MULTIVARIATE APPROXIMATION AND INTERPOLATION WITH APPLICATIONS 2013

A class of Laplacian multi-wavelets bases for high-dimensional data

Nir Sharon

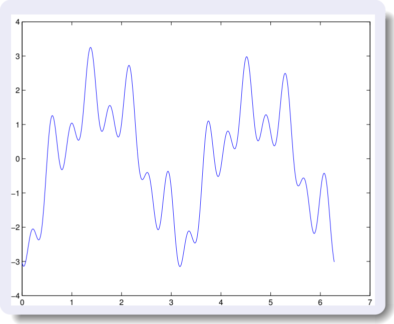
Tel-Aviv University

Joint work with Yoel Shkolnisky

A part of PhD thesis under the supervision of Yoel Shkolnisky and Nira Dyn

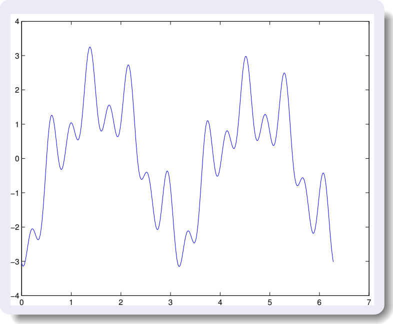
September 26, 2013

Representing signals



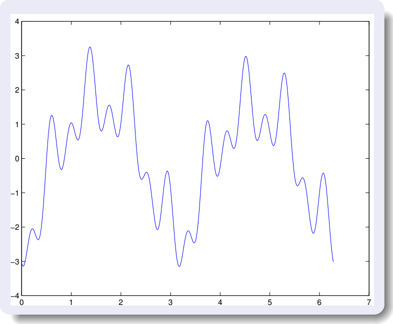
- 1D signals – Fourier basis, wavelets, polynomials,. . .

Representing signals



- 1D signals – Fourier basis, wavelets, polynomials,. . .
- What to do in higher dimensions?

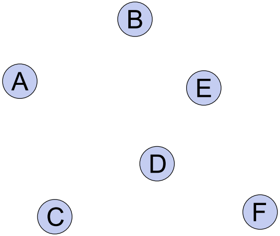
Representing signals



- 1D signals – Fourier basis, wavelets, polynomials,. . .
- What to do in higher dimensions?
- What to do for general data - images, documents, gene arrays, . . . ?

What is general data?

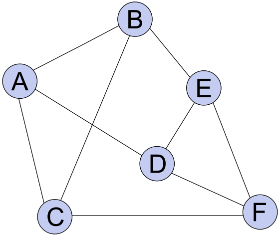
Let $\mathcal{X} = \{x_i\}_{i=1}^N$, $x_i \in \mathbb{R}^D$, be a set of N points with two requirements:



What is general data?

Let $\mathcal{X} = \{x_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^D$, be a set of N points with two requirements:

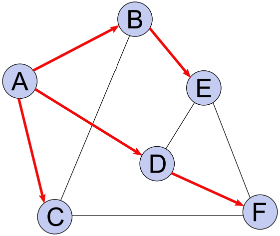
1. The set $\mathcal{X}$ is associated with a kernel function $K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_+$, and with the graph structure induced by K.



What is general data?

Let $\mathcal{X} = \{x_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^D$, be a set of N points with two requirements:

1. The set $\mathcal{X}$ is associated with a kernel function $K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_+$, and with the graph structure induced by K.
2. $\mathcal{X}$ has an associated tree structure – analog of a dydic partition.



The goal

Find N functions

$$\{\phi_n\}_{n=1}^N, \quad \phi_n : \mathcal{X} \mapsto \mathbb{R},$$

such that $\langle \phi_n, \phi_m \rangle = \delta_{n,m}$.

