The Nonuniform FFT and its applications

Leslie Greengard
Center for Computational Mathematics, Flatiron Institute
& Courant Institute, New York University

with *Z. Gimbutas, S. Inati, J.-Y. Lee, L. Fleysher, R. Fleysher*

New York Scientific Data Summit 2020
October 23, 2020

COURANT INSTITUTE OF MATHEMATICAL SCIENCES

FLATIRON INSTITUTE

Thanks to:

Alok Dutt, Vladimir Rokhlin

Alex H. Barnett, Jeremy F. Magland, Ludvig af Klinteberg,
Yu-hsuan (Melody) Shih, Andrea Malleo, Libin Lu,
Joakim Andén

fiNUFFT

Greg Beylkin, Charles Epstein, Hannah Lawrence, Patrick Lin

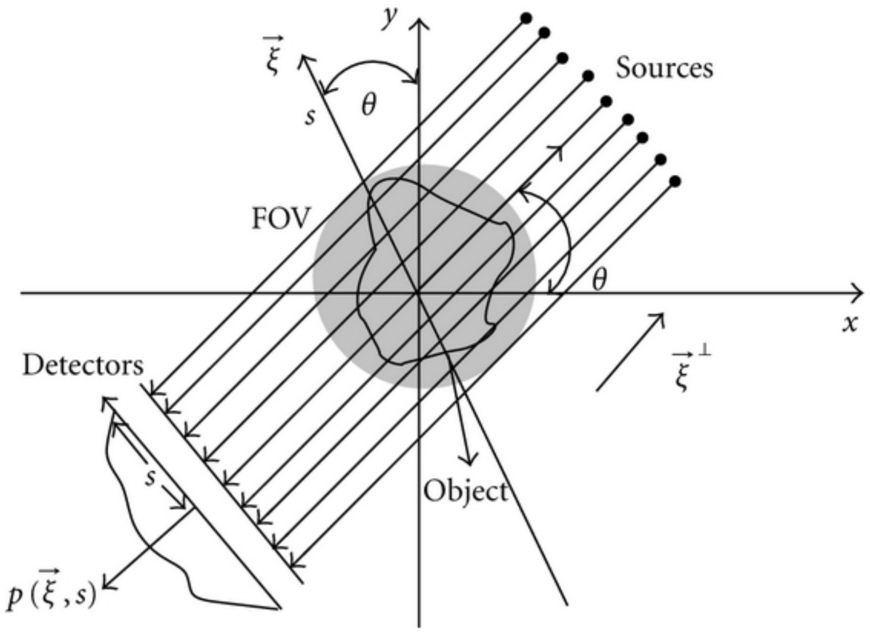
NYU COURANT INSTITUTE OF MATHEMATICAL SCIENCES

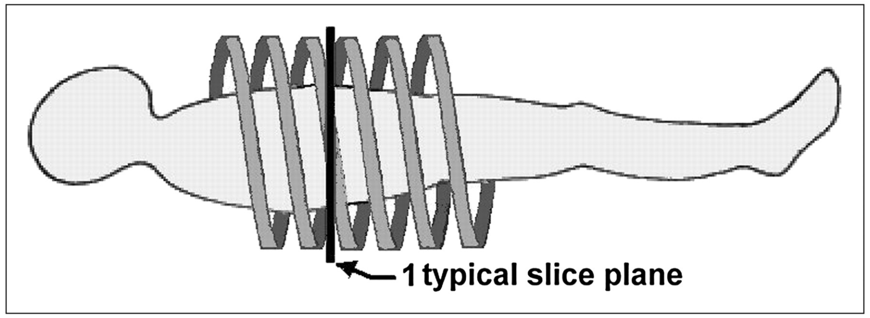
FLATIRON INSTITUTE

X-Ray CT Imaging

1980 era scanner (Science Museum, London)

Modern spiral scanner (Siemens)





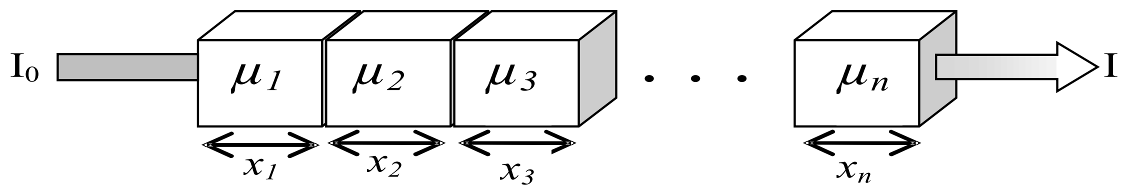


Figure 1: Attenuation of radiation by different attenuation coefficients

Olliveira et al., International Nuclear Atlantic Conference - INAC 2011

$$I = I_0\, e^{-(\mu_1 x_1 + \mu_2 x_2 + \ldots \mu_N x_N)}$$

$$\Leftrightarrow \sum_{n=1}^{N} \mu_n x_n = ln \frac{I_0}{I}$$

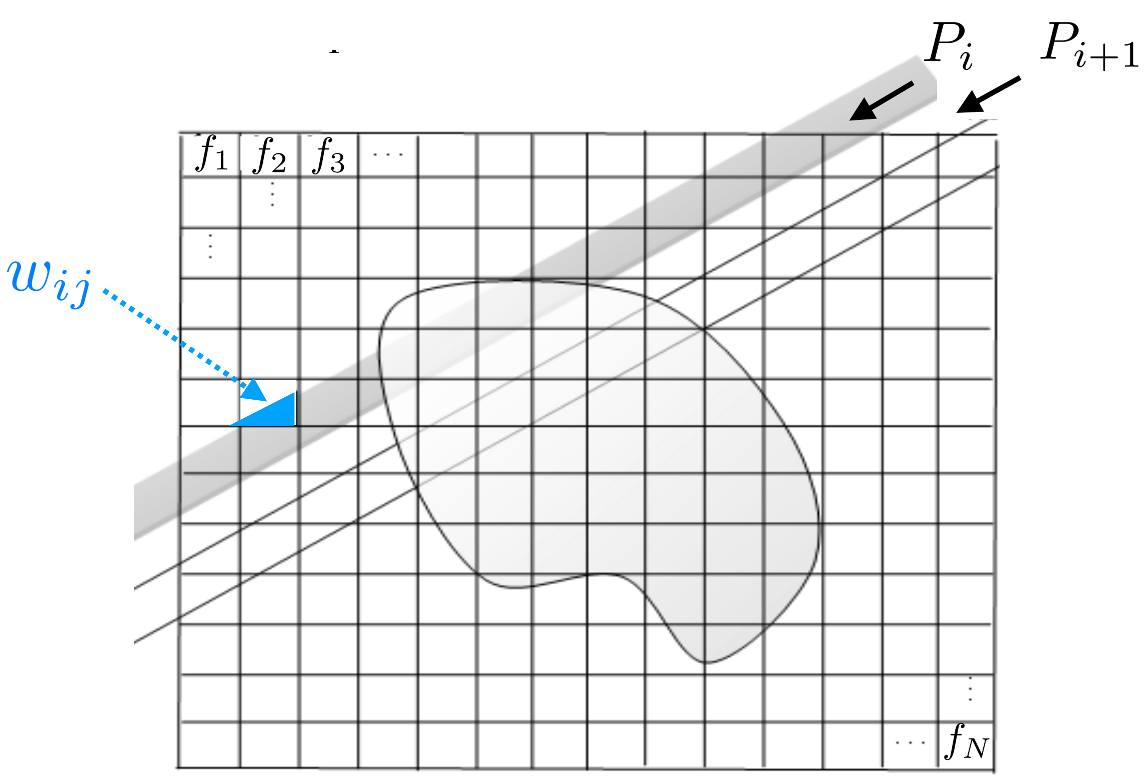
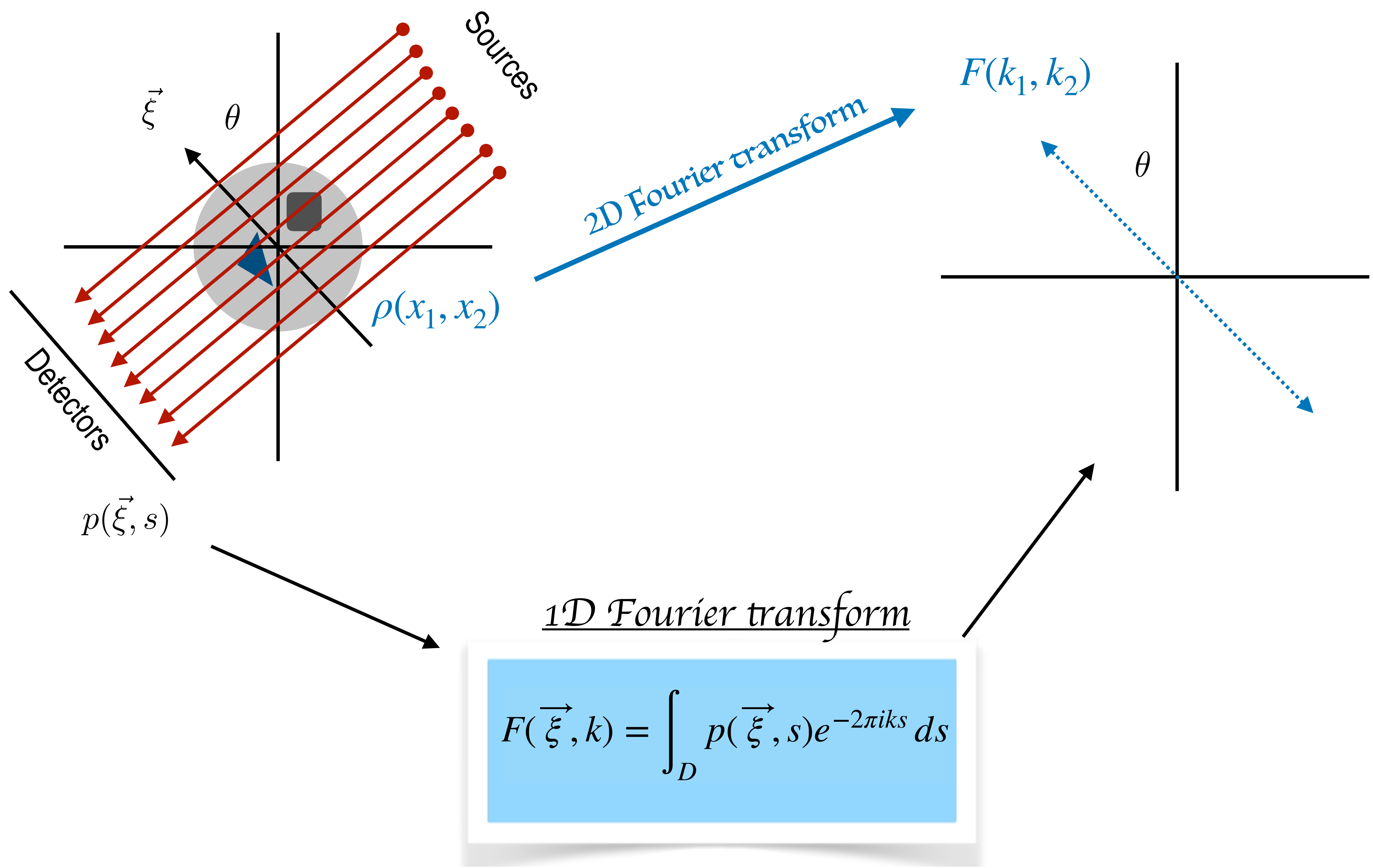
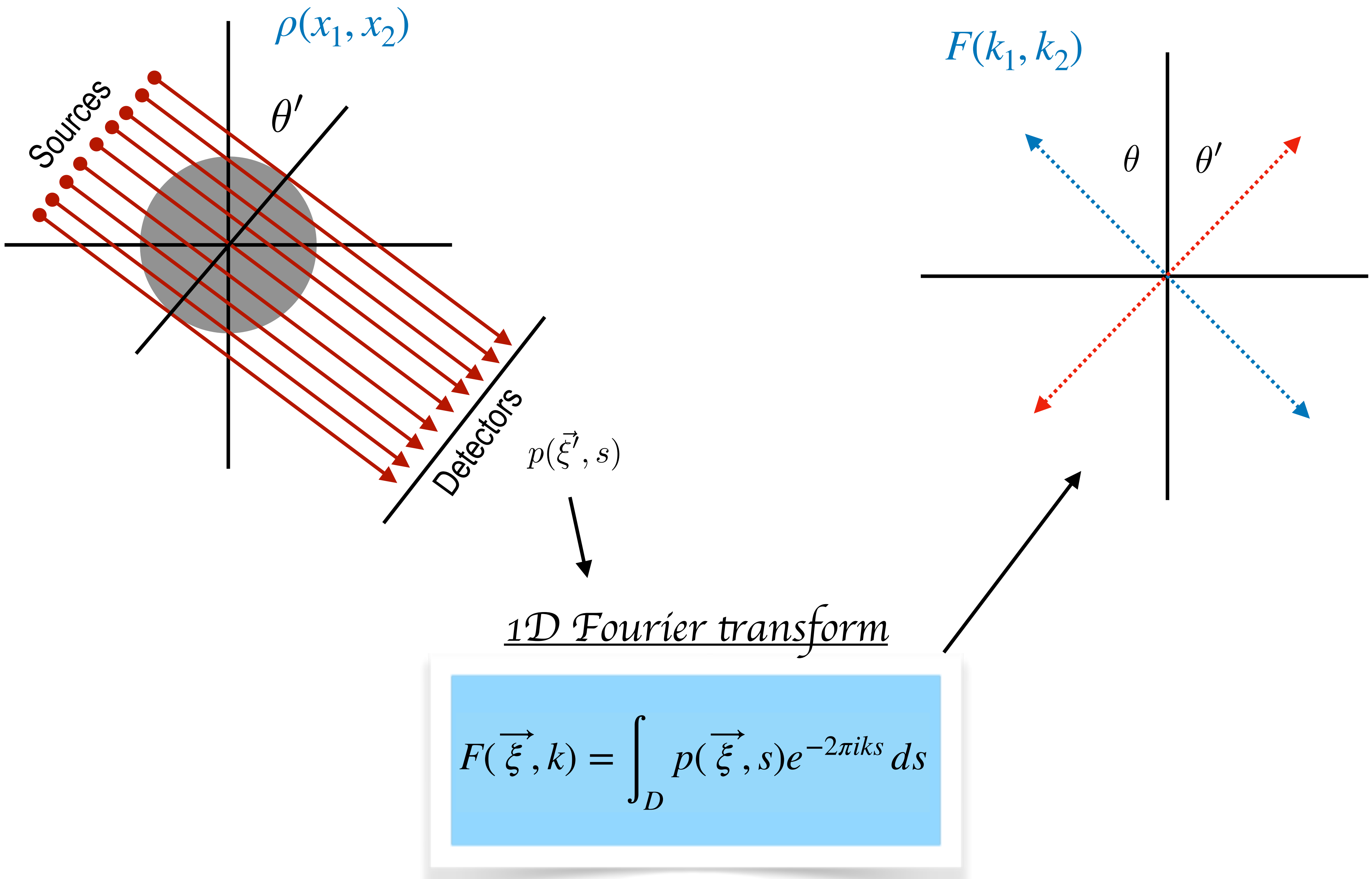


