PNFFT Users Manual

for version 1.0.3-alpha August 3, 2015

Michael Pippig
Chemnitz University of Technology
Department of Mathematics
09107 Chemnitz, Germany

Download Parallel Nonequispaced Fast Fourier Transform Software Library at
http://www.tu-chemnitz.de/~mpip/software

michael.pippig@mathematik.tu-chemnitz.de

Contents

1 Introduction

1.1 Introduction

- NFFT consists of 3 steps: ...
- describe the data decomposition
- make clear that 3d-decomposition is possible but 2d-decomposition is more natural
- Details on the algorithms can be found in [**?**].
- Details on the underlying parallel FFT can be found in [**?**].

local NFFT specific variables

- `ptrdiff_t local_M`
- `ptrdiff_t local_N[3]`, `ptrdiff_t local_N_start[3]`
- `double lower_border[3]`, `double upper_border[3]`

global NFFT specific variables

- `ptrdiff_t N[3]`
- `ptrdiff_t n[3]`
- `int m`
- `double x_max[3]`

2d data decomposition with non-transposed Fourier coefficients

$$\hat{N}_0/P_0 \times \hat{N}_1/P_1 \times \hat{N}_2 \overset{\mathsf{NFFT}}{\rightarrow} C_0/P_0 \times C_1/P_1 \times C_2$$

2d data decomposition with transposed Fourier coefficients

$$\hat{N}_1/P_0 \times \hat{N}_2/P_1 \times \hat{N}_0 \overset{\mathsf{NFFT}}{\rightarrow} C_0/P_0 \times C_1/P_1 \times C_2$$

3d data decomposition with non-transposed Fourier coefficients

$$\hat{N}_0/P_0 \times \hat{N}_1/P_1 \times \hat{N}_2/P_2 \overset{\mathsf{NFFT}}{\rightarrow} C_0/P_0 \times C_1/P_1 \times C_2/P_2$$

3d data decomposition with transposed Fourier coefficients with $P_2 = Q_0 Q_1$

$$\hat{N}_1/(P_0 Q_0) \times \hat{N}_2/(P_1 Q_1) \times \hat{N}_0 \overset{\mathsf{NFFT}}{\rightarrow} C_0/P_0 \times C_1/P_1 \times C_2/P_2$$

1.1.1 Parallel NFFT Workflow

- create a simple test program and describe it here
- get block distribution of Fourier coefficients and nodes
- call PNFFT planner
- init Fourier coefficients and nodes
- precomputations that depend on the nodes
- execute PNFFT plan
- read results
- finalize PNFFT

1.2 Special Features

1.2.1 Transposed Fourier coefficients

A parallel transpose FFT algorithm typically ends up with a transposed order of the output array. Start with

Similar to PFFT, our parallel NFFT supports an optimization flag that disables the backward transpositions. Therefore, one must work on a transposed array of Fourier coefficients.

1.2.2 Truncated Torus

PNFFT support the special case, where the nodes $\boldsymbol{x}_j$ fulfill the restriction $\boldsymbol{x}_j \in \left[-\frac{C_0}{2}, \frac{C_0}{2}\right]$

Copy from PNFFT paper:

In addition, we pay special attention to the case where all the nonequispaced nodes $\boldsymbol{x}_j$ are contained in a special subset of the torus $\mathbb{T}^3$. For $\boldsymbol{C} = (C_0, C_1, C_2)^\top \in \mathbb{R}^3$ with $0 < C_0, C_1, C_2 \leq 1$ we define the truncated torus $\mathbb{T}^3_{\boldsymbol{C}} := [-\frac{C_0}{2}, \frac{C_0}{2}) \times [-\frac{C_1}{2}, \frac{C_1}{2}) \times [-\frac{C_2}{2}, \frac{C_2}{2})$. For the parallel NFFT we assume $\boldsymbol{x}_j \in \mathbb{T}^3_{\boldsymbol{C}}$ for every $j = 1, \ldots, M$. Obviously, for $C_0 = C_1 = C_2 = 1$ this corresponds to the serial NFFT, where the nodes $\boldsymbol{x}_j$ are contained in the whole three-dimensional torus $\mathbb{T}^3$. This slight generalization is necessary in order to assure a load balanced distribution of nodes $\boldsymbol{x}_j$ whenever the nodes are concentrated in the center of the box.

1.3 Download and Install

- download, configure, make
- Advice for developers: install new version of autotools (add the script here)

2 Fortran Interface

- uses Fortran 2003 iso C bindings
- all function names are the same as in C interface with one exception: `get_n` becomes `get_nos`, since Fortran is not case sensitive and we need to distinguish between `get_N` and `get_n`
- all C-pointer must be converted to Fortran-pointers but freed as C-pointers

The order of array dimensions flip, since C uses a row-major memory layout, whereas Fortran uses column-major memory order.

