Fourier Frame approximations: Algorithms and Applications

Roel Matthysen

May 23, 2016

Joint work with Daan Huybrechs
Department of Computer Science, KU Leuven

Goal

Given a function f as N data points on an equispaced grid, find an expansion in Fourier basis functions that

- converges fast
- is easily computable ($O(N)$ operations).

Example: approximate $\exp(x)$ for increasing dof:

(a) Fourier interpolation

(b) Fourier Frame

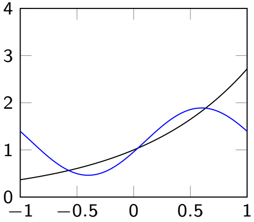
A set of Fourier basisfunctions restricted to a subdomain constitutes a Frame.

Goal

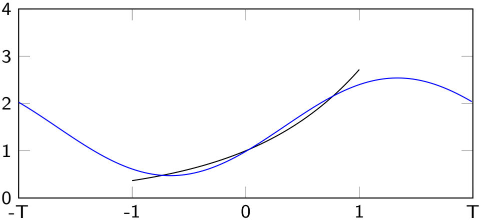
Given a function f as N data points on an equispaced grid, find an expansion in Fourier basis functions that

- converges fast
- is easily computable ($O(N)$ operations).

Example: approximate $\exp(x)$ for increasing dof:



(a) Fourier interpolation



(b) Fourier Frame

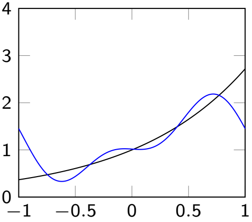
A set of Fourier basisfunctions restricted to a subdomain constitutes a Frame.

Goal

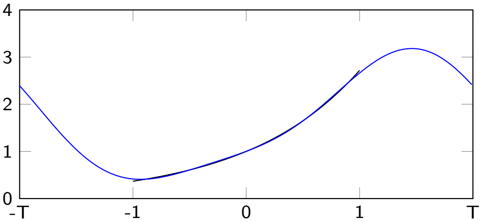
Given a function f as N data points on an equispaced grid, find an expansion in Fourier basis functions that

- converges fast
- is easily computable ($O(N)$ operations).

Example: approximate $\exp(x)$ for increasing dof:



(a) Fourier interpolation



(b) Fourier Frame

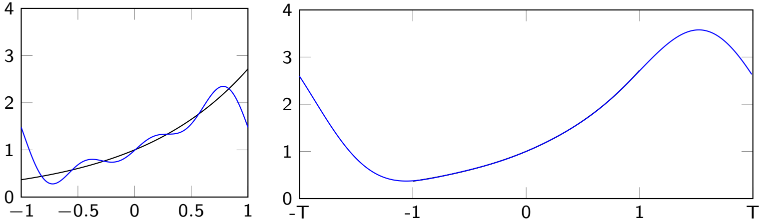
A set of Fourier basisfunctions restricted to a subdomain constitutes a Frame.

Goal

Given a function f as N data points on an equispaced grid, find an expansion in Fourier basis functions that

- converges fast
- is easily computable ($O(N)$ operations).

Example: approximate $\exp(x)$ for increasing dof:



(a) Fourier interpolation (b) Fourier Frame

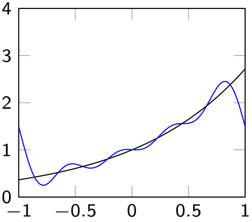
A set of Fourier basisfunctions restricted to a subdomain constitutes a Frame.

Goal

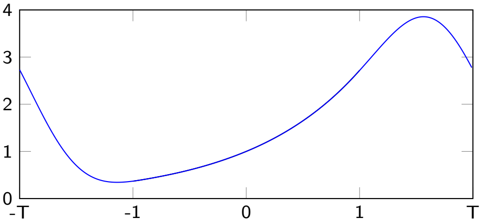
Given a function f as N data points on an equispaced grid, find an expansion in Fourier basis functions that

- converges fast
- is easily computable ($O(N)$ operations).

Example: approximate $\exp(x)$ for increasing dof:



(a) Fourier interpolation



(b) Fourier Frame

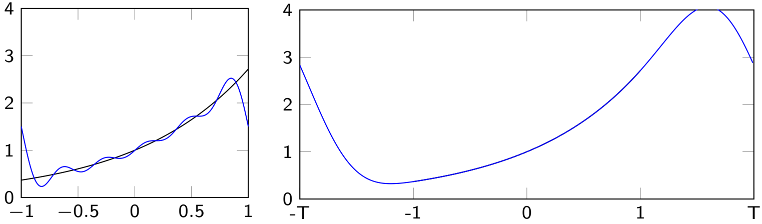
A set of Fourier basisfunctions restricted to a subdomain constitutes a Frame.

