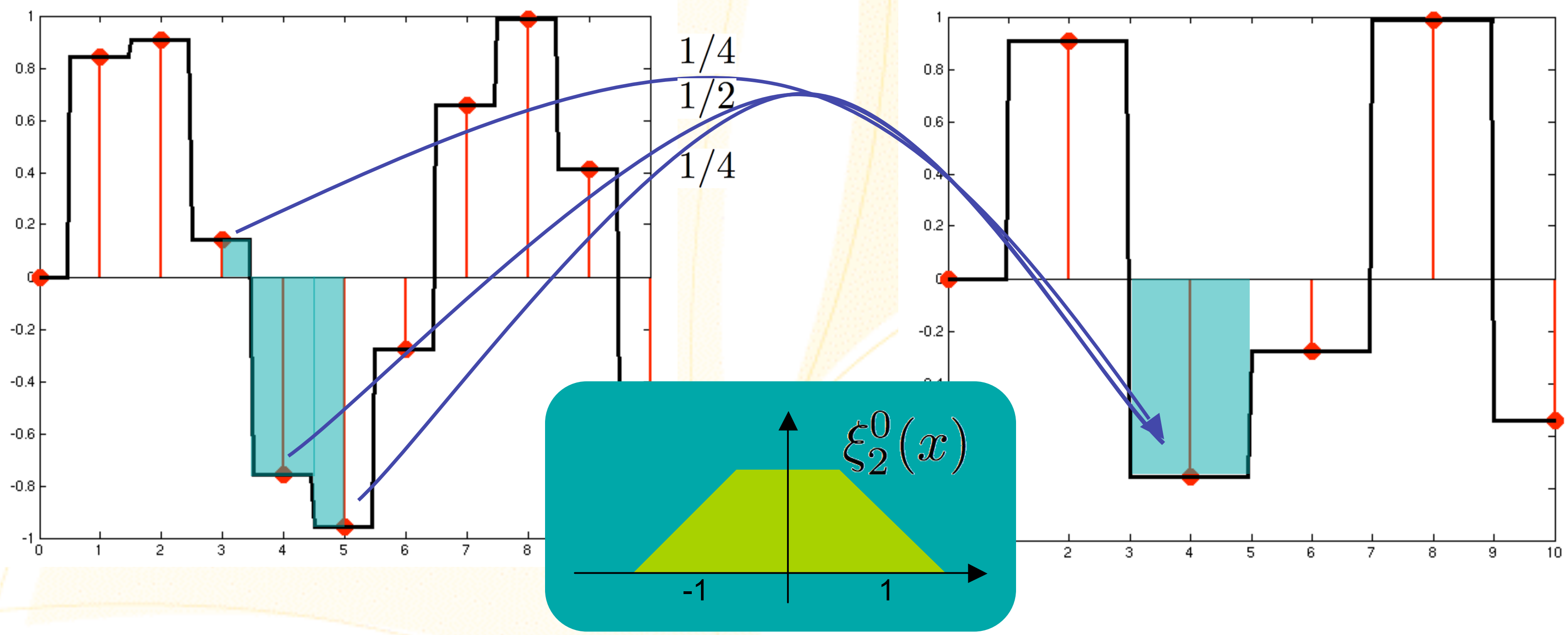
3. obtain samples of new spline representation

$$s_\kappa[k] = \left(d_\kappa * \left(b_1^{2n+1}\right)^{-1}\right)[k]$$

[Unser et al. 1995]

Least-square spline

- Special case: “surface projection”
  - first-order B-splines on source and target grid
  - weight of sample = overlap between B-splines’ support



Quantitative approximation quality

- Best approximation in a space?

analog input $f(x)$ → filtering & sampling →

analysis function at scale *a*

$$c[k] = \langle f, \tilde{\varphi}(\cdot/a - k) \rangle$$

$$s(x) = \sum_{k \in \mathbb{Z}} c[k] \varphi(x/a - k)$$

- Orthogonal projection $\min_{s \in V_a} \|f - s\|_{L_2}^2$

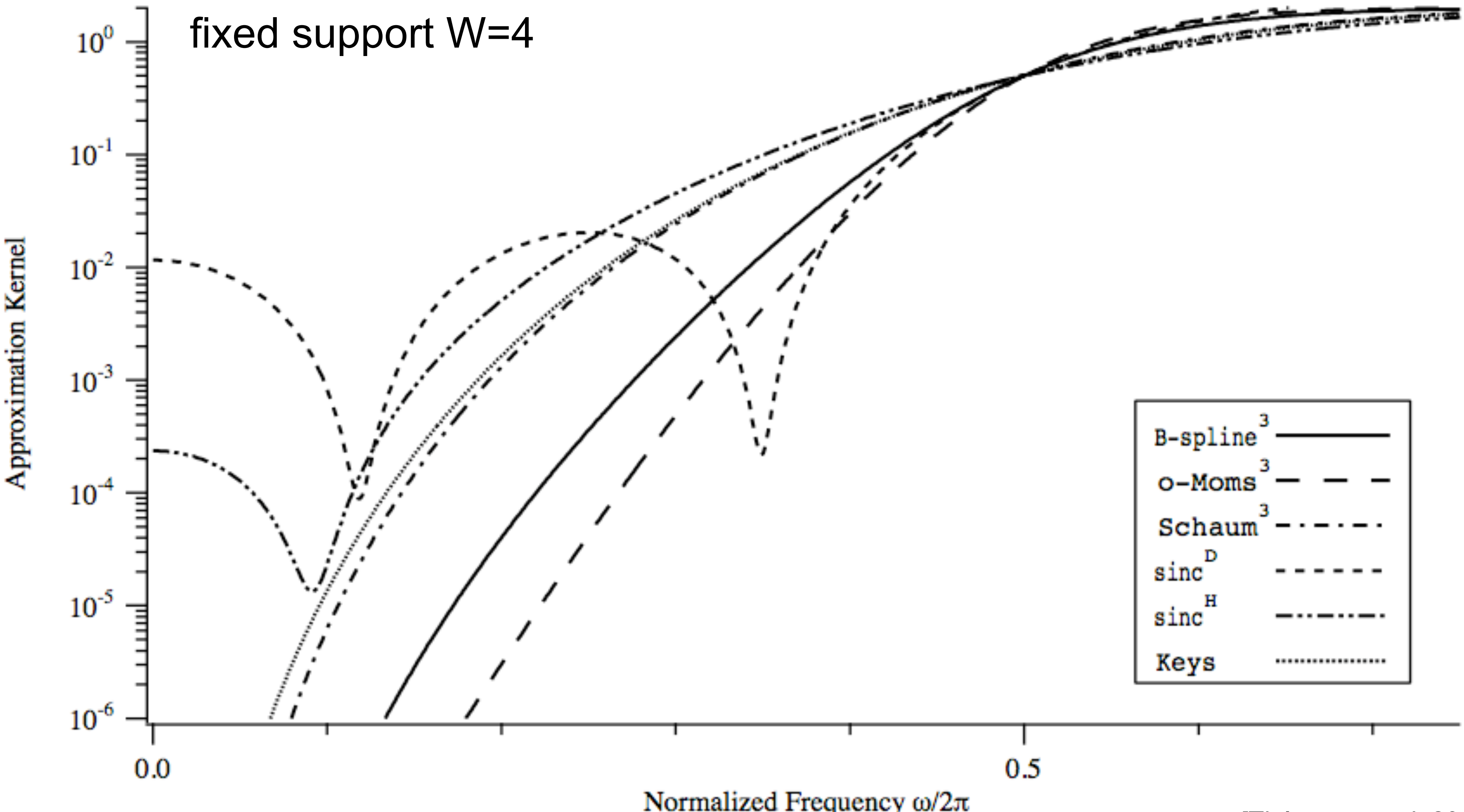
$$\|f - s\|_{L_2}^2 = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \left| \hat{f}(\omega) \right|^2 E(a\omega) \mathrm{d}\omega$$

