Parallel processing toolbox in Matlab

Xing Chen
x.chen@nin.knaw.nl

- Bumping up against limits in computational power?
- Computing speed is the limiting factor in your analyses?
- Consider upgrading equipment and using parallel processing techniques

- Initial investment of time and energy to learn how to use Parallel Processing Toolbox convert code
- Example from my work:
- Process data from 20 data files, daily, over the course of 10 days.
- 20*10 = 200 data files
- Each file takes 10 min
- 200*10 = 2000 min = 34 hours, i.e. more than a day

- Computer with a dual quad core: 2*4 = 8 cores
- Job can be divided between 8 workers
- 34/8 = 4 hours

- A saving of 30 hours! I can have the results ready by today, instead of tomorrow

How does it work?

- It’s not magic

- Good news: basic implementation only involves two simple functions

Lining up at the post office

- Only one counter open ☹

- More counters open ☺

Random lady at post office

- She pays a bill and

gets a receipt

- I post a letter

- Independent processes

Separate counters

- Parallel transactions

My mum in front of me

- I need to enclose the receipt from her transaction in my letter

- Processing is sensitive to the task order

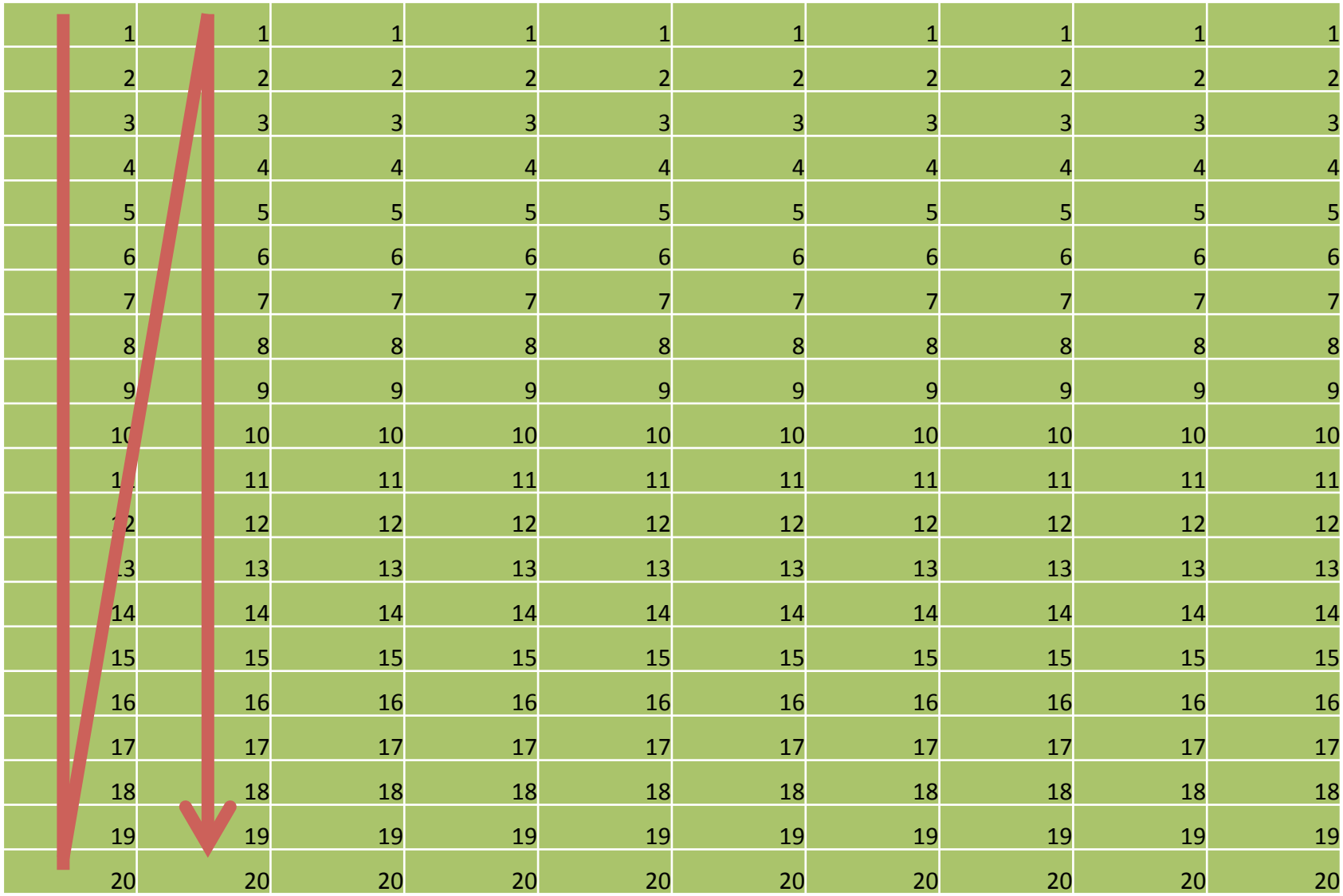
Days

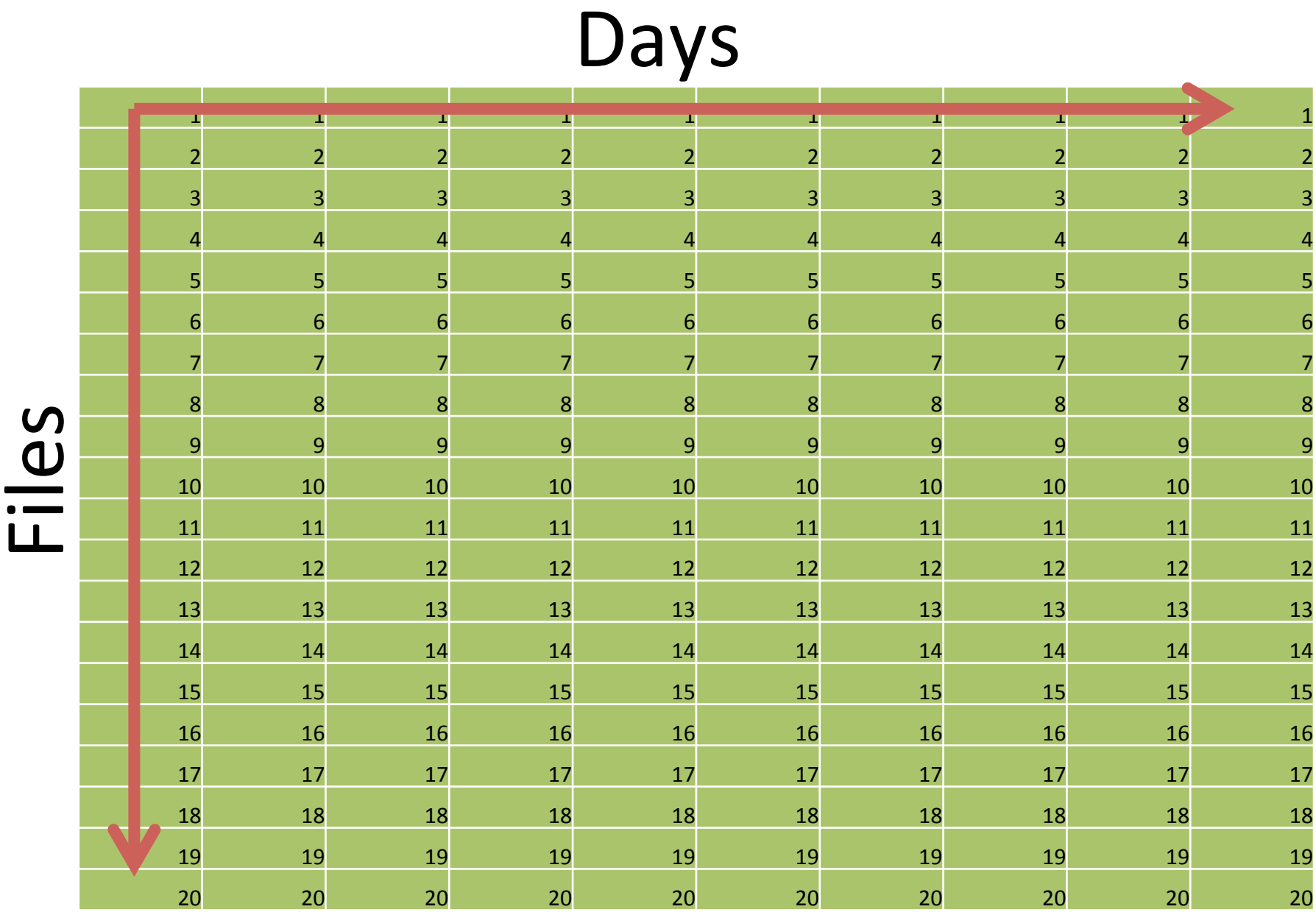
Files

1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9	9
10	10	10	10	10	10	10	10	10	10
11	11	11	11	11	11	11	11	11	11
12	12	12	12	12	12	12	12	12	12
13	13	13	13	13	13	13	13	13	13
14	14	14	14	14	14	14	14	14	14
15	15	15	15	15	15	15	15	15	15
16	16	16	16	16	16	16	16	16	16
17	17	17	17	17	17	17	17	17	17
18	18	18	18	18	18	18	18	18	18
19	19	19	19	19	19	19	19	19	19
20	20	20	20	20	20	20	20	20	20

