Testing Guide for the GenPPI Python Interface

This document provides a simplified guide to installing and testing the GenPPI Python interface. It focuses on the most common use cases.

1 Environment Setup

Follow these steps to set up your test environment.

1.1 Installation (Development Mode)

To test and modify the code, install the package in development mode. This allows your source code changes to be reflected immediately without needing to reinstall.

```
# Navigate to the project's root directory (where setup.py is)
cd /path/to/genppi_py

# Install in development mode
pip install -e .
```

1.2 Installation (Production Version)

To install the stable version from TestPyPI:

```
# Install version 0.1.2 from TestPyPI (with dependencies from official PyPI)
pip install --index-url https://test.pypi.org/simple/ --extra-index-url https://pypi.org/simple/ genppi-py==0.1.2

# Verify critical dependencies
python -c "import py7zr; print('py7zr:', py7zr.__version__)"
python -c "import multivolumefile; print('multivolumefile OK')"
```

Important Note: Version 0.1.2 exclusively uses the py7zr library to extract multi-volume `model.7z` files. The dependencies `py7zr>=0.20.0` and `multivolumefile>=0.2.3` are installed automatically.

Note on TestPyPI: Since TestPyPI does not host all dependencies, you must use the `-extra-index-url` flag. This allows pip to fetch dependencies (like py7zr and multivolumefile) from the official PyPI.

1.3 Downloading Test Data

The package includes a command to easily download the required sample files for testing.

```
# This command will create a 'samples' folder in your current directory
genppi-download-samples
```

This will download the necessary protein files and save them in the `samples/` folder.

1.4 Verifying the Installation

After installation, the `genppi.py` command should be available in your terminal. Check it by running the help command:

```
genppi.py --help
```

This should display a list of all available parameters, confirming a successful installation.

2 Running the Main Tests

GenPPI is designed to be user-friendly. The following tests cover over 90% of use cases.

2.1 Test 1: Standard Execution (Most Common)

The most common usage is to simply provide the directory with the protein files. **GenPPI automatically runs conserved neighborhood and phylogenetic profiles (Method 1) when multiple genomes are present.**

```
# Run GenPPI on the sample data (CN + PP Method 1 are automatic)
genppi.py -dir samples

# Run with gene fusion as well
genppi.py -dir samples -genefusion
```

The program will process the files and generate results in the `samples/` folder.

2.2 Test 2: Running with Machine Learning (Higher Accuracy)

For a more accurate analysis, you can enable the Machine Learning model with the `-ml` flag. This is the second most common use case.

Prerequisite: The `model.dat` file must be available. On first run, GenPPI will try to download and extract it automatically. You can also download it manually:

```
genppi-download-model
```

Running the test:

```
# Run GenPPI with the Machine Learning model
genppi.py -dir samples -ml
```

2.3 Test 3: Testing Phylogenetic Profile Methods

GenPPI offers 7 different methods for phylogenetic profile prediction. Method 1 is the default, but you can choose others using `-ppmethod`:

```
# Method 1: No filters (default - these commands are equivalent)
genppi.py -dir samples
genppi.py -dir samples -ppmethod 1

# Method 2: Only conserved neighborhood interactions
genppi.py -dir samples -ppmethod 2

# Method 5: Filter by threshold (requires additional parameters)
genppi.py -dir samples -ppmethod 5 -threshold 5 -plusminus ">"
```

2.4 Test 4: Running with Low Memory Usage (`-use-db`)

If you are working with many genomes or on a machine with limited memory, use the `-use-db` flag. This makes GenPPI save intermediate data to disk instead of keeping it in memory.

```
# Combine with standard execution or with -ml
genppi.py -dir samples -ml --use-db
```

3 Verifying the Results

After running any of the tests above, check that the result files were created correctly.

```
# List the report and network files
ls -l samples/ppi-report/
ls -l samples/ppi-files/
```

You should see a `report.txt` file and several network files (like `.sif` and `.dot`) inside these folders.

4 Testing Automatic Recovery

An important feature of the package is its ability to download missing components.

4.1 Testing the Executable Download

1. **Remove the executable:**

```
# (Adjust the path if your structure is different)
rm -f genppi_py/genppi_py/bin/genppi32g-Linux

```

2. **Run a simple command:**

```
genppi.py --help

```

The package should silently download the missing executable and then display the help message.

4.2 Testing the ML Model Download

1. **Remove the model file:**

```
# (Adjust the path if your structure is different)
rm -f genppi_py/genppi_py/bin/model.dat

```

2. **Run a command that requires the model:**

```
genppi.py -dir samples -ml

```

You will see messages indicating that `model.dat` was not found and that the download and extraction will begin.

5 Common Troubleshooting

If something goes wrong, check the following points:

- **Permission Issues (Linux/macOS):** If you get a "Permission denied" error, the executables may lack execution permissions. Fix it with:

```
chmod +x genppi_py/genppi_py/bin/genppi*

```

- **Failed to Extract `model.dat`:** Version 0.1.2 uses the py7zr library for multi-volume extraction. If automatic extraction fails:

  1. **Check py7zr installation:**

```
# Check if py7zr is installed correctly
python -c "import py7zr; print('py7zr version:', py7zr.
    __version__)"

# If needed, reinstall a specific version
pip install --upgrade --force-reinstall "py7zr>=0.20.0,<1.0.0"

```

  2. **Manual extraction:** If automatic extraction still fails, extract it manually:

```
# Download the 7z file parts manually
wget https://github.com/santosardr/genppi/raw/master/src/model
    .7z.001
wget https://github.com/santosardr/genppi/raw/master/src/model
    .7z.002
wget https://github.com/santosardr/genppi/raw/master/src/model
    .7z.003

# Extract using py7zr
python -c "
import py7zr
from multivolumefile import MultiVolume
with MultiVolume('model.7z', mode='rb') as vol:
    with py7zr.SevenZipFile(vol, mode='r') as archive:
        archive.extractall(path='.')
"

```

  3. After manual extraction, move the `model.dat` file to the package's bin directory.

With this guide, you can quickly and efficiently validate the core functionalities of your GenPPI package.