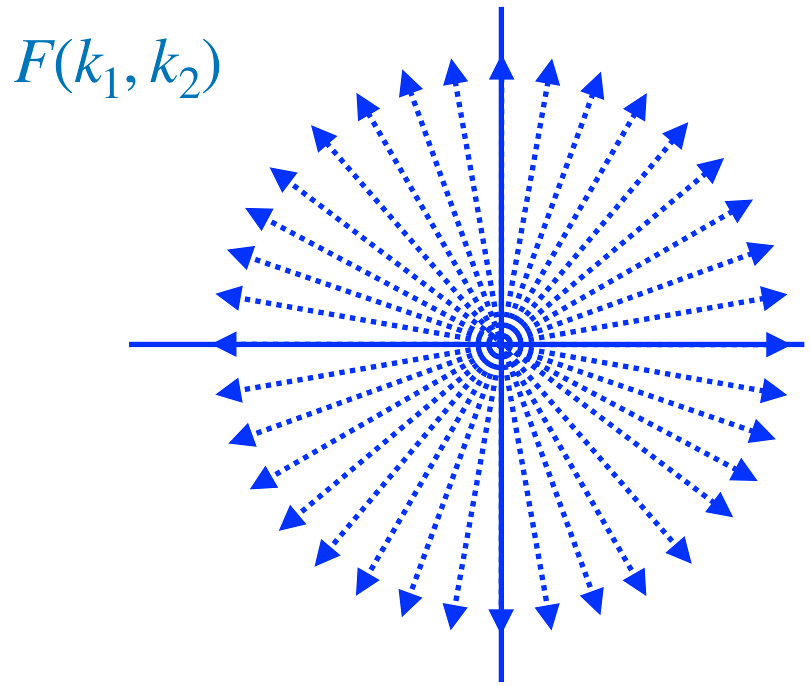
Figure 2: Discretization of the irradiated section

$$P_i = \sum_{j=1}^{N} f_j\, w_{ij}, \;\; i = 1, 2, \ldots, M$$

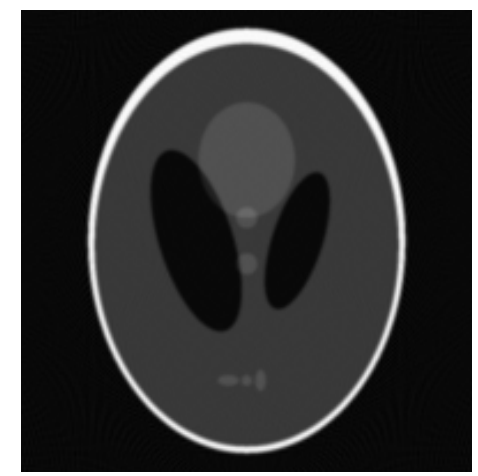

Sources
$\vec{\xi}$
θ
$\rho(x_1, x_2)$
Detectors
$p(\vec{\xi}, s)$
2D Fourier transform
$F(k_1, k_2)$
θ
1D Fourier transform
$F(\overrightarrow{\xi}, k) = \int_D p(\overrightarrow{\xi}, s) e^{-2\pi i k s}\, ds$


$\rho(x_1, x_2)$
Sources
θ'
Detectors
$p(\vec{\xi}', s)$
$F(k_1, k_2)$
θ
θ'
1D Fourier transform
$F(\overrightarrow{\xi}, k) = \int_D p(\overrightarrow{\xi}, s) e^{-2\pi iks} \, ds$

Continue process to "fill in k-space"



Repeat for multiple angles until desired resolution is achieved



Shepp Logan phantom

<u>*2D Fourier transform*</u>

$$F(k_1, k_2) = \iint \rho(x_1, x_2) e^{-2\pi i (k_1 x_1 + k_2 x_2)}\, dx_1 dx_2$$

$$\rho(x_1, x_2) = \iint F(k_1, k_2) e^{2\pi i (k_1 x_1 + k_2 x_2)}\, dk_1 dk_2$$

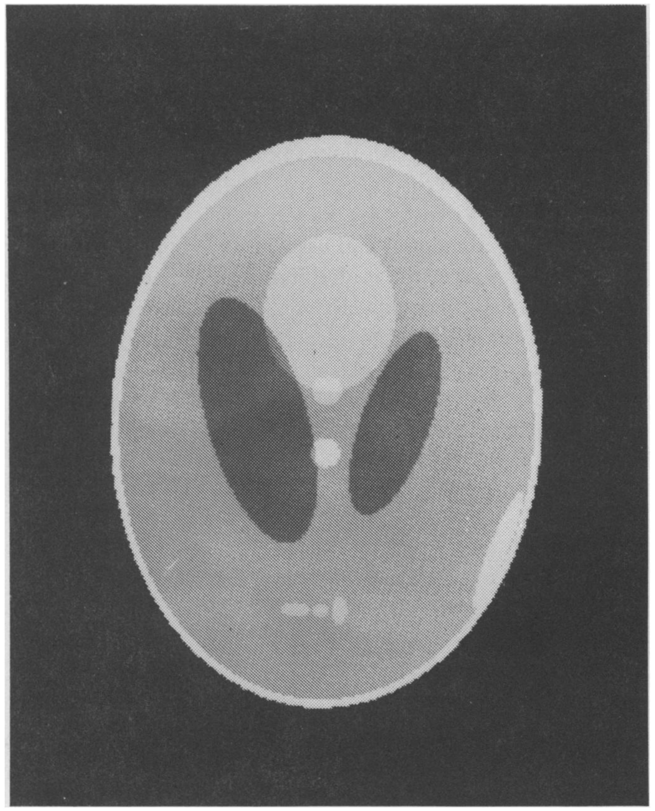


Fig. 1. Simulation of human head using 11 ellipses. The density of the skull is 2.0 and of the ventricles, tumors, etc. is 1.0-1.05 (see [20] for more details).

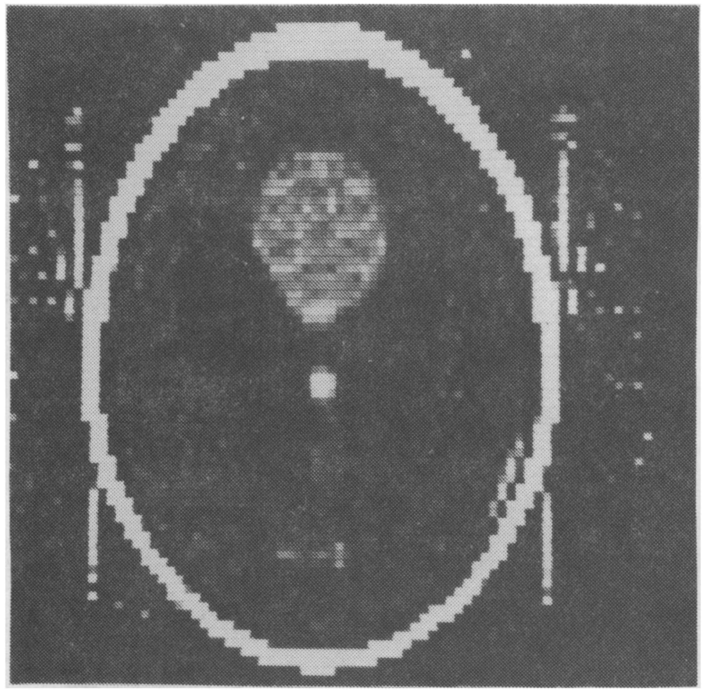


Fig. 2. Reconstruction using the algorithm embodied in the first commercial machine (EMI Ltd.) from 180×160 strip projection data obtained by exact calculation from Fig. 1.