- We use

$$\langle f, g \rangle = \sum_{x \in \mathcal{X}} f(x) g(x), \quad \forall f, g : \mathcal{X} \mapsto \mathbb{R}.$$

The goal

Find N functions

$$\{\phi_n\}_{n=1}^{N}, \quad \phi_n : \mathcal{X} \mapsto \mathbb{R},$$

such that $\langle \phi_n, \phi_m \rangle = \delta_{n,m}$.

- We use

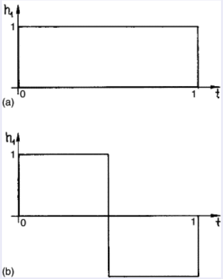
$$\langle f, g \rangle = \sum_{x \in \mathcal{X}} f(x)g(x), \quad \forall f, g : \mathcal{X} \mapsto \mathbb{R}.$$

- Further requirements
  - The construction must be applicable in cases where D (the dimension of each point in $\mathcal{X}$) is very large.
  - It should allow for a sparse representation of a large family of functions.
  - It must have a fast and numerically stable algorithm.

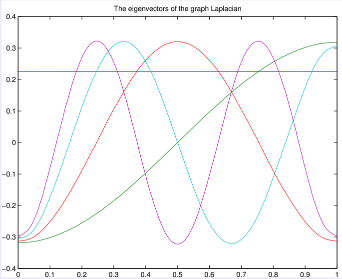
Known solutions

- Two known solutions for general data
  - Haar basis – Piecewise constant functions
  - Fourier basis – Eigenvectors of the (graph) Laplacian

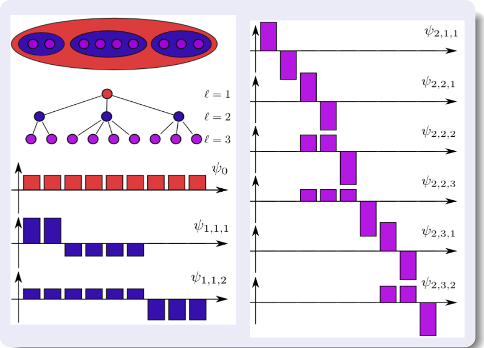
Haar basis



Fourier basis

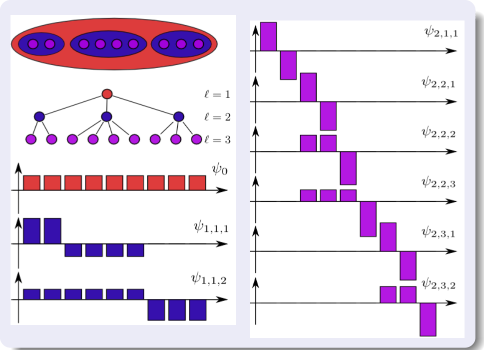


Haar basis – general data



Haar-like on graphs (Gavish, Nadler, and Coifman)

Haar basis – general data



Haar-like on graphs (Gavish, Nadler, and Coifman)

Pros
- Simple, fast.
- Applicable to high dimensional data.

Cons
- Poor representations of smooth functions.

Fourier basis

- Eigenfunctions of the Laplacian, e.g., $\varphi'' = -\lambda\varphi$.

Fourier basis

- Eigenfunctions of the Laplacian, e.g., $\varphi'' = -\lambda\varphi$.
- How to generalize? – Eigenvectors of the "graph Laplacian".

Fourier basis

- Eigenfunctions of the Laplacian, e.g., $\varphi'' = -\lambda\varphi$.
- How to generalize? – Eigenvectors of the "graph Laplacian".
- The graph Laplacian, in a nutshell:

Fourier basis

- Eigenfunctions of the Laplacian, e.g., $\varphi'' = -\lambda\varphi$.
- How to generalize? – Eigenvectors of the "graph Laplacian".
- The graph Laplacian, in a nutshell:

1. For any set of points (in $\mathbb{R}^D$, on a manifold,...), use kernel K to construct a graph

$$W_{i,j} = K\left(\|x_i - x_j\|^2/2\varepsilon\right).$$

Fourier basis

- Eigenfunctions of the Laplacian, e.g., $\varphi'' = -\lambda\varphi$.
- How to generalize? – Eigenvectors of the "graph Laplacian".
- The graph Laplacian, in a nutshell:

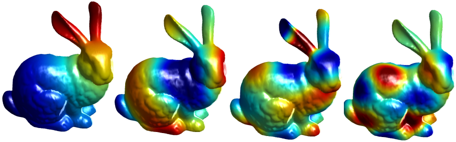
1. For any set of points (in $\mathbb{R}^D$, on a manifold, . . .), use kernel K to construct a graph
$$W_{i,j} = K\left(\|x_i - x_j\|^2/2\varepsilon\right).$$
2. Normalize, e.g.,
$$L = I - B^{-1}W, \quad B_{i,i} = \sum_{j=1}^{N} W_{i,j}.$$

Fourier basis

- Eigenfunctions of the Laplacian, e.g., $\varphi'' = -\lambda\varphi$.
- How to generalize? – Eigenvectors of the "graph Laplacian".
- The graph Laplacian, in a nutshell:

1. For any set of points (in $\mathbb{R}^D$, on a manifold,...), use kernel K to construct a graph
$$W_{i,j} = K\left(\|x_i - x_j\|^2/2\varepsilon\right).$$
2. Normalize, e.g.,
$$L = I - B^{-1}W, \quad B_{i,i} = \sum_{j=1}^{N} W_{i,j}.$$
3. Compute the eigenvectors.

Graph Laplacian basis



Graph Laplacian's eigenvectors on meshes (Gabriel Peyré)

Pros
- Efficient representation for smooth functions.
- Applicable to high dimensional (almost arbitrary) data.

Cons
- Poor representation of non-smooth functions/rapidly changing functions.
- Global basis functions.

Let's construct a new family of bases

Let's construct a new family of bases

- Orthogonal

Let's construct a new family of bases

- Orthogonal
- Multi-scale – basis elements of varying support.

Let's construct a new family of bases

- Orthogonal
- Multi-scale – basis elements of varying support.
- A family of bases parameterized by k – controls the localization of the basis elements.
- Extreme cases
  - $k = 1 \implies$ Haar basis
  - $k = N \implies$ Fourier basis

Let's construct a new family of bases

- Orthogonal
- Multi-scale – basis elements of varying support.
- A family of bases parameterized by k – controls the localization of the basis elements.
- Extreme cases
  - $k = 1 \implies$ Haar basis
  - $k = N \implies$ Fourier basis
- Stable $\mathcal{O}(k^2 N \log N + T(N,k)\log(N))$ algorithm, where $T(N,k)$ is the complexity of computing k top eigenvectors. Usually $N \gg k$.

Let's construct a new family of bases

- Orthogonal
- Multi-scale – basis elements of varying support.
- A family of bases parameterized by k – controls the localization of the basis elements.
- Extreme cases
  - $k = 1 \implies$ Haar basis
  - $k = N \implies$ Fourier basis
- Stable $\mathcal{O}(k^2 N \log N + T(N,k)\log(N))$ algorithm, where $T(N,k)$ is the complexity of computing k top eigenvectors. Usually $N \gg k$.
- Building blocks: graph Laplacian and multi-resolution analysis.

Construction overview

Two phases:

Construction overview

Two phases:

1. Define the vectors which span the approximation spaces

$$V_0 \subset V_1 \subset \cdots \subset V_j,$$

where $V_j = \mathbb{R}^N$, with N the number of data points.

Construction overview

Two phases:

1. Define the vectors which span the approximation spaces

$$V_0 \subset V_1 \subset \cdots \subset V_j,$$

where $V_j = \mathbb{R}^N$, with N the number of data points.

