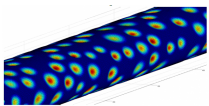
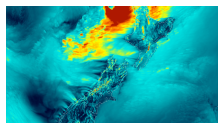
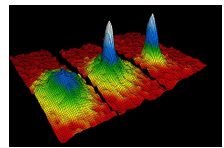
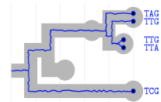
How NeSI helps Manaaki Whenua - Landcare Research monitor land cover changes

Alexander Pletzer, James Shepherd (MWLR), Mandes Schönherr and Chris Scott
(alexander.pletzer@nesi.org.nz)
eResearch 2019, Auckland
18-20 Feb 2019

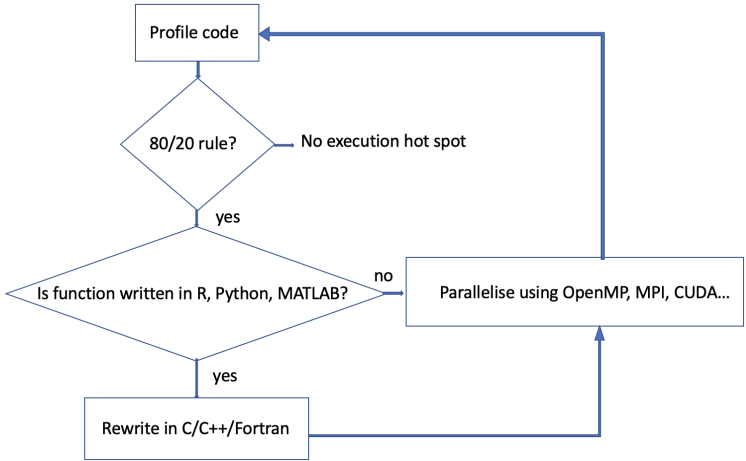
New Zealand eScience Infrastructure

NeSI has a free consultancy service

- to help users run better and faster on NeSI platforms
- https://www.nesi.org.nz/services/consultancy
  - profile and optimise code
  - implement algorithms
  - leverage GPUs, OpenMP, MPI
  - add visualisation
  - introduce “good” engineering practices

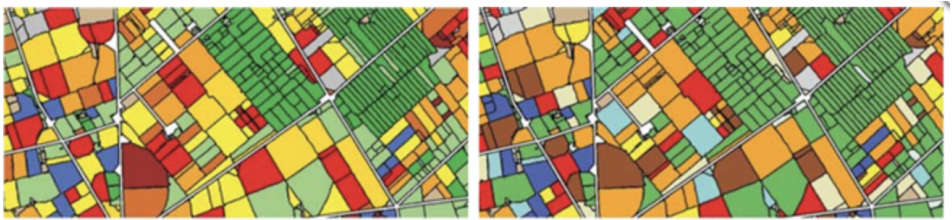



,


Typical workflow for getting your code to run faster



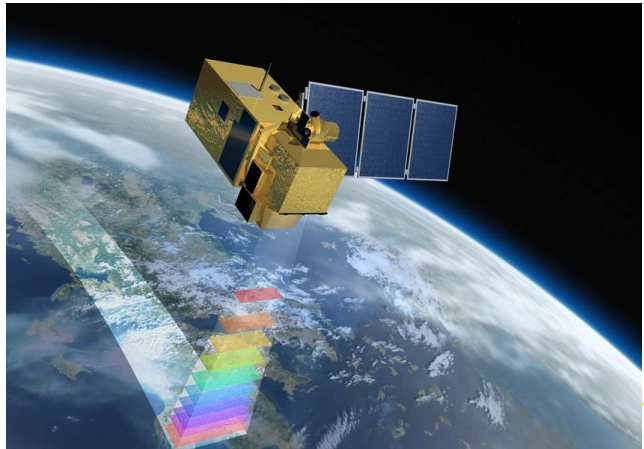
Example of a NeSI consultancy: monitor land-use changes

- help determine land cover patterns (forests, urban areas, agriculture)
- type of soil
- distribution of crops in time
- used for environmental modelling