Goal

Given a function f as N data points on an equispaced grid, find an expansion in Fourier basis functions that

- converges fast
- is easily computable ($O(N)$ operations).

Example: approximate $\exp(x)$ for increasing dof:



(a) Fourier interpolation

(b) Fourier Frame

A set of Fourier basisfunctions restricted to a subdomain constitutes a Frame.

Fourier Frame

Fourier Frame on $[-T, T]$

- Basis functions $\phi_m(x) = \frac{1}{\sqrt{2T}} e^{i\frac{\pi m}{T}x}, m = -M, \ldots, M$
- Approximation $g(x) = \sum_{k=-M}^{M} a_k \phi_k(x)$

Fitting the data on $[-1, 1]$

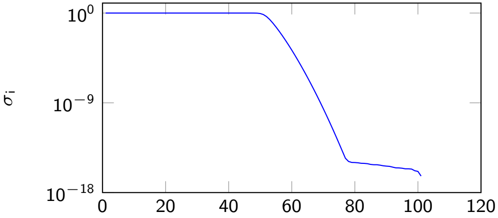
- Collocation at equidistant points $x_j = \frac{j}{N}, j = -N, \ldots, N$.
- Least squares problem $A_{ij} = \phi_i(x_j),\ b_j = f(x_j)$

$$Aa = b$$

- A is DFT-subblock
- Some oversampling $N = \eta M$ required, A is a tall matrix ($\eta \sim 2$)
- Fast solver for $Aa = b$ needed
- Fast algorithm exists for $T = 2$ (Lyon 2011)

A closer look at A

Singular values of A



- Spectrum has a plateau shape (Slepian 1978, Wilson 1987)
- Size of transition/plunge/problematic region

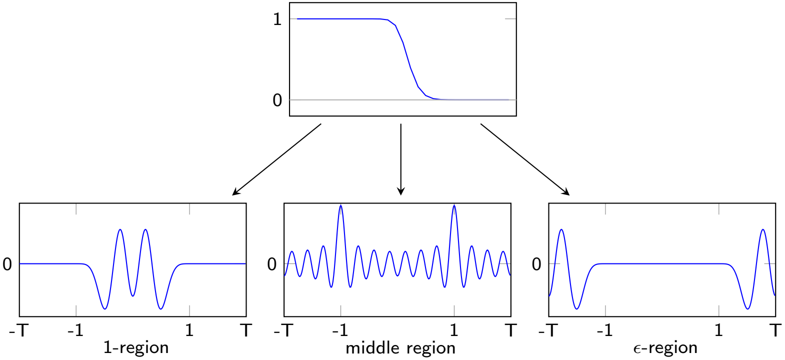
$$1 - \epsilon > \theta_i > \epsilon$$

  grows as $O(\log N)$ for 1D problems.
- System matrix is extremely ill-conditioned

A closer look at A

Ill-conditioning explained

Fourier series corresponding to the right singular vectors of A:



- Solving with Truncated SVD yields approximation to machine accuracy (Adcock, Huybrechs, Martin-Vaquero 2014)

Solving the least-squares system

Truncated SVD

- $A = U\Sigma V'$
- $x = V\Sigma^{-1}U'b$
- $x = \underbrace{V_1\Sigma_1^{-1}U_1'b_1}_{x_1} + \underbrace{V_{mid}\Sigma_{mid}^{-1}U_{mid}'b_{mid}}_{x_{mid}} + \underbrace{V_\epsilon\Sigma_\epsilon^{-1}U_\epsilon'b_\epsilon}_{x_\epsilon}$

Assumption: b_ϵ is negligible (discrete Picard condition)

x_1 is easy when x_{mid} is known

- $b_1 = b - b_{mid} = b - Ax_{mid}$
- $x_1 = V_1\Sigma_1^{-1}U_1'b_1 \approx V_1\Sigma_1U_1'b_1 = A'b_1$

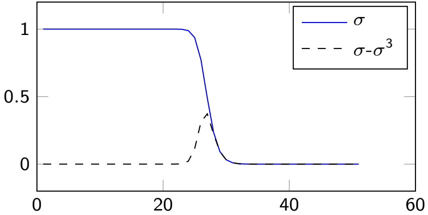
All you need is one application of A and A' (fast!)

How to find x_{mid}?

Isolating the problematic singular values

- 'Projection operator' $I - AA'$ isolates the middle singular values

$$(I - AA')A = U(\Sigma - \Sigma^3)V'$$



The numerical nullspace of $(I - AA')A$ contains both the 1 and ϵ-regions.