Results for:
- Fixed scale
- Asymptotically

with error kernel $E(\omega) = 1 - \frac{|\hat{\varphi}(\omega)|^2}{\sum_{n \in \mathbb{Z}} |\hat{\varphi}(\omega + 2\pi n)|^2}$

[Blu & Unser 1999]

Quantitative approximation quality

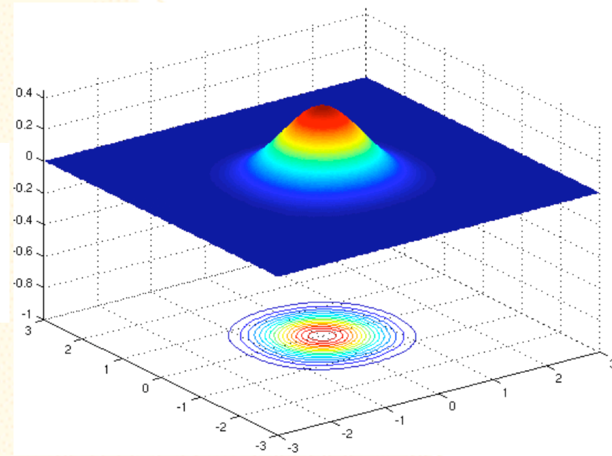


[Thévenaz et al. 2000]

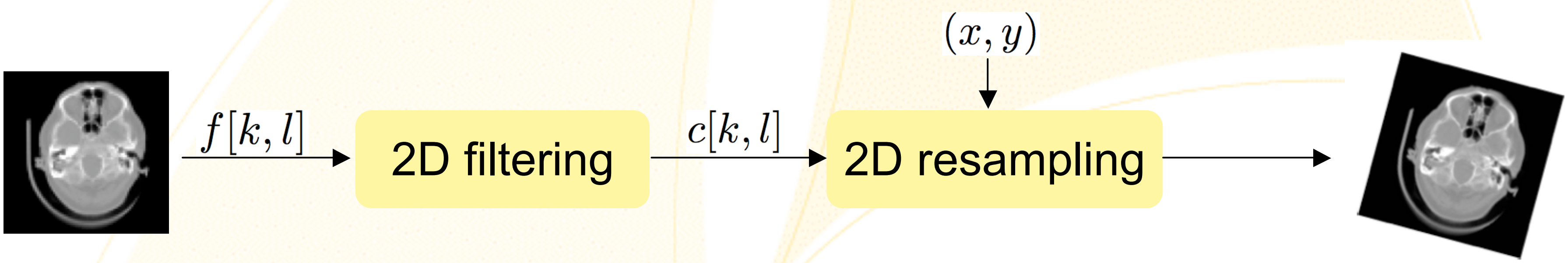
B-spline interpolation in 2D

- 2D separable model

$$f(x,y) = \sum_{k=k_1}^{k_1+n+1} \sum_{l=l_1}^{l_1+n+1} c[k,l]\, \beta^n(x-l)\, \beta^n(y-l)$$

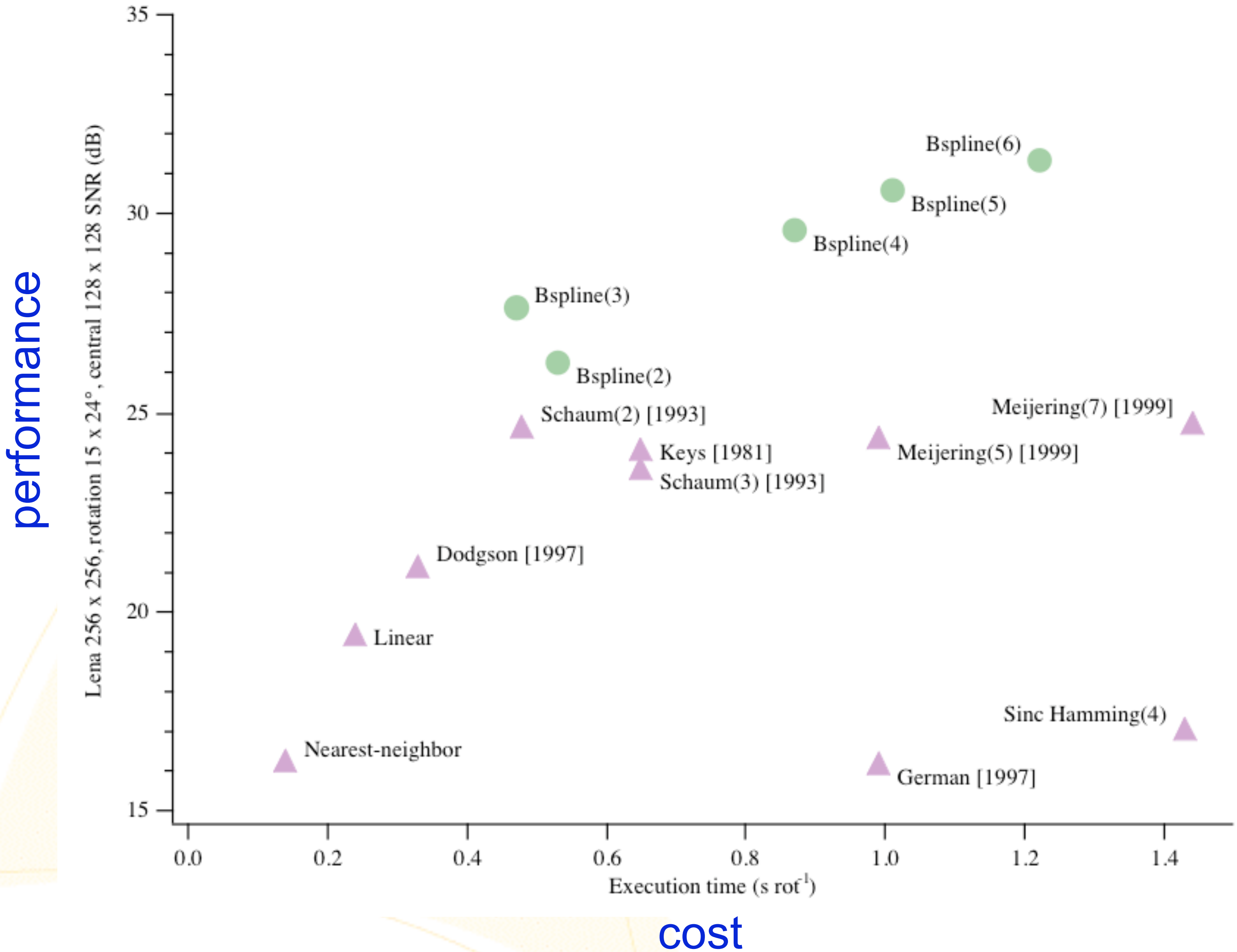


- Geometric transformations



- Applications
  - zooming, rotation, resizing, warping

High-quality image interpolation



[Thévenaz et al. 2000]

