Bioinformatics Containerization Tutorial
Session 4: SLURM + Apptainer Workflow (KUACC)

Dr. İsmail K. Sağlam

November 15, 2025

Contents

1 Overview

In this session you will:

- Wrap Apptainer runs inside SLURM batch jobs on KUACC.
- Reuse the same bind layout as in Session 3.
- Run:
  - `01_call_genotypes.sh` (ANGSD) as a SLURM job.
  - `02_pcangsd_pipeline.sh` (PCAngsd + selection) as a SLURM job.
- Optionally chain these jobs using `-dependency`.

Your scripts and `.sif` image remain unchanged; you only add SLURM wrappers.

2 Minimal SLURM + Apptainer structure

Every batch script follows this pattern:

1. SLURM header: job name, partition, CPUs, memory, time.
2. `cd` into `~/oulu`.
3. `module load apptainer/1.4.1`.
4. `apptainer exec` with consistent `-bind` layout:
   - `~/oulu/data` → `/data`
   - `/userfiles/.../new_bams` → `/data/bams`
   - `/userfiles/.../references` → `/data/ref`
   - `~/oulu/results` → `/results`
   - `~/oulu/scripts` → `/workspace/scripts`
5. Execute the appropriate script inside the container.

3 SLURM job for ANGSD genotype calling

Create `~/oulu/hpc/run_angsd_genotypes.sbatch`:

```
#!/bin/bash
#SBATCH --job-name=angsd_genotypes
#SBATCH --output=slurm-angsd-%j.out
#SBATCH --error=slurm-angsd-%j.err
#SBATCH --partition=short
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=8
#SBATCH --mem=32G
#SBATCH --time=08:00:00

# 1) Go to working directory (where sif, data, results, scripts live)
cd ~/oulu

# 2) Load Apptainer module
```

```
module load apptainer/1.4.1

# 3) Run the ANGSD genotypes script inside the container
apptainer exec \
  --bind ~/oulu/data:/data:ro \
  --bind /userfiles/utopalan22/isophya/new_bams:/data/bams:ro \
  --bind /userfiles/utopalan22/isophya/references:/data/ref:ro \
  --bind ~/oulu/results:/results \
  --bind ~/oulu/scripts:/workspace/scripts:ro \
  --pwd /workspace \
  ~/oulu/isophya-course_0.1.sif \
  bash -lc './scripts/01_call_genotypes.sh'
```

3.1 Submit and monitor

From `~/oulu`:

```
cd ~/oulu
sbatch hpc/run_angsd_genotypes.sbatch
```

Example response:

```
Submitted batch job 1234567
```

Monitor:

```
squeue -u iksaglam
tail -f slurm-angsd-1234567.out
tail -f slurm-angsd-1234567.err
```

Inspect results:

```
ls -R ~/oulu/results
```

4 SLURM job for PCAngsd + selection pipeline

Create `~/oulu/hpc/run_pcangsd_pipeline.sbatch`:

```
#!/bin/bash
#SBATCH --job-name=pcangsd_pipeline
#SBATCH --output=slurm-pcangsd-%j.out
#SBATCH --error=slurm-pcangsd-%j.err
#SBATCH --partition=short
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=8
#SBATCH --mem=32G
#SBATCH --time=08:00:00

cd ~/oulu
module load apptainer/1.4.1
```

```
apptainer exec \
  --bind ~/oulu/data:/data:ro \
  --bind /userfiles/utopalan22/isophya/new_bams:/data/bams:ro \
  --bind /userfiles/utopalan22/isophya/references:/data/ref:ro \
  --bind ~/oulu/results:/results \
  --bind ~/oulu/scripts:/workspace/scripts:ro \
  --pwd /workspace \
  ~/oulu/isophya-course_0.1.sif \
  bash -lc './scripts/02_pcangsd_pipeline.sh'
```

Submit:

```
cd ~/oulu
sbatch hpc/run_pcangsd_pipeline.sbatch
```

Check queue and logs:

```
squeue -u iksaglam
tail -f slurm-pcangsd-<jobid>.out
ls -R ~/oulu/results
```

5 Chaining jobs with dependencies

Typical pattern:

1. Submit ANGSD job.
2. Submit PCAngsd job with `afterok` dependency.

Example:

```
cd ~/oulu

# Step 1: submit ANGSD job
jid1=$(sbatch hpc/run_angsd_genotypes.sbatch | awk '{print $4}')
echo "ANGSD job id: $jid1"

# Step 2: submit PCAngsd job that waits for ANGSD to finish successfully
sbatch --dependency=afterok:$jid1 hpc/run_pcangsd_pipeline.sbatch
```

6 Unified SLURM script with task selector (optional)

Instead of two separate files, you can use one script with a variable:

```
#!/bin/bash
#SBATCH --job-name=isophya_task
#SBATCH --output=slurm-isophya-%x-%j.out
#SBATCH --error=slurm-isophya-%x-%j.err
#SBATCH --partition=short
#SBATCH --nodes=1
#SBATCH --ntasks=1
```

```
#SBATCH --cpus-per-task=8
#SBATCH --mem=32G
#SBATCH --time=08:00:00

# TASK can be: angsd or pcangsd
TASK="${TASK:-angsd}"

cd ~/oulu
module load apptainer/1.4.1

case "$TASK" in
  angsd)
    SCRIPT="./scripts/01_call_genotypes.sh"
    ;;
  pcangsd)
    SCRIPT="./scripts/02_pcangsd_pipeline.sh"
    ;;
  *)
    echo "Unknown TASK: $TASK" >&2
    exit 1
    ;;
esac

echo "Running TASK=${TASK} using ${SCRIPT}"

apptainer exec \
  --bind ~/oulu/data:/data:ro \
  --bind /userfiles/utopalan22/isophya/new_bams:/data/bams:ro \
  --bind /userfiles/utopalan22/isophya/references:/data/ref:ro \
  --bind ~/oulu/results:/results \
  --bind ~/oulu/scripts:/workspace/scripts:ro \
  --pwd /workspace \
  ~/oulu/isophya-course_0.1.sif \
  bash -lc "${SCRIPT}"
```

Submit:

```
# ANGSD:
sbatch --export=TASK=angsd hpc/run_isophya_task.sbatch

# PCAngsd:
sbatch --export=TASK=pcangsd hpc/run_isophya_task.sbatch
```

7 Summary

After Session 4 you can:

- Submit containerized analyses as SLURM jobs.
- Use consistent bindings from laptop → Apptainer → SLURM.
- Chain jobs via dependencies to build simple two-step pipelines.