L. Shepp and J. B. Kruskal,
Computerized Tomography: The New Medical X-Ray Technology,

[*The American Math. Monthly, 1978*]

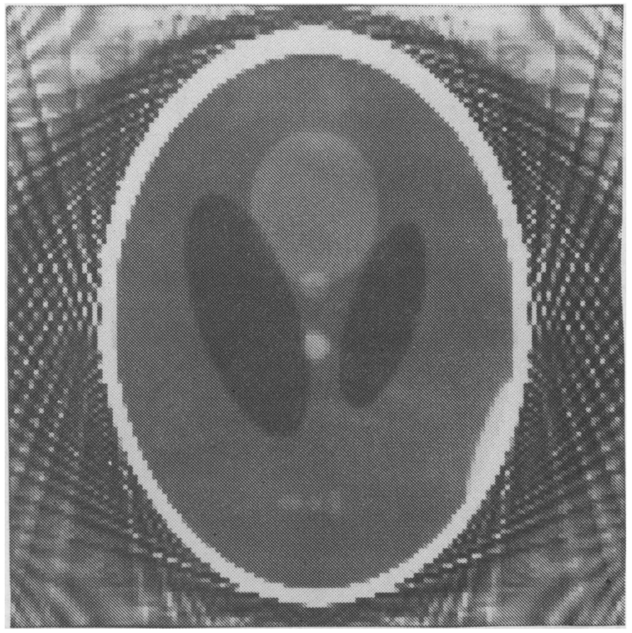
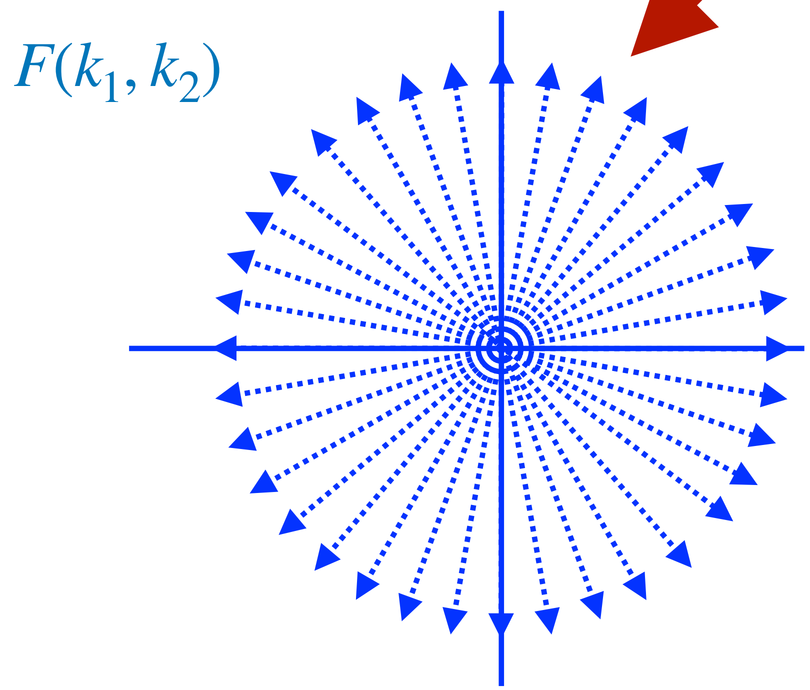
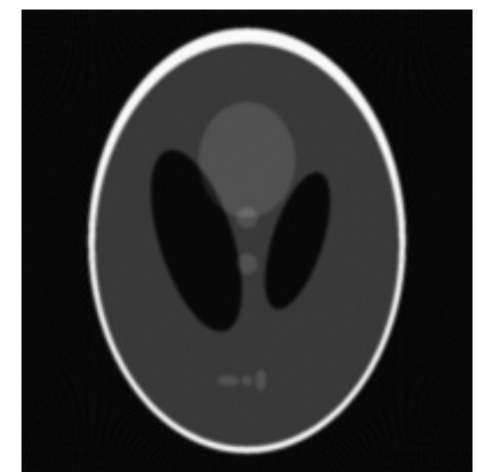


Fig. 3. Reconstruction from the same data using the Fourier based algorithm of Shepp [20] (see [20] for more details).

Points are sampled on radial grid
and the FFT does not apply





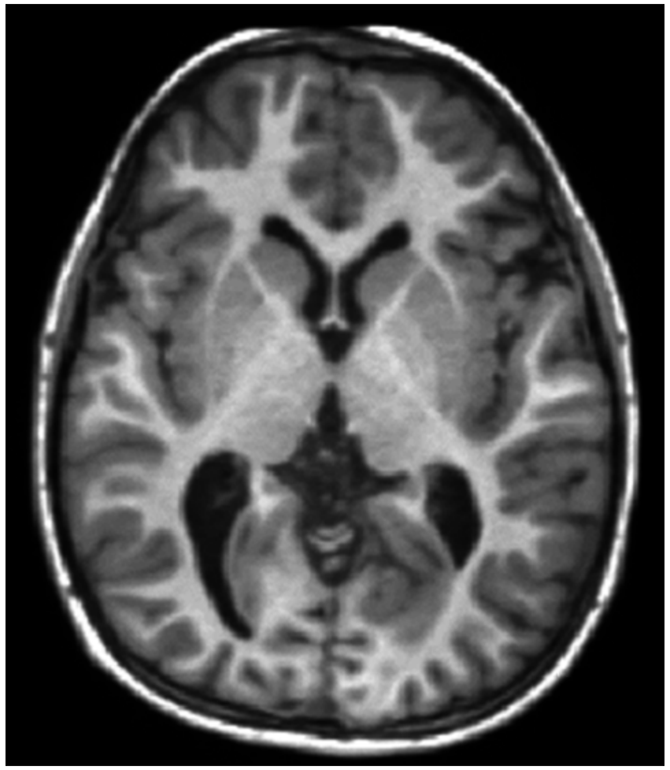
Shepp Logan phantom

2D Fourier transform

$$F(k_1, k_2) = \int\int \rho(x_1, x_2) e^{-2\pi i(k_1 x_1 + k_2 x_2)}\, dx_1 dx_2$$

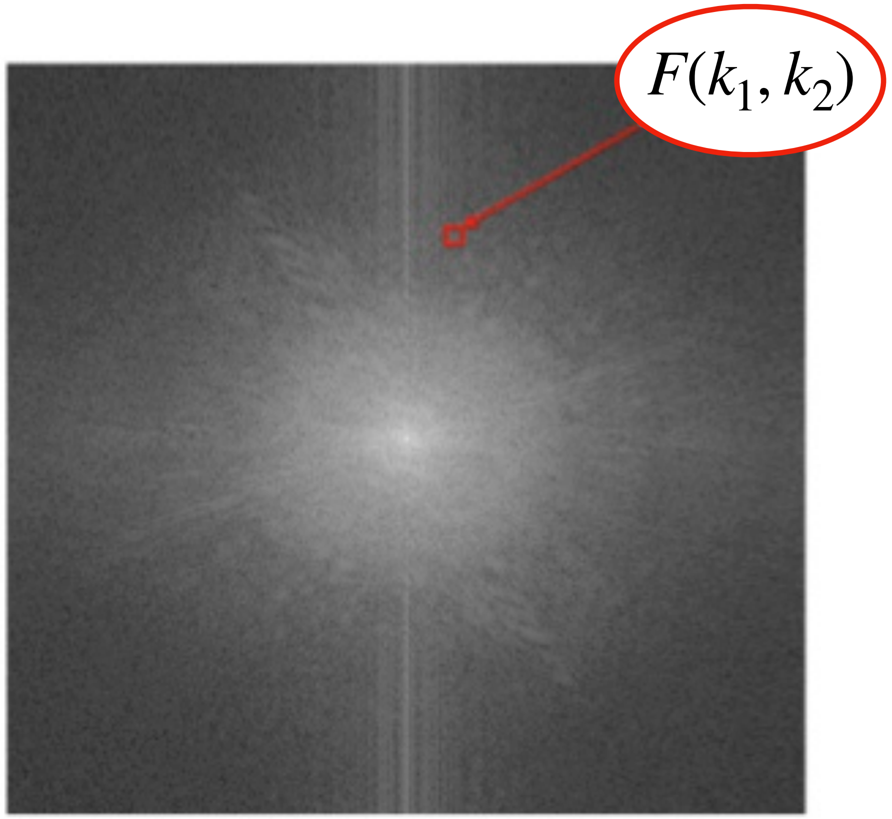
$$\rho(x_1, x_2) = \int\int F(k_1, k_2) e^{2\pi i(k_1 x_1 + k_2 x_2)}\, dk_1 dk_2$$

Magnetic Resonance Imaging



$\mathscr{F}$

$\rightarrow$



$$F(k_1, k_2) = \int\int \rho(x_1, x_2) e^{-2\pi i (k_1 x_1 + k_2 x_2)} \, dx_1 dx_2$$

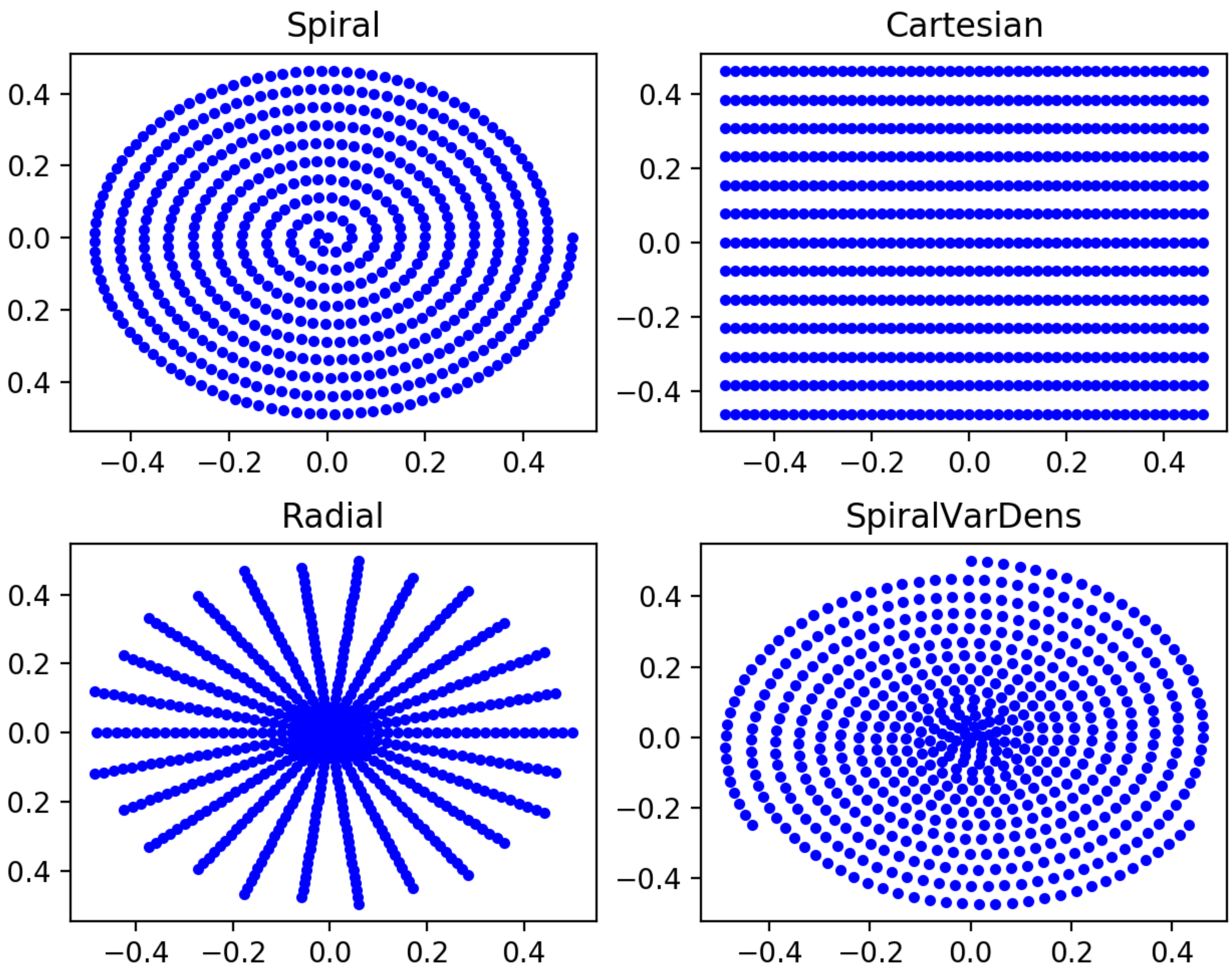
$$s(t) = \int\int \rho(x, y) e^{-2\pi i (k_1(t) x_1 + k_2(t) x_2)} \, dx_1 dx_2$$

Signal Equation

$$s(t) = \int\int \rho(x, y) e^{-2\pi i (k_1(t) x_1 + k_2(t) x_2)} e^{-i\phi(x_1, x_2) t} \, dx_1 dx_2$$

