VASP: Best Practices

Weine Olovsson

National Supercomputer Centre (NSC)

@ Å80109, Ångström Lab., Uppsala University, Uppsala, 11th June 2019

LINKÖPINGS UNIVERSITET

My background

- PhD in Physics 2005 @UU
- Application Expert @NSC, 2011 (50%), 2016 (90%)
- 10% theoretical spectroscopy @IFM, LiU
- Electronic structure calculations
- VASP, WIEN2k, Quantum Espresso, GPAW, Siesta, …

Introduction

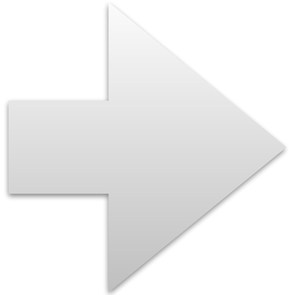
- Focus on practical aspects of running VASP
  …at specific supercomputer centres
- Influential parameters, NPAR/NCORE, ALGO, NSIM, KPAR, …
- Memory usage
- Benchmarks, examples
- Common problems

… clickable links are underlined

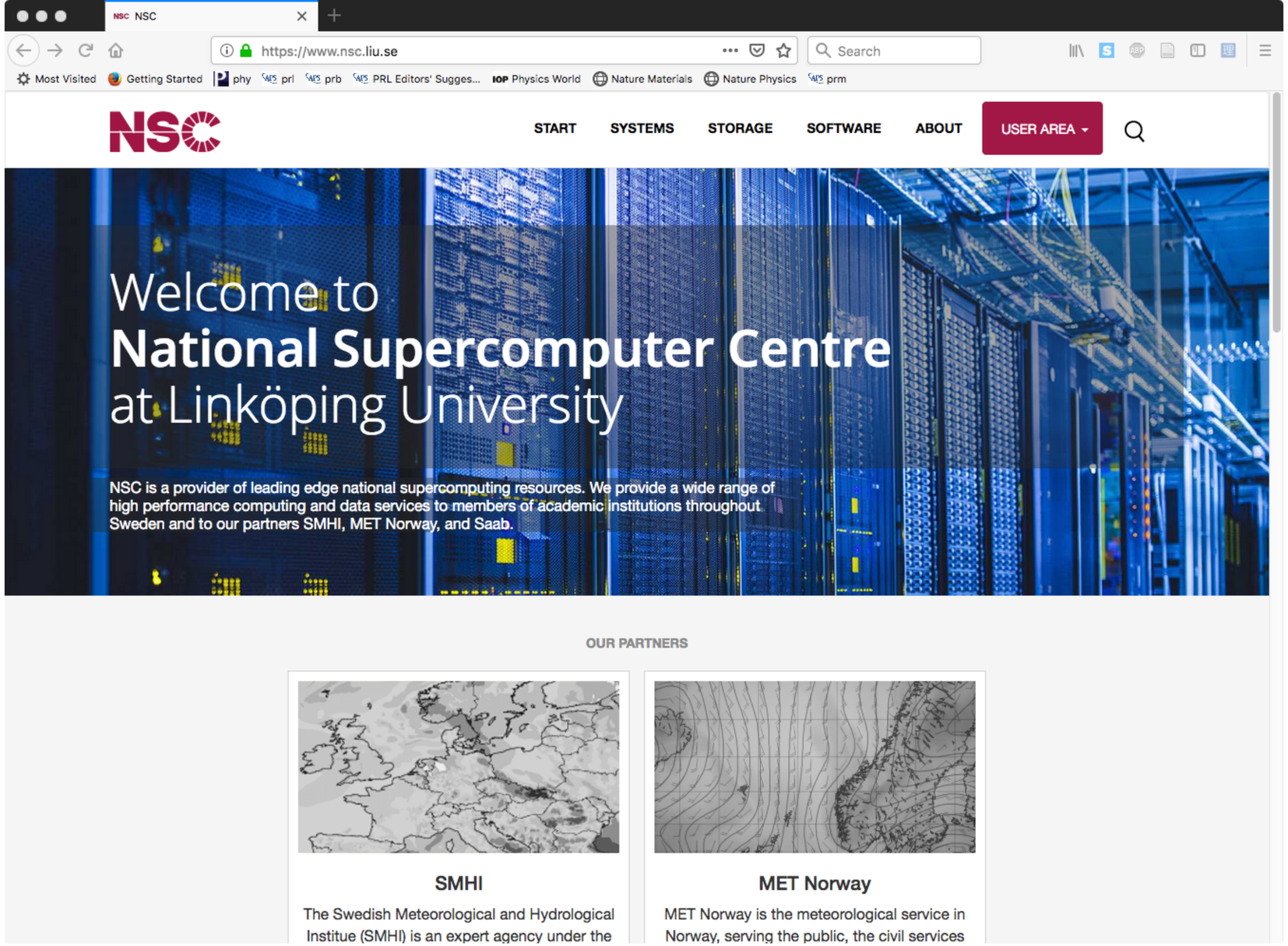

b-initio
VASP
ackage
imulation
ienna

VASP background

- PAW-method
- DFT, **post-DFT** (HSE06, GW, …)
- Born-Oppenheimer Molecular Dynamics
- widely used in Academia/Industry
  - Efforts from Intel & Nvidia for optimization
- 20-25% of Tetralith usage



http://vasp.at/



https://www.nsc.liu.se/ Software > Installed software > Tetralith & Sigma software list > VASP

https://www.nsc.liu.se/software/installed/tetralith/vasp/

NSC NSC

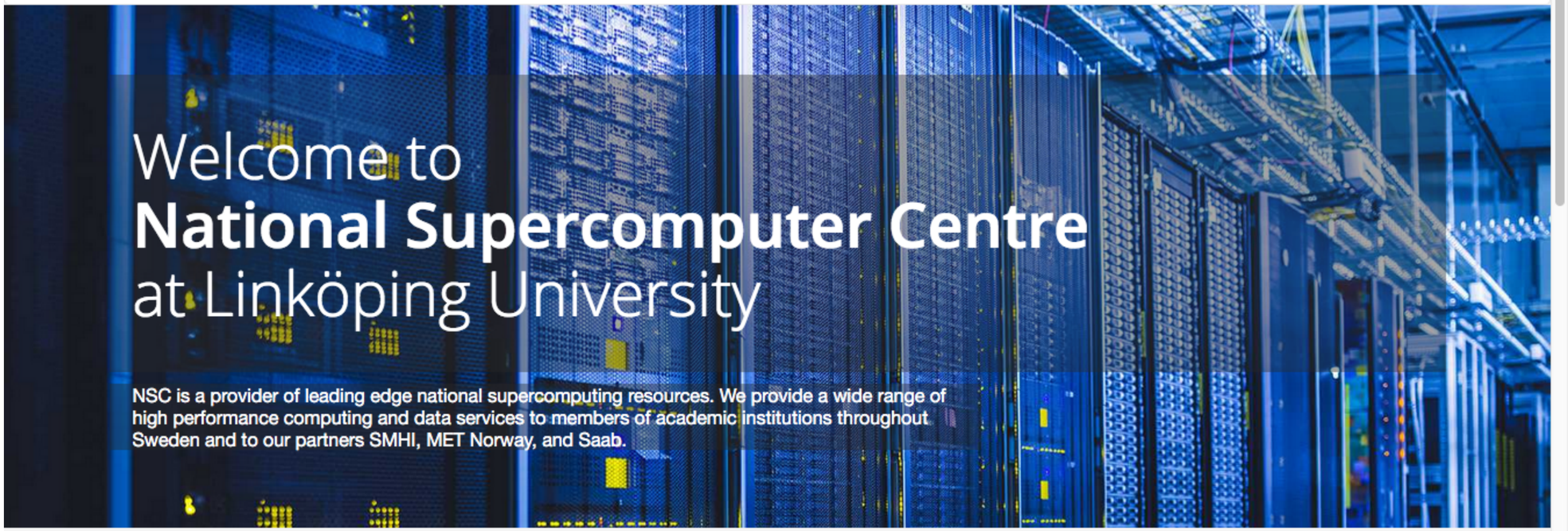
https://www.nsc.liu.se

Search

Most Visited | Getting Started | phy | prl | prb | PRL Editors' Sugges... | Physics World | Nature Materials | Nature Physics | prm

