Technische Universität Chemnitz
Franziska Nestler and Michael Pippig
Jülich Research Center
CECAM Summer School

Programming with Nonequispaced FFT

Lab 2 **Parallel FFT and NFFT Hands On**

Introduction:
The following steps must be done, before you can start with the exercises:

1. Login to Judge via `ssh userid@judge.fz-juelich.de`
2. Copy the tests from `/homea/hpclab/train006/fourier_lab2` to your home directory via `cp -r /homea/hpclab/train006/fourier_lab2 $HOME`
3. Go into the test directory and build the tests via `cd ~/fourier_lab2 && make`
4. Execute `msub -I -l nodes=1:ppn=8,walltime=00:10:00` in order to allocate 8 processes in interactive mode.

Hint: A pdf version of this worksheet and the solutions are available at http://www.tu-chemnitz.de/~mpip/lehre.php.

Exercise 1 (Testing PFFT):
Lookup and open the source file `pfft_check.c` found in `~/fourier_lab2/`. Skim through the `main` routine. Try to understand what it does. Then, run the actual executable `pfft_check.x` found in `~/fourier_lab2/build` with 8 processes, i.e.,
`mpiexec -np 8 ~/fourier_lab2/build/pfft_check.x`

Exercise 2 (Testing PNFFT):
Lookup and open the source file `pnfft_test.c` found in `~/fourier_lab2/`. Skim through the `main` routine. Try to understand what it does. Then, run the actual executable `pnfft_test.x` found in `~/fourier_lab2/build` with 8 processes, i.e.,
`mpiexec -np 8 ~/fourier_lab2/build/pnfft_test.x`

Where is the difference to Exercise 1?

Exercise 3 (Accuracy check for PNFFT):
Since the approach from exercise 2 did not yield a check for the PNFFT, we try something different.

Lookup and open the source file `pnfft_check.c` found in `~/fourier_lab2/`. First, skim through the subroutine `pnfft_perform_guru`. Try to understand what it does. Then, have a look at the two calls of this subroutine in the `main` routine. What is the idea behind this error check? Then, run the actual executable `pnfft_check.x` found in `~/fourier_lab2/build` with 8 processes, i.e.,
`mpiexec -np 8 ~/fourier_lab2/build/pnfft_check.x`

Explore the behavior of the PNFFT error for different parameters. Therefore, you can start the executable `pnfft_check.x` again with some additional command line arguments:

- Change the real space cutoff with the argument `-pnfft_m` followed by one number between 1 and 8.
- Change the number of Fourier coefficients with the argument `-pnfft_N` followed by 3 even numbers.
- Change the size of the FFT grid with the argument `-pnfft_n` followed by three even numbers.
- Change the process mesh with the argument `-pnfft_np` followed by the three dimensions of the mesh.

For example you can call

```
mpiexec -np 2 ~/fourier_lab2/build/pnfft_check.x -pnfft_m 3 -pnfft_N 8 8 8 -pnfft_np 1 2 1
```

If the check fails, think about the choice of parameters.

Exercise 4 (Advanced: Fix the check from Exercise 2):
In exercise 2 we found out, that the subroutine `init_random_nodes` in `pnfft_test.c` was used to initialize random nodes to calculate the NFFT.

Lookup and open the source file `pnfft_test_adv.c` found in `~/fourier_lab2/`. There is little difference to `pnfft_test.c`. Now, subroutine `init_equispaced_nodes` is used to initialize the nodes, but the body of this subroutine is missing. Add the missing lines to complete the check. Do not forget to execute `make` after every change in the source file. Then, run the actual executable `pnfft_test_adv.x` found in `~/fourier_lab2/build` with 8 processes, i.e.,

```
mpiexec -np 8 ~/fourier_lab2/build/pnfft_test_adv.x
```

Hint 1: We have to choose the "nonequispaced" nodes equal to the grid points of an equispaced FFT. But how can we compute the needed parameters from the inputs of `init_equispaced_nodes`?

Hint 2: PNFFT uses a parallel data decomposition of the cube $\left[\frac{1}{2}, \frac{1}{2}\right)^3$ into equal blocks. Distribute a mesh of appropriate size in the same way and compute the corresponding nodes within $\left[\frac{1}{2}, \frac{1}{2}\right)^3$.