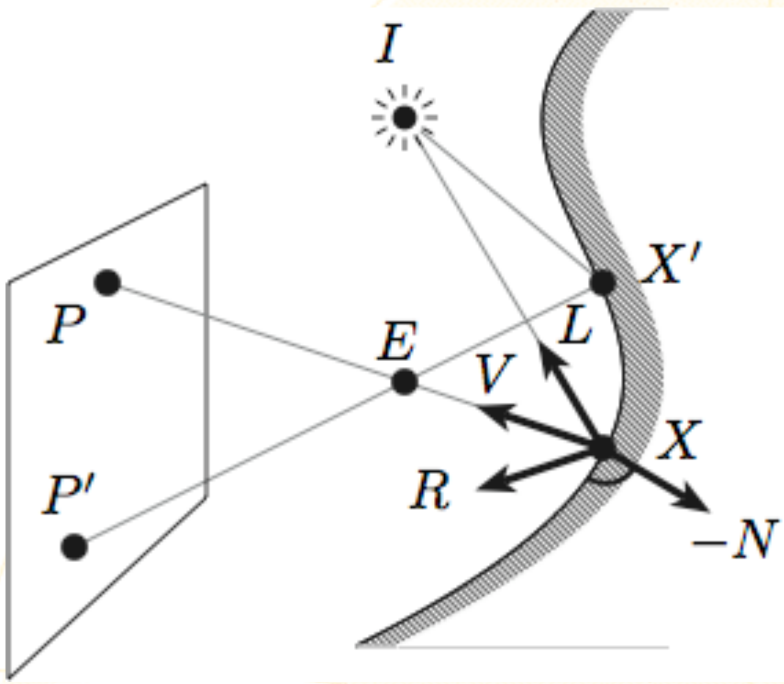
Interpolation benchmark

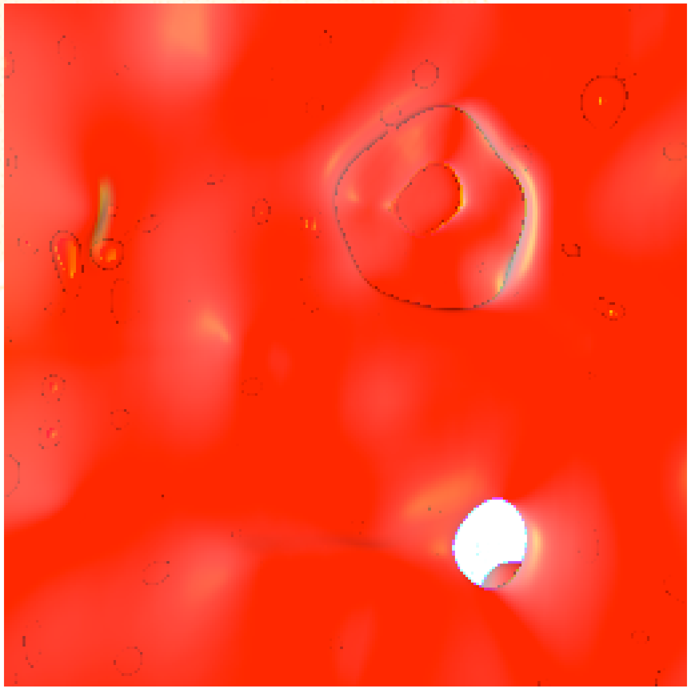
- Cumulative interpolation experiment:
  the best algorithm wins…

bilinear windowed sinc cubic spline

High-quality isosurface rendering

- 3D B-spline representation of volume data
- Isosurface
  - analytical knowledge of normal vectors





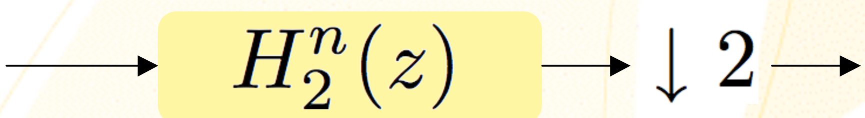
[Thévenaz et al. 2000]

Multi-resolution approximation

- *m*-scale relation

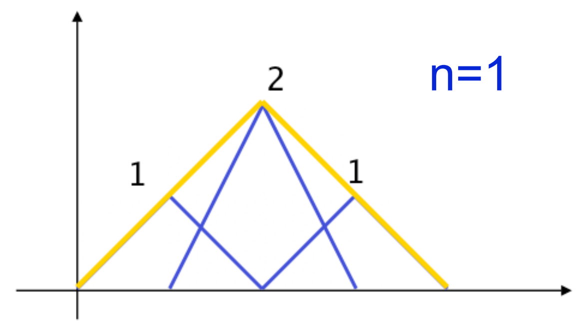
$$\beta_+^n(x/m) = \sum_{k\in\mathbb{Z}} h_m^n[k]\beta^n(x-k) \quad \text{with} \quad H_m^n(z) = \frac{1}{m^n}\left(\sum_{k=0}^{m-1} z^{-k}\right)^{n+1}$$

- Pyramid or tree algorithms ($m = 2^i$)
  - fast evaluation of $\left(f(\cdot) * \beta_+^n(\cdot/2^i)\right)(k)$