2. Apply a fast orthogonalization process to obtain

$$V_j = V_0 \oplus W_0 \oplus W_1 \oplus \cdots \oplus W_{j-1},$$

with $W_p \perp V_p$, $W_p \oplus V_p = V_{p+1}$ for $p \geq 0$.

Phase one – $V_0 \subset V_1 \subset \cdots \subset V_j$,

Phase one – $V_0 \subset V_1 \subset \cdots \subset V_j$,

- Define the first approximation space V_0 as

$$V_0 = \text{span}\left\{\text{first } k \text{ eigenvectors of the (global) graph Laplacian}\right\}$$

Phase one – $V_0 \subset V_1 \subset \cdots \subset V_j$,

- Define the first approximation space V_0 as

$$V_0 = \text{span}\{\text{first } k \text{ eigenvectors of the (global) graph Laplacian}\}$$

- To construct V_j, $j \geq 1$ we use

  1. Restriction operator on V_{j-1}.
  2. Local graph Laplacian and its first eigenvectors.

Phase one – $V_0 \subset V_1 \subset \cdots \subset V_j$,

- Define the first approximation space V_0 as

$$V_0 = \text{span}\,\{\text{first } k \text{ eigenvectors of the (global) graph Laplacian}\}$$

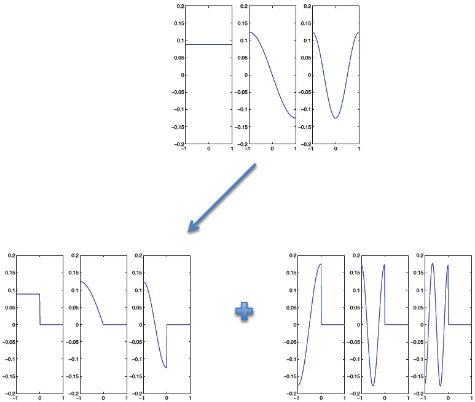
- To construct V_j, $j \geq 1$ we use

  1. Restriction operator on V_{j-1}.
  2. Local graph Laplacian and its first eigenvectors.

- This construction is repeated until V_j satisfies $\dim(V_j) = N$.

Approximation spaces – an example

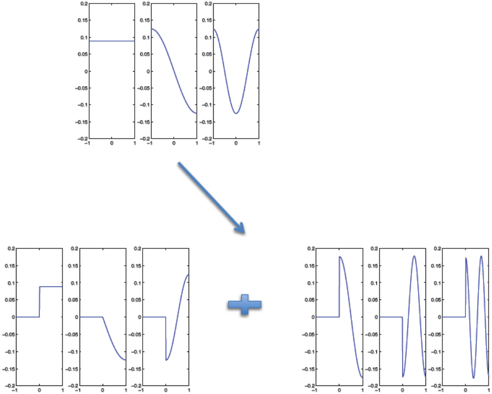
Constructing $V_1 = V_{1,0} + V_{1,1}$



Restriction Local eigenvectors

Approximation spaces – an example

Constructing $V_1 = V_{1,0} + V_{1,1}$



Restriction Local eigenvectors

A few additional remarks

A few additional remarks

1. Restriction operator and the tree ensure that the total number of nonzeros in each level is kN (sparsity).

A few additional remarks

1. Restriction operator and the tree ensure that the total number of nonzeros in each level is kN (sparsity).

2. Balanced tree means $\mathcal{O}(\log N)$ levels (or tree depth). Therefore, we can "pack" the nested spaces in a sparse matrix of $\mathcal{O}(kN \log(N))$ nonzeros.

A few additional remarks

1. Restriction operator and the tree ensure that the total number of nonzeros in each level is kN (sparsity).

2. Balanced tree means $\mathcal{O}(\log N)$ levels (or tree depth). Therefore, we can "pack" the nested spaces in a sparse matrix of $\mathcal{O}(kN\log(N))$ nonzeros.