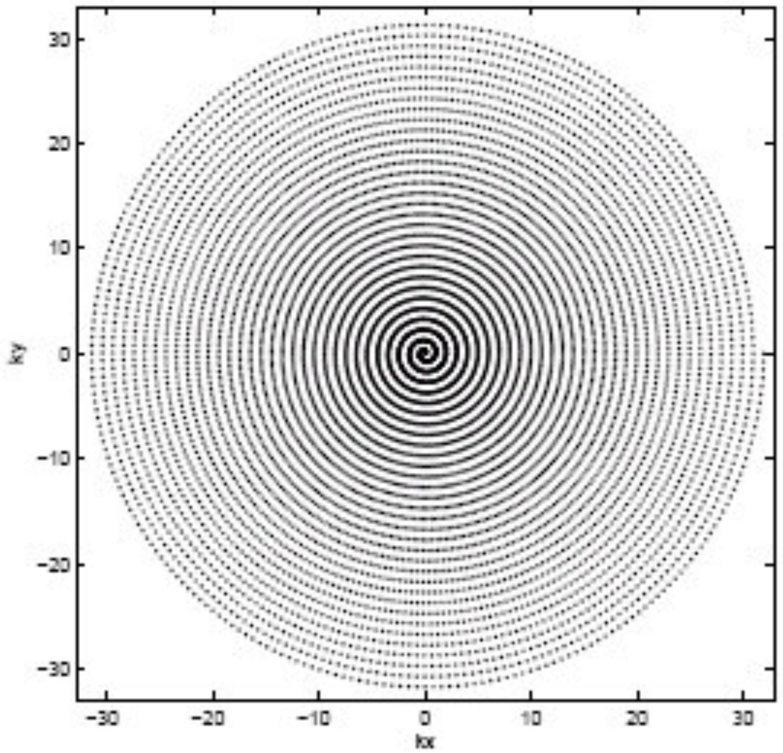
Field Inhomogeneity

Many Possible Trajectories



From MRiReco.jl, Julia MRI package

Image reconstruction



(a) k-space trajectory

$$s(t) = \iint \rho(x,y) e^{-2\pi i (k_1(t)x_1 + k_2(t)x_2)}\, dx_1 dx_2$$

$\Rightarrow \quad s(t) = F(k_1(t), k_2(t))$

$\Rightarrow \rho(x_1, x_2)$ can be recovered from

$$\rho(x_1, x_2) = \iint F(k_1, k_2) e^{2\pi i (k_1 x_1 + k_2 x_2)}\, dk_1 dk_2$$

1) Collect N samples : $F(k_1^q, k_2^q) \; = s(t_q)$

2) Compute $\rho(x_1, x_2)$

$$\rho(x_1, x_2) \approx \sum_q F(k_1^q, k_2^q) e^{2\pi i (k_1^q x_1 + k_2^q x_2)} w_q$$

Fourier Transform/Reconstruction

$$\rho(\mathbf{x}) = \int\int F(\mathbf{k}) e^{2\pi i \mathbf{k}\cdot\mathbf{x}} \, d\mathbf{k}$$

There are three distinct issues involved

- **Acquisition of data $F(\mathbf{k})$ at N points $\mathbf{k}_j$**
- **Selection of quadrature weights w_j**
- **A fast algorithm for computing the discrete approximation at a collection of N points $\mathbf{x}_l$.**

$$\rho(\mathbf{x}_l) \approx \sum_{j=1}^{N} F(\mathbf{k}_j) e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j} w_j$$

The Nonuniform FFT

allows such sums to be computed in $O(N \log N)$ time *with complete control of precision*. Dutt and Rokhlin (1993) provided the first complete analysis and introduced what are now called transforms of types 1,2 and 3.

$$F_n = \sum_{j=1}^{N} \rho_j \, e^{-inx_j} \quad n = -N/2, \ldots, N/2 \qquad \text{(Type 1)}$$

$$\text{(Type 2)} \qquad \rho_j = \sum_{n=-N/2}^{N/2-1} F_n \, e^{inx_j}, \quad j = 1, \ldots, N$$

$$F_n = \sum_{j=1}^{N} \rho_j \, e^{-ik_n x_j}, \quad n = 1, \ldots, N \qquad \text{(Type 3)}$$

Brief & *Incomplete* History

Fast Fourier Transforms for Nonequispaced data. A. Dutt and V. Rokhlin. SIAM J. Sci. Comput. 14, 1368 (1993). (Dutt, Yale Tech. Rpt 841, 1991).

On the Fast Fourier Transform of Functions with Singularities, G. Beylkin, Applied and Comput. Harmonic Analysis 2 (4) (1995) 363–381. → *USFFT*

Fast Fourier transforms for nonequispaced data: A tutorial, D. Potts, G. Steidl, and M. Tasche,, in Modern Sampling Theory, Birkhauser, Boston 2001, ch. 12, pp. 249–274. → *NFFT* (C, Matlab, Julia)

Nonuniform fast Fourier transforms using min- max interpolation, J. A. Fessler and B. P. Sutton, IEEE Trans. Signal Process., 51 (2003), pp. 560–574. → *NUFFT* (Matlab, Julia)

Non-equispaced fast Fourier transforms with applications to tomography. K. Fourmont. J. Fourier Anal. Appl. 9(5) 431-450 (2003).

Accelerating the nonuniform fast Fourier transform, L. Greengard, J.-Y. Lee,, SIAM Rev. 46 (2004) 443–454. → *NUFFT* (Fortran, Matlab)

A parallel non-uniform fast Fourier transform library based on an ``exponential of semicircle" kernel. A. H. Barnett, J. F. Magland, and L. af Klinteberg. SIAM J. Sci. Comput. 41(5), C479-C504 (2019). → *fiNUFFT* (C++, C, Fortran, MATLAB, Octave, Python, Julia)

Earlier/Concurrent work

Interpolation and Fourier transformation of fringe visibilities, A. R. Thompson and R. N. Bracewell,, Astronom. J., 79 (1974), pp. 11–24. (1)

A fast sinc function gridding algorithm for Fourier inversion in computer tomography, J. D. O'Sullivan,, IEEE Trans. Med. Imag., MI-4 (1985), 200–207. (1)

F*ast algorithm for spectral analysis of unevenly sampled data*. W. H. Press and G. B. Rybicki, Astrophys. J. 338 (1989), 227-280. (1)

Selection of a convolution function for Fourier inversion using gridding, J. I. Jackson, C. H. Meyer, D. G. Nishimura, and A. Macovski,, IEEE Trans. Med. Imag., 10 (1991), 473–478. (1)

Multilevel computations of integral transforms and particle interactions with oscillatory kernels, A. Brandt, Comput. Phys. Commun. 65 (1991), 24–38. (3)

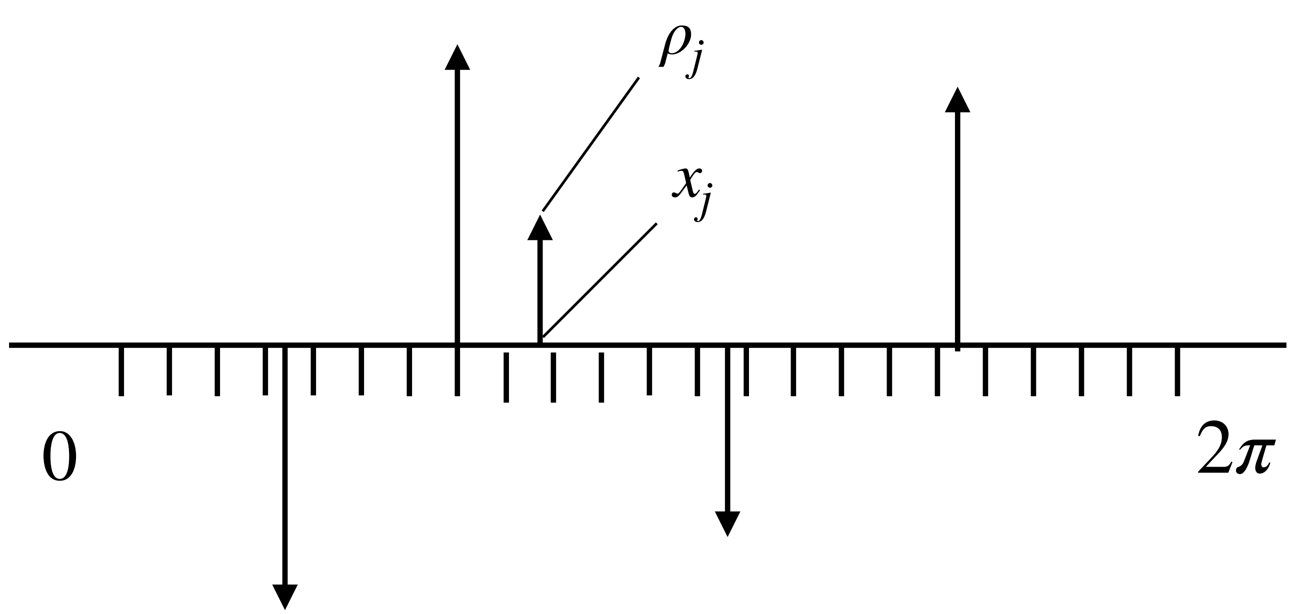
A Fast Algorithm for Chebyshev, Fourier, and Sine Interpolation onto an Irregular Grid, J. P. Boyd, J. Comput. Phys 103 (1992), 243-257. (2)

The type 1 transform

$$F_n = \sum_{j=1}^{N} \rho_j \, e^{-inx_j} \quad n = -N/2,\ldots,N/2, \;\; x_j \in [0,2\pi]\,.$$

Exact Fourier transform of the function

$$\rho(x) = \sum_{j=1}^{N} \rho_j \delta(x - x_j)$$

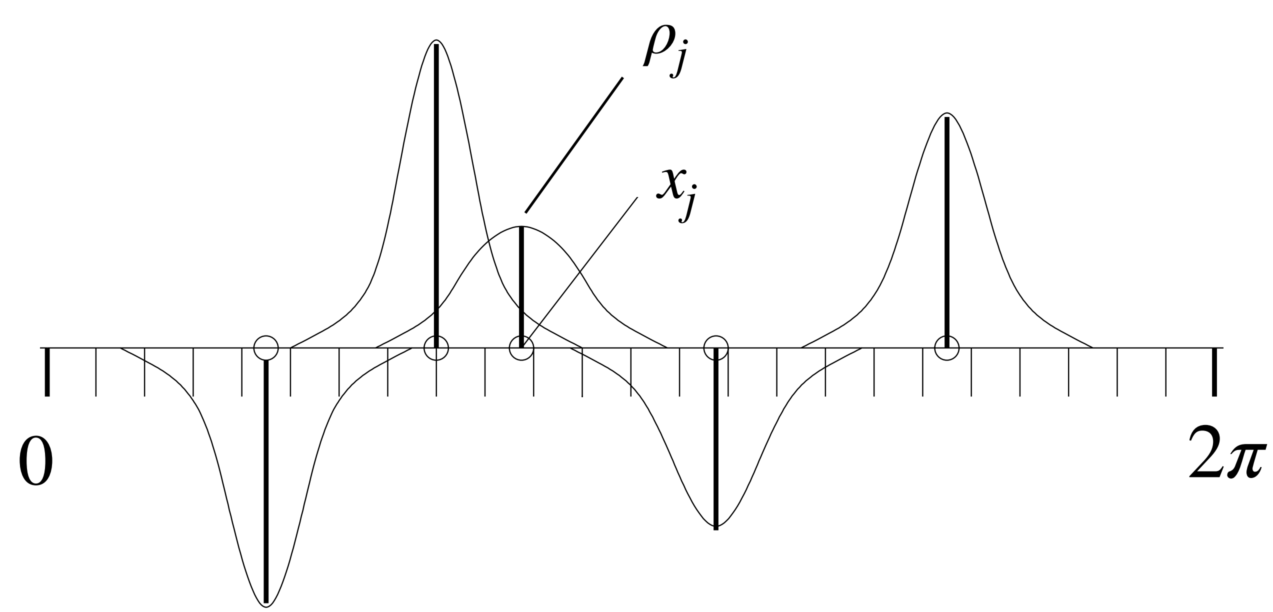


Step 1: **Convolve**

$$\rho(x) = \sum_{j=1}^{N} \rho_j \delta(x - x_j)$$

with a localized spreading function:

$$\rho_{sm}(x) = \rho(x) * g_{sm}(x)$$

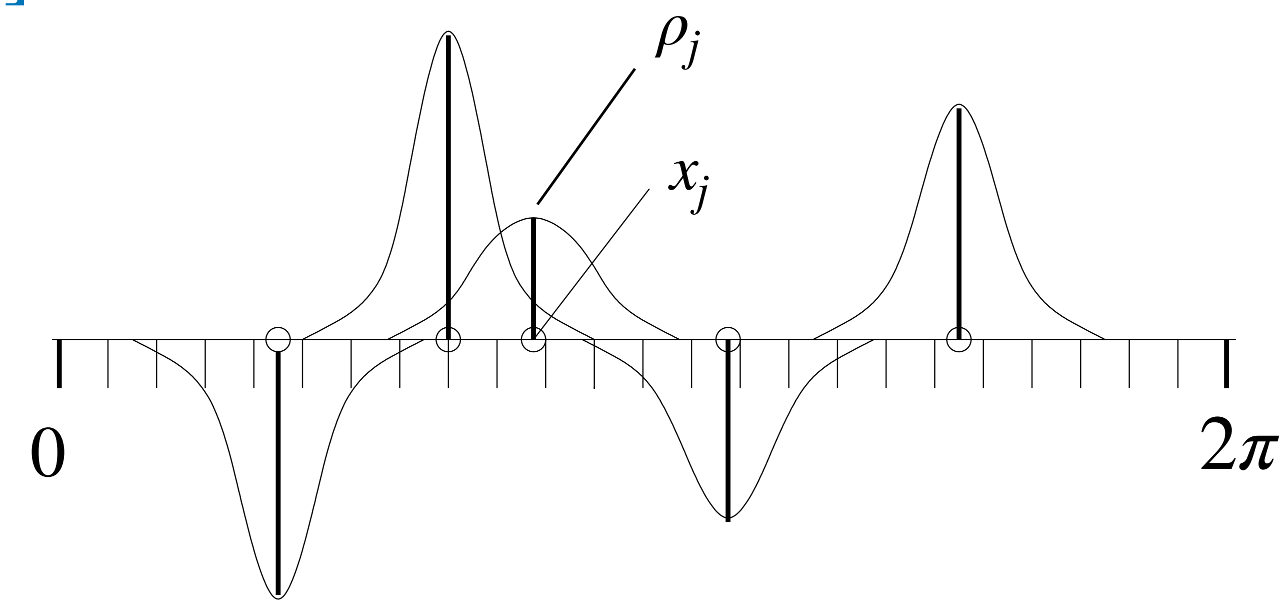


Sample on a grid with 2N points (2x oversampling)