NSC

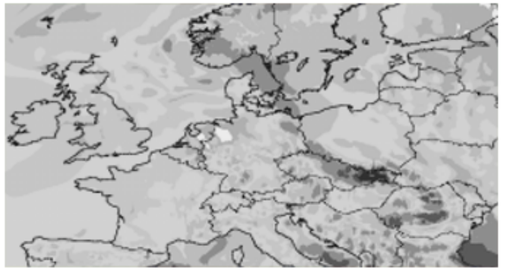
START SYSTEMS STORAGE SOFTWARE ABOUT USER AREA



Welcome to National Supercomputer Centre at Linköping University

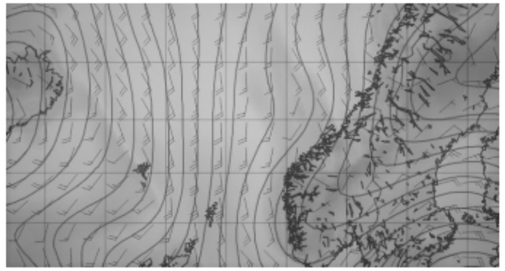
NSC is a provider of leading edge national supercomputing resources. We provide a wide range of high performance computing and data services to members of academic institutions throughout Sweden and to our partners SMHI, MET Norway, and Saab.

OUR PARTNERS



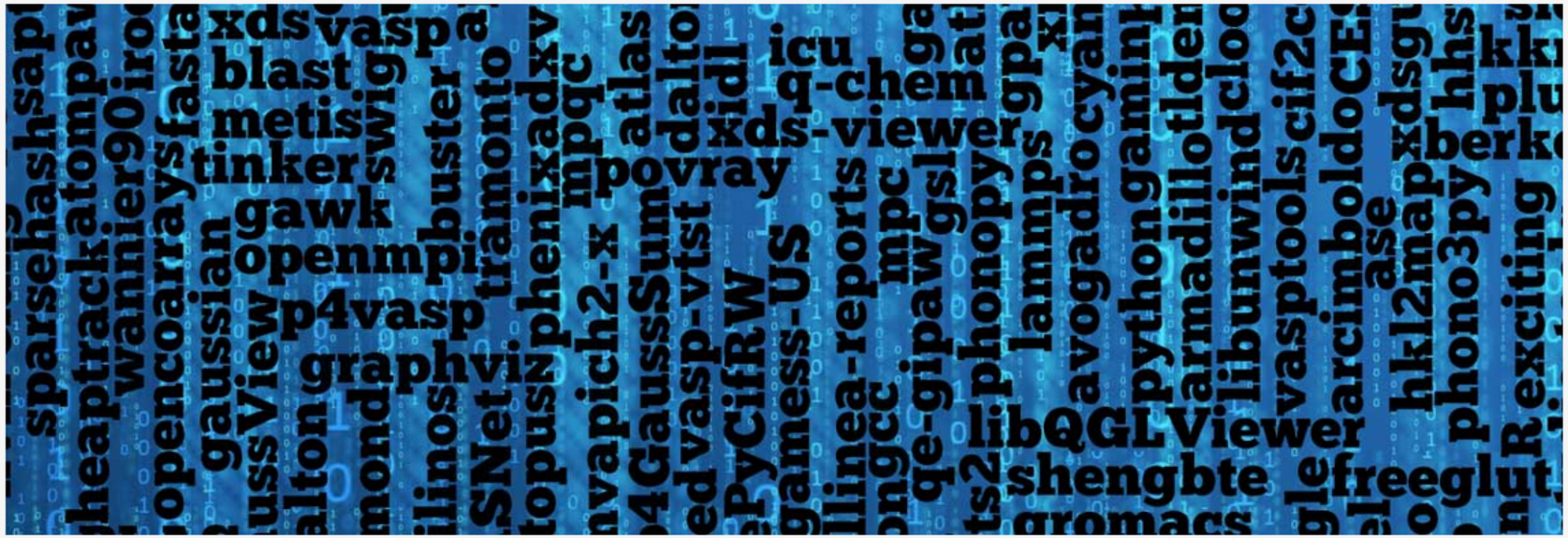
SMHI

The Swedish Meteorological and Hydrological Institue (SMHI) is an expert agency under the



MET Norway

MET Norway is the meteorological service in Norway, serving the public, the civil services



Software

Explore this part to find out about all software environment related matters. Please proceed to the installed software page to find out which scientific software we have on our clusters. If you are going to compile software from source code, we suggest that you read the NSC build environment introduction and the compilers section.

Installed software

Which software is available on what systems and how to run it

Software installation policy

What to do if software you need isn't installed

Software licensing

How we handle software licensing of commercial software.

Installed software

NSC has a large number of software installations available, often in multiple versions to suit the needs of various user communities. For a list of installed software, please see the corresponding resource page below. If you need software that is presently not installed, please see our software installation policy.

Software portfolios by cluster

- Tetralith & Sigma Software List.
- Bi Software List. This is the main resource for finding out what software is available on Bi.

Module system

You can also query the module system for available software and recommendations on what versions to use, e.g:

```
module avail
module add vasp/recommendation
```

SNIC knowledge base

Information on software and availability for all of SNIC is also available in the SNIC knowledge base software section. There is specific information for these NSC resources:

Tetralith & Sigma Software

A list of software installed on Tetralith and Sigma

Tetralith & Sigma Software

Tetralith & Sigma Software List

DISCLAIMER: Please note that the software catalogue is a work in progress! If your application is missing, please request it by sending e-mail to NSC Support

The following scientific applications have been installed centrally under `/software/sse/` . This list may not always be 100% up to date. The most reliable source is running the command `module avail` while logged into Tetralith or Sigma, possibly augmented by `ls /software/sse/manual/` to show additional manually performed installations without modules. Please note that some of this software is licensed, and may not be available for everyone. You need ask NSC for access, which is typically granted upon some proof of having a license.

The list was last updated: 2019-02-21

Electronic structure

- Abinit
- ASE
- Elk
- EPW
- exciting
- GPAW
- phonopy
- phono3py
- p4vasp
- Quantum Espresso
- vasptools
- VASP (licensed)
- WIEN2k (licensed)

VASP Installations on Tetralith & Sigma

First of all, VASP is licensed software, your name needs to be included on a VASP license in order to use NSC's centrally installed VASP binaries. Read more about how we handle licensing of VASP at NSC.

Some problems which can be encountered running VASP are described at the end of this page.

How to run: quick start

A minimum batch script for running VASP looks like this:

```
#!/bin/bash
#SBATCH -J jobname
#SBATCH -N 4
#SBATCH --ntasks-per-node=32
#SBATCH -t 4:00:00
#SBATCH -A SNIC-xxx-yyy

module add VASP/5.4.4.16052018-nsc1-intel-2018b-eb
mpprun vasp_[std/gam/ncl]
```

This script allocates 4 compute nodes with 32 cores each, for a total of 128 cores (or MPI ranks) and runs VASP in parallel using MPI. Note that you should edit the jobname and the account number before submitting.

Software

General information about VASP

Licenses

General information about VASP

System	Available versions
Beskow	5.4.4-wannier90, 5.4.4, 5.4.1.patched, 5.3.5-vtst3.1, 5.3.5-31Mar14
Tegner	5.4.1, 5.3.5

The Vienna Ab initio Simulation Package (VASP) is a computer program for atomic scale materials modelling, e.g. electronic structure calculations and quantum-mechanical molecular dynamics, from first principles.

For more information see: http://vasp.at

Licenses

VASP is not free software and requires a software license. VASP licenses are managed in SUPR. All people who want to use VASP should have SUPR accounts and be a member of a VASP group in SUPR. VASP groups have owners, typically a principal investigator of a project, and that owner can add and remove people using the SUPR interface. If you are Ph.D student, we suggest that you check with your supervisor.

Using VASP 5.4.4 on Beskow

General observations

- VASP is not helped by hyper-threading (64 virtual cores per compute node).
- No GPU/OpenMP-support.
- Running on fewer than 32 cores per node allocates more memory to each MPI task. This can in some cases improve performance and is necessary if your job crashes with an OOM error. See the example submit script below on how to do this correctly.

