Optimization and Data Science

Lecture 4: Fast Fourier Transformation and Applications

Prof. Dr. Thomas Slawig

Kiel University - CAU Kiel
Dep. of Computer Science

Summer 2020

Contents

Contents

Recall: DFT and inverse DFT as matrix-vector product

- The DFT mapping $z \mapsto c$ is given by a matrix-vector multiplication

$$c = \frac{1}{m} M z \quad \text{with } M := \left(e^{-i \frac{2\pi kj}{m}} \right)_{k,j=0}^{m-1} \in \mathbb{C}^{m \times m}.$$

- The inverse DFT mapping $c \mapsto z$ is performed with the inverse matrix:

$$z = m M^{-1} c.$$

  Returns values $z_j = f(j \frac{L}{m})$ of function f at equidistant points from its Fourier coefficients.
- The inverse matrix is given by

$$M^{-1} = \frac{1}{m} \left(e^{i \frac{2\pi kj}{m}} \right)_{k,j=0}^{m-1} \in \mathbb{C}^{m \times m}.$$

- Note: Some literature/algorithms define the DFT coefficients c_k without factor $\frac{1}{m}$. Then the factor m has to be omitted in the inverse DFT.

Contents

Fast Fourier Transformation (FFT)

- What is it?

  Efficient implementation of the DFT

  Only $\mathcal{O}(m \log m)$ operations instead of $\mathcal{O}(m^2)$ (standard matrix-vector product)

- Why are we studying this?

  Named one of the Top Ten algorithms of the 20th century,
  in *Computing in Science & Engineering* 2000,
  American Institute of Physics, IEEE Computer Society,
  see `https://archive.siam.org/pdf/news/637.pdf`

- How does it work?

  "Divide and conquer"

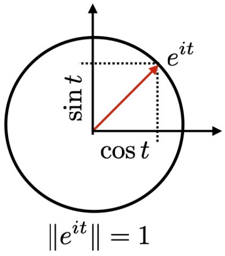
- What if we can use it?

  Significant acceleration of Fourier analysis and synthesis

Fast Fourier Transformation

- We exploit the periodicity of the function

$$t \mapsto e^{it} = \cos t + i \sin t.$$



- Defining $\omega_m := e^{-i\frac{2\pi}{m}}$, we get

$$\omega_m^{kj} = \left(e^{-i\frac{2\pi}{m}}\right)^{kj} = e^{-i\frac{2\pi kj}{m}}.$$

- The Transformation matrix can be written as:

$$M_m = \left(e^{-i\frac{2\pi kj}{m}}\right)_{k,j=0}^{m-1} = \left(\omega_m^{kj}\right)_{k,j=0}^{m-1} \in \mathbb{C}^{m\times m}$$

- ... and the DFT as

$$c_k = \frac{1}{m}\sum_{j=0}^{m-1} z_j e^{-i\frac{2\pi kj}{m}} = \frac{1}{m}\sum_{j=0}^{m-1} z_j \omega_m^{kj}, \quad k = 0, \dots, m-1.$$

Fast Fourier Transformation

- Transformation matrix

$$M_m = \left(e^{-i\frac{2\pi kj}{m}}\right)_{k,j=0}^{m-1} = \left(\omega_m^{kj}\right)_{k,j=0}^{m-1}$$

- We compute some special values:

$$\begin{aligned} kj = m: \quad & \omega_m^m = e^{-i\frac{2\pi m}{m}} = e^{-i2\pi} = \cos(-2\pi) + i\,\sin(-2\pi) = 1 + i \cdot 0 = 1 \\ kj = m+\ell: \quad & \omega_m^{m+\ell} = \omega_m^m \omega^\ell = 1 \cdot \omega_m^\ell = \omega_m^\ell. \end{aligned} \tag{1}$$

- ... and if $m = 2n, n \in \mathbb{N}$:

$$\begin{aligned} kj = \frac{m}{2} = n: \quad & \omega_m^n = e^{-i\frac{2\pi m}{2m}} = e^{-i\pi} = \cos(-\pi) + i\,\sin(-\pi) = -1 + i \cdot 0 = -1 \\ kj = n+\ell: \quad & \omega_m^{n+\ell} = \omega_m^n \omega^\ell = -\omega_m^\ell \end{aligned} \tag{2}$$