Step 2: Compute FFT of size 2N for function

$$\rho_{sm}(x) = [\rho * g_{sm}](x)$$

sampled on $[0,2\pi]$



to obtain $$F_{sm}(n) = \int \rho_{sm}(x)\ e^{-2\pi i n x}\ dx$$

Theorem: $F_{sm}(n)$ is computed with spectral accuracy for $n = -N/2, \ldots, N/2$

Step 3: **Correct for the spreading step by deconvolution:**

$$F_n = F_{sm}(n)/G_{sm}(n)$$

where $G_{sm}(n) = \int g_{sm}(x)\ e^{-2\pi inx}\ dx$

This follows immediately from the convolution theorem

Using a Gaussian kernel for spreading, the variance can be chosen so that spreading to 24 points yields 12 digits of accuracy *for arbitrarily located points x_j.* (12 point spreading yields 6 digits.) *(D&R, 1993)*

Using KB or ES (fiNUFFT) kernel (Barnett et al.): 13 points -> 12 digits

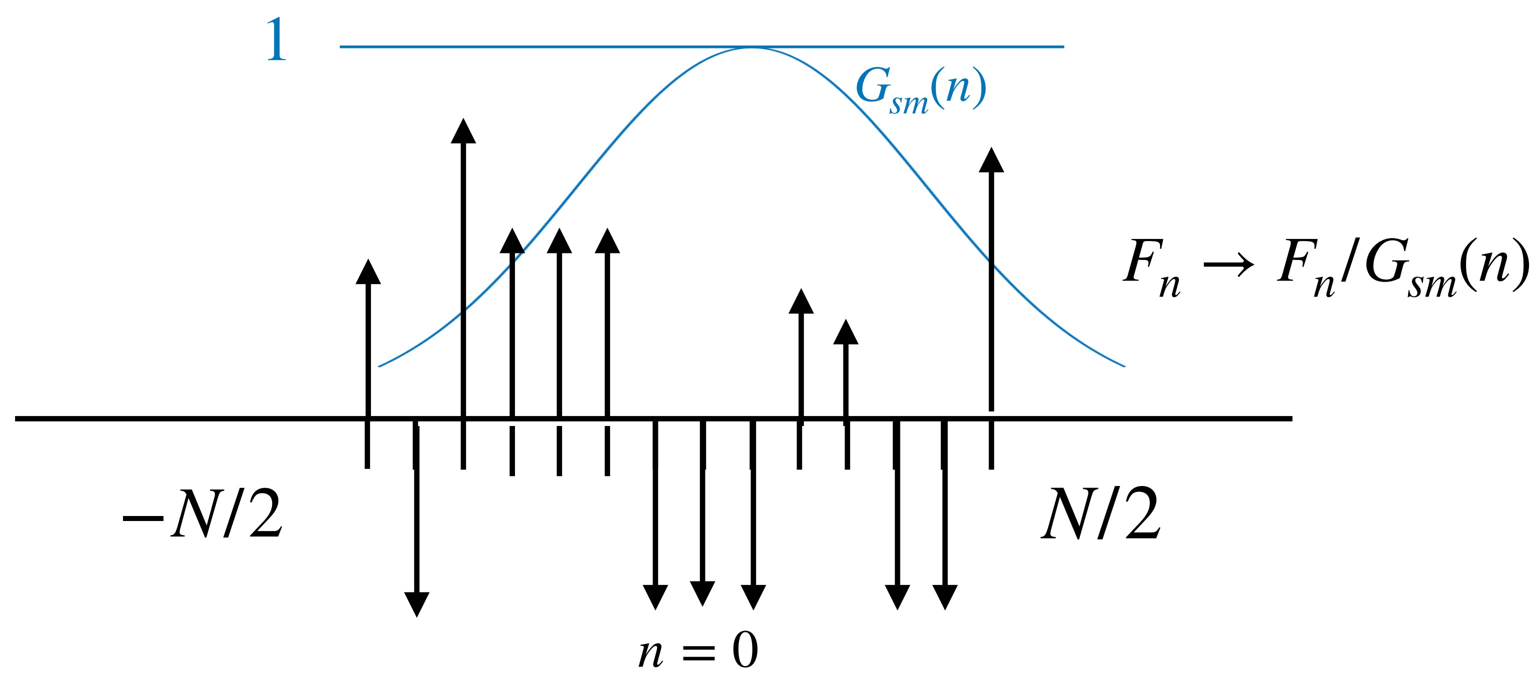
The type 2 transform

$$\rho_j = \sum_{n=-N/2}^{N/2-1} F_n \, e^{inx_j}, \quad j = 1, \ldots, N$$

cf. Boyd,...

Evaluation of Fourier series at arbitrary points

Same algorithm, but in reverse!



Step 1: Amplify the Fourier coefficients

$$F_{amp} = F_n / G_{sm}(n)$$

where $G_{sm}(n) = \int g_{sm}(x)\, e^{-2\pi inx}\, dx$ (again)

Step 2: Zero-pad F_{amp} to size 2N and Compute FFT

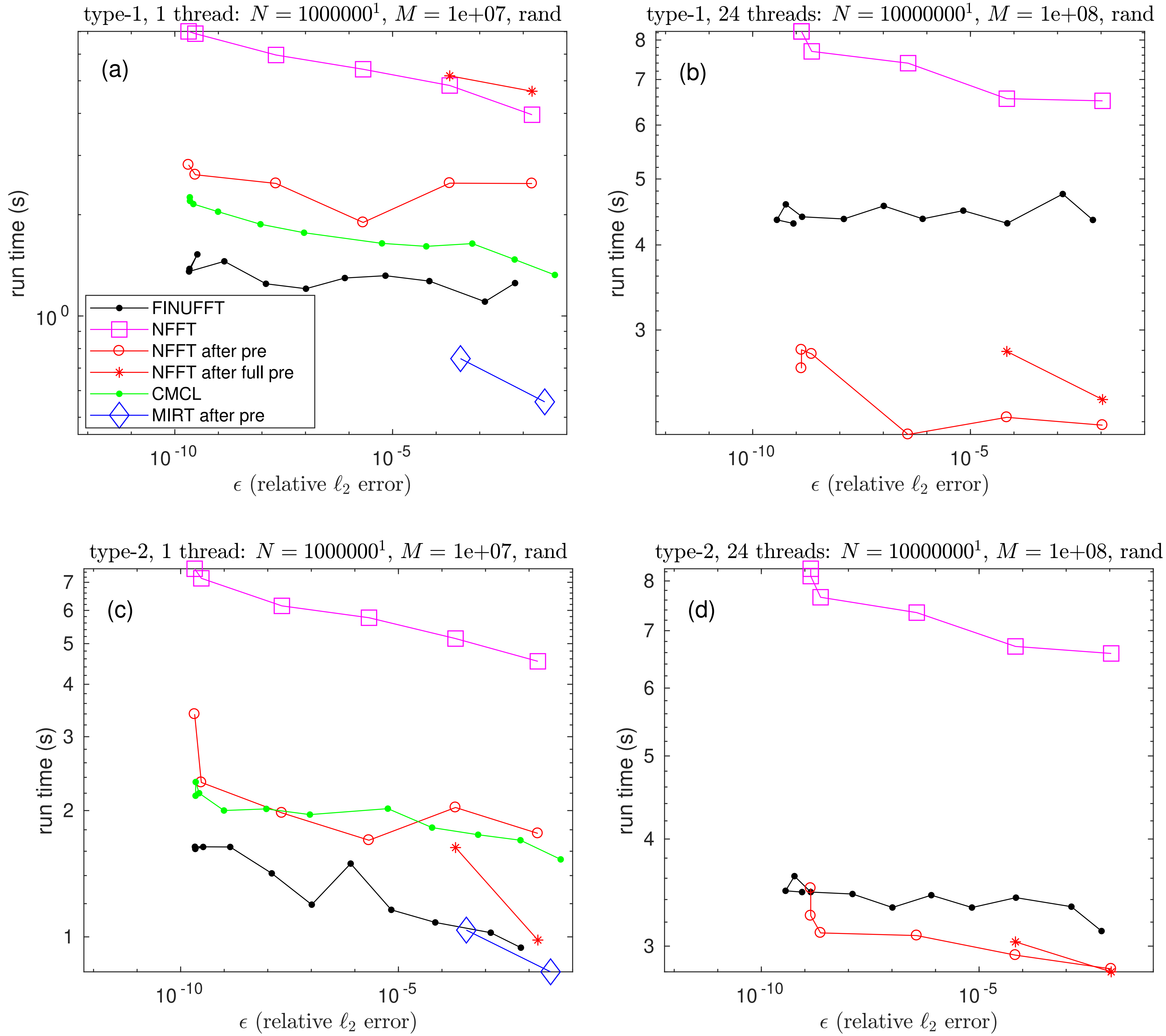
to yield ρ_{amp}

Step 3: Compute $\rho(x_j)$ at target points by

convolving ρ_{amp} with g_{sm} : $\rho(x_j) = [\rho_{amp} * g_{sm}](x_j)$

Cost: using contributions from nearest 24 points yields 12 digits of accuracy. (D&R, 1993)