Days

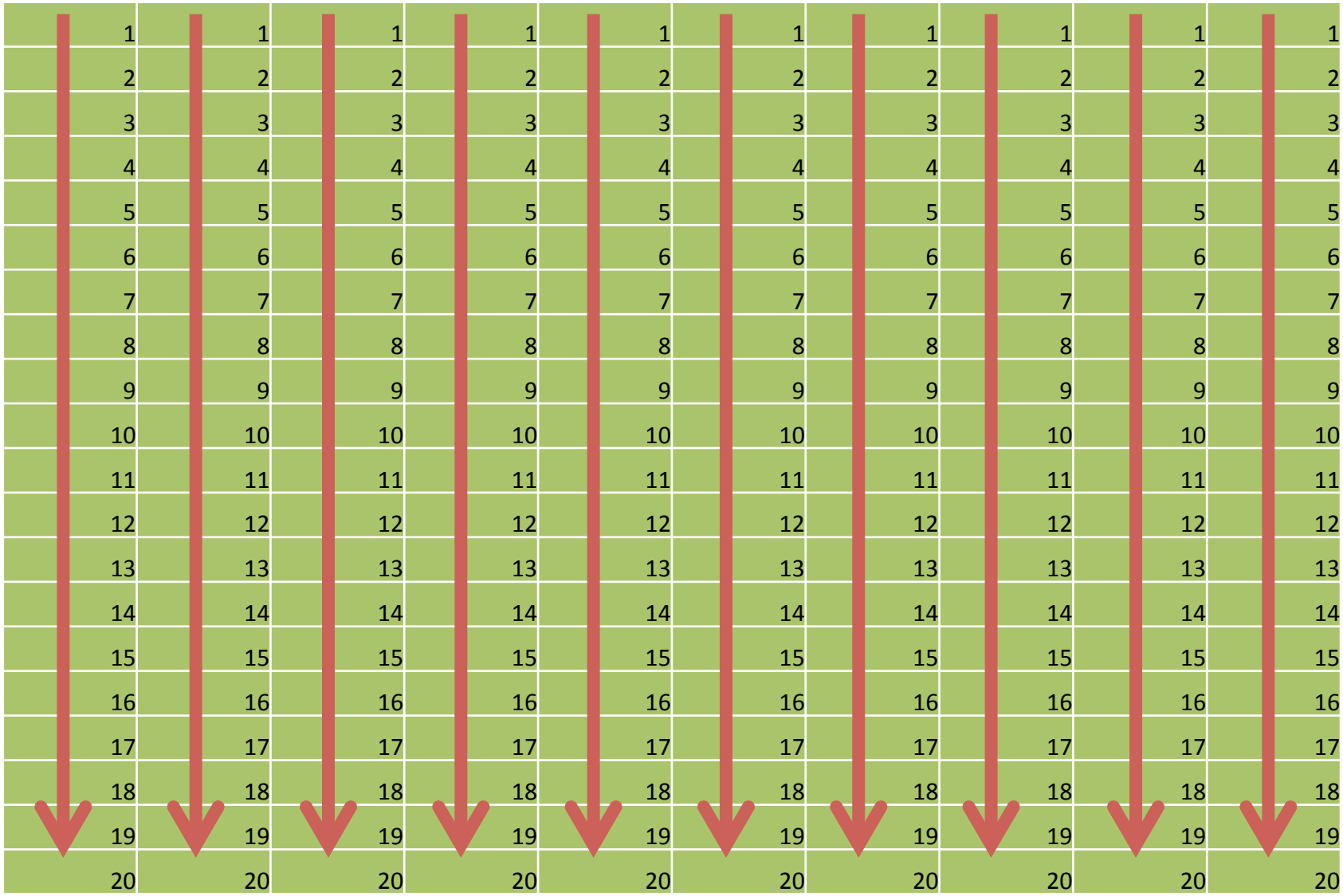
Files

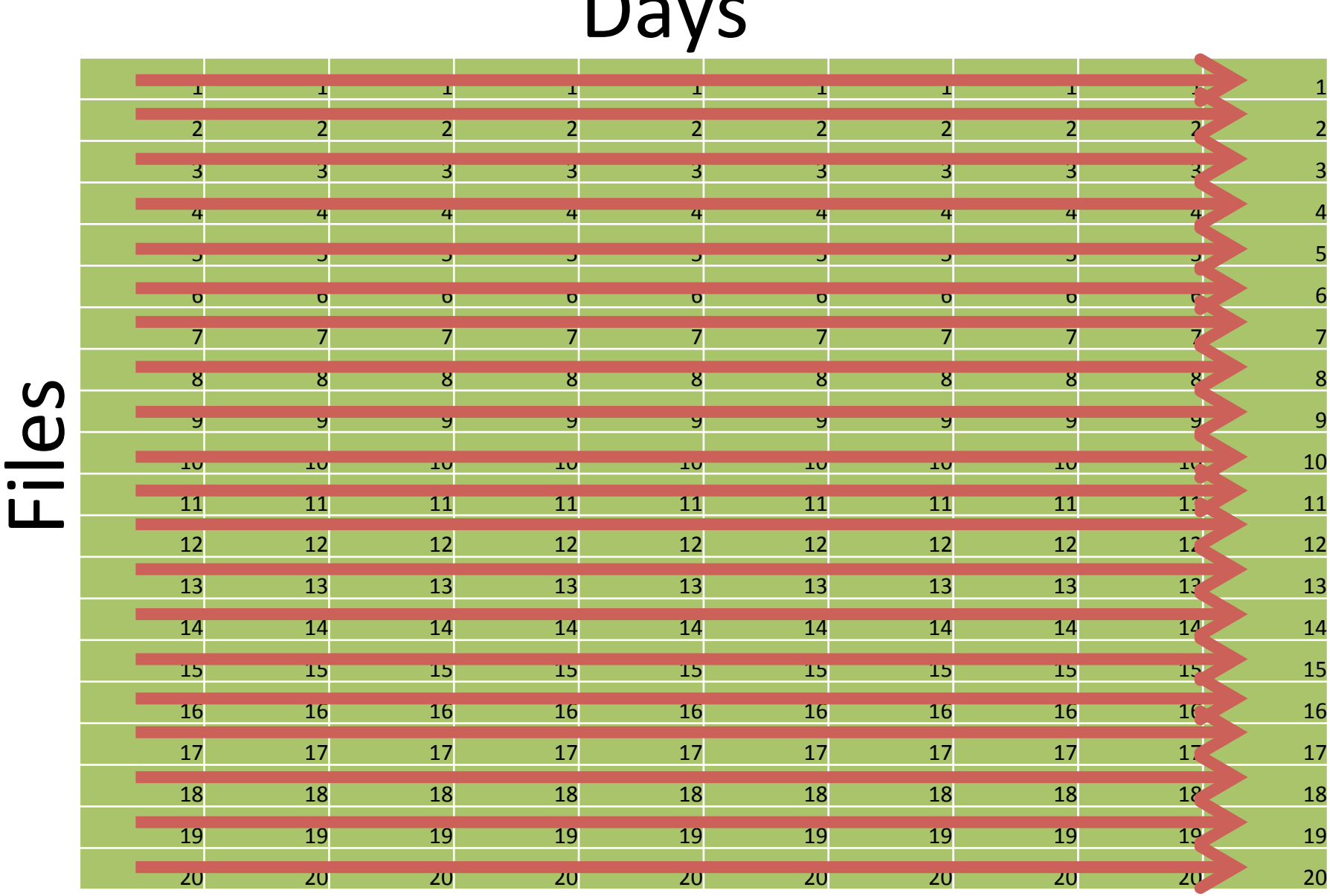

1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6
7 7 7 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9 9
10 10 10 10 10 10 10 10 10 10
11 11 11 11 11 11 11 11 11 11
12 12 12 12 12 12 12 12 12 12
13 13 13 13 13 13 13 13 13 13
14 14 14 14 14 14 14 14 14 14
15 15 15 15 15 15 15 15 15 15
16 16 16 16 16 16 16 16 16 16
17 17 17 17 17 17 17 17 17 17
18 18 18 18 18 18 18 18 18 18
19 19 19 19 19 19 19 19 19 19
20 20 20 20 20 20 20 20 20 20



- Parallel processing: How to carve up your data set?

Independent of day?




Days
Files

Matlab code

- Find the highest level of computation at which you can ‘split’ the data.
- Instead of using ‘for’ loops,
- Use ‘parfor’ loops.
- Refer to parfor_sample_code.m to get an idea

Recap

- Open a matlabpool (*matlabpool open*)
- Use a *parfor* loop for the parallel processing

Best tasks/stages for parallelization

- Those that are repetitive,
- Those that can be carried out simultaneously and independently by multiple workers, and