Contents

Example ($m = 4$)

- Transformation matrix:

$$M_4 = \left(\omega_4^{kj}\right)_{k,j=0}^{3} = \begin{array}{rcl} & \begin{pmatrix} \omega_4^0 & \omega_4^0 & \omega_4^0 & \omega_4^0 \\ \omega_4^0 & \omega_4^1 & \omega_4^2 & \omega_4^3 \\ \omega_4^0 & \omega_4^2 & \omega_4^4 & \omega_4^6 \\ \omega_4^0 & \omega_4^3 & \omega_4^6 & \omega_4^9 \end{pmatrix} & \begin{array}{l} \leftarrow \quad k=0 \\ \leftarrow \quad k=1 \\ \leftarrow \quad k=2 \\ \leftarrow \quad k=3 \end{array} \\ & \begin{array}{cccc} \uparrow & \uparrow & \uparrow & \uparrow \\ 0 & 1 & 2 & 3 \end{array} & \\ j = & & \end{array}$$

- Periodicity: (1) on page 8 $\Rightarrow \omega_4^0 = \omega_4^4 = 1, \omega_4^{4+\ell} = \omega_4^{\ell}$ gives

$$M_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4 & \omega_4^2 & \omega_4^3 \\ 1 & \omega_4^2 & 1 & \omega_4^2 \\ 1 & \omega_4^3 & \omega_4^2 & \omega_4 \end{pmatrix}$$

Example ($m = 4$)

- Transformation, using periodicity:

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \frac{1}{4} M_4 \begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4 & \omega_4^2 & \omega_4^3 \\ 1 & \omega_4^2 & 1 & \omega_4^2 \\ 1 & \omega_4^3 & \omega_4^2 & \omega_4 \end{pmatrix} \begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix}$$

- Exchange second and third row (means: split w.r.t. even and odd indices):

$$\begin{pmatrix} c_0 \\ c_2 \\ \hline c_1 \\ c_3 \end{pmatrix} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4^2 & 1 & \omega_4^2 \\ 1 & \omega_4 & \omega_4^2 & \omega_4^3 \\ 1 & \omega_4^3 & \omega_4^2 & \omega_4 \end{pmatrix} \begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix}$$

Example ($m = 4$)

- Transformation matrix with exchanged rows can be written as product of two matrices:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4^2 & 1 & \omega_4^2 \\ 1 & \omega_4 & \omega_4^2 & \omega_4^3 \\ 1 & \omega_4^3 & \omega_4^2 & \omega_4 \end{pmatrix} = \left(\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 1 & \omega_4^2 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & \omega_4^2 \end{array}\right) \left(\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & \omega_4^2 & 0 \\ 0 & \omega_4 & 0 & \omega_4^3 \end{array}\right)$$

where we used (see page 8):

$$\omega_4^4 = e^{-i\frac{2\pi 4}{4}} = e^{-i2\pi} = \cos(-2\pi) + i\,\sin(-2\pi) = 1 + i \cdot 0 = 1.$$

and

$$\omega_4^5 = \omega_4^4 \omega_4^1 = 1 \cdot \omega_4^1 = \omega_4.$$

Example ($m = 4$)

- Transformation matrix with exchanged rows was written as product of two matrices:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4^2 & 1 & \omega_4^2 \\ 1 & \omega_4 & \omega_4^2 & \omega_4^3 \\ 1 & \omega_4^3 & \omega_4^2 & \omega_4 \end{pmatrix} = \left(\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 1 & \omega_4^2 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & \omega_4^2 \end{array}\right) \left(\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & \omega_4^2 & 0 \\ 0 & \omega_4 & 0 & \omega_4^3 \end{array}\right).$$

- The second block matrix can be simplified using (2) on page 8: $\omega_4^2 = -1, \omega_4^3 = -\omega_4$:

$$\left(\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & \omega_4^2 & 0 \\ 0 & \omega_4 & 0 & \omega_4^3 \end{array}\right) = \left(\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & -1 & 0 \\ 0 & \omega_4 & 0 & -\omega_4 \end{array}\right) = \begin{pmatrix} I_2 & I_2 \\ D_2 & -D_2 \end{pmatrix}$$

with the identity matrix I_2 and the diagonal matrix $D_2 = \operatorname{diag}(1, \omega_4)$.

