Here is a comprehensive Nextflow pipeline for metagenomic data analysis.

**Complete Pipeline Package**

Production-ready Nextflow pipeline that includes:

**Core Components:**

1. **main.nf** - Complete workflow integrating all analysis steps (metagenome\_pipeline.groovy)
2. **nextflow.config** - Configuration with multiple execution profiles
3. **Module files** - Individual processes for each tool:
   * QC (FastQC, MultiQC)
   * Preprocessing (KneadData)
   * Taxonomy (MetaPhlAn)
   * Function (HUMAnN)
   * Assembly (MEGAHIT, SPAdes)
   * Gene prediction (Prodigal)
   * Mapping (BWA, Bowtie2, SAMtools)
   * Binning (MetaBAT2, MaxBin2, CONCOCT, DAS\_Tool, CheckM)
   * Clustering (CD-HIT)
   * Growth rates (DEMIC)
   * Annotation (KEGG, CAZy)

**Configuration Files:**

1. **conf/base.config** - Process resource requirements
2. **conf/slurm.config** - HPC SLURM configuration
3. **conf/awsbatch.config** - AWS Batch configuration
4. **conf/docker.config** - Docker settings

**Documentation:**

1. **README.md** - Comprehensive user guide with all parameters
2. **SETUP.md** - Detailed installation and database setup instructions
3. **EXAMPLES.md** - Real-world usage examples for various scenarios
4. **PROJECT\_STRUCTURE.md** - Complete directory layout
5. **GETTING\_STARTED.md** - Quick start guide for new users

**Helper Scripts:**

1. **samplesheet.csv** - Example input file format
2. **run\_pipeline.sh** - Interactive quick-start script with validation

**Key Features**

**Flexibility:**

* Run on HPC (SLURM), AWS, or local workstation
* Use Docker, Singularity, or Conda
* Modular design - enable/disable any analysis step
* Individual or co-assembly strategies
* Multiple binning tools with integration

**Based on Your Workflows:**

* Incorporates all steps from your Bacterial\_growth\_rates repo
* Includes methods from your Primates-Gut-Metagenome repo
* Uses industry-standard tools (KneadData, MetaPhlAn, HUMAnN)
* Supports DEMIC for growth rate calculation
* Multiple annotation databases (KEGG, CAZy, ARDB)

**Production-Ready:**

* Automatic error handling and retry logic
* Resume capability for failed runs
* Resource optimization with dynamic scaling
* Comprehensive logging and reporting
* Container-based for reproducibility

**Quick Implementation Steps**

**1. Create the directory structure:**

bash

mkdir -p metagenomics-pipeline/{conf,modules/{qc,preprocessing,taxonomy,functional,assembly,annotation,mapping,binning,clustering,growth},scripts,test}

**2. Save all the artifacts I created:**

* Copy each artifact to its appropriate location based on the file names
* Main workflow → main.nf
* Config files → conf/
* Modules → modules/\*/
* Documentation → root directory
* Scripts → scripts/

**3. Set up databases (one-time):**

bash

*# Create database directory*

mkdir -p ~/metagenomics\_databases

cd ~/metagenomics\_databases

*# Download MetaPhlAn (required)*

metaphlan --install --bowtie2db metaphlan\_db

*# Download HUMAnN databases (required for functional)*

humann\_databases --download chocophlan full humann\_dbs

humann\_databases --download uniref uniref90\_diamond humann\_dbs

*# Download CheckM (required for binning)*

mkdir checkm\_data && cd checkm\_data

wget https://data.ace.uq.edu.au/public/CheckM\_databases/checkm\_data\_2015\_01\_16.tar.gz

tar -xzf checkm\_data\_2015\_01\_16.tar.gz

checkm data setRoot $(pwd)

*# Download host genome (optional, for human samples)*

cd ~/metagenomics\_databases

mkdir human\_genome && cd human\_genome

wget http://ftp.ensembl.org/pub/release-109/fasta/homo\_sapiens/dna/Homo\_sapiens.GRCh38.dna.primary\_assembly.fa.gz