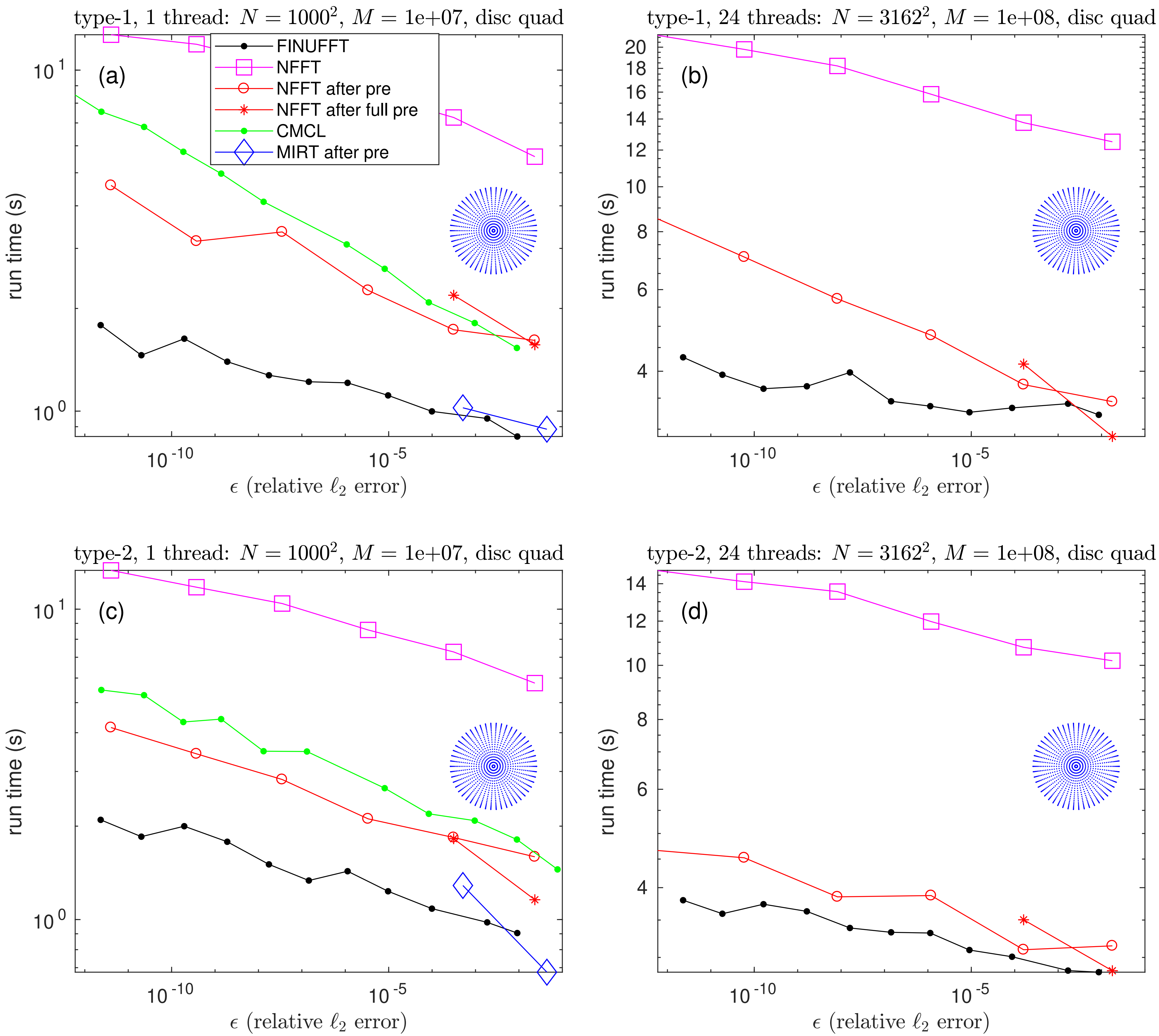
Using KB or ES (fiNUFFT) kernel (Barnett et al.): 13 points -> 12 digits



$M \to N$

$N \to M$

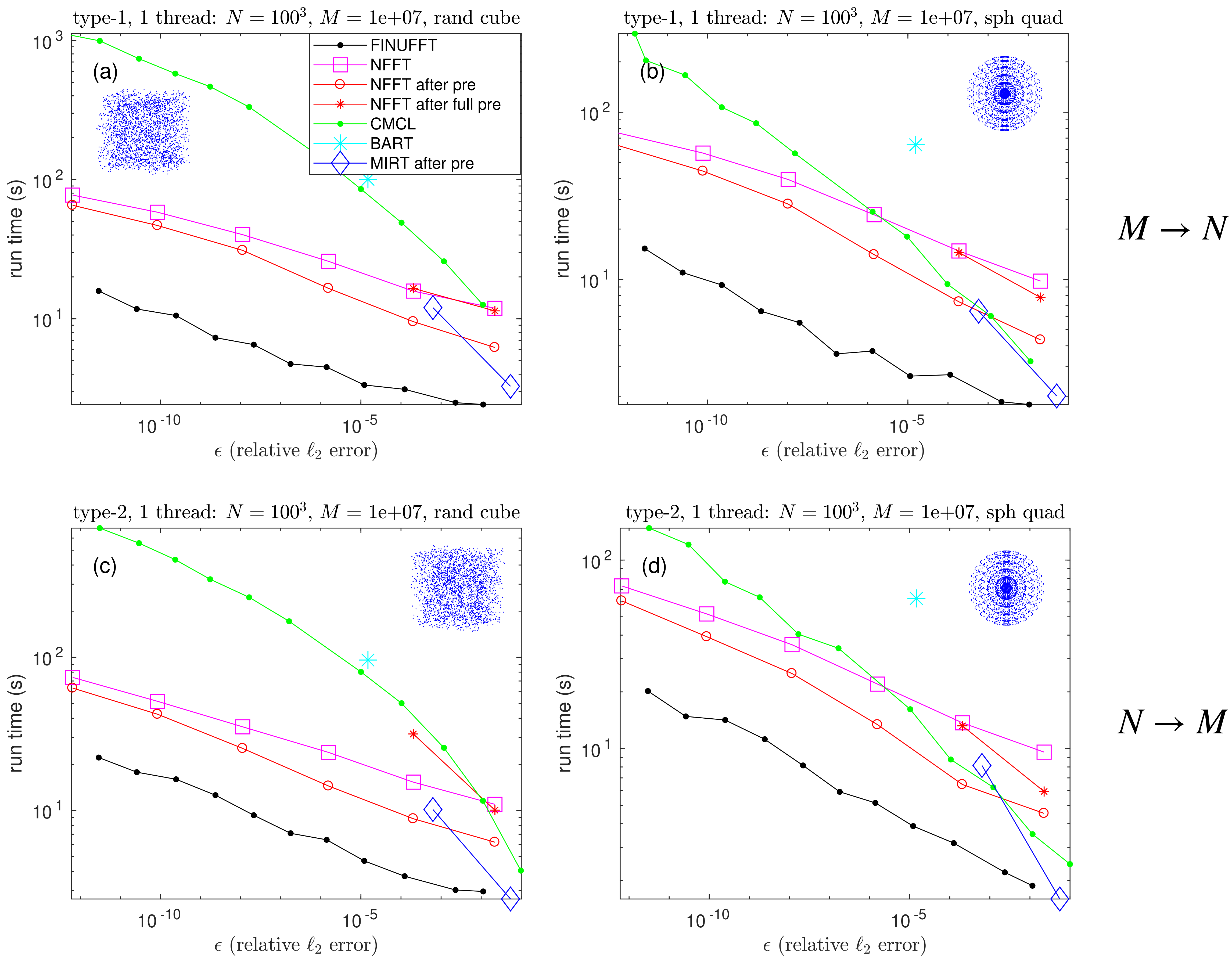
Barnett, Magland, af Klintberg (2019)



$M \to N$

$N \to M$

Barnett, Magland, af Klintberg (2019)



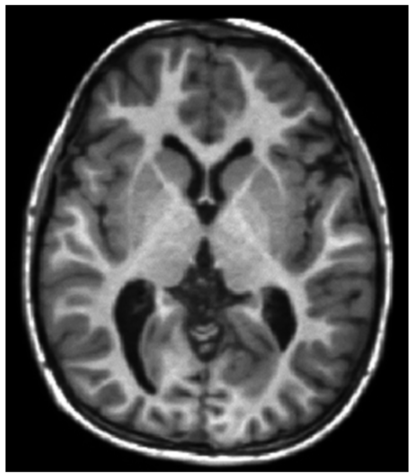
Barnett, Magland, af Klintberg (2019)

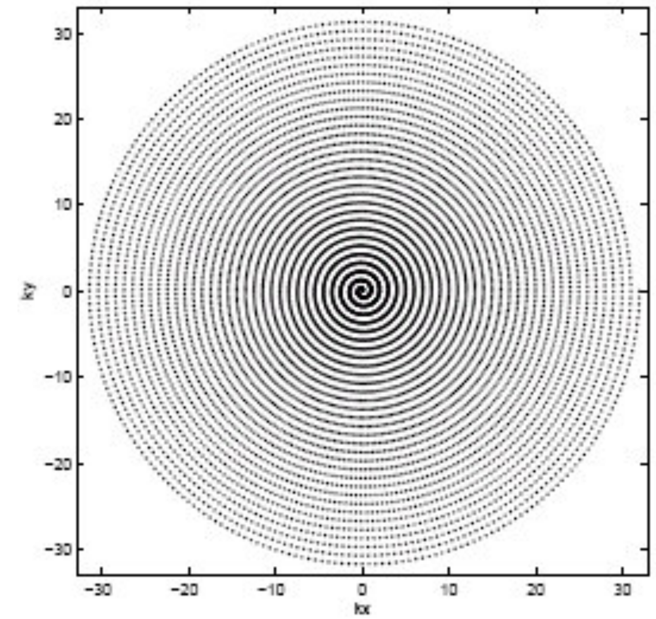
fiNUFFT

https://github.com/flatironinstitute/finufft

- multi-threaded, for multi-core shared-memory machines
- one, two, or three dimensional transforms
- typically achieves $10^6 - 10^8$ points/second/core
- written in C++, simple interfaces to (C, Fortran, MATLAB, octave, python, and Julia)
- OpenMP, uses FFTW
- released under Apache v. 2.

Back to MRI





Sample points $\mathbf{k}_j = (k^j{}_1, k^j{}_2)$

(Discrete) Signal Equation

$$F(\mathbf{k}_j) = \int\int \rho(x, y) e^{-2\pi i (k^j{}_1 x_1 + k^j{}_2 x_2)} \, dx_1 dx_2$$

$$= \mathscr{H} \rho$$ **(Continuous to discrete map)**

$$\rho \approx \mathscr{H}^+ F = \mathscr{H}^\dagger (\mathscr{H} \mathscr{H}^\dagger)^+ F$$ **Pseudoinverse**

$$\rho \approx \mathscr{H}^{+}F = \mathscr{H}^{\dagger}(\mathscr{H}\mathscr{H}^{\dagger})^{+}F$$

Pseudoinverse (minimum L^2 norm solution)

It is well-known that $(\mathscr{H}\mathscr{H}^{\dagger})_{m,n} = \mathrm{sinc}(\mathbf{k}_m - \mathbf{k}_n)$ **so,**

letting $\mathbf{a} = (a_1, \ldots, a_N) = (\mathscr{H}\mathscr{H}^{\dagger})^{+}F$ **:** (Solve time?)

$$\rho(\mathbf{x}_l) \approx \sum_{j=1}^{N} e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j} a_j$$

(Type 1 or Type 3) NUFFT

$$\rho \approx \mathscr{H}^{+}F = \mathscr{H}^{\dagger}(\mathscr{H}\mathscr{H}^{\dagger})^{+}F$$

Pseudoinverse (minimum L^2 norm solution)

It is well-known that $(\mathscr{H}\mathscr{H}^{\dagger})_{m,n} = \mathrm{sinc}(\mathbf{k}_m - \mathbf{k}_n)$ so,

letting $\mathbf{a} = (a_1, \ldots, a_N) = \underbrace{(\mathscr{H}\mathscr{H}^{\dagger})^{+}F}$:

$$\rho(\mathbf{x}_l) \approx \sum_{j=1}^{N} e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j}\, a_j$$

(Type 1 or Type 3) NUFFT

$$\rho(\mathbf{x}_l) \approx \mathscr{H}^{\dagger}DF = \sum_{j=1}^{N} e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j}\, w_j F(\mathbf{k}_j)$$

Computing the inverse Fourier transform by quadrature can be viewed as a diagonal approximation of the pseudo-inverse.

$$\rho \approx \mathscr{H}^{+}F = \mathscr{H}^{\dagger}(\mathscr{H}\mathscr{H}^{\dagger})^{+}F$$

Pseudoinverse (minimum L^2 norm solution)

It is well-known that $(\mathscr{H}\mathscr{H}^{\dagger})_{m,n} = \mathrm{sinc}(\mathbf{k}_m - \mathbf{k}_n)$ so,

letting $\mathbf{a} = (a_1, \ldots, a_N) = \underbrace{(\mathscr{H}\mathscr{H}^{\dagger})^{+}F}$:

$$\rho(\mathbf{x}_l) \approx \sum_{j=1}^{N} e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j} a_j$$

(Type 1 or Type 3) NUFFT

$$\rho(\mathbf{x}_l) \approx \mathscr{H}^{\dagger}DF = \sum_{j=1}^{N} e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j} \overbrace{w_j F(\mathbf{k}_j)}$$

Computing the inverse Fourier transform by quadrature can be viewed as a diagonal approximation of the pseudo-inverse.

$$\rho \approx \mathscr{H}^{+}F = \mathscr{H}^{\dagger}(\mathscr{H}\mathscr{H}^{\dagger})^{+}F$$

Pseudoinverse (minimum L^2 norm solution)