Example ($m = 4$): Summary

- Transformation matrix with exchanged rows can be written as product of two matrices:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4^2 & 1 & \omega_4^2 \\ 1 & \omega_4 & \omega_4^2 & \omega_4^3 \\ 1 & \omega_4^3 & \omega_4^2 & \omega_4 \end{pmatrix} = \left(\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 1 & \omega_4^2 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & \omega_4^2 \end{array}\right) \left(\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & -1 & 0 \\ 0 & \omega_4 & 0 & -\omega_4 \end{array}\right)$$

$$= \begin{pmatrix} M_2 & 0_2 \\ 0_2 & M_2 \end{pmatrix} \begin{pmatrix} I_2 & I_2 \\ D_2 & -D_2 \end{pmatrix}$$

where all appearing block matrices have size (2×2):

$$M_2 : \text{the transformation matrix of half size}, \quad 0_2 : \text{zero matrix},$$

$$I_2 : \text{identity matrix}, \quad D_2 = \operatorname{diag}(1, \omega_4) : \text{diagonal matrix}.$$

$\rightsquigarrow$ We can solve the problem of size $m = 4$ by solving two problems of size $\frac{m}{2} = 2$.

Contents

Generalization ($m = 2n, n \in \mathbb{N}$)

- We use again (1) on page 8:

$$\omega_m^{2\ell n} = \omega_m^{\ell m} = (\omega_m^m)^\ell = 1^\ell = 1$$

and

$$\omega_m^2 = e^{-i\frac{4\pi}{m}} = e^{-i\frac{4\pi}{2n}} = e^{-i\frac{2\pi}{n}} = \omega_n$$

- ... for the Fourier coefficients with even indices $k = 2\ell$:

$$c_{2\ell} = \frac{1}{m}\sum_{j=0}^{2n-1} z_j\omega_m^{2\ell j} = \frac{1}{m}\left(\sum_{j=0}^{n-1} z_j\omega_m^{2\ell j} + \sum_{j=0}^{n-1} z_{n+j}{\omega_m}^{2\ell(n+j)}\right)$$

$$= \frac{1}{m}\sum_{j=0}^{n-1}(z_j + z_{n+j})\omega_m^{2\ell j} = \frac{1}{m}\sum_{j=0}^{n-1}(z_j + z_{n+j})\omega_n^{\ell j}.$$

Generalization ($m = 2n, n \in \mathbb{N}$)

- Summary:

$$c_{2\ell} = \frac{1}{m} \sum_{j=0}^{n-1} (z_j + z_{n+j}) \omega_n^{\ell j}, \quad \ell = 0, \dots, n-1.$$

- Thus for the even indices:

$$c_{even} := \begin{pmatrix} c_0 \\ \vdots \\ c_{2(n-1)} \end{pmatrix} = \frac{1}{m} M_n \begin{pmatrix} z_0 + z_n \\ \vdots \\ z_{n-1} + z_{2(n-1)} \end{pmatrix} = \frac{1}{m} M_n \left(\begin{array}{c|c} I_n & I_n \end{array} \right) z$$

with I_n being the identity matrix of half size $n = \frac{m}{2}$.

Generalization ($m = 2n, n \in \mathbb{N}$)

- We use, see (2) on page 8:

$$\omega_m^{(2\ell+1)(n+j)} = \omega_m^{2\ell n}\omega_m^{n+(2\ell+1)j} = \omega_m^{n+(2\ell+1)j} = -\omega_m^{(2\ell+1)j}$$

for the odd indices $k = 2\ell + 1$:

$$\begin{aligned} c_{2\ell+1} = \frac{1}{m}\sum_{j=0}^{2n-1} z_j\omega_m^{(2\ell+1)j} &= \frac{1}{m}\left(\sum_{j=0}^{n-1} z_j\omega_m^{(2\ell+1)j} + \sum_{j=0}^{n-1} z_{n+j}\omega_m^{(2\ell+1)(n+j)}\right) \\ &= \frac{1}{m}\sum_{j=0}^{n-1} z_j\omega_m^{(2\ell+1)j} - \sum_{j=0}^{n-1} z_{n+j}\omega_m^{(2\ell+1)j} \\ &= \frac{1}{m}\sum_{j=0}^{n-1}(z_j - z_{n+j})\omega_m^j\omega_m^{2\ell j} = \frac{1}{m}\sum_{j=0}^{n-1}(z_j - z_{n+j})\omega_m^j\omega_n^{\ell j}. \end{aligned}$$

