Population Genomics Pipeline - WSL Terminal Commands

Step-by-Step Setup and Execution

1. Initial Setup (Run Once)

```bash
# Create and run the setup script
curl -O https://raw.githubusercontent.com/your-repo/setup.sh
chmod +x setup.sh
./setup.sh
```

OR manually create setup.sh:

```bash
# Create setup script
cat > setup.sh << 'EOF'
# Copy the entire setup.sh content from the artifact above
EOF

chmod +x setup.sh
./setup.sh
```

2. Navigate to Project Directory

```bash
cd population_genomics_pipeline
```

3. Copy Source Code from Artifacts

Copy Cargo.toml:

```bash
# The Cargo.toml is already created by setup script
# Verify it contains all dependencies
cat Cargo.toml
```

Copy Rust source files:

bash

```bash
# Copy generate_data.rs content to:
nano src/bin/generate_data.rs
# Paste the entire generate_data.rs artifact content

# Copy population_analysis.rs content to:
nano src/bin/population_analysis.rs
# Paste the entire population_analysis.rs artifact content
```

Copy Nextflow pipeline:

bash

```bash
# Copy main.nf content:
nano main.nf
# Paste the entire main.nf artifact content
```

4. Compile Rust Code

bash

```bash
# Install Rust dependencies and compile
cargo build --release
```

Expected output:

```
   Compiling population_genomics_pipeline v0.1.0
    Finished release [optimized] target(s) in 45.2s
```

5. Test Pipeline with Small Dataset

bash

```bash
# Quick test run
./scripts/test_pipeline.sh
```

OR manually:

bash

```bash
# Test with minimal data
nextflow run main.nf \
    --individuals 100 \
    --snps 1000 \
    --populations 2 \
    --output_dir test_results
```

6. Run Full Pipeline

```bash
# Standard run
./scripts/run_pipeline.sh
```

OR with custom parameters:

```bash
./scripts/run_pipeline.sh \
    --individuals 2000 \
    --snps 20000 \
    --populations 5 \
    --output results_large
```

OR directly with Nextflow:

```bash
nextflow run main.nf \
    --individuals 1000 \
    --snps 10000 \
    --populations 3 \
    --maf_threshold 0.05 \
    --missing_threshold 0.1 \
    --num_pcs 10 \
    --hwe_test true \
    --population_structure true \
    --output_dir results
```

7. Generate Data Only (for testing)

```bash
# Generate synthetic data only
./scripts/generate_data_only.sh 1000 10000 3

# OR manually
cargo build --release --bin generate_data
./target/release/generate_data \
    --individuals 1000 \
    --snps 10000 \
    --populations 3 \
    --output data
```

8. Run Individual Analysis Components

Data Generation:

```bash
./target/release/generate_data \
    --individuals 1000 \
    --snps 10000 \
    --populations 3 \
    --output data
```

Population Analysis:

```bash
./target/release/population_analysis \
    --genotypes data/genotypes.csv \
    --metadata data/sample_metadata.csv \
    --output results \
    --maf-threshold 0.05 \
    --missing-threshold 0.1 \
    --num-pcs 10 \
    --hwe-test \
    --population-structure
```

Expected Directory Structure After Setup

```
population_genomics_pipeline/
├── Cargo.toml                # ✅ Created by setup
├── main.nf                   # ⚠️  Copy from artifact
├── nextflow.config           # ✅ Created by setup
├── src/bin/
│   ├── generate_data.rs      # ⚠️  Copy from artifact
│   └── population_analysis.rs # ⚠️  Copy from artifact
├── scripts/
│   ├── run_pipeline.sh       # ✅ Created by setup
│   ├── test_pipeline.sh      # ✅ Created by setup
│   └── generate_data_only.sh # ✅ Created by setup
├── data/                     # 📁 Will contain generated data
├── results/                  # 📁 Will contain analysis results
├── target/                   # 📁 Rust compilation output
├── work/                     # 📁 Nextflow work directory
├── README.md                 # ✅ Created by setup
└── LICENSE                   # ✅ Created by setup
```

Output Files Location

After successful run, find results in:

```bash
# Main results directory
ls -la results/

# Quality control
cat results/qc/qc_report.txt

# PCA results
head results/pca/pca_eigenvalues.csv
head results/pca/pca_scores.csv

# Population structure (if enabled)
head results/population_structure/fst_matrix.csv
head results/population_structure/population_allele_frequencies.csv

# Hardy-Weinberg tests (if enabled)
head results/hwe/hwe_tests.csv

# Final report
firefox results/final_report.html  # Or use your preferred browser
```

Troubleshooting Commands

Check system requirements:

```bash
# Check Rust installation
rustc --version
cargo --version

# Check Nextflow installation
nextflow -version

# Check Java installation
java -version

# Check available memory
free -h

# Check disk space
df -h
```

Clean and rebuild:

```bash
# Clean Rust build
cargo clean
cargo build --release

# Clean Nextflow work directory
rm -rf work/
rm -rf .nextflow*

# Clean previous results
rm -rf results/ test_results/ data/
```

Debug Rust compilation issues:

```bash
# Install additional dependencies if needed
sudo apt update
sudo apt install -y build-essential pkg-config libssl-dev libblas-dev liblapack-dev

# Check for specific errors
cargo build --release --bin generate_data 2>&1 | grep error
cargo build --release --bin population_analysis 2>&1 | grep error
```

Monitor pipeline execution:

```bash
# Watch Nextflow execution
tail -f .nextflow.log

# Monitor system resources
htop

# Check individual process logs
ls -la work/*/
```

Performance Tuning

For large datasets, modify nextflow.config:

```bash
nano nextflow.config

# Increase memory allocation:
process {
    memory = '8 GB'
    cpus = 4

    withName:PCA_ANALYSIS {
        memory = '16 GB'
        cpus = 8
    }
}
```

Use cluster execution (if available):

```bash
nextflow run main.nf -profile cluster  # Add cluster profile to nextflow.config
```

Example Complete Workflow

```bash
# 1. Setup (one time)
./setup.sh

# 2. Navigate to project
cd population_genomics_pipeline

# 3. Copy source code from artifacts (manual step)
# Copy each artifact content to respective files

# 4. Test compilation
cargo build --release

# 5. Quick test
./scripts/test_pipeline.sh

# 6. Full analysis
./scripts/run_pipeline.sh --individuals 2000 --snps 20000 --populations 5

# 7. View results
ls -la results/
cat results/summary_statistics.txt
firefox results/final_report.html
```

This pipeline provides a complete, robust foundation for population genomics analysis with synthetic data generation, comprehensive QC, PCA, population structure analysis, and Hardy-Weinberg testing.