NPAR, NCORE and NSIM

From initial testing, we recommend:

- NPAR = number of compute nodes
- NCORE = cores / node, typically 16,24 or 32.
- NSIM = 2
- KPAR = number of compute nodes (if applicable)

How to choose the number of cores

Rule of thumb:

- 1 atom per core = Good
- 0.5 atom per core = Could work (but bad efficiency and time wasted)

First: day-to-day tools

• less / vi	reading/editing files	
• grace / gnuplot	plotting tools	
• Bash	simple scripts	
• cif2cell	convert from .cif	
• p4vasp	analysis of VASP output	
• xcrysden / vesta	view structure	
• Schrödinger	create/view structure	Tetralith (license for users)

Also of interest:

• Python / R	analysis etc. (check out e.g. using jupyter)
• ASE	different tools and Python modules
• Matlab / Octave	analysis etc.

Check out presentations for Tetralith training

Computation - considerations

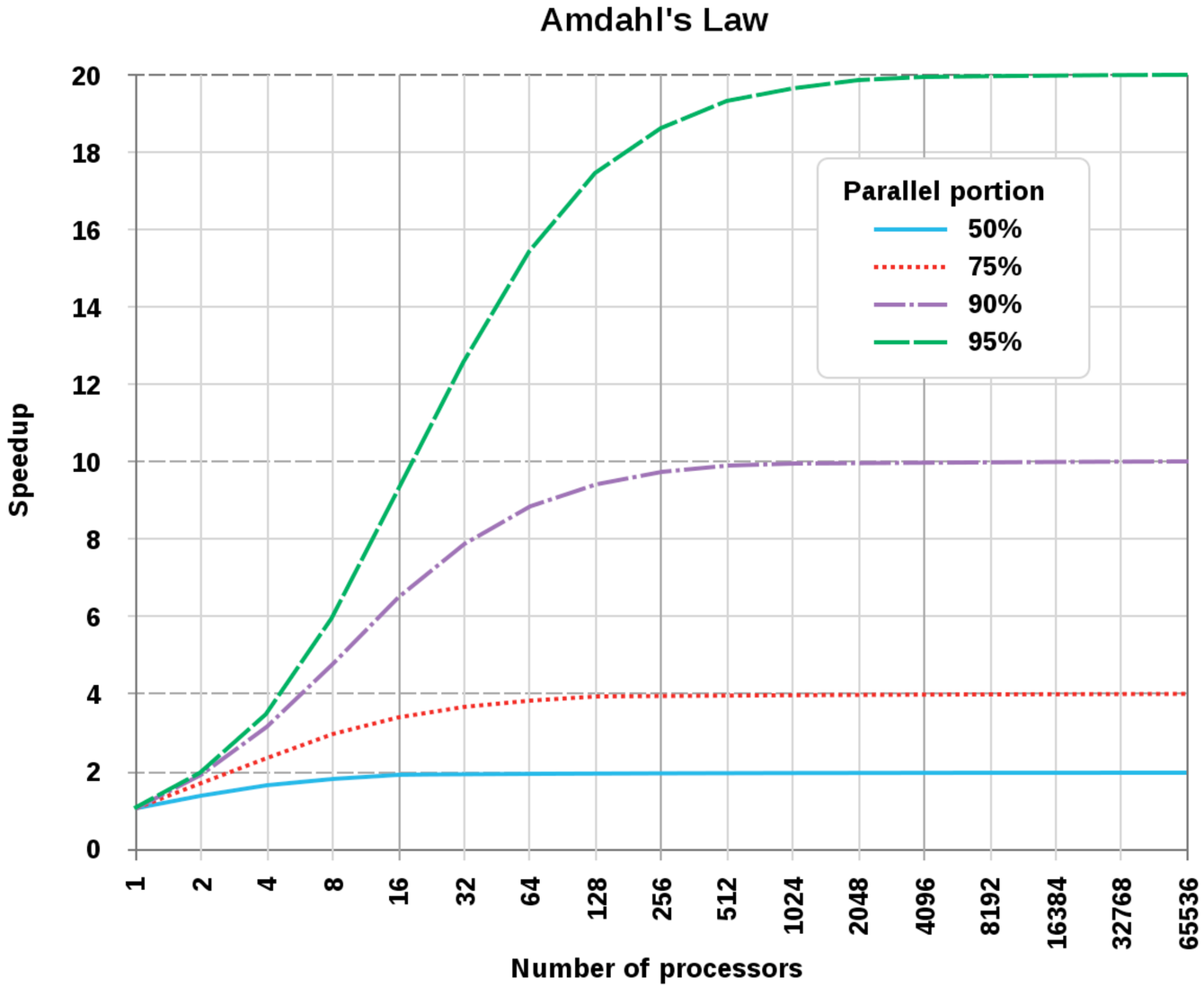
Efficiency:
Running as many jobs as possible for a given allocation of computer time

Speed:
The amount of time (real, "human time") to run a specific simulation from when it starts

Time-to-solution:
Speed + the time waiting in queue

@Tetralith: wall-time limit **7 days**,
@Beskow: **24h**, preferably ≥ **4 nodes**

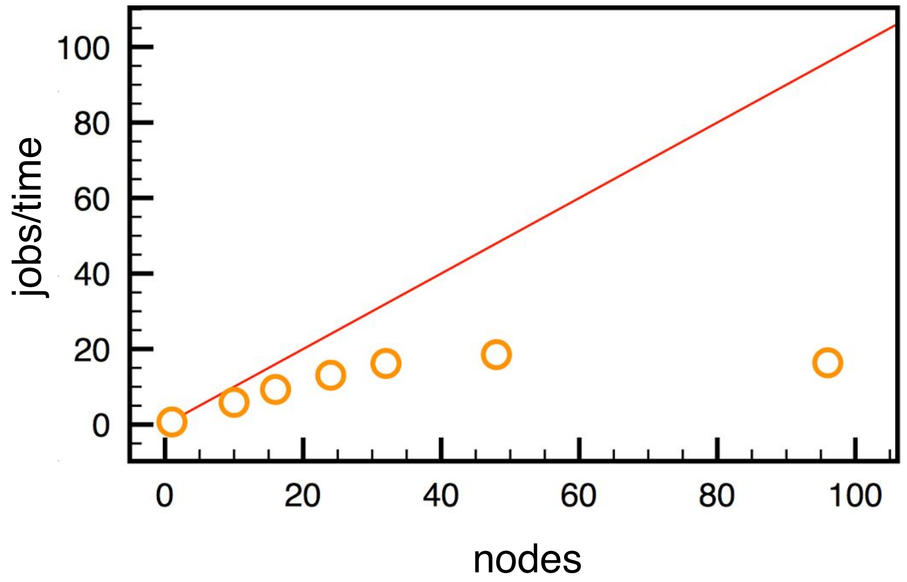
Parallelization - limitations

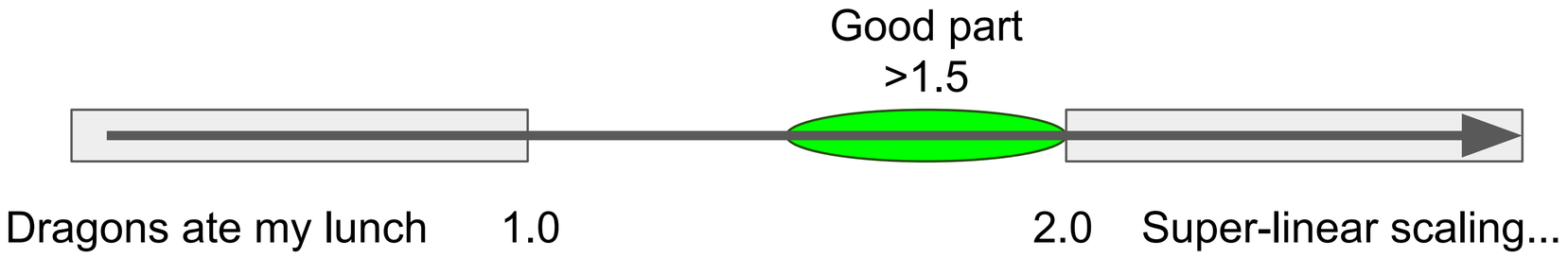


Simple scaling analysis

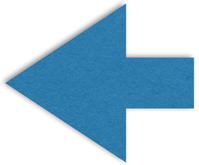
A minimal scaling analysis can save lots of allocated core hours…

1. Tool your runscript to time your simulation
2. Run an initial best guess number of cores (n)
3. Run the same test on half the number of cores (n/2)
4. Score = time(n/2) / time(n)