- Those that are not affected by the order in which they are processed by the workers.

Additional strategies

- Monitor computer's performance through the Performance tab on Task Manager (not sure what it is on other OSs), to ensure that you maximise CPU usage.

Additional strategies

- Scenario: you're certain that your code largely works (e.g. you've tested it on a subset of your data and it's fine), but not sure whether it will work on every single data file (e.g. in case some files are corrupted)
- You want to run the processing overnight, and don't want your code to crash/ get stuck if it encounters errors, but to move on to next files
- Use 'try-catch ME' statement within *parfor* loop, write errors to log file for later viewing

Points to note

- When running parallel routines using Matlab Parallel Toolbox, these routines are run in what is termed ‘headless’ mode.
- This means that the plotting of images onscreen is suppressed- you can save figures to file, but you cannot view them as they are being generated.

Points to note

- Debugging is not allowed in *parfor* loops. You can keep track of what your workers are doing indirectly, by printing strings to screen
- However, if a bug is present in your code, have to exit the parallel mode (by simply changing your *parfor* loop to a regular *for* loop) in order to fix it.
- You should ensure that your code is fully functional in non-parallel mode, before running it in parallel mode.

- Catch-22 scenario:
- you are trying to modify old code or write new code that implements parallel processing, but you are unable to debug it to uncover the source of your problems.

- Don’t worry, this is not a debilitating problem- if the bug is Parallel-Toolbox-related, Matlab will provide a description of the error. Flags will also appear in your code, drawing attention to violations of parallel processing laws.
- Problems tend to occur when allocation of data to workers is non-independent and causes conflicts.
- With a little practise, you will get the hang of how to allocate data correctly and optimally.