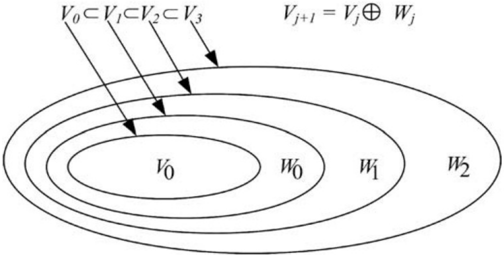
3. We do not assume the tree is binary nor a complete tree.

Phase two

Phase two

1. Recall that

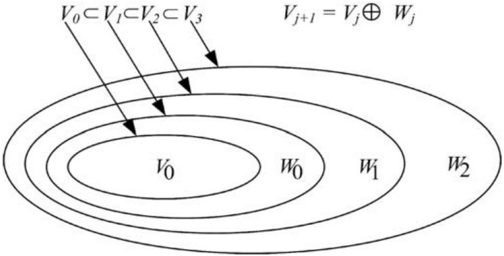
$$V_j \subset V_J, \quad V_J \perp W_J \implies V_j \perp W_J, \quad 0 \le j < J.$$



Phase two

1. Recall that

$$V_j \subset V_J, \quad V_J \perp W_J \implies V_j \perp W_J, \quad 0 \leq j < J.$$

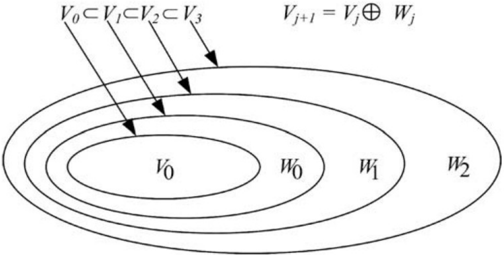


2. Due to sparsity, every complement space W_j is calculated with $\mathcal{O}(k^2 N)$ operations.

Phase two

1. Recall that

$$V_j \subset V_J, \quad V_J \perp W_J \implies V_j \perp W_J, \quad 0 \leq j < J.$$



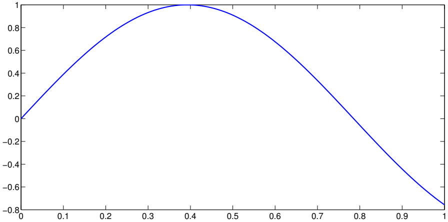
2. Due to sparsity, every complement space W_j is calculated with $\mathcal{O}(k^2 N)$ operations.
3. Overall complexity for this phase is $\mathcal{O}(k^2 N \log N)$. Usually $N \gg k$.

Representing functions (synthetic data)

The 1D case: taking 128 equally spaced on $[0,1]$. Compare the Haar ($k=1$), Laplacian ($k=N$), and an intermediate case ($1<k<N$)

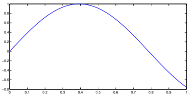
Representing functions (synthetic data)

The function:

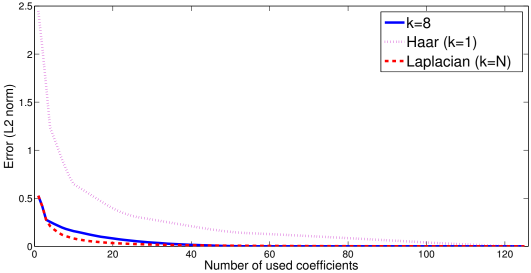


$\sin(4x)$

Representing functions (synthetic data)



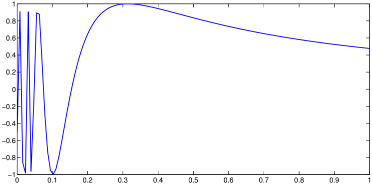
$\sin(4x)$



L_2 approximation error

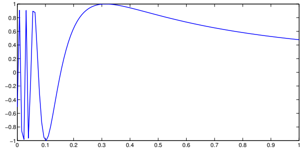
Representing 1D functions – oscillatory function