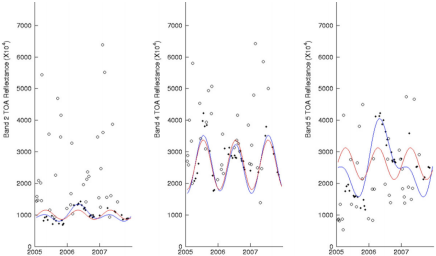
Input is satellite imagery

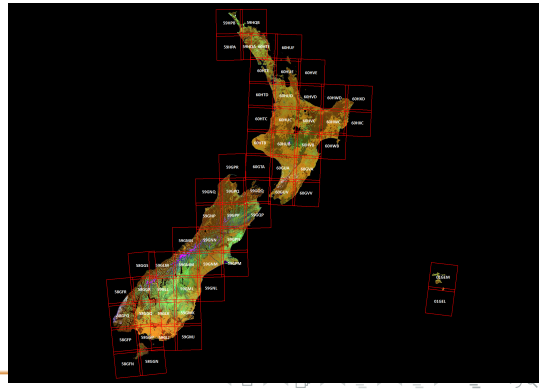
- using near infrared (NIR) and short wave infrared (SWIR)
- provides reflectance
- cloud, cloud shadows and snow are a problem



TMASK code analyses multiyear NIR/SWIR Sentinel satellite imagery

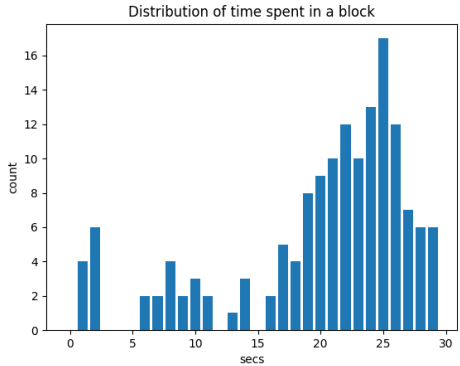
- written in C, computes five Fourier coefficients to account for seasonal variation
- uses a regression model implemented in the Gnu Scientific Library (GSL)
- takes 10 hours to compute the coefficients per tile & satellite band (3 bands)





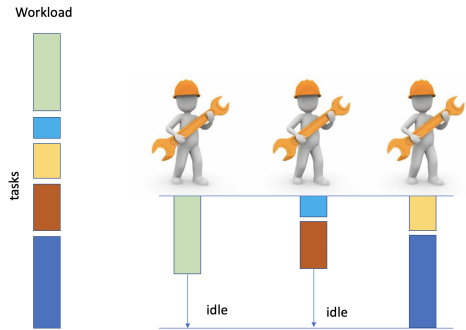
Good news! TMASK's computation is embarrassingly parallel

- computation of coefficients is independent for each pixel, but...
- the time it takes can vary significantly
- shown below is the CPU time it takes to compute the coefficients for a 256x256 block



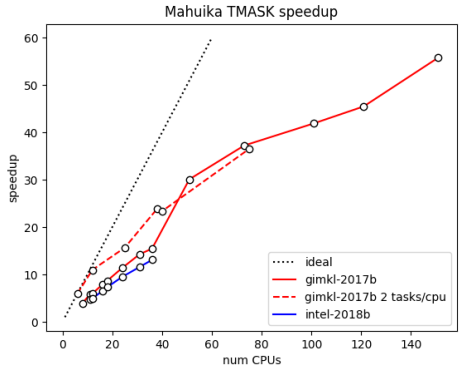
Master-worker paradigm addresses load balancing problem

- master goes through list of tasks. Each task involves computing 5 coefficients for a block of pixels (typically 256x256)



Up to 53x speedup on NeSI's Mahuika cluster

- not limited by communication, load balancing is the limiting factor
- best efficiency when each process handles $\sim$ 3 tasks
- GNU compiler worked best
- assigning two MPI tasks per CPU gives up to 20 percent extra boost



Summary

- reduced the wall clock time of test case from 55 min to less than one minute (151 MPI processes for 150 blocks)
- best to have about 10 tasks per process
- further potential for parallelisation in satellite bands
- manager consumes one process so works best when number of processes $\gg 1$
- memory considerations may dictate optimal number of MPI tasks
- can spread the computation across multiple nodes (unlike OpenMP and other parallelisations)
- Other serial codes in the NZ research community may benefit from a similar approach

CPUs aren't getting faster

- Don't count (too much) on the next processor to run faster
- The end of the road for serial code?

How can we help you run better on NeSI's supercomputers?

- talk to us
  - Tue 1:30pm: Visualisation capabilities of NeSI's new high performance computers
  - Tue 1:30pm: A day in the life of NeSI's Apps Support
  - Wed afternoon: Bring your own code workshop
- to request help send email to: support@nesi.org.nz
- check out NeSI's case studies at https://www.nesi.org.nz/case-studies
- Thanks!