Introduction to Languages for Scientific Computing

Prof. **Paolo Bientinesi**

pauldj@aices.rwth-aachen.de

Part 1: 2d-FFT

$$Y := P_{m,\frac{m}{2}}(I_2 \otimes F_{\frac{m}{2}})D_{2,\frac{m}{2}} \left(\begin{array}{c|c} I_{\frac{m}{2}} & I_{\frac{m}{2}} \\ \hline I_{\frac{m}{2}} & -I_{\frac{m}{2}} \end{array}\right) X \left(\begin{array}{c|c} I_{\frac{n}{2}} & I_{\frac{n}{2}} \\ \hline I_{\frac{n}{2}} & -I_{\frac{n}{2}} \end{array}\right) D_{2,\frac{n}{2}}(I_2 \otimes F_{\frac{n}{2}})P^T_{n,\frac{n}{2}}$$

- Write a Matlab function `my2dFFT( X )`.
  The input argument is a matrix `X` of size `m`×`n`.
  The output is the matrix `Y`, containing `X`'s 2d FFT.
- The objective is **elegant** and **smart** code!

Rules:

- `Y` must be computed by the recursive formula above.
- Assume that both `m` and `n` are even.
- No matrices of size `m`×`n` are allowed. Only exception: Y.
- The command `kron` cannot be used.
- If needed, write the functions for `D`, `P` and `F` yourself; include them in the same file as `my2dFFT`.
- For recursive calls to a 2d FFT, use Matlab's `fft2`.

Hints

- Math first, programming later.
- No need to further decompose $F_{\frac{m}{2}}$ or $F_{\frac{n}{2}}$.
- Work your way from X towards the boundaries, until you reach

$$P_{m,\frac{m}{2}}(I_2 \otimes F_{m/2})\tilde{X}(I_2 \otimes F_{n/2})P^T_{n,\frac{n}{2}}.$$

- Use math, NOT brute force.
- ...make sure your computation is correct.

Part 2: Image compression

- Write a Matlab function `DROP`.
  Inputs: a 3-dimensional array `Img` (an image), and a real number `k` $\in [0, 1]$.
  Output: a "compressed" image.
  Rationale: `DROP` discards `k`*100% of `Img`'s spectrum.

- For each layer of `Img`, `DROP` implements these steps.

  1) compute the 2D FFT;

  2) shift the frequencies so that the lowest ones are in the middle of the spectrum;

  3) drop all the frequencies outside a disk whose radius is identified by `k`.

     If `k == 0`, no frequency is dropped; if `k == 1`, no frequency is kept;

  4) shift the frequencies back;

  5) compute the inverse 2D FFT and return the so-obtained image.

Experiment with different images and different values of `k`.

The objective is **clear**, **readable** code.

Submission rules

- Individual assignment.
- Prepare two files: `my2dFFT.m` and `DROP.m`.
  Put your name in each of them.
- Archive the files as `your_name.zip` or `your_name.tgz` and submit.
- Submission by email to `pauldj@aices.rwth-aachen.de`
- Email's subject: ‘‘`LSC-15 challenge3 <your last name>`’’
- **Deadline: Sunday, December 6, midnight.**