In general

- Read the documentation!
- VASP default settings  good starting point
- Caution: “inherited” starting files
- Avoid overly complex INCAR
- Possible differences in centres installations

refer to respective webpages / documentation

VASP versions & utilities

- **Latest:** patch.5.4.4.16052018 for 18Apr17
  - Check centre webpages for details!
- wannier90: maximally localized wannier functions
- VTST: transition state tools for VASP
- VASPsol: solvation model for VASP
- Beef: Bayesian error estimation functionals
- constrained relaxation (NSC)

- On many clusters: `$ module avail vasp`

VASP binaries

- vasp_std - regular version
- vasp_gam - one k-point (Gamma), faster
- vasp_ncl - noncollinear magnetism
- + modifications

Starting files

- INCAR - input parameters
- POSCAR - structure (generate using *e.g.* cif2cell)
- POTCAR - PAW potentials (which ones?)
- KPOINTS - k-mesh (or list)
- + job script

Example of job script - Tetralith (NSC)

```
#!/bin/bash
#SBATCH -A snic2019-3-203
#SBATCH -J test
#SBATCH -t 3:59:00
#SBATCH -N 2

module load VASP/5.4.4.16052018-nsc1-intel-2018a-eb
mpprun vasp
```

Recommended to load module in job script,
e.g. due to possible runtime fixes

https://www.nsc.liu.se/software/installed/tetralith/vasp/

Example of job script - Beskow (PDC)

```
#!/bin/bash -l
#SBATCH -A 2019-3-203
#SBATCH -J test
#SBATCH -t 3:59:00
#SBATCH --nodes=2

module unload cray-mpich/7.0.4
module load vasp/5.4.4

aprun -n 48 -N 24 vasp
#aprun -n 64 -N 32 vasp
```

https://www.pdc.kth.se/software/software/VASP/beskow/5.4.4/index.html#running-vasp

POTCAR

- Check recommendations
- PBE, LDA
- for short bonds: _h
- for GW: _GW

Note several choices, e.g.:
Ga, Ga_d, Ga_d_GW,
Ga_GW, Ga_h, Ga_sv_GW

```
$ grep PAW POTCAR

$ grep ENMAX POTCAR
```

INCAR parameters

- PREC - “precision”, ENCUT and FFT grids
- ENCUT - plane wave energy cutoff
- ALGO - wf optimisation
- NBANDS - if not set, auto-determined
- NSIM - for RMM-DIIS algorithm (ALGO)
- NCORE or NPAR - bands treated in parallel
- KPAR - k-point parallel

INCAR parameters

- PREC - “precision”, ENCUT and FFT grids
- ENCUT - plane wave energy cutoff

Recommended to set!

- ALGO - wf optimisation
- NBANDS - if not set, auto-determined

Must be the same for Etot comparison!

- NSIM - for RMM-DIIS algorithm (ALGO)
- NCORE or NPAR - bands treated in parallel
- KPAR - k-point parallel

Hardware - affects best practices

- **Tetralith** (NSC), Intel Xeon Gold 6130 2.1GHz

  1 node = 32 cores (96GB RAM, fat node 384GB)

- **Beskow** (PDC), Cray XC40, Intel Xeon E5v3 2.3GHz

  1 node = 32 cores (64GB RAM), no local disk; use 24 cores/node (large jobs)

- **Kebnekaise** (HPC2N)

  1 node (Intel Xeon E5-2690v4) = 28 cores (128GB RAM)

  + 2xGPU (NVidia K80) ← **vasp-gpu version!**

  1 KNL (Intel Xeon Phi 7250) node = 68 cores (192GB RAM)

different best practices →

PREC

- PREC = “precision”, sets ENCUT and FFT grids
- PREC = Normal, **default**
- PREC = Accurate, highly accurate forces
- Recommended to set ENCUT by hand

NBANDS

- **Easy to divide**, 2^n, 4, 8, 12, 16, …
- NBANDS = 511 or 512?
- Max limit, 1 band/core
- **Affects Etot!**

Run e.g. quick job to check NBANDS:

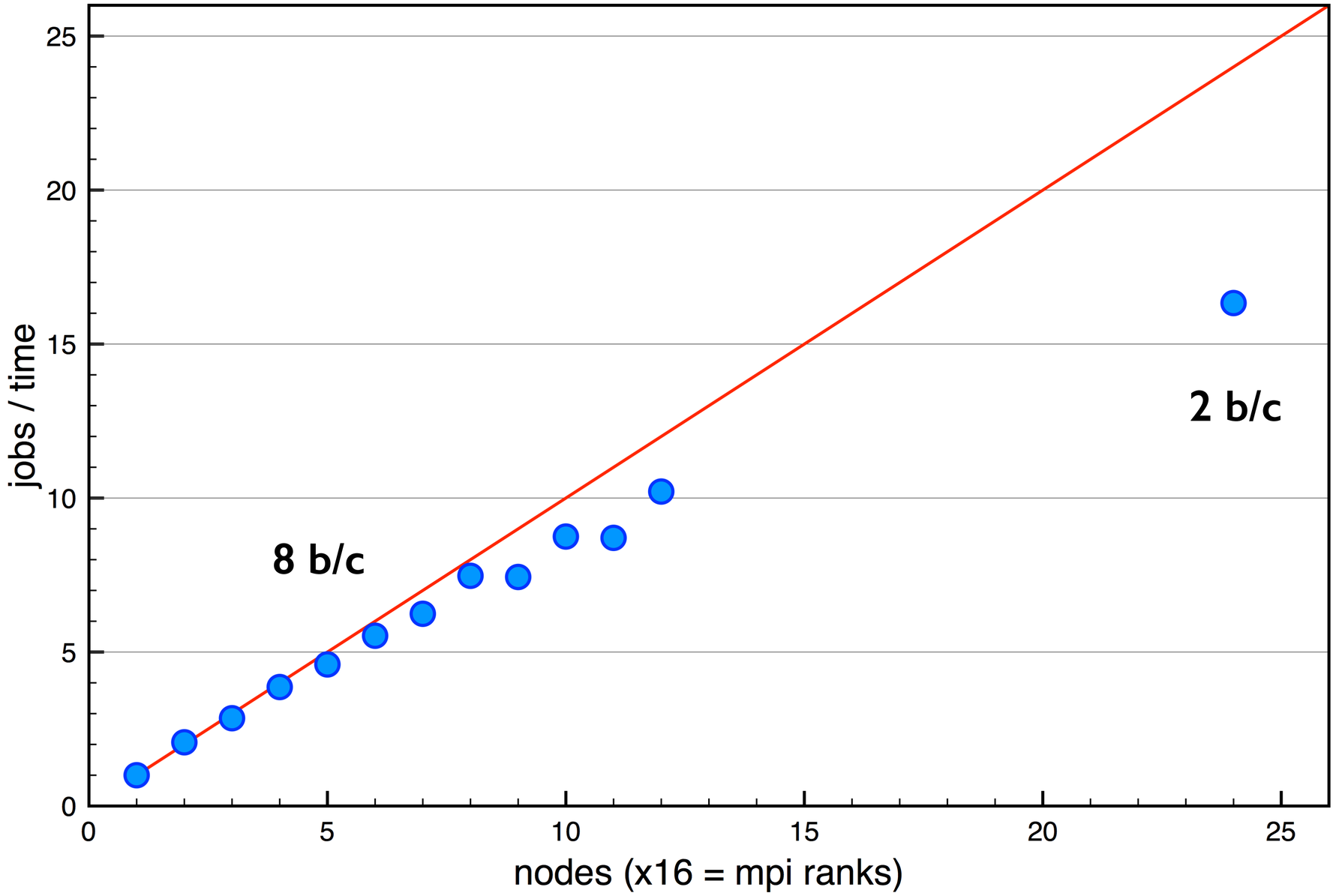
```
#SBATCH --reservation=devel
```

```
$ grep NBANDS OUTCAR
```

How many cores - efficient and/or fast?