The function:

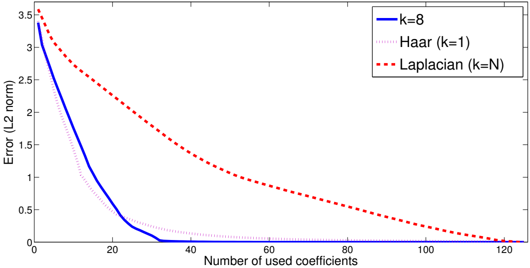


$$\sin\left(\frac{1}{0.01+2x}\right)$$

Representing 1D functions – oscillatory function

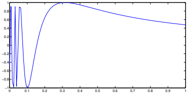


$\sin\left(\frac{1}{0.01+2x}\right)$

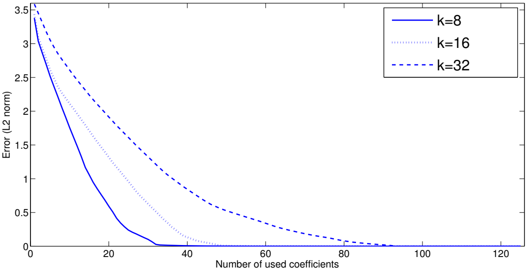


L_2 approximation error

Representing 1D functions – oscillatory function (cont.)

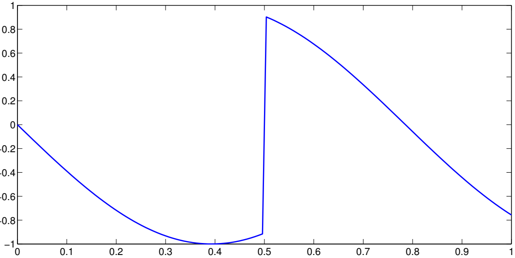


$\sin\left(\frac{1}{0.01+2x}\right)$



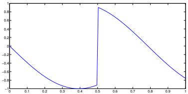
L_2 approximation error

Representing 1D functions – piecewise smooth function

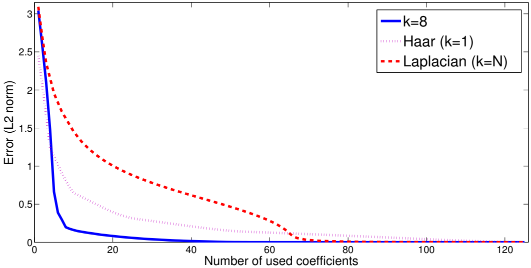


$$\text{sign}(x - \tfrac{1}{2}) \sin(4x)$$

Representing 1D functions – piecewise smooth function



$\mathrm{sign}(x - \frac{1}{2})\sin(4x)$

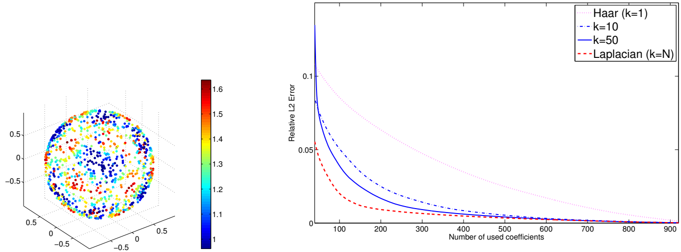


L_2 approximation error

A smooth function on $S^2 \subset \mathbb{R}^3$

A 3D case - 1000 data points distributed on the sphere. Compare between $k = 1, 10, 50, 1000$.