binomial filter

  - for high *n* ~ Gaussian filter

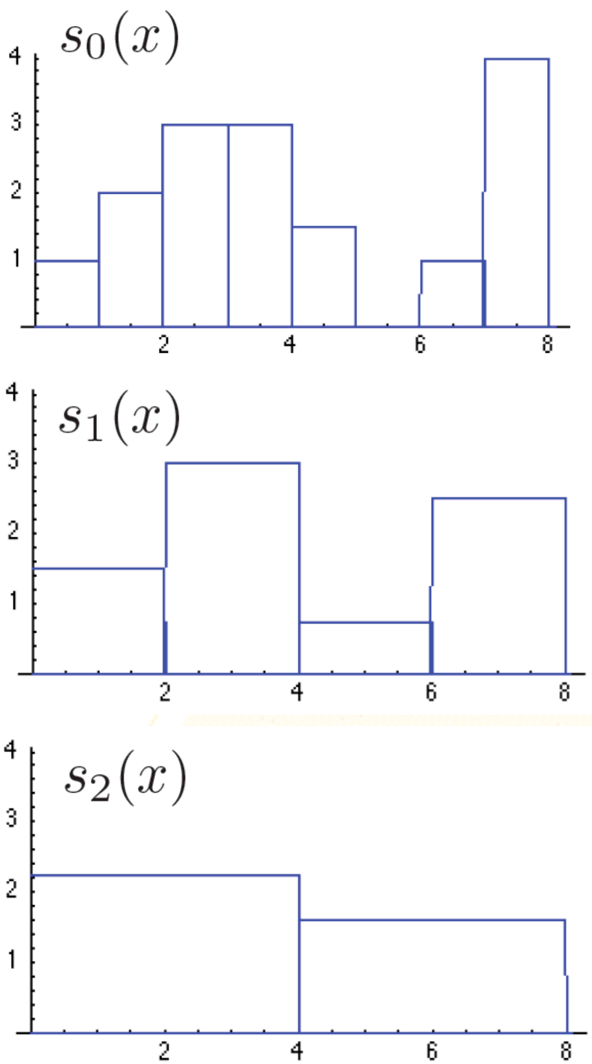


Wavelets

- Admissible scaling function ("father wavelet")
  - Riesz basis conditions
  - partition of unity
  - two-scale relation
- B-splines are perfect candidates
- Then there exists a wavelet $\psi(x/2) = \sum_{k\in\mathbb{Z}} g[k]\varphi(x-k)$ such that $\left\{2^{-i/2}\psi\left(\frac{x-2^i k}{2^i}\right)\right\}_{i\in\mathbb{Z},k\in\mathbb{Z}}$ forms a Riesz basis of L_2

[Mallat-Meyer 1989]

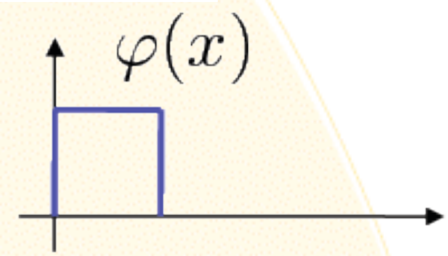
Haar wavelet transform revisited



- Signal representation

$$s_0(x) = \sum_k c_k \varphi(x-k)$$

basis function:



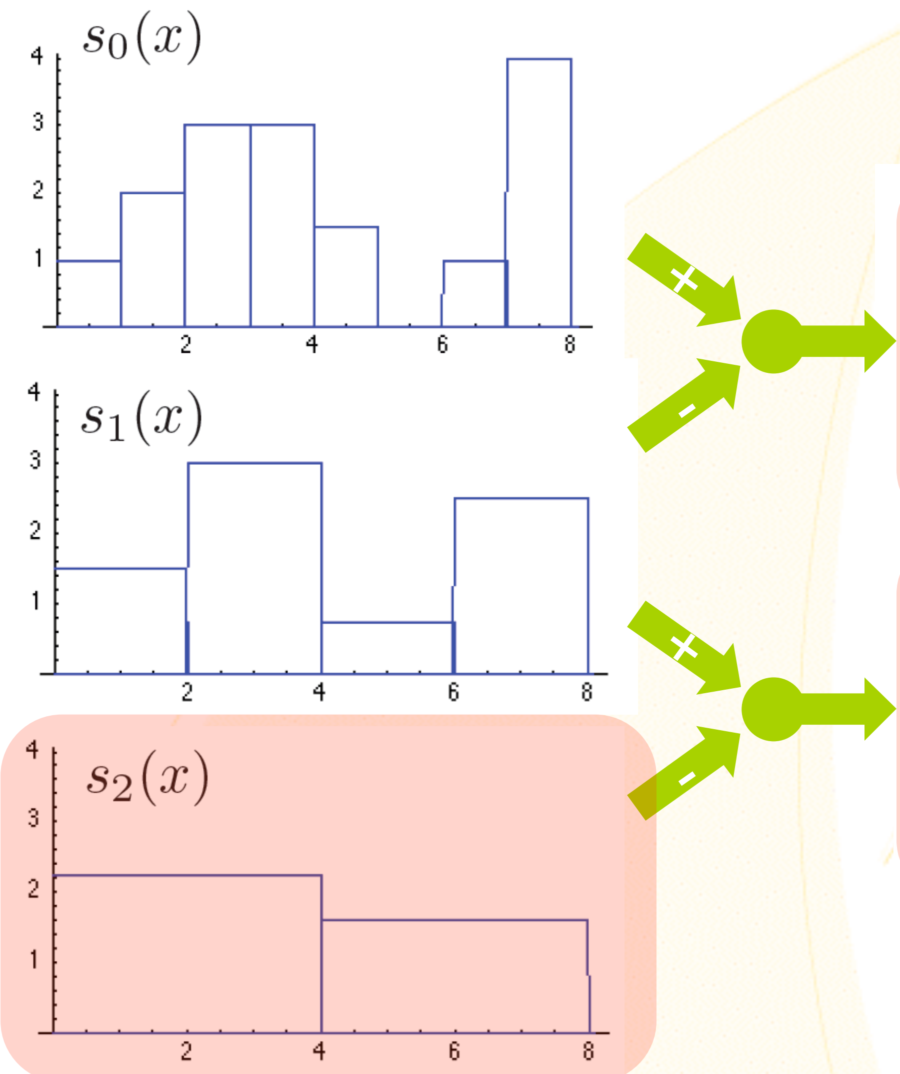
- Multi-scale signal representation

$$s_i(x) = \sum_k c_{i,k} \varphi_{i,k}(x)$$

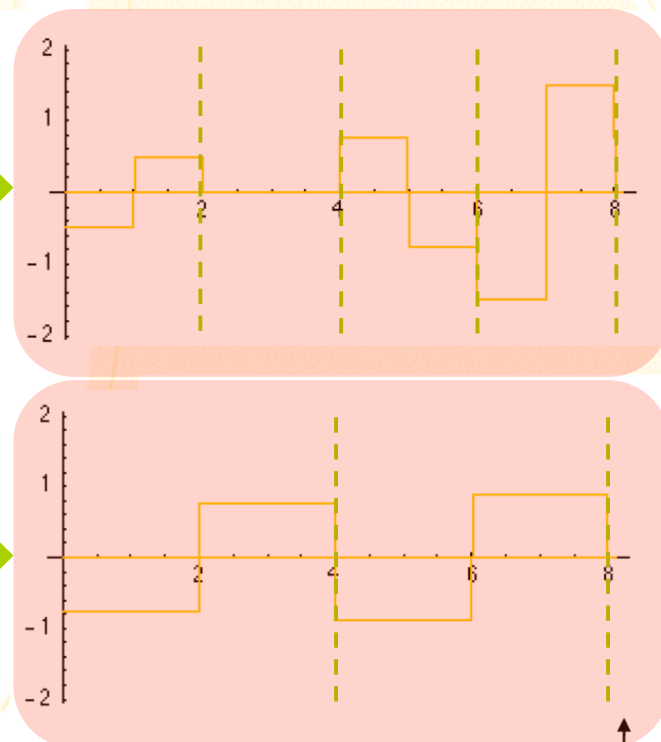
multi-scale basis function:

$$\varphi_{i,k}(x) = \varphi\left(\frac{x - 2^i k}{2^i}\right)$$

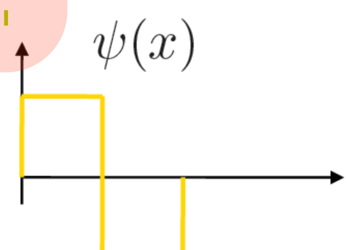
Haar wavelet transform revisited



$$r_i(x) = s_{i-1}(x) - s_i(x)$$



Wavelet:

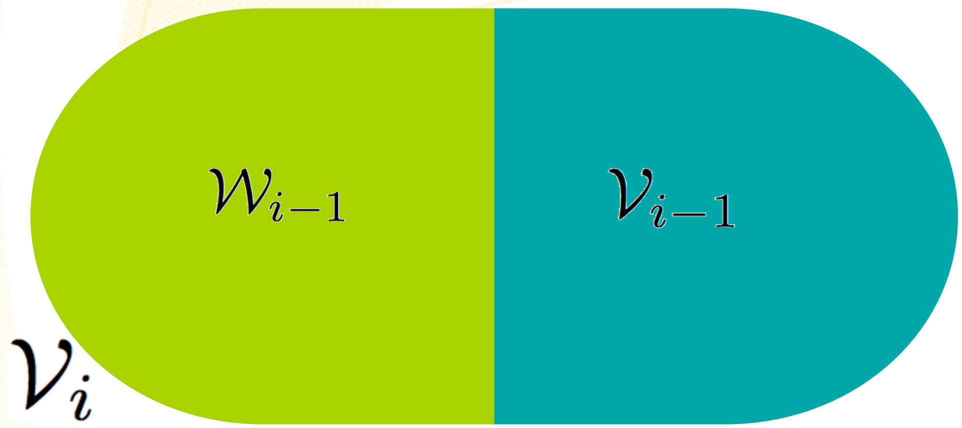


Semi-orthogonal wavelets

- Scaling and wavelet spaces

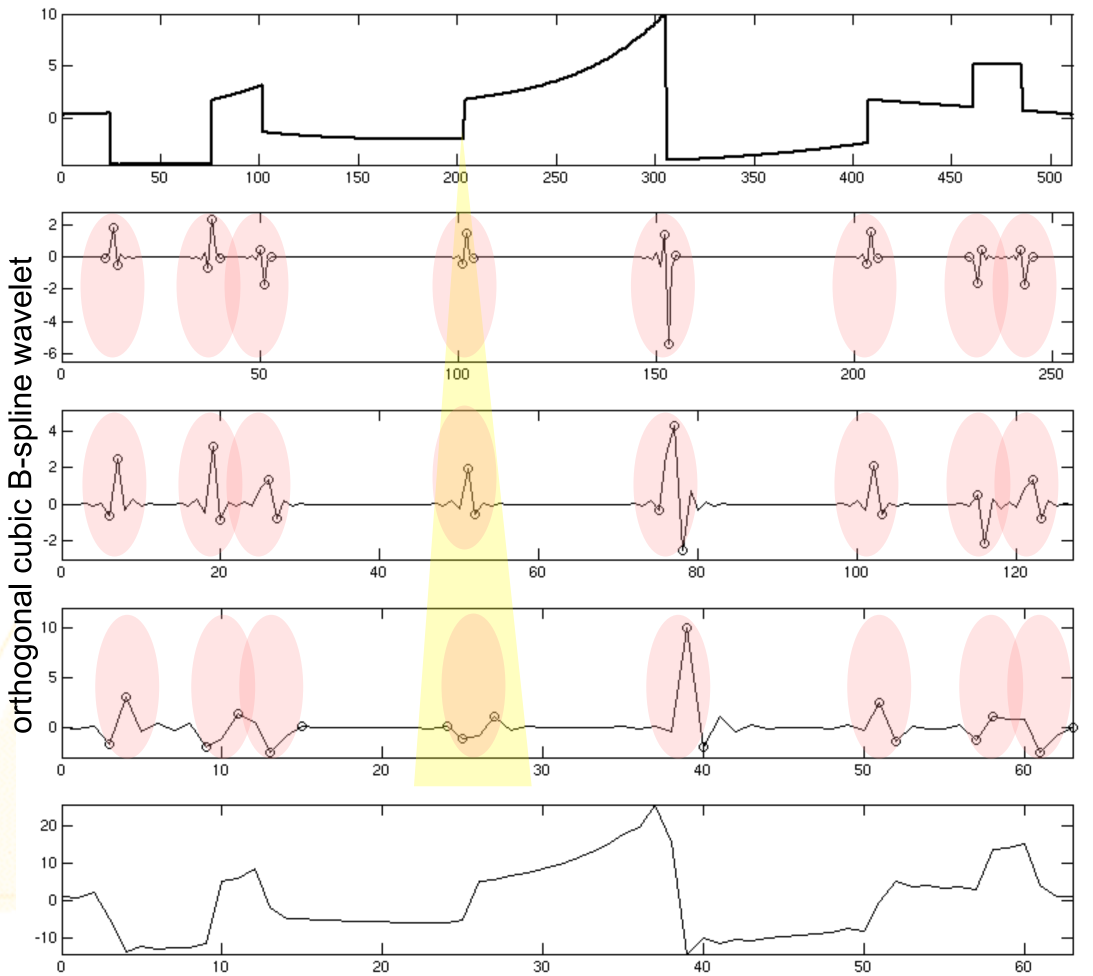
$$\begin{aligned}\mathcal{V}_i &= \mathrm{span}_{n\in\mathbb{Z}}\left\{\varphi\left(\frac{x}{2^i}-n\right)\right\}\\ \mathcal{W}_i &= \mathrm{span}_{n\in\mathbb{Z}}\left\{\psi\left(\frac{x}{2^i}-n\right)\right\}\end{aligned}$$

- Semi-orthogonality conditions
  1. $\mathcal{W}_i \subset \mathcal{V}_{i-1}$
  2. $\mathcal{W}_i \perp \mathcal{V}_i$



Wavelets

- Wavelets act as differentiators



Effect on transient features:

1) locality

2) sparsity (vanishing moments)

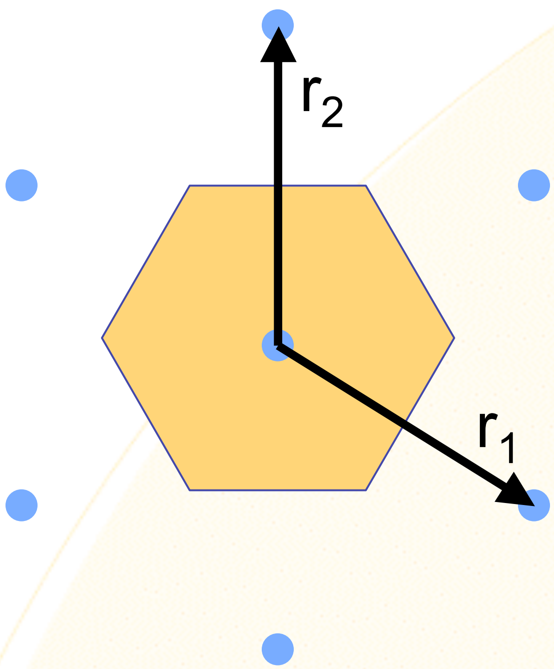
Wavelets and differentiation

- Fundamental property: multiscale differentiator

$$\hat{\psi}(\omega) \propto |\omega|^{\gamma} \quad \text{when } \omega \to 0$$

- Responsible for
  - vanishing moments
  - decorrelation
- Very successful for coding applications
  - JPEG2000

