PBS equivalent or similar	SLURM directive	Purpose	Example
-l walltime=hhh:mm:ss	--time=hhh:mm:ss	Requests the maximum amount of time that the job can be running. The job will be aborted when this time is reached. If no time is specified, a default value will be assigned (currently 20min).	#SBATCH –-time=15:20:00 requests 15 hours and 20 minutes
–l nodes=N:ppn=n	--nodes=N --tasks-per-node=n	Requests the number of nodes and the number of tasks/processes per node that will be allocated to the job. By default SLURM will allocate 1 core per task, unless --cpus-per-task is specified.	#SBATCH --nodes=10 #SBATCH --tasks-per-node=4 requests 10 nodes with 4 cores each.
N/A	--nodes=N --tasks-per-node=n --cpus-per-task=c	Requests the number of nodes, the number of tasks/processes per node, and the number of cores per task that will be allocated to the job. This job will be allowed to start n processes on each node, but each process will be able to run on c different cores (with c threads for example).	#SBATCH --nodes=10 #SBATCH --tasks-per-node=4 #SBATCH --cpus-per-task=2 requests 10 nodes with 4 tasks/processes each. Each process could use 2 cores. (this job will be allocated with 80 cores in total)
nodes=N:ppn=n:gpus=g	--nodes=N --tasks-per-node=n --gres=gpu:g	Requests the number of nodes, the number of cores per node, and the number of GPUs per node that will be allocated to the job.	#SBATCH --nodes=2 #SBATCH --gres=gpu:1 #SBATCH --tasks-per-node=8 requests 2 nodes with 1 GPU and 8 cores each
-l procs=X	--ntasks=X	Requests the number of tasks/processes that will be allocated to the job in an unspecified number of nodes (SLURM could allocate the cores in any number of nodes from 1 to X so that the job starts as soon as possible.	#SBATCH --ntasks=100 requests 100 cores that will be distributed across the cluster

–l pmem=m	--mem-per-cpu=m	Requests the maximum amount of memory in megabytes that will be assigned to any core in the job. m must be an integer. It can also be requested in gigabytes adding G at the end. Default value is currently 1940MB.	#SBATCH --mem-per-cpu=4G requests 4 gigabytes per core
-l mem=M	N/A		
N/A	--mem=M	Requests the maximum amount of memory in megabytes that must be allocated per node. M must be an integer. It can also be requested in gigabytes adding G at the end.	#SBATCH --mem=18G requests 18 gigabytes of memory on each node allocated to the job
-N <jobname>	--job-name=jobname	Sets the name that your job will have for the scheduler system. This name will be showed, for example, when using qstat or squeue. Default name of the job is the name of the submission script.	#SBATCH --job-name=exp1 sets exp1 as the name of the job
-o <outputname>	--output=outputname	Specifies the name of the file where SLURM will direct the standard output and standard error of the job. Default name of output log file is slurm-<jobID>.out	#SBATCH --output=results1 sets results1 as the name of the output log file
-j oe	N/A	In SLURM, the standard output and error are joined in the same file by default. This can be modified if --error is specified	
N/A	--error=errorname	Specifies the name of the file where SLURM will direct the error output of the job. Default name is slurm-<jobID>.out	#SBATCH --error=errors1 sets errors1 as the name of the error log file

PBS_O_WORKDIR	SLURM_SUBMIT_DIR	This environment variable contains the directory from which the job was submitted. By default, a job will start running on this directory, so there is no need anymore to cd $SLURM_SUBMIT_DIR (or cd $PBS_O_WORKDIR)	
PBS_JOBID	SLURM_JOB_ID	This environment variable contains the ID of the job allocation	
qsub <my_script>	sbatch <my_script>	Submits the job described by the batch script my_script	sbatch experiment1 submits script experiment1
mpirun <my_mpi_code>	srun <my_mpi_code>	srun is the native launcher of parallel processes in slurm. **We strongly recommend to use srun instead of mpirun or mpiexec**. Note that there is no need to specify the number of processes, this info will be obtained from the job request.	#SBATCH --nstaks=25 #SBATCH --cpus-per-task=3 srun vasp will start 25 vasp processes across the cluster, and allows each process to fork into 3 parallel threads
pbsdsh –v <my_script> pbsdsh2 –v <my_script>	srun <my_script>	srun can also run multiple copies of a script or binary in parallel. Use the environment variable SLURM_PROCID (the equivalent of PBS_VNODENUM) to know the number of process assigned to each copy	#SBATCH --nodes=2 #SBATCH --tasks-per-node=8 srun exp1 will run 16 copies of the script exp1 (8 instances on each node). The variable SLURM_PROCID will take values 0,1,2…15
qstat	squeue	Show a list of all jobs in the queues. With squeue the jobs are ordered by priority, not by order of submission as with qstat.	squeue

qstat –f <jobID>	scontrol show job <jobID> scontrol show job <jobID> -dd	List detailed information of the job with identification number jobID	scontrol show job 321 obtains detailed information of job 321
qstat –u <nsid>	squeue –u <nsid>	List all jobs of user nsid	squeue –u abc123 lists all jobs of user abc123 in the queues
showbf showq	N/A	There are no direct equivalence to these commands in SLURM, but rather different options to obtain the same information. Some of these options are described below	
	sprio	Shows the priority of queued jobs and the different factors that compose it.	sprio
	sshare	Shows the usage of an user and its group. It also shows the fairshare value that will be used to calculate the fairshare component of the priority of jobs submitted by the user.	sshare
	sinfo --states=idle	Shows a list of all idle nodes per queue	sinfo --states=idle
	squeue --start	Shows the list of all jobs queued with its **estimated** start times	squeue --start –u abc123 lists all queued jobs of user abc123 with its estimated start times
	sbatch --test-only <my_script>	Shows the estimated start time of the job described in my_script without submitting it.	sbatch --test-only experiment1 determines the estimated start time for the job described in experiment1 as if it were submitted now. The job is never submitted.