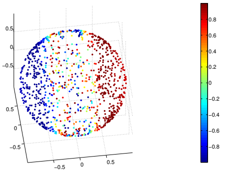
A smooth function on $S^2 \subset \mathbb{R}^3$



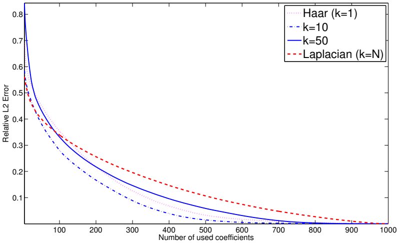
(a) $C(x) = \|\cos(2x)\|$ (b) L_2 relative error

Figure: Representing a smooth function on the sphere.

A rapidly changing function on $S^2 \subset \mathbb{R}^3$



(a) $R(x) = \sin\left((x^T x_0 + 0.2)^{-1}\right)$



(b) L_2 relative error

Figure: R oscillates rapidly in regions on the sphere where x close to be orthogonal to x_0.

Compression of hyper spectral images

Compression of hyper spectral images

- Remote-sensing platforms are often comprised of a cluster of different spectral range detectors or sensors to benefit from the spectral identification capabilities of each range.

Compression of hyper spectral images

- Remote-sensing platforms are often comprised of a cluster of different spectral range detectors or sensors to benefit from the spectral identification capabilities of each range.
- In this example, hyperspectral image of visible spectral region:

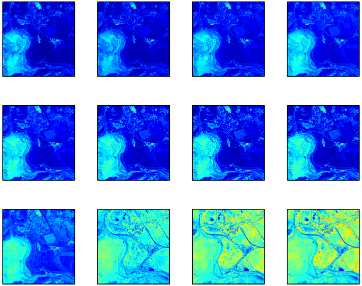
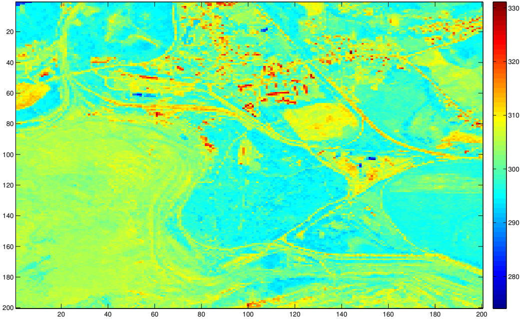


Figure: The 12 different wave length images given as the data.

Compression of surface temperature

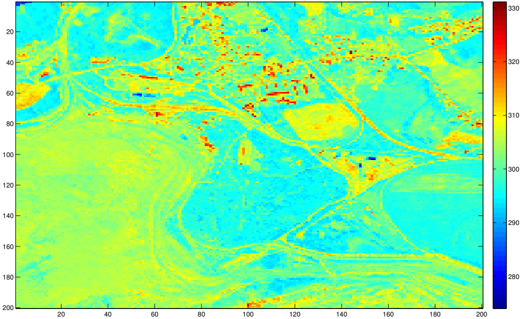
Compression of surface temperature

- The surface temperature is derived form sensors in non-visible spectral region - long-wave infrared sensors. Measured in Kelvin.



Compression of surface temperature

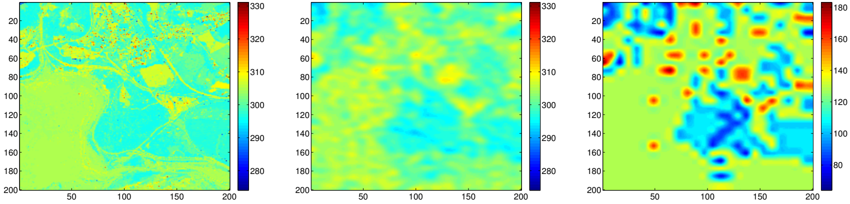
- The surface temperature is derived form sensors in non-visible spectral region - long-wave infrared sensors. Measured in Kelvin.



- We compare the compression with two (non-adaptive) benchmarks: DCT and JPEG2000.

Compression results

Using 200 coefficients, that is 0.5%:



(a) LMW (b) DCT (c) JPEG2000

Thank you !

Questions ?