- Start from # of bands, **NBANDS**
- 1 band/core: typically inefficient
- 2 bands/core: ~50% parallel efficiency
- 8 bands/core: good starting point
  - try e.g. **cores ≈ NBANDS / 8**

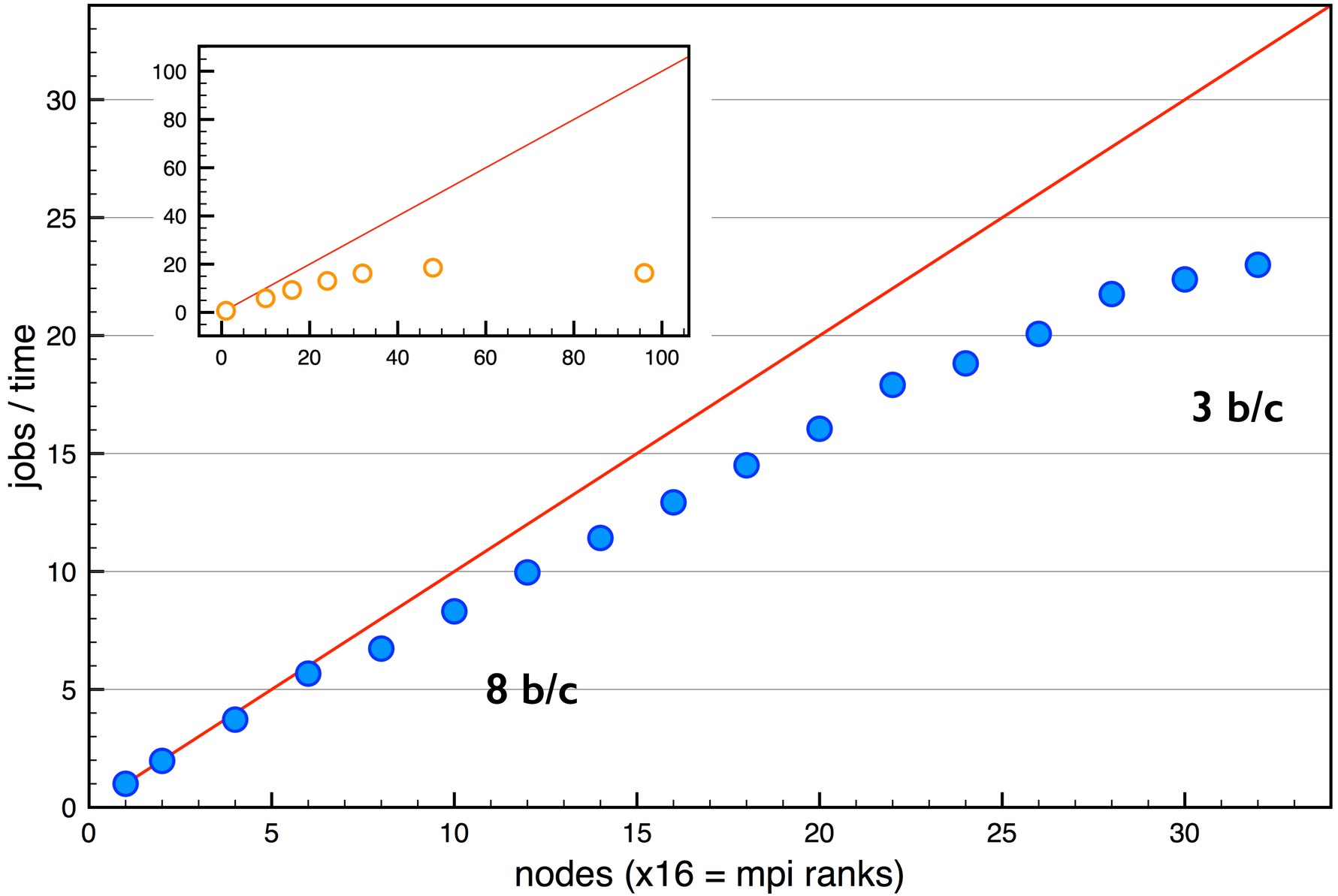
Si-H/Ag(111) 129 atoms, VASP PBE @Triolith (old)



NBANDS=750
4 k-points

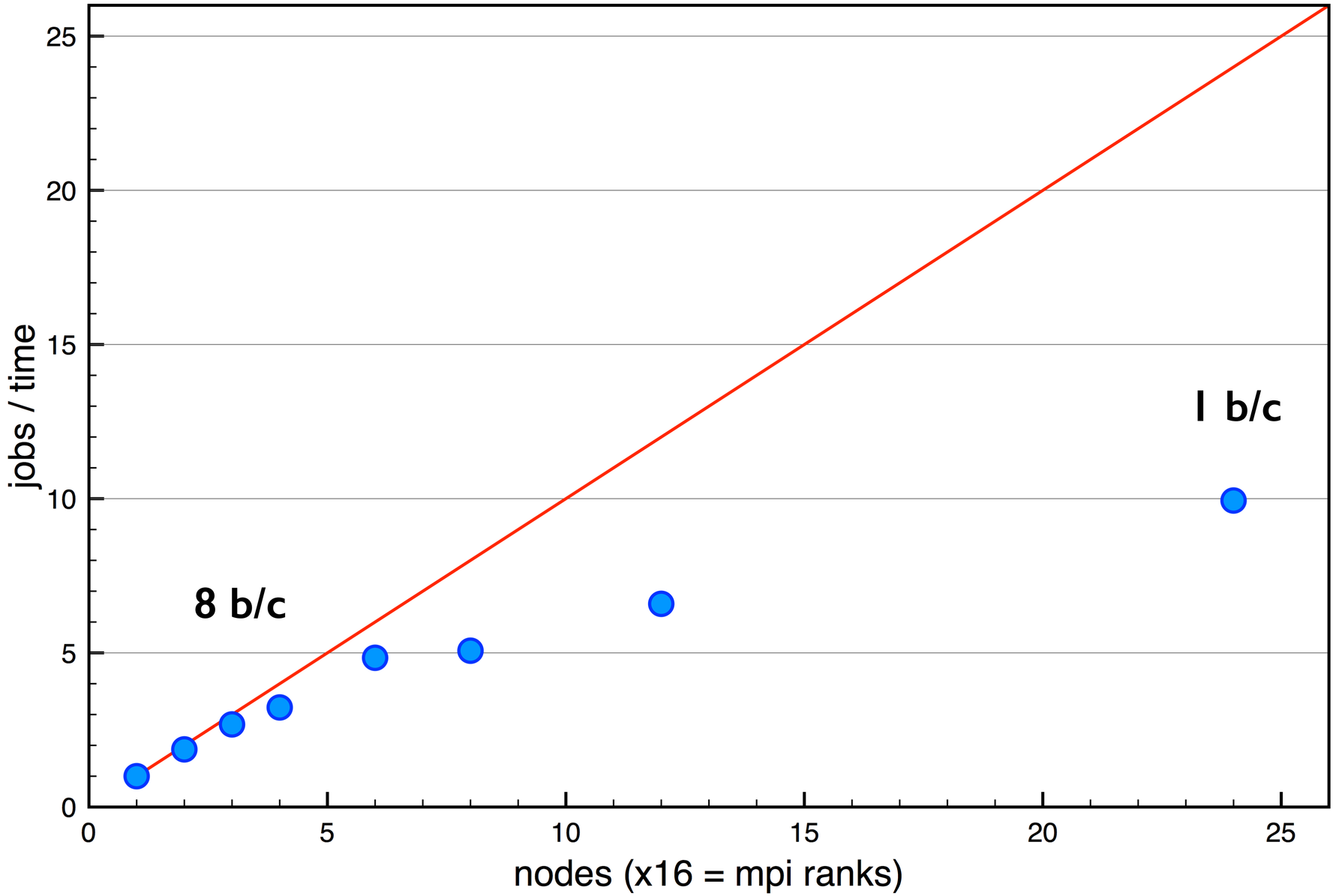
Triolith had **16** c/node
Tetralith: **32** c/node

GaAsBi 512 atoms, VASP PBE @Triolith (old)



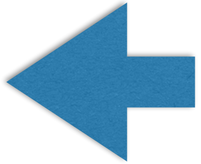
NBANDS=1536
4 k-points

GaAsBi 128 atoms, VASP HSE06 @Triolith (old)