It is well-known that $(\mathscr{H}\mathscr{H}^{\dagger})_{m,n} = \mathrm{sinc}(\mathbf{k}_m - \mathbf{k}_n)$ so,

letting $\mathbf{a} = (a_1, \ldots, a_N) = \underbrace{(\mathscr{H}\mathscr{H}^{\dagger})^{+}F}$:

$$\rho(\mathbf{x}_l) \approx \sum_{j=1}^{N} e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j} a_j$$

(Type 1 or Type 3) NUFFT

$$\rho(\mathbf{x}_l) \approx \mathscr{H}^{\dagger}DF = \sum_{j=1}^{N} e^{2\pi i \mathbf{x}_l \cdot \mathbf{k}_j} \overbrace{w_j F(\mathbf{k}_j)}$$

"Optimal" weights can be determined by finding best diagonal approximation of pseudo-inverse $(\mathscr{H}\mathscr{H}^{\dagger})^{+}$ in Frobenius norm:
$min_W \| I - \mathscr{H}\mathscr{H}^{\dagger} W \|_F$

$$w_m = \frac{1}{\sum_n \mathrm{sinc}^2(\mathbf{k}_m - \mathbf{k}_n)}$$

<u>Theorem</u>: G-, Lee, Inati (2005), Choi, Munson (1998)

Fast Sinc Transform

$$G_l = \sum_{n=1}^{N} q_n \mathrm{sinc}(\mathbf{k}_n - \mathbf{k}_l)$$

$$H_l = \sum_{n=1}^{N} q_n \mathrm{sinc}^2(\mathbf{k}_n - \mathbf{k}_l)$$

Naive summation requires $O(N^2)$ work

Note that the weight formula $w_m = \dfrac{1}{\sum_n \mathrm{sinc}^2(\mathbf{k}_m - \mathbf{k}_n)} = \dfrac{1}{H_l}$

assuming all $q_n = 1$

Fast Sinc Transform (G, Lee, Inati, 2005)

$$G_l = \sum_{n=1}^{N} q_n \operatorname{sinc}(\mathbf{k}_n - \mathbf{k}_l) = G(\mathbf{k}_l),$$ where

$$G(\mathbf{v}) = \int_{\mathbb{R}^2} \operatorname{sinc}(\mathbf{v} - \mathbf{k}) P(\mathbf{k})\, d\mathbf{k}, \quad P(\mathbf{k}) = \sum_{n=1}^{N} q_n \delta(\mathbf{k} - \mathbf{k}_n)$$

From convolution theorem,

$$G(\mathbf{v}) = \int_{\mathbb{R}^2} \hat{G}(\mathbf{x}) e^{-2\pi i \mathbf{x}\cdot\mathbf{v}}\, d\mathbf{x}\,,$$ **where**

$$\hat{G}(\mathbf{x}) = \mathscr{H}^{-1}\operatorname{sinc}(\mathbf{k}) \cdot \mathscr{H}^{-1} P(\mathbf{k})$$

Fast Sinc Transform (G, Lee, Inati, 2005)

$$G_l = \sum_{n=1}^{N} q_n \text{sinc}(\mathbf{k}_n - \mathbf{k}_l) = G(\mathbf{k}_l), \text{ where}$$

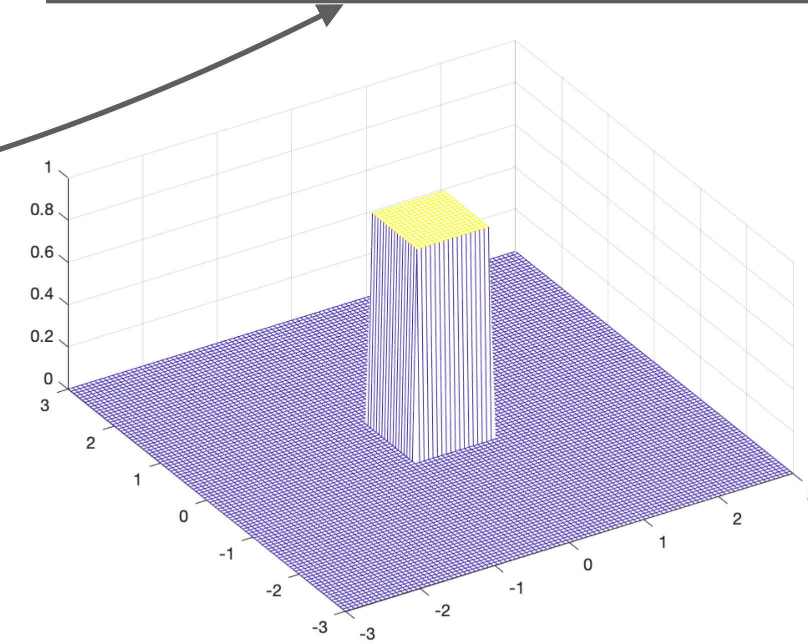
$$G(\mathbf{v}) = \int_{\mathbb{R}^2} \text{sinc}(\mathbf{v} - \mathbf{k}) P(\mathbf{k})\, d\mathbf{k}, \quad P(\mathbf{k}) = \sum_{n=1}^{N} q_n \delta(\mathbf{k} - \mathbf{k}_n)$$

From convolution theorem,

$$G(\mathbf{v}) = \int_{\mathbb{R}^2} \hat{G}(\mathbf{x}) e^{-2\pi i \mathbf{x}\cdot\mathbf{v}}\, d\mathbf{x}\,, \text{ where}$$

$$\boxed{\hat{P}(\mathbf{x}) = \mathscr{H}^{-1} P(\mathbf{k}) = \sum_{n=1}^{N} q_n e^{2\pi i \mathbf{x}\cdot\mathbf{k}_n}}$$

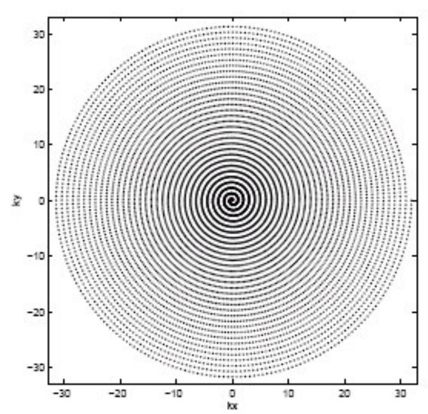
$$\hat{G}(\mathbf{x}) = \underbrace{\mathscr{H}^{-1}\text{sinc}(\mathbf{k})} \cdot \underbrace{\mathscr{H}^{-1} P(\mathbf{k})}$$



$$G(\mathbf{k}_n) = \int_{-1/2}^{1/2}\int_{-1/2}^{1/2} \hat{P}(\mathbf{x}) e^{-2\pi i \mathbf{x}\cdot\mathbf{k}_n}\, d\mathbf{x}$$

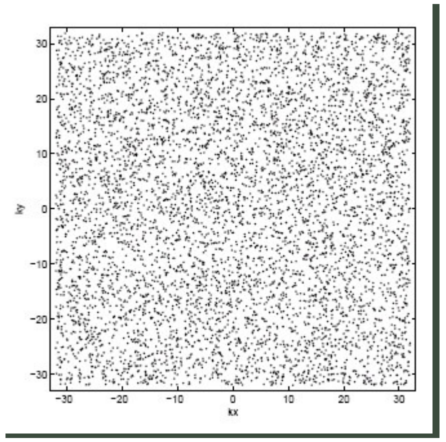
Computed using the type 3 NUFFT after quadrature

Fig. 1. Spiral Scan

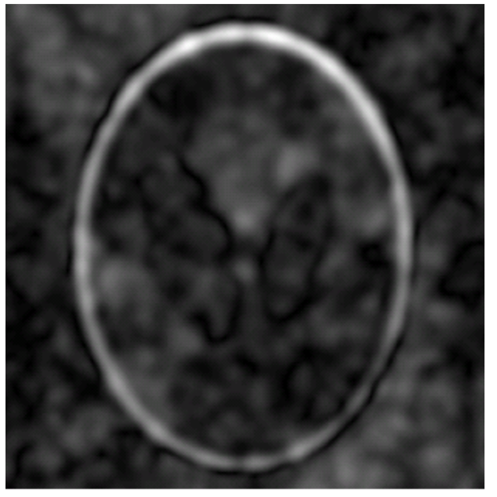


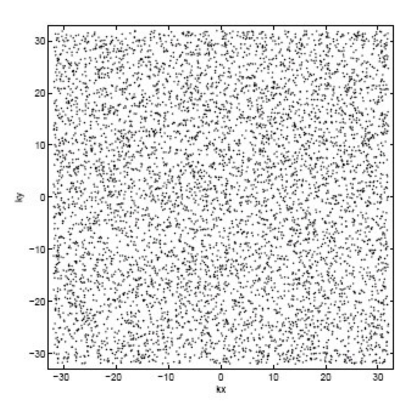
(a) k-space trajectory

(c) Original band limited object

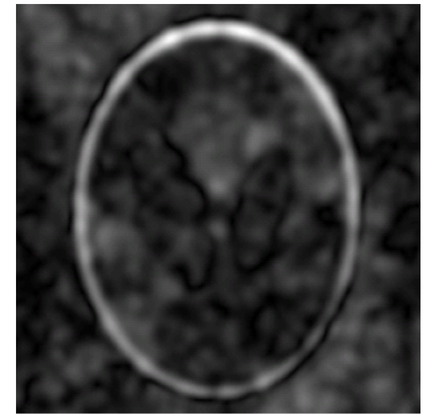


(c) Original band limited object

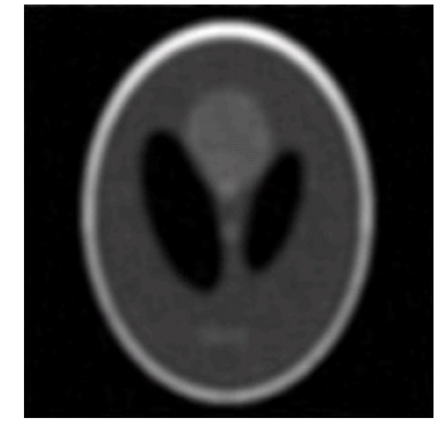




Original band-limited



Weights+NUFFT



Fast Pseudo-inverse

$$\rho \approx \mathscr{H}^{+}F = \mathscr{H}^{\dagger}(\mathscr{H}\mathscr{H}^{\dagger})^{+}F$$

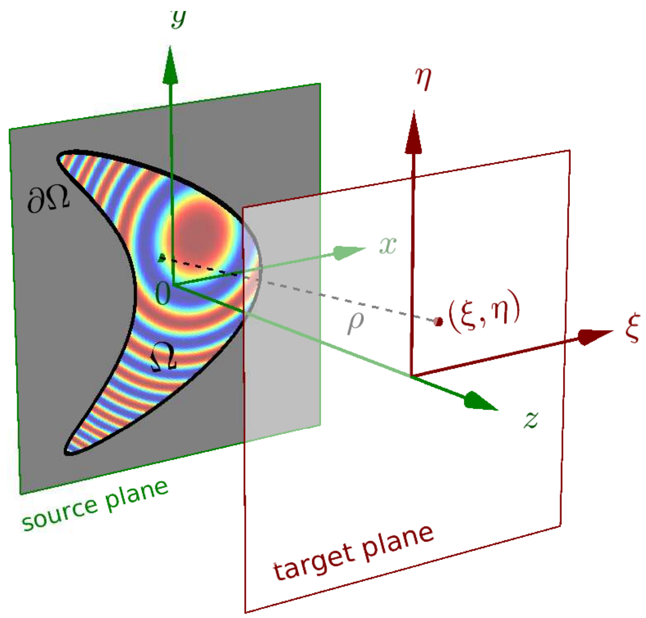
$$(\mathscr{H}\mathscr{H}^{\dagger})_{m,n} = \mathrm{sinc}(\mathbf{k}_m - \mathbf{k}_n)$$

Fast Pseudo-inverse construction:

(Inati, Lee, Fleysher, Fleysher, G—, 2006)

