cufft

http://docs.nvidia.com/cuda/cufft/index.html

- Algorithme de complexité O(n log(n))
- Optimisé pour des tailles $2^a3^b5^c7^d$ mais plus le facteur est petit mieux c'est (i.e. puissances de 2).
- Transformations :
    - Real to complex R2C
    - Complex to real C2R
    - Complex to complex C2C
- 1D, 2D, 3D
- *In-place* et *out-of-place*
- Compatibilités avec FFTW

Plan d’exécution

- Create a plan !
  - Basic : Use cufftPlan1D / cufftPlan2D / cufftPlan3D with a cufftHandle as argument
  - Or cufftCreate(&plan) then cufftMakeplan
- Different identifiers for simple and double precision
  - R : real simple
  - D : real double
  - C : complex simple
  - Z : complex double
- Call the function
  - cufftExecC2C() / cufftExecZ2Z()
  - cufftExecR2C() / cufftExecD2Z()
  - cufftExecC2C() / cufftExecZ2D()
- Free the plan with cufftDestroy()

Normalisation

- Don’t forget to normalize after the direct transform

Thread safety

- Use different *plans* for different *host threads* and datas must be disjoined

Gestion d’erreurs

- Every function return a cufftResult

Concurrency

- Can be streamed via cufftSetStream(cufftHandle plan, cudaStream_t stream)

Compilation et utilisation

- Inclure le header <cufft.h>
- Chemin du header : -I/usr/local/cuda/inc généralement
- Inclure la librairie avec -L/path/to/library -lcufft (généralement /usr/local/cuda/lib)

Memory layout

- Real to complex (R2C) transforms and back : uses hermitian symmetry : only half the complex plan is represented (the leading dimension it the one cutted)

TP

Exercice 1 : 1D

- Generate a signal, combination of two frequencies (5 and 10).
- Copy the data to device memory
- Perform a forward and inverse in-place fourier transform (don’t forget to rescale)
- Print the result to check

Exercice 2 : 2D

- Generate a 2D field sin(x) * cos(y) over a box of size (2*pi)^2
- Copy it to device memory
- Perform a forward 2D fourier
- Computes its Laplacian
- Perform an inverse fourier transform