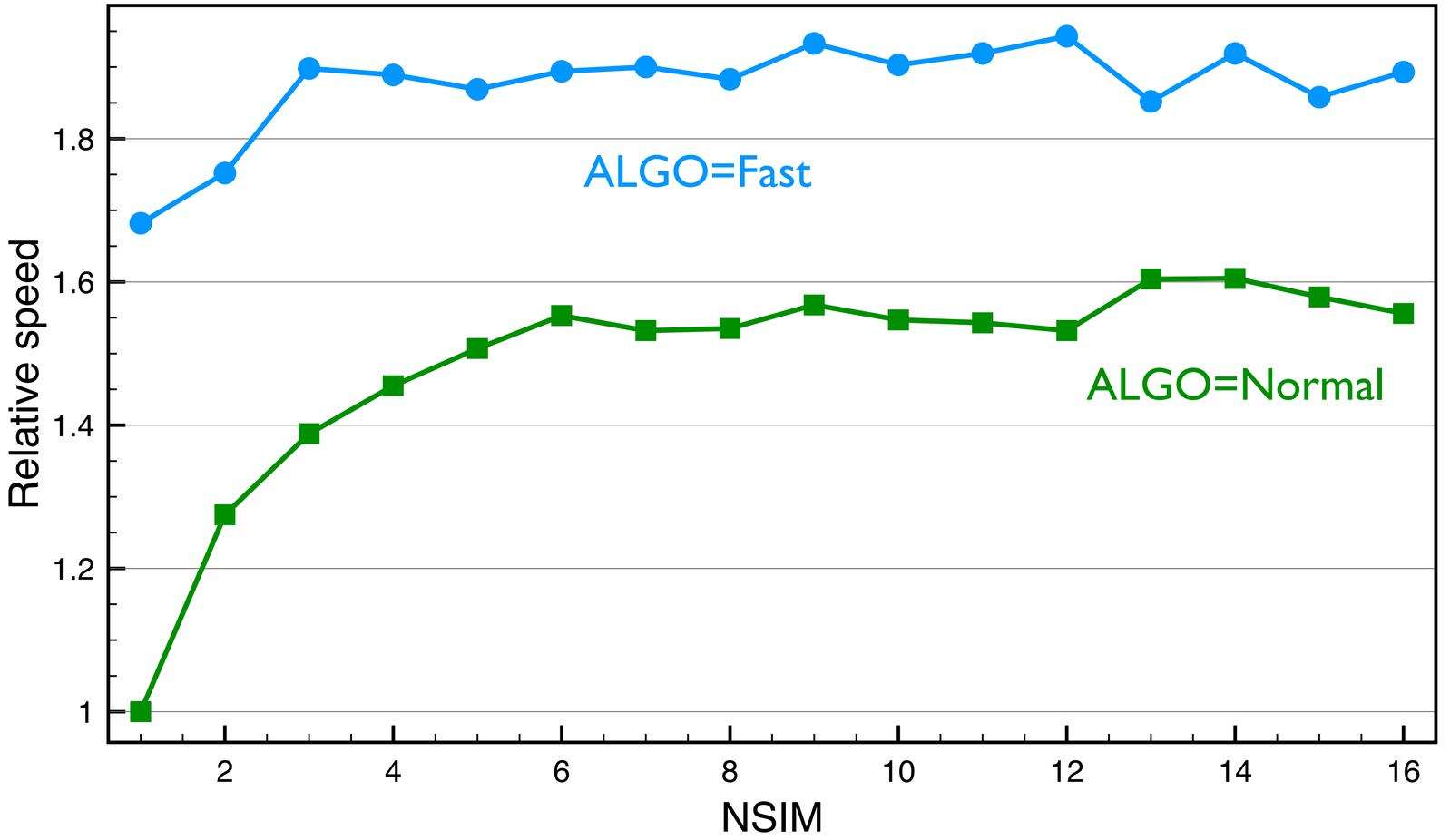


NBANDS=384
12 k-points

ALGO & NSIM

- Blocking mode for RMM-DIIS algorithm
- ALGO = Fast (Dav + R-D) / VeryFast (R-D)
- ALGO = Normal (Davidson algorithm), **default**
- **not for hybrid-DFT**, HSE06 (Damped, All, Normal)
- NSIM = 4, **default**  usually good
- Tetralith: NSIM = 4 (or higher)
- Beskow: NSIM = 2

Si-H/Ag(111) 129 atoms, VASP PBE @Triolith (old)



NBANDS=750, 4 k-points

NCORE or NPAR

(default)

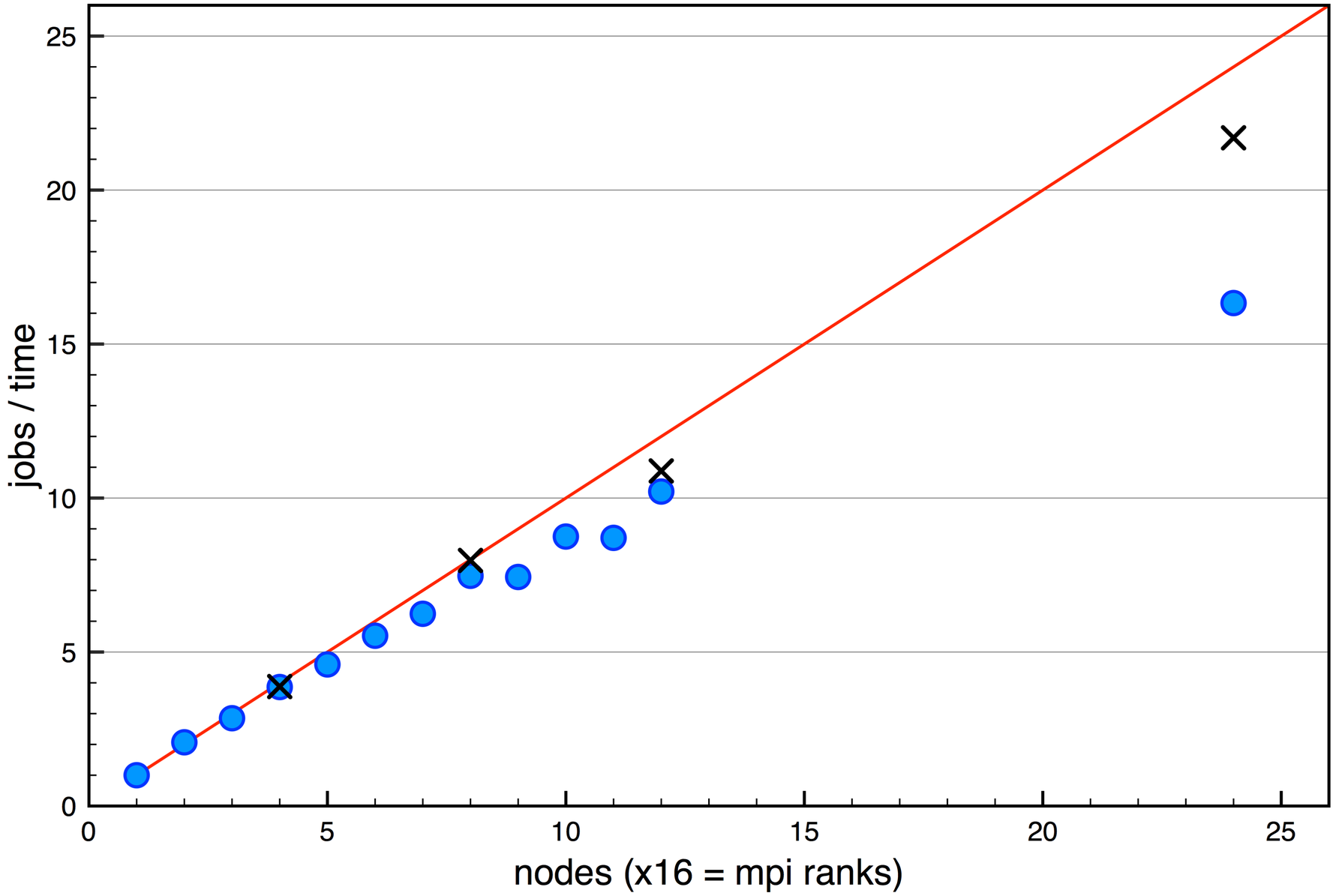
- cores per orbital / bands treated in parallel
- Davidson & RMM-DIIS algorithm
- ALGO = Normal & Fast, VeryFast
- NPAR = 1, saves memory
- NPAR = number of compute nodes
- NCORE = cores per compute node (or socket)