2d data decomposition with non-transposed Fourier coefficients

$$\hat{N}_2 \times \hat{N}_1/P_1 \times \hat{N}_0/P_0 \overset{\mathsf{NFFT}}{\to} C_2 \times C_1/P_1 \times C_0/P_0$$

2d data decomposition with transposed Fourier coefficients

$$\hat{N}_0 \times \hat{N}_2/P_1 \times \hat{N}_1/P_0 \overset{\mathsf{NFFT}}{\to} C_2 \times C_1/P_1 \times C_0/P_0$$

3d data decomposition with non-transposed Fourier coefficients

$$\hat{N}_2/P_2 \times \hat{N}_1/P_1 \times \hat{N}_0/P_0 \overset{\mathsf{NFFT}}{\to} C_2/P_2 \times C_1/P_1 \times C_0/P_0$$

3d data decomposition with transposed Fourier coefficients with $P_2 = Q_0 Q_1$

$$\hat{N}_0 \times \hat{N}_2/(P_1 Q_1) \times \hat{N}_1/(P_0 Q_0) \overset{\mathsf{NFFT}}{\to} C_2/P_2 \times C_1/P_1 \times C_0/P_0$$

How to Deal with FFT Shifts in Parallel

PFFT calculates

$$h_l = \sum_{k=0}^{N-1} \hat{h}_k \mathrm{e}^{-2\pi \mathrm{i} k l/n}, \quad l = 0, \ldots, L$$

with $n \geq N$ and $n \geq L$. Step 2 of our PNFFT algorithm requires the shifted index sets

$$g_l = \sum_{k=-N/2}^{N/2-1} \hat{g}_k \mathrm{e}^{-2\pi \mathrm{i} k l/n}, \quad l = -L/2, \ldots, L/2-1$$

A common technique to deal with this problem is to call FFT shift, i.e., set

$$\hat{h}_k = \begin{cases} g_k, & k = 0, \ldots, N/2-1 \\ g_{k+N}, & k = N/2, \ldots, N-1 \end{cases}$$

$$g_l = \begin{cases} h_{l+N}, & l = -N/2, \ldots, -1 \\ h_l, & l = 0, \ldots, N/2-1 \end{cases}$$

However, because of the parallel data decomposition, this involves explicit data communication. Instead we apply the FFT shifts in frequency domain. Use the translation property

$$\hat{h}_k = \hat{g}_k \mathrm{e}^{+2\pi \mathrm{i} k L/2/n}$$

and the modulation property

$$h_{l-N/2} = \mathrm{e}^{-2\pi \mathrm{i} l N/2/n}$$

of the discrete Fourier transform.

User can choose to shift the input (`PNFFT_SHIFT_INPUT`) and/or to shift the output (`PNFFT_SHIFT_OUTPUT`)

One-dimensional example

$$\begin{aligned}
f(x) &= \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \hat{f}_k \mathrm{e}^{-2\pi \mathrm{i} k x} \\
&\approx \sum_{l=-\frac{n}{2}}^{\frac{n}{2}-1} \left(\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \frac{\hat{f}_k}{\varphi_k} \mathrm{e}^{-2\pi \mathrm{i} k \frac{l}{n}} \right) \varphi\left(x - \frac{l}{n}\right) \\
&= \sum_{l=-\frac{n}{2}+\frac{n_o}{2}}^{\frac{n}{2}+\frac{n_o}{2}-1} \left(\sum_{k=0}^{N-1} \frac{\hat{f}_{(k-\frac{N}{2})}}{\varphi_{(k-\frac{N}{2})}} \mathrm{e}^{-2\pi \mathrm{i} (k-\frac{N}{2}) \frac{(l-\frac{n_o}{2})}{n}} \right) \varphi\left(\left(x + \frac{n_o}{2n}\right) - \frac{l}{n}\right) \\
&= \sum_{l=-n/2+n_o/2}^{n/2+n_o/2-1} \left[\mathrm{e}^{+\pi \mathrm{i} N \frac{l}{n}} \sum_{k=0}^{N-1} \left(\frac{\hat{f}_{(k-\frac{N}{2})}}{\hat{\varphi}_{(k-\frac{N}{2})}} \mathrm{e}^{+\pi \mathrm{i} (k-\frac{N}{2}) \frac{n_o}{n}} \right) \mathrm{e}^{-2\pi \mathrm{i} k \frac{l}{n}} \right] \varphi\left(x - \frac{l - \frac{n_o}{2}}{n}\right)
\end{aligned}$$

$$\begin{aligned}
\mathrm{lo} &= \frac{\mathrm{local_no_start} - \mathrm{no}/2}{\mathrm{n}} \\
\mathrm{up} &= \frac{\mathrm{local_no_start} + \mathrm{local_no} - \mathrm{no}/2}{\mathrm{n}} = \mathrm{lo} + \frac{\mathrm{local_no}}{\mathrm{n}}
\end{aligned}$$

Bibliography