Generalization ($m = 2n, n \in \mathbb{N}$)

- Summary:

$$c_{2\ell+1} = \frac{1}{m} \sum_{j=0}^{n-1} (z_j - z_{n+j}) \omega_m^j \omega_n^{\ell j}, \quad \ell = 0, \ldots, n-1.$$

- Thus, for the odd indices:

$$c_{odd} := \begin{pmatrix} c_1 \\ \vdots \\ c_{2n-1} \end{pmatrix} = \frac{1}{m} M_n \begin{pmatrix} (z_0 - z_n)\omega_m^0 \\ \vdots \\ (z_{n-1} - z_{2n-1})\omega_m^{n-1} \end{pmatrix} = \frac{1}{m} M_n \left(\begin{array}{c|c} D_n & -D_n \end{array} \right) z$$

with the diagonal matrix

$$D_n = \operatorname{diag}\left(\omega_{2n}^0, \omega_{2n}^1, \omega_{2n}^2, \ldots, \omega_{2n}^{n-1}\right).$$

Algorithmic realization in the general case ($m = 2n, n \in \mathbb{N}$)

- Transformation process $z \mapsto c$ for $c, z \in \mathbb{R}^{2n}$, written as matrix-vector product

$$c = \frac{1}{2n} M_{2n} z, \quad M_{2n} \in \mathbb{R}^{2n \times 2n},$$

- ... can be written as:

$$\begin{pmatrix} c_{even} \\ c_{odd} \end{pmatrix} = \frac{1}{2n} \begin{pmatrix} M_n & 0_n \\ 0_n & M_n \end{pmatrix} \begin{pmatrix} I_n & I_n \\ D_n & -D_n \end{pmatrix} z$$

- Last matrix-vector product is multiplication with diagonal matrix $\rightsquigarrow$ effort $\mathcal{O}(n)$.

$\rightsquigarrow$ Effort of problem of size $2n \approx 2\times$effort of problem of size n.

$\rightsquigarrow$ Recursive application (“divide and conquer”)

$\rightsquigarrow$ $m = 2^k, k = \log_2 m$: effort $\mathcal{O}(m \log m)$ instead of $\mathcal{O}(m^2)$ for standard matrix-vector product.

Contents

FFT for real-valued data

For real-valued data, complex coefficients are transformed into real ones using Euler's formula

$$e^{it} = \cos t + i \sin t.$$

Theorem

Let $m = 2n, n \in \mathbb{N}, z \in \mathbb{R}^m$ and $c = (c_k)_{k=0}^{m-1} \in \mathbb{C}^m$ be the corresponding complex Fourier coefficients. Then

$$\frac{a_0}{2} + \sum_{k=1}^{n-1} \left(a_k \cos \frac{2\pi k t_j}{L} + b_k \sin \frac{2\pi k t_j}{L} \right) + \frac{a_n}{2} \cos \frac{2\pi n t_j}{L} = z_j, t_j = j\frac{L}{m}, j = 0, \ldots, m-1, \quad (3)$$

where

$$a_k = 2\, Re\, c_k, k = 0, \ldots, n, \quad b_k = -2\, Im\, c_k, k = 1, \ldots, n-1.$$

$\mathrm{Re}\, z \in \mathbb{R}$ and $\mathrm{Im}\, z \in \mathbb{R}$ denote the real and imaginary part of $z \in \mathbb{C}$, i.e., $z = \mathrm{Re}\, z + i\, \mathrm{Im}\, z$.