I find it easier to use NCORE, e.g. on Tetralith:
NCORE=32

KPAR

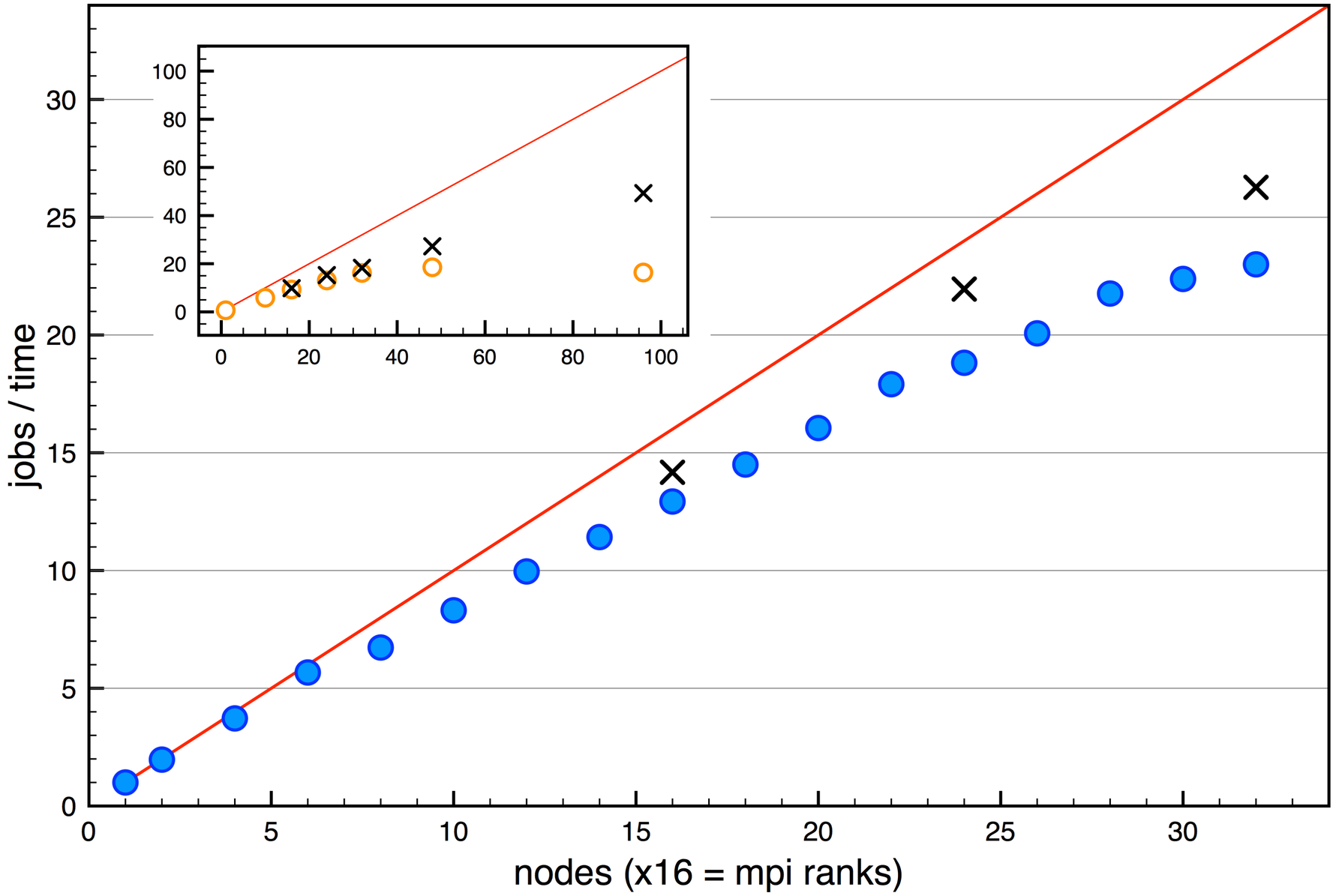
- KPAR = number of k-points treated in parallel
- in particular, good for **hybrid-DFT** jobs
- increase cores at least 2x
- try **KPAR = min (nodes, k-points)**

Si-H/Ag(111) 129 atoms, VASP PBE @Triolith (old)



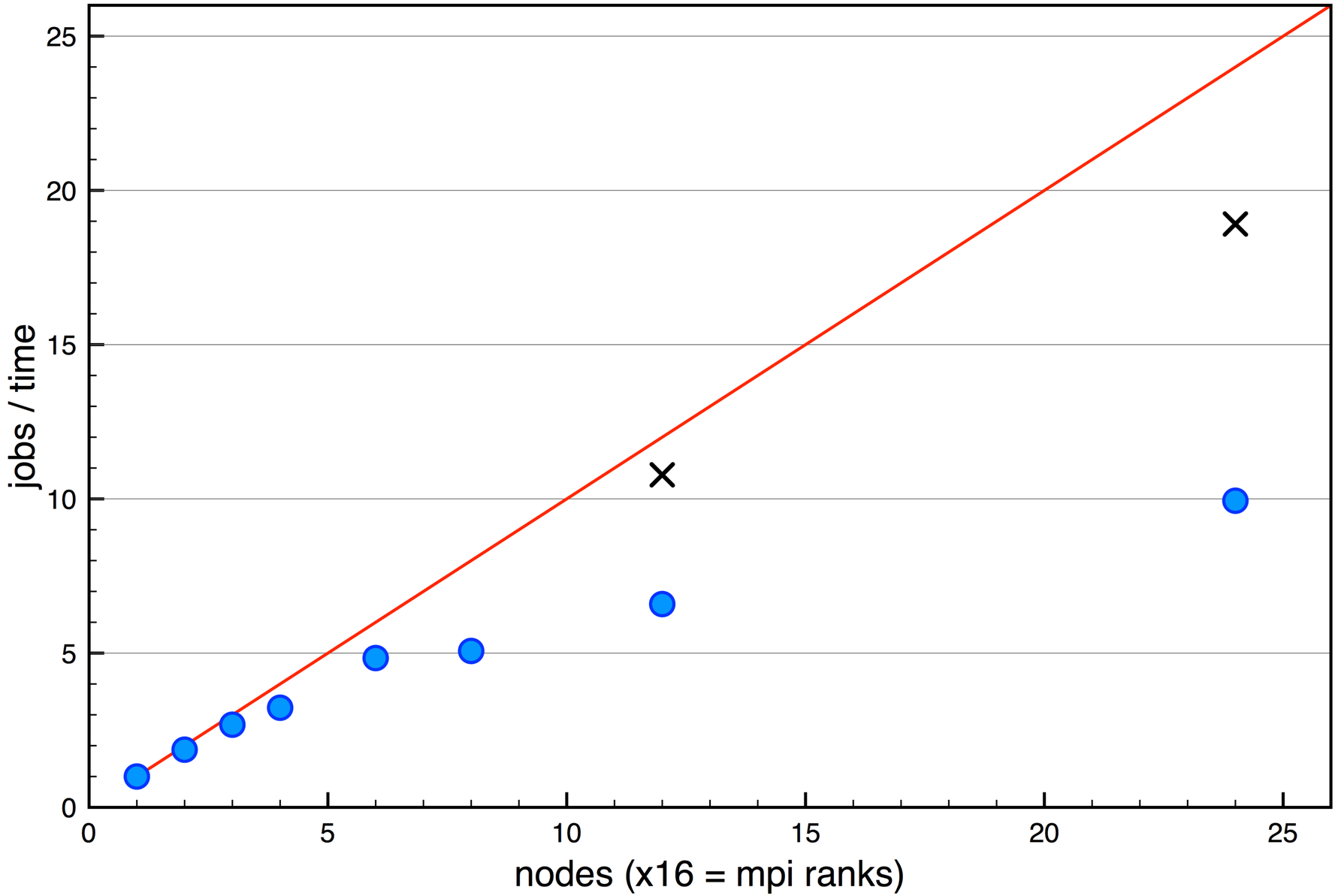
NBANDS=750
4 k-points

GaAsBi 512 atoms, VASP PBE @Triolith (old)



