

## Automatic Parallelization/OpenMP

The instructions for automatic parallelization below use the GCC compiler. If you want to use the Studio compiler, take a look at the notes overleaf. For the OpenMP implementations, both compilers can be used.

**Note: the OpenMP part is meant for the session in the afternoon!**

### Exercise 1:

Write a C/Fortran code to calculate  $\pi$  using the formula

$$\pi = \int_0^1 \frac{4}{1+x^2} dx \approx \frac{1}{N} \sum_{i=1}^N \frac{4}{1 + \left(\frac{i-0.5}{N}\right)^2}$$

Implement the integrand as a function call!

1. Check your serial version.
2. GCC compilers (both C and Fortran) support limited automatic parallelization. You can use the option `-ftree-parallelize-loops=t`, where `t` is the number of threads to be used for the parallel execution. This value is a compile option, i.e. if you want to change the number of threads, you have to recompile with a different value for `t`! To be able to parallelize your code, the GCC compiler might need to apply advanced transformations as well, i.e. you need probably to add `-ffast-math` as well.
3. Why is `clock()` not useful as a timer in parallel programs?
4. Now implement the parallelism using OpenMP (see the lecture slides), and repeat step 2. Is your OpenMP comparable to the autopar version?

### Exercise 2:

Write a program that calculates matrix times vector - you can use the code shown in the lecture as a template.

1. Check your serial version.
2. Use the automatic parallelization options of the compiler to generate a parallel version.
3. Now implement the parallelism using OpenMP (see the lecture slides), and repeat step 3. Is your OpenMP comparable to the autopar version? What should you do to get similar behaviour so as with autopar, regarding small problem sizes.

### Using the Studio compiler for automatic parallelization

- Automatic parallelization in the Studio compiler is enabled with the `-xautopar` `-xreduction` `-xloopinfo` options.
- To control the number of threads used, set `OMP_NUM_THREADS` to a number larger than 1, e.g. 2, 4. Does the program run in parallel for all values of N? Use `/bin/time` to compare user and real time.