FFT for real-valued data

- The method above, i.e., treating real-valued data $z \in \mathbb{R}^m$ as complex values $z \in \mathbb{C}^m$ with zero imaginary part, wastes storage (m complex $\hat{=}\, 2m$ real numbers for m real data).
- Alternative: given $z = (z_j)_{j=0}^{m-1} \in \mathbb{R}^m$, $m = 2n$, create complex data $\tilde{z} \in \mathbb{C}^n$ as

$$\tilde{z}_\ell := z_{2\ell} + iz_{2\ell+1}, \quad \ell = 0, \dots, n-1.$$

- Apply (complex) DFT to $\tilde{z}$, giving coefficients $c = (c_k)_{k=0}^{n-1} \in \mathbb{C}^n$.
- Then, the coefficients a_k, b_k used in (3) on page 22 can be computed as

$$a_k = \mathrm{Re}\left(\frac{1}{2}(c_k + \bar{c}_{n-k}) + \frac{1}{2i}(c_k - \bar{c}_{n-k})e^{-\frac{ik\pi}{n}}\right), \quad k = 0, 1, \dots, n,$$
$$b_k = -\mathrm{Im}\left(\frac{1}{2}(c_k + \bar{c}_{n-k}) + \frac{1}{2i}(c_k - \bar{c}_{n-k})e^{-\frac{ik\pi}{n}}\right), \quad k = 1, \dots, n,$$

  with $c_n = c_0$.
- Here, $\bar{w}$ denotes the **complex conjugate** of $w = x + iy \in \mathbb{C}$, i.e., $\bar{w} := x - iy \in \mathbb{C}$.

Contents

Interpretation and application

- Assume we have given data $z \in \mathbb{R}^m$
- ... and computed the real Fourier coefficients a, b.
- Interpretation of the formula of the Theorem on page 22:

$$\frac{a_0}{2} + \sum_{k=1}^{n-1} \left(a_k \cos \frac{2\pi k t_j}{L} + b_k \sin \frac{2\pi k t_j}{L} \right) + \frac{a_n}{2} \cos \frac{2\pi n t_j}{L} = z_j, \quad t_j = j\frac{L}{m}, j = 0, \ldots, m-1.$$

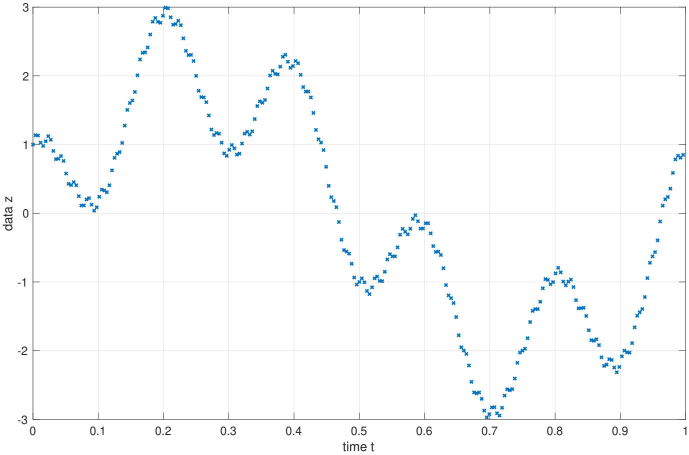
- We can deduce which frequencies k are dominant.
- We can omit frequencies with small Fourier coefficients $\leadsto$ data compression.
- We can detect and omit high frequencies (which might be random perturbations).

Example (last lecture)

- Given: dataset

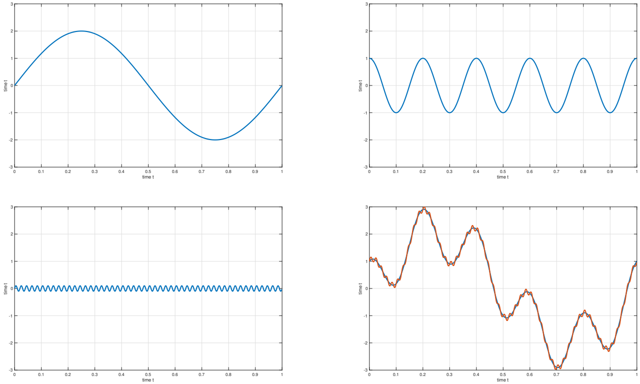
  $$(t_j, z_j)_{j=0,\ldots,m-1}, t_j, z_j \in \mathbb{R}.$$

- Example: t time, z measurements.
- We see "some" structure ...
- How to analyze this?



Example

- Here we have the Fourier coefficients $b_1 = 2, a_5 = 1, b_{50} = 0.1$.
- The Fourier coefficients are the amplitudes of the periodic parts in the data corresponding to the the frequency k.