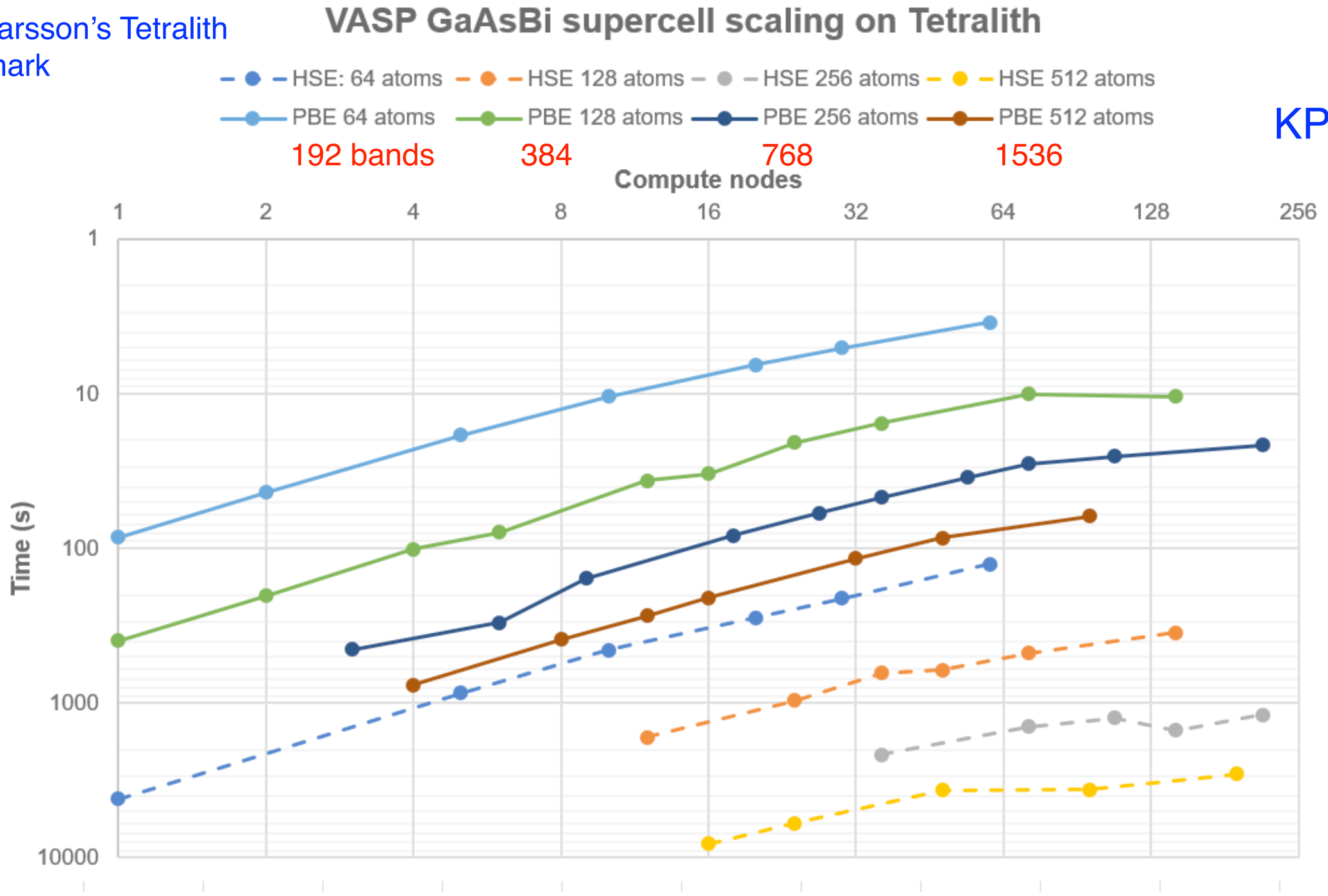
NBANDS=1536
4 k-points

GaAsBi 128 atoms, VASP HSE06 @Triolith (old)



NBANDS=384
12 k-points

Peter Larsson's Tetralith benchmark



As a practical example, let us calculate how many core hours that would be required to run 10,000 full SCF cycles (say 100 geometry optimizations, or a few molecular dynamics simulations). The number of nodes has been chosen so that the parallel efficiency is > 90%:

Atoms	Bands	Nodes	Core hours
64	192	5	8,000
128	384	12	39,000
256	768	18	130,000
512	1536	16	300,000

The same table for 10,000 SCF cycles of HSE06 calculations looks like:

Atoms	Bands	Nodes	Core hours
64	192	10	400,000
128	384	36	2,000,000
256	768	36	6,900,000
512	1536	24	13,000,000

For comparison, a typical large SNAC project might have an allocation of 100,000-1,000,000 core hours per month with several project members, while a smaller personal allocation might be 5,000-10,000 core hours/month. **Thus, while it is technically possible to run very large VASP calculations quickly on Tetralith, careful planning of core hour usage is necessary, or you will exhaust your project allocation.**

Quick comparison

GaAsBi 512 atoms, VASP PBE, NBANDS = 1536, 4 k-points

@Tetralith, 6 nodes, NCORE=32, NSIM=30: 576s
4: 625s

@Beskow, 6 nodes, NCORE=32, NSIM=4: 1074s
24 2: 1593s

GaAsBi 128 atoms, VASP HSE06, NBANDS = 384, 3 k-points

@Beskow, 6 nodes on 24c, NSIM=1: 2146s
on 32c : 2044s

Memory issues

In particular going from PBE to HSE06, GW

2x2x2 k-mesh 4x4x4 k-mesh x 8 k-points

ENCUT = 400 eV ENCUT = 600 eV x 1.8

$$n_{pw} = \propto \texttt{ENCUT}^{3/2}$$

Memory issues …solutions

- Reduce cores/node, e.g. 24c/node, 16c/node
- Fat memory nodes (Tetralith)

```
#SBATCH -C fat
```

- More nodes (and reduce cores)
- Reduce k-mesh, ENCUT?
- Simplify system?

```
#SBATCH --ntasks-per-node=16
```

```
INCAR: NCORE=16
```

VASP problems

- Check output for clues
- Out of memory (OOM)?
- Run using VASP “vanilla” version
- Reduce complexity of INCAR
- POSCAR correct? Check POTCAR

support@nsc.liu.se

- complicated INCAR…
- structure (POSCAR)
- k-mesh (KPOINTS)
- NCORE/NPAR, KPAR
- VASP version
- cores
- memory

support@nsc.liu.se

ALGO=N

- complicated INCAR… simplify & try again!
- structure (POSCAR) reasonable/correct?
- k-mesh (KPOINTS) Γ-centered?
- **NCORE**/~~NPAR~~, KPAR simplify (possibly remove)!
- VASP version try latest (possibly "vanilla" version)!

```
$ module add VASP/5.4.4.16052018-nsc1-intel-2018a-eb
```

- cores too few/many?
- memory
  - ENCUT
  - k-mesh

larger memory nodes:

```
#SBATCH -C fat
```

reduce cores/node:

```
#SBATCH --ntasks-per-node=16
```

```
INCAR: NCORE=16
```

Summary: “rules of thumb”

- job size (max): total cores ≈ NBANDS / 8
- NSIM = 4 (default), (? Tetralith, 2 Beskow)
- NCORE = cores/node (32 Tetralith, 24? Beskow)
- KPAR = min (nodes, k-points)
  - In particular useful for **hybrid-DFT**
- In general, INCAR default settings OK
- GPU: important to increase NSIM

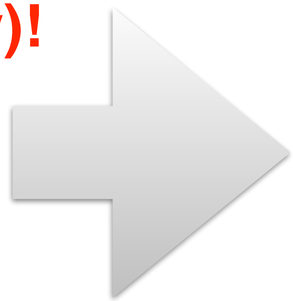
Resources: theory

http://vasp.at/

- Nice presentations by Marsman and Blöchl (click for links)
- Blöchl PRB **50**, 17953 (1994)
- Blöchl *et al.* https://arxiv.org/abs/cond-mat/0201015v2
- Kresse & Joubert PRB **59**, 1758 (1999)
- Holzwarth *et al.* PRB **55**, 2005 (1997)
- Martin, *Electronic Structure*, Chapter 11.1, 13.2

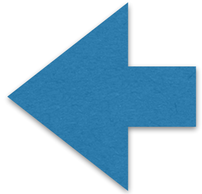
Resources

- Manual
  Read all (really)!
- Wiki
  examples, presentations
- Forum



Find the links:
http://vasp.at/

- Peter Larsson's old blog at NSC:
  https://www.nsc.liu.se/~pla/



info & discussion

Questions? support@nsc.liu.se