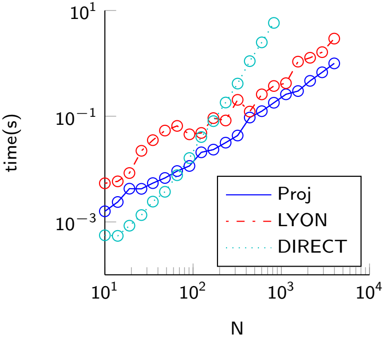
- Solving $(I - AA')Ax = (I - AA')b$ yields x_{mid}
- $(I - AA')A$ has numerical rank $O(\log N)$
- This can be solved efficiently in $O(N \log^2 N)$

Implementation

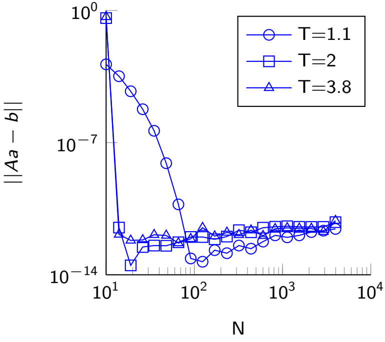
Algorithm

$$(I - AA')Ax_\beta = (I - AA')b$$
$$x_\alpha = A'(b - Ax_\beta)$$
$$x = x_\alpha + x_\beta$$

Results



(a) timings

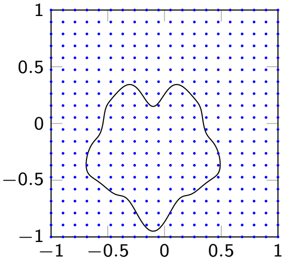


(b) convergence

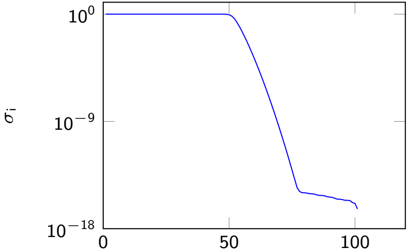
2D Frames

Fourier Frame

- A consists of selected rows of the 2D DFT matrix.



(a) Masked Grid

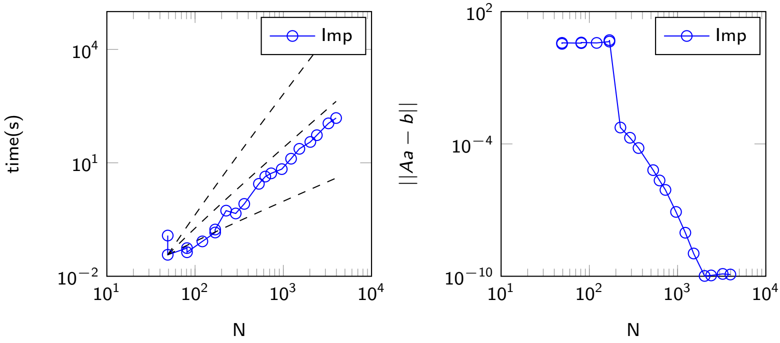


(b) Singular values of A

- Partially proven: the problematic region grows as $O(\sqrt{N})$ for N total points.
- The algorithm complexity becomes $O(N^2)$

2D Frames

Experimental results for 2D extensions:

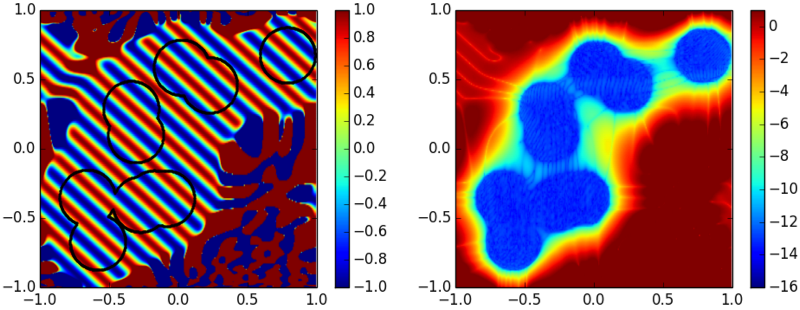


(a) Timings (b) Error

- Complexity is $O(N^2)$, as expected
- Convergence speed is seemingly conserved

2D Fourier Frame Approximation

$$f(x, y) = cos(20x + 22y)$$



(a) Fourier⊗Chebyshev Extension (b) $\log_{10}(error)$

Possible extensions

Observation: You can add extra rows and/or columns to A while preserving the plateau shape. Let $\bar{A} = \begin{bmatrix} A & b \end{bmatrix}$. Then because of the interlacing property of singular values

$$\bar{\sigma}_1 > \sigma_1 > \bar{\sigma}_2 > \dots$$

the spectra are similar:

$$sp(A) : 1 > \sigma_1 > \dots > \sigma_k > (1-\epsilon) > \sigma_{k+1} > \dots > \sigma_l > \epsilon > \sigma_{l+1} > \dots > 0.$$
$$sp(\bar{A}) : 1 > \bar{\sigma}_2 > \dots > \bar{\sigma}_k > (1-\epsilon) > \bar{\sigma}_{k+2} > \dots > \bar{\sigma}_l > \epsilon > \bar{\sigma}_{l+2} > \dots > 0.$$