Multi-dimensional Fourier Transformation

- ... is sequentially applied for each dimension.
- Example: color pictures are 3-D data
  - Image needs 1 Byte per color and pixel
  - Apply DFT for each color (e.g., 3 times for RGB pictures)
  - Apply for each row in a 2-D picture
  - Apply for the resulting columns
  - JPEG includes FFT as one step.
- Compression: delete the Fourier coefficients $c_k \in \mathbb{C}$ with $|c_k| < \epsilon$, a given threshold.
- Noise elimination: delete high frequencies

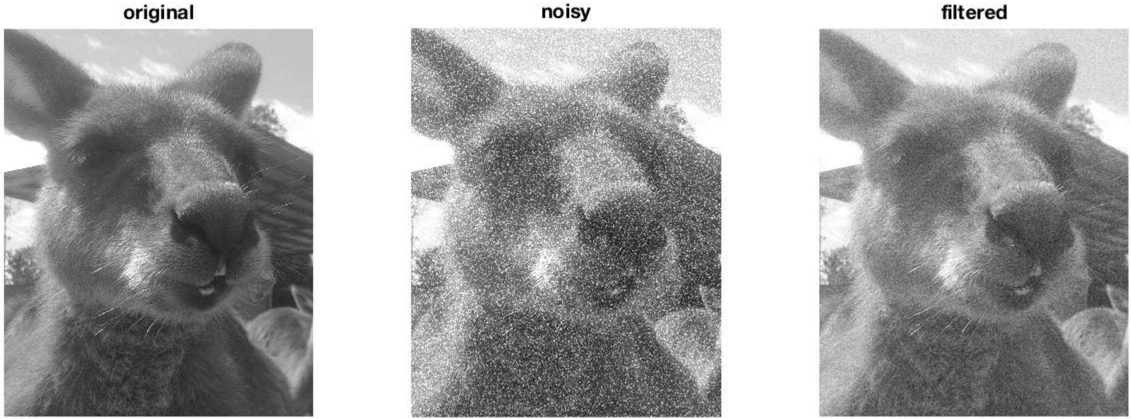
Example: Image compression with DFT

original size: 1060759

compressed, #fft coeff: 107264

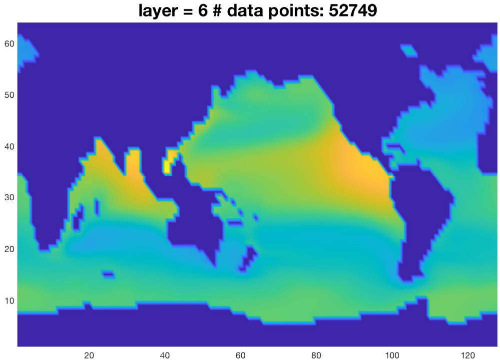
Example: Image compression with DFT

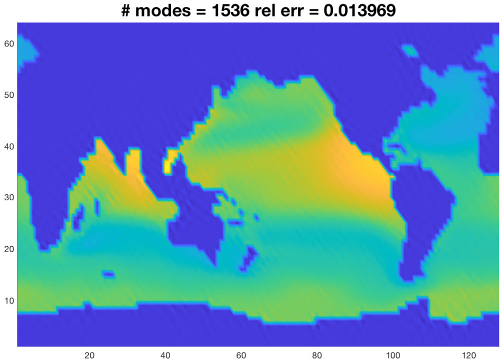
Example: Denoising with DFT



Example: 3-dimensional data compression

Compression of 3-D data (simulation of nutrients in the ocean):





Fourier Analysis: What is important

- Fourier analysis is an important and powerful tool to analyse, compress and denoise data in arbitrary dimensions.
- It performs a transformation of the data into the frequency domain.
- It is based on a transformation of the data considered as complex numbers.
- This transformation is a matrix-vector multiplication.
- Exploiting periodicity properties and applying the divide-and-conquer principle, a very efficent algorithm (the FFT) was developed.
- The FFT reduces the effort to $\mathcal{O}(m \log m)$, where m is the dimension of the data.
- Real-valued data are considered as complex-valued data, or considered as real and imaginary parts of complex numbers (to obtain even higher efficiency).
- Hardware-optimized implementations of the FFT are available in nearly all languages.
- It does not make sense to write another one, but to use a library function instead.