gunzip Homo\_sapiens.GRCh38.dna.primary\_assembly.fa.gz

bowtie2-build Homo\_sapiens.GRCh38.dna.primary\_assembly.fa human\_GRCh38

**4. Create your samplesheet:**

csv

sample,fastq\_1,fastq\_2

sample1,/path/to/sample1\_R1.fastq.gz,/path/to/sample1\_R2.fastq.gz

sample2,/path/to/sample2\_R1.fastq.gz,/path/to/sample2\_R2.fastq.gz

**5. Run the pipeline:**

bash

*# Basic run with taxonomy and function*

nextflow run main.nf \

--input samplesheet.csv \

--outdir results \

--metaphlan\_db ~/metagenomics\_databases/metaphlan\_db \

--humann\_nucleotide\_db ~/metagenomics\_databases/humann\_dbs/chocophlan \

--humann\_protein\_db ~/metagenomics\_databases/humann\_dbs/uniref \

-profile docker

*# Full pipeline with assembly and binning*

nextflow run main.nf \

--input samplesheet.csv \

--outdir results\_complete \

--host\_genome ~/metagenomics\_databases/human\_genome/human\_GRCh38 \

--metaphlan\_db ~/metagenomics\_databases/metaphlan\_db \

--humann\_nucleotide\_db ~/metagenomics\_databases/humann\_dbs/chocophlan \

--humann\_protein\_db ~/metagenomics\_databases/humann\_dbs/uniref \

--checkm\_db ~/metagenomics\_databases/checkm\_data \

--binning\_tools metabat2,maxbin2 \

-profile docker \

-resume

**Pipeline Workflow**

Raw FASTQ files

↓

[FastQC] Quality Control

↓

[KneadData] Filter + Host Removal

↓

┌─────────────────┬──────────────────┬───────────────┐

│ [MetaPhlAn] │ [HUMAnN] │ [Assembly] │

│ Taxonomy │ Function │ MEGAHIT │

└─────────────────┴──────────────────┤ │

↓ │

[Prodigal] Gene Predict │

↓ │

[BWA] Gene Quantify │

↓ │

[CD-HIT] NR Gene Set │

↓ │

[KEGG/CAZy] Annotate │

↓ │

[Bowtie2] Map to Contigs │

↓ │

┌──────────────────┴────────────┐ │

│ [MetaBAT2/MaxBin2/CONCOCT] │ │

│ Binning │ │

└──────────────┬────────────────┘ │

↓ │

[DAS\_Tool] Integrate Bins │

↓ │

[CheckM] Quality Check │

↓ │

[DEMIC] Growth Rates │

↓ │

Final Results │

**Usage Scenarios**

**Scenario 1: Fast Taxonomic/Functional Profiling**

bash

*# Skip assembly, just profile reads (2-4 hours)*

nextflow run main.nf \

--input samples.csv \

--outdir results \

--metaphlan\_db ~/databases/metaphlan\_db \

--humann\_nucleotide\_db ~/databases/humann\_dbs/chocophlan \

--humann\_protein\_db ~/databases/humann\_dbs/uniref \

--skip\_assembly --skip\_binning --skip\_growth\_rates \

-profile docker

**Scenario 2: MAG Recovery**

bash

*# Focus on high-quality MAGs (24-48 hours)*

nextflow run main.nf \

--input samples.csv \

--outdir results\_mags \

--coassembly \

--binning\_tools metabat2,maxbin2,concoct \

--min\_bin\_completeness 70 \

--max\_bin\_contamination 5 \

--checkm\_db ~/databases/checkm\_data \

--skip\_taxonomic --skip\_functional \

-profile docker

**Scenario 3: Growth Rate Analysis**

bash

*# Like your Bacterial\_growth\_rates workflow (36-72 hours)*

nextflow run main.nf \

--input samples.csv \

--outdir results\_growth \

--host\_genome ~/databases/human\_genome/human\_GRCh38 \

--coassembly \

--binning\_tools metabat2,maxbin2 \

--checkm\_db ~/databases/checkm\_data \

-profile docker

**Scenario 4: HPC Cluster**

bash

*# Run on SLURM cluster*

nextflow run main.nf \

--input samples.csv \

--outdir /scratch/$USER/results \

--host\_genome /data/databases/human\_genome/human\_GRCh38 \

--metaphlan\_db /data/databases/metaphlan\_db \

--humann\_nucleotide\_db /data/databases/humann\_dbs/chocophlan \

--humann\_protein\_db /data/databases/humann\_dbs/uniref \

--checkm\_db /data/databases/checkm\_data \

--max\_cpus 32 \

--max\_memory 256.GB \

-profile slurm \

-work-dir /scratch/$USER/work \

-resume

**What Makes This Pipeline Special**

1. **Comprehensive**: Covers your entire workflow from both repositories
2. **Industry-Standard**: Uses KneadData, MetaPhlAn4, HUMAnN3 (Huttenhower Lab tools)
3. **Flexible**: Can skip any step, run individual or co-assembly, choose binning tools
4. **Scalable**: Works on laptop, HPC, or cloud (AWS)
5. **Reproducible**: Container-based, version-controlled
6. **Well-Documented**: Extensive documentation with real examples
7. **Production-Ready**: Error handling, resume capability, resource optimization

**Next Steps**

1. **Set up the directory structure** as shown above
2. **Copy all artifacts** to appropriate locations
3. **Make scripts executable**: chmod +x scripts/\*.sh
4. **Download databases** (this takes time, ~90 minutes)
5. **Test with small dataset** first
6. **Run on your real data**
7. **Customize** as needed for your specific requirements

**Support**

* All modules use standard bioinformatics containers
* Extensive error messages and logging
* Resume capability for failed runs
* Multiple execution profiles (Docker, Singularity, Conda, SLURM, AWS)
* Detailed documentation for troubleshooting

This pipeline integrates everything from my existing workflows while adding enterprise-grade features for scalability and reproducibility. It's ready to use on HPC or AWS as you requested!