Problematic region contains a maximum of 3 extra singular values. This might be useful for

- Some extra functions added to the frame
- Boundary conditions when solving differential equations

Differential Equations

Derivative is diagonal operator

$$f(x) = \sum_{m=-M}^{M} a_m \phi_m(x), \qquad f'(x) = \sum_{m=-M}^{M} \frac{a_m i \pi m}{T} \phi_m(x)$$

A differential equation

$$\Delta A + k^2 A = f$$

becomes

$$(D_x^2 + D_y^2 + k^2 I)\boldsymbol{c}_A = \boldsymbol{c}_f$$

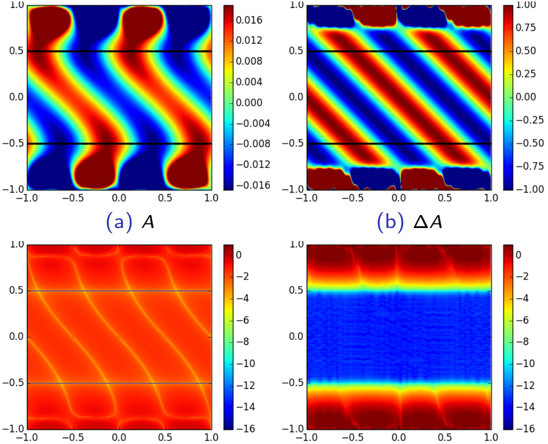
Implementation

- Boundary conditions are sampled at the boundary and added as extra rows in the system matrix. Complexity remains $O(N \log N)/O(N^2)$.
- If the diagonal operator is not invertible, care has to be taken of the DC coefficient

Differential Equations

Poisson Equation

$$\Delta A = \cos(2\pi(x+y)), \qquad \delta A(x,y)/\delta y = 0, \quad (x,y) \in \delta\Omega$$



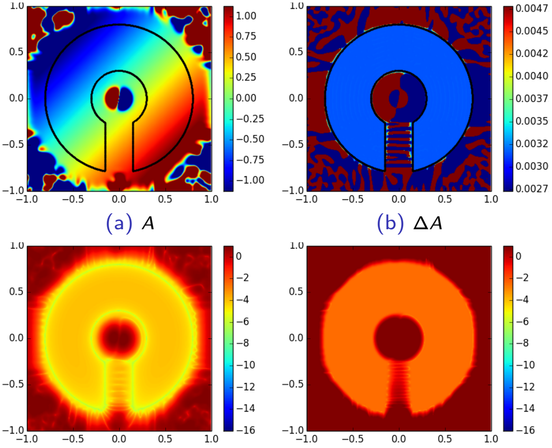
(a) A (b) ΔA

(c) $\log_{10}(\text{error}_{boundary})$ (d) $\log_{10}(\text{error}_{domain})$

Differential Equations

Laplace Equation

$$\Delta A = 0, \qquad A(x,y) = x - y, \quad (x,y) \in \delta\Omega$$



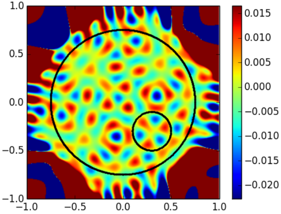
(a) A (b) ΔA

(c) $\log_{10}(\text{error}_{boundary})$ (d) $\log_{10}(\text{error}_{domain})$

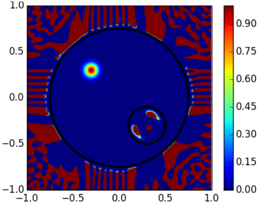
Differential Equations

Helmholtz Equation

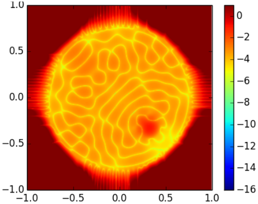
$$\Delta A + 35^2 A = e^{-200((x-0.3)^2+(y+0.3)^2)}, \qquad A(x,y) = 0, \quad (x,y) \in \delta\Omega$$



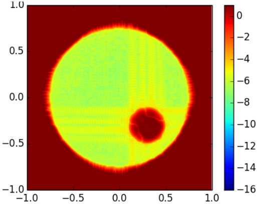
(a) A



(b) $\Delta A + k^2 A$



(c) $\log_{10}(\text{error}_{boundary})$



(d) $\log_{10}(\text{error}_{domain})$

The end

Julia code available at `https://github.com/daanhb/Framefuns.jl`