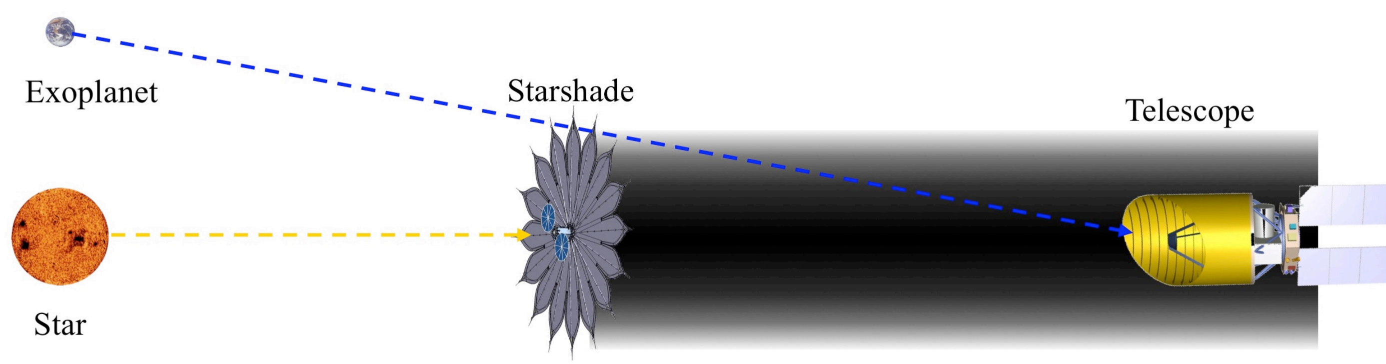
1) **Solve $(\mathscr{H}\mathscr{H}^{\dagger})^{+}F$ iteratively, using Fast Sinc Transform, with optimal weights as preconditioner**
2) **$O(N \log N)$ work and without need for further regularization**

Fresnel Diffraction/Starshades (Barnett, 2020)



$$u^{ap}(\xi, \eta) = \frac{1}{i\lambda z} \int\int_{\Omega} e^{\frac{i\pi}{\lambda z}[(\xi - x)^2 + (\eta - y)^2]}\, dx dy$$

Kirchhoff diffraction approximation to Maxwell equations



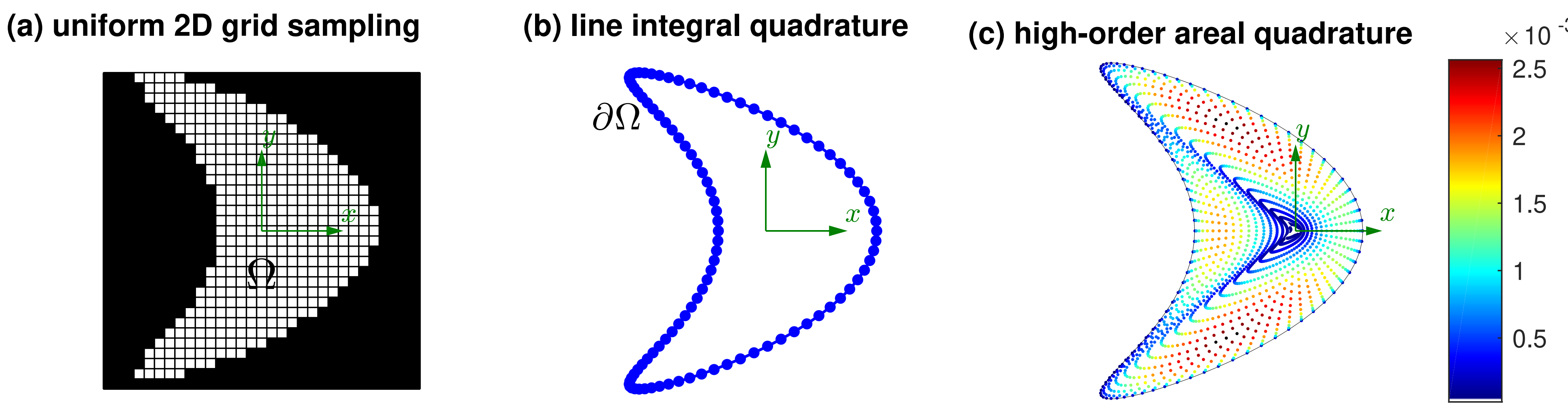
$$u^{oc}(\xi, \eta) = 1 - u^{ap}(\xi, \eta)$$

Design problem: Create *starshade* to block direct light from a star, allowing much dimmer exoplanets to be imaged.

From: The Search For Habitable Worlds: 1. The Viability of a Starshade Mission
M. C. Turnbull , T. Glassman , A. Roberge , W. Cash , C. Noecker , A. Lo , B. Mason , P. Oakley , & J. Bally (arXiv:1204.6063, 2012)

Efficient high-order accurate Fresnel diffraction via areal quadrature and the nonuniform FFT (Barnett, arXiv:2010.05978, 2020)

$$u^{ap}(\xi,\eta) = \frac{1}{i\lambda z}\int\!\!\int_{\Omega} e^{\frac{i\pi}{\lambda z}[(\xi-x)^2+(\eta-y)^2]}\,dxdy$$

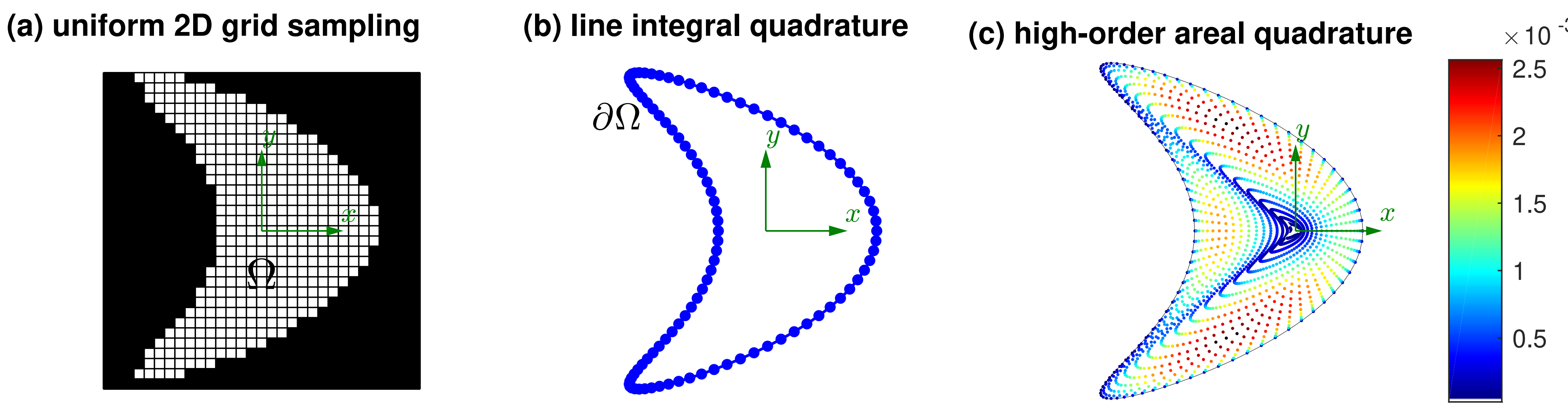


$$u^{ap}(\xi,\eta) \approx \frac{1}{i\lambda z}\sum_{j=1}^{N} e^{\frac{i\pi}{\lambda z}[(\xi-x_j)^2+(\eta-y_j)^2]}w_j$$

$$\approx \frac{e^{\frac{i\pi}{\lambda z}(\xi^2+\eta^2)}}{i\lambda z}\sum_{j=1}^{N} e^{\frac{-2\pi i}{\lambda z}(\xi x_j+\eta y_j)}\underbrace{e^{\frac{i\pi}{\lambda z}(x_j^2+y_j^2)}w_j}_{W_j}$$

Efficient high-order accurate Fresnel diffraction via areal quadrature and the nonuniform FFT (Barnett, arXiv:2010.05978, 2020)

$$u^{ap}(\xi,\eta) = \frac{1}{i\lambda z}\iint_{\Omega} e^{\frac{i\pi}{\lambda z}[(\xi - x)^2 + (\eta - y)^2]}\,dxdy$$



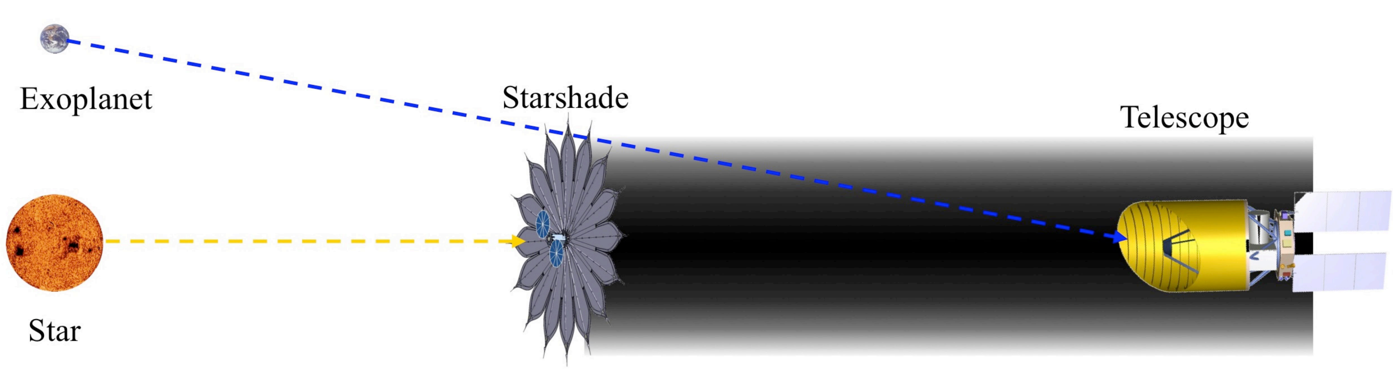
$$u^{ap}(\xi,\eta) \approx \frac{1}{i\lambda z}\sum_{j=1}^{N} e^{\frac{i\pi}{\lambda z}[(\xi - x_j)^2 + (\eta - y_j)^2]} w_j$$

$$\approx \frac{e^{\frac{i\pi}{\lambda z}(\xi^2+\eta^2)}}{i\lambda z}\sum_{j=1}^{N} e^{\frac{-2\pi i}{\lambda z}(\xi x_j + \eta y_j)}\,\underbrace{e^{\frac{i\pi}{\lambda z}(x_j^2+y_j^2)} w_j}_{W_j}$$

design	λ (m)	z (m)	$\mathfrak{f}$	m (petal)	total nodes	M (targets)	method	CPU time
NI2	5e-7	3.72e7	9.1	6000	n=192000	10^6, grid	BDWF	5361 s
				400	N=499200		NUFFT t1 (ε=10^{-8})	0.076 s
HG	5e-7	8e7	24	60	n=2048	10^6, grid	BDWF	80.5 s
				60	N=37440		NUFFT t1 (ε=10^{-8})	0.042 s

Table 2 Parameters and CPU times for the proposed NUFFT t1 and the BDWF edge-integral to complete the same diffraction tasks, for two starshades. See Fig. 5 for comparisons of their answers. The Fresnel number $\mathfrak{f}$ uses the maximum radius R in (2). N is the number of areal quadrature nodes, while n the number of boundary nodes.

From *Efficient high-order accurate Fresnel diffraction via areal quadrature and the nonuniform FFT* (Barnett, arXiv:2010.05978, 2020)



From: The Search For Habitable Worlds: 1. The Viability of a Starshade Mission
M. C. Turnbull , T. Glassman , A. Roberge , W. Cash , C. Noecker , A. Lo , B. Mason , P. Oakley , & J. Bally (arXiv:1204.6063, 2012)

Summary

- *The NUFFT is a powerful extension of the FFT*. It provides much greater flexibility in a variety of applications of Fourier analysis where uniform discretization in either the space/time or frequency domain is needed.
- There are several existing libraries which provide high performance implementations (fiNUFFT, NFFT, etc.)
- *Inverse problems* where the forward model involves the Fourier transform can be solved either via the inverse transform with suitable quadrature weights, or by inverting the (often ill-conditioned) forward problem.
- <u>*Caveat*</u>: Fourier methods (as a general rule) cannot cope with spatial adaptivity which introduces high frequency content (*Heisenberg*).
- There is a some confusion in the literature where the issues of sampling, selection of a quadrature rule, and discrete fast algorithms are conflated.
- Out of date but accessible intro: *Accelerating the nonuniform fast Fourier transform*, L. Greengard, J.-Y. Lee,, SIAM Rev. 46 (2004) 443–454.

fiNUFFT

https://github.com/flatironinstitute/finufft

NYU COURANT INSTITUTE OF MATHEMATICAL SCIENCES

FLATIRON INSTITUTE