Hexagonal lattices



lattice matrix: $\mathbf{R} = [r_1 \; r_2]$

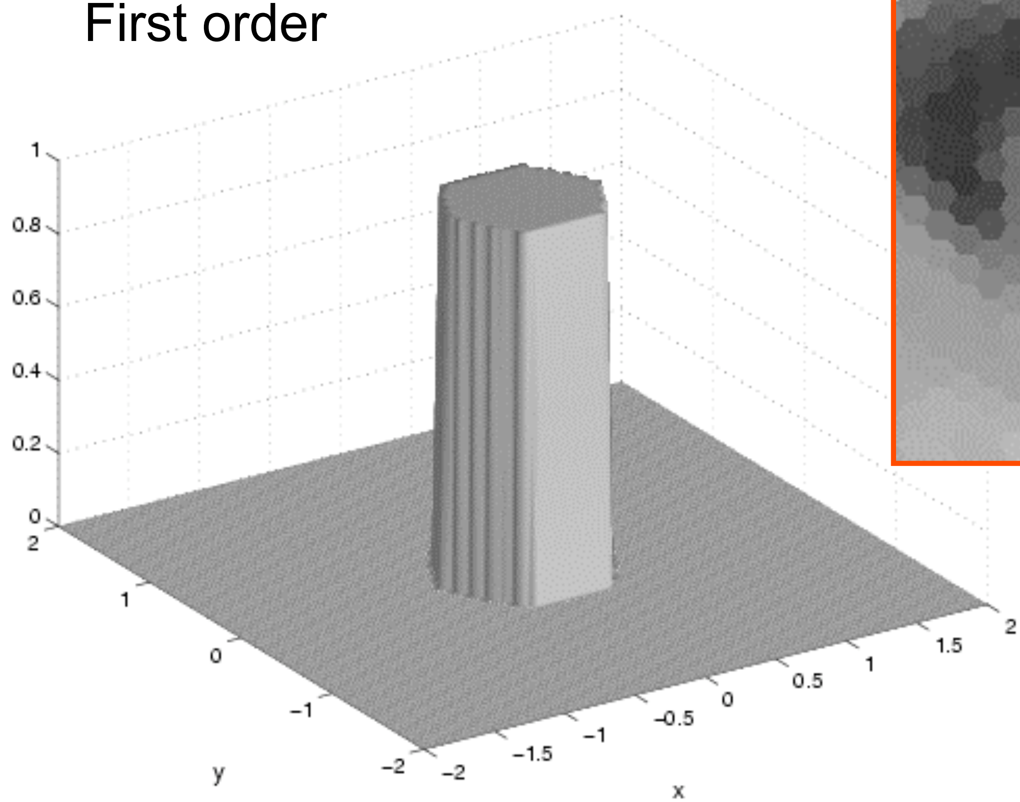
Voronoi cell =
"best" tessellation:

- Six equivalent neighbours
- Twelve-fold symmetry
- High isotropy

Hex-splines

➤ Basis functions for hexagonal grids

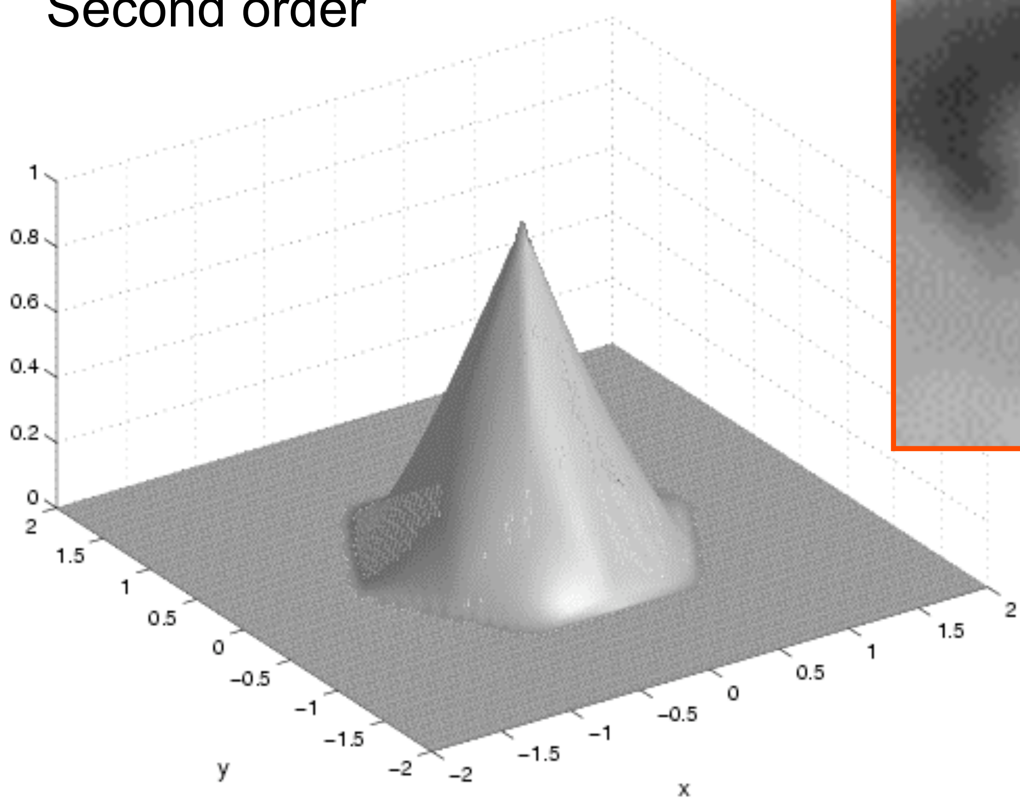
First order



Hex-splines

- Basis functions for hexagonal grids

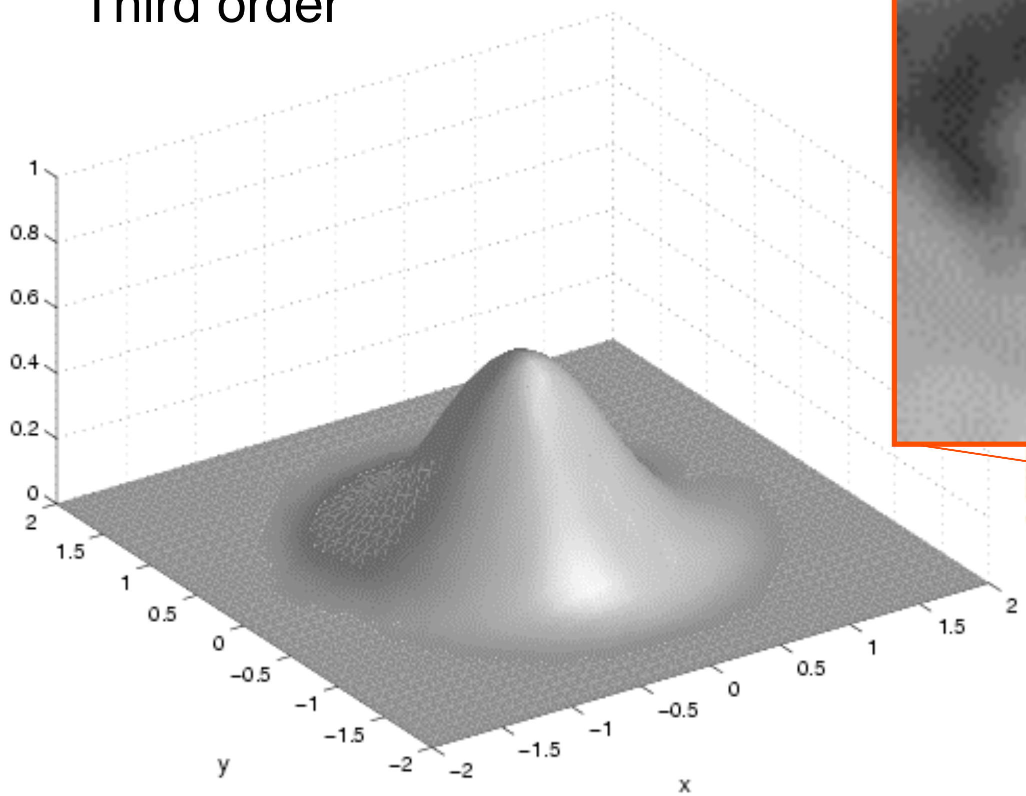
Second order



Hex-splines

- Basis functions for hexagonal grids

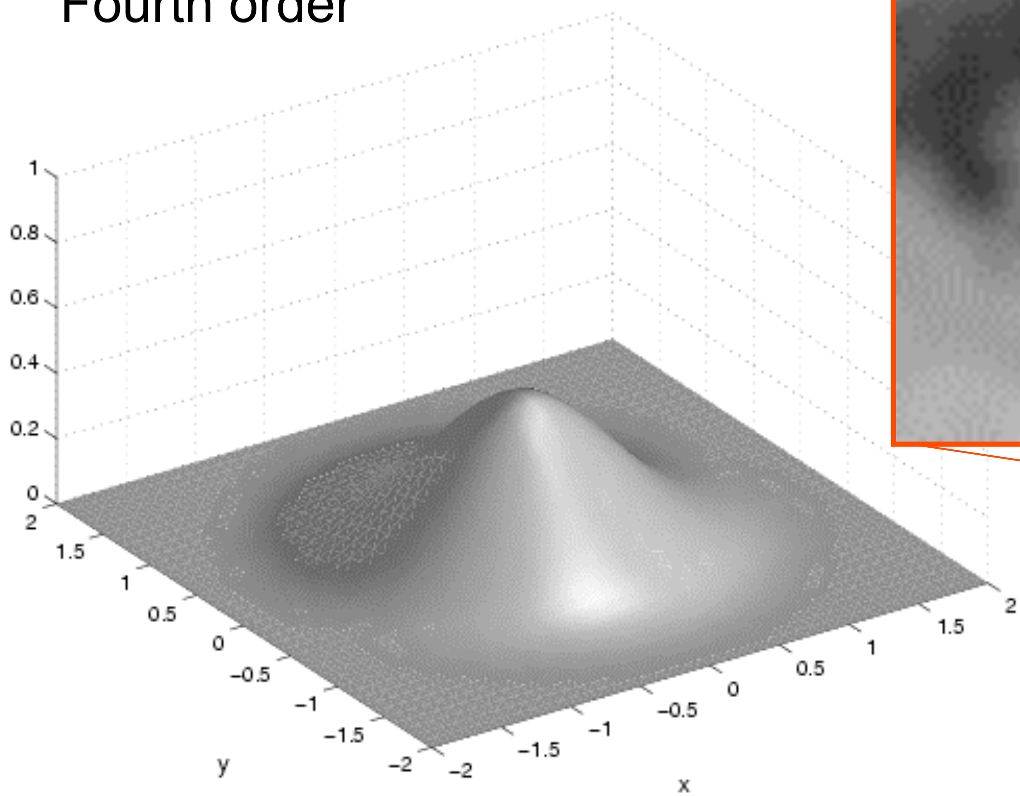
Third order



Hex-splines

- Basis functions for hexagonal grids

Fourth order

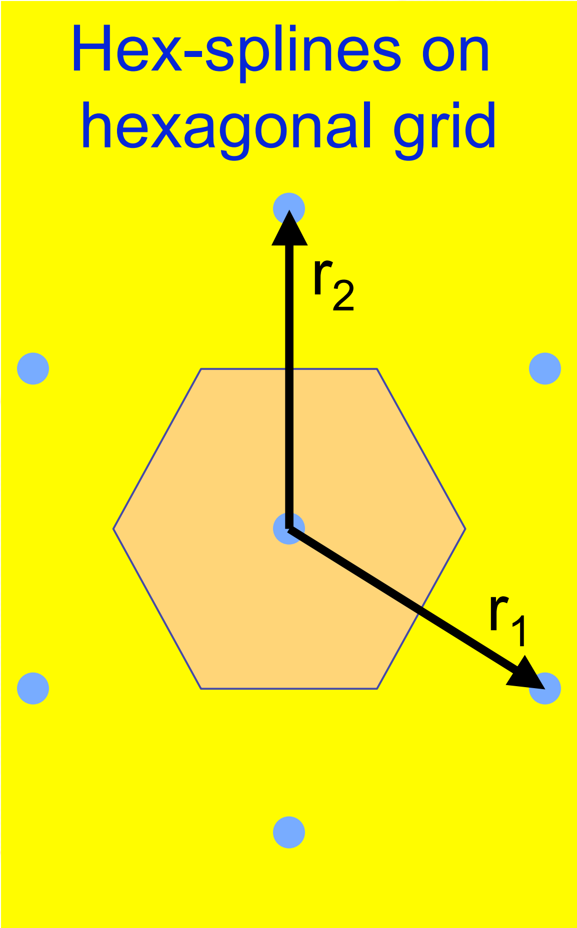


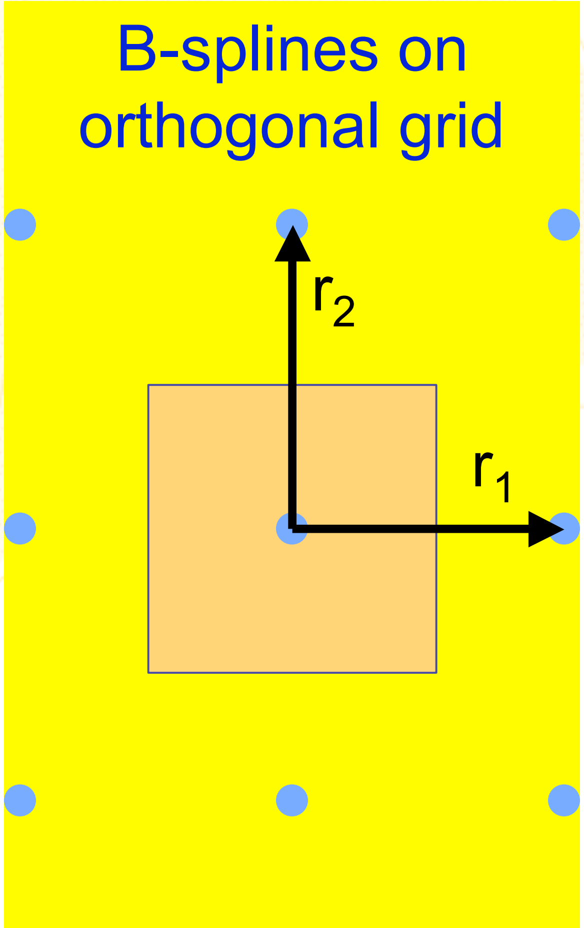
Hex-splines

- B-spline-like construction algorithm:
  - generating functions (~ differentiation operator in 3 directions)
  - localization operators (~ discrete versions of the operators)
- B-spline-like properties:
  - convolution property (by construction)
  - positivity, partition of unity, compact support
  - convergence to Gaussian
- Hex-splines exist for all periodic lattices
  - coincide with separable B-splines for cartesian lattice
- Fitting: interpolation, smoothing, least-squares
- But…
  - no two-scale relation

Hex-splines versus B-splines

- Keep sampling density equal: det($\mathbf{R}$)=Ω

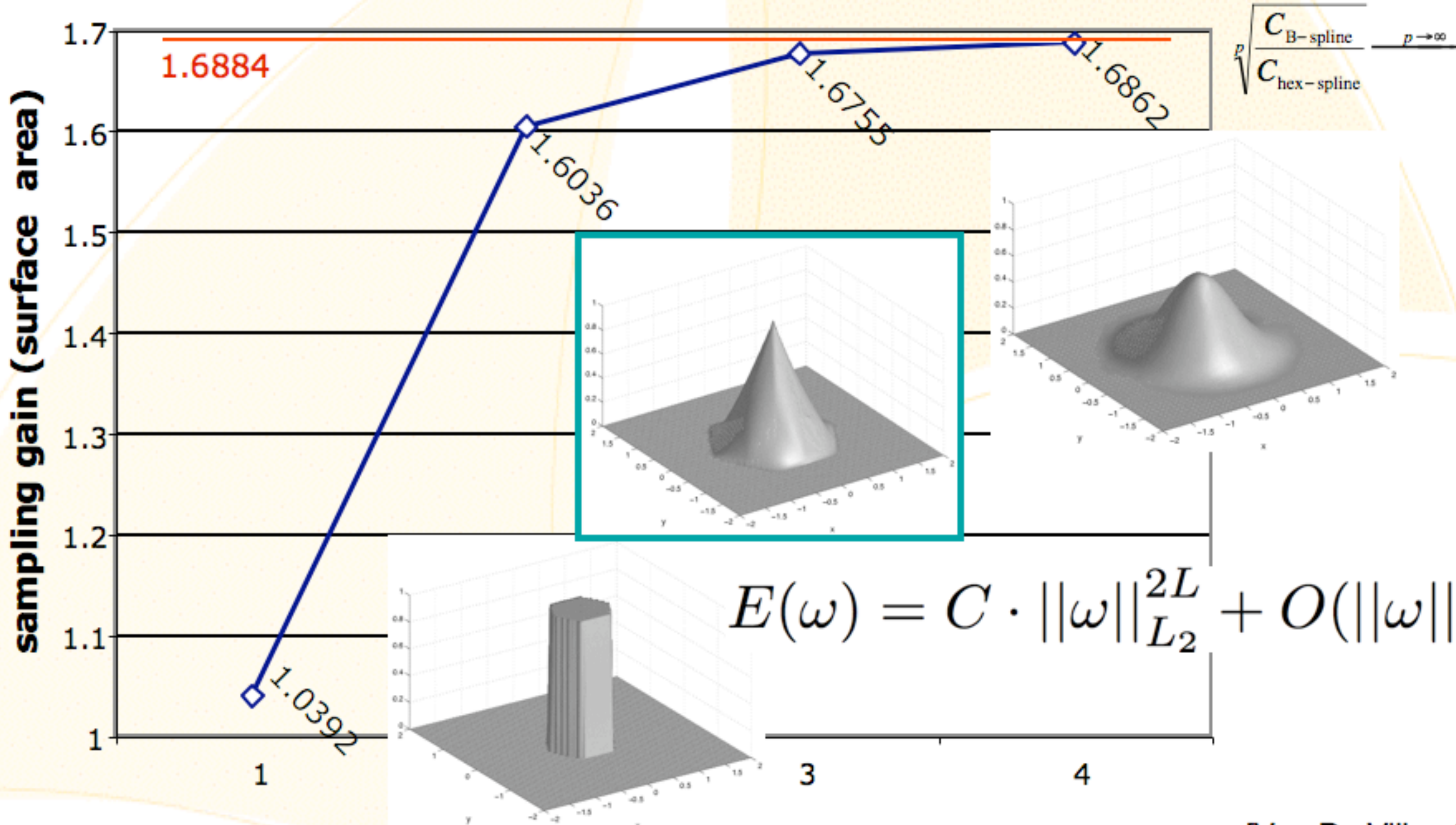




Hex-splines versus B-splines

➤ Extra samples so approximation quality B-splines equals that one of hex-splines (asymptotical result)

$$\sqrt[p]{\frac{C_{\text{B-spline}}}{C_{\text{hex-spline}}}} \xrightarrow{p\to\infty} \frac{8\pi^2}{27\sqrt{3}}$$



$$E(\omega) = C \cdot ||\omega||_{L_2}^{2L} + O(||\omega||_{L_2}^{2L+1})$$

[Van De Ville et al. 2005]

Hex-splines versus B-splines

- Classical result:
  - isotropic band-limited signals are better approximated on hexagonal lattices [Mersereau, 1979]
- Here, result for non-bandlimited signals
  - first order (nearest neighbor) on hexagonal lattices does not pay
  - at least second order (linear-like) hex-splines should be used; second-order still have easy analytical characterization

Conclusions

- B-splines are a great tool for interpolation and approximation: link continuous and discrete!
  - short support; analytical expression; tunable degree
  - fundamentally linked to differential operators
- Shift-invariant spaces due to *uniform sampling* brings along
  - fast algorithms (filtering, FFT-based,…)
  - powerful theoretical results (error kernel)
- Multi-resolution
  - *m*-scale relation for pyramids and wavelets
- Multi-dimensional extensions and variations
  - tensor-product, hex-splines, box-splines (see later)

And finally

- Many thanks go to

Michael Unser

Thierry Blu

Philippe Thévenaz

- Papers, demonstrations, source code:
http://bigwww.epfl.ch/

- The Wavelet Digest: (22000+ subscribers)
http://www.wavelet.org/