1. Verify dependencies of read\_dewesoft.py and check with dependencies/requirements.txt

>File\_utils.py dependencies and visualization/plots.py

1. Bear in mind, that “correct metadata time” is now part of the read\_dewesoft and might cause trouble later when called but not existent. Function call needs to be deleted. Check the return and the function call from main
2. .gitignore is somewhere and I used mv gitingore.txt .gitignore … and I repasted gitignore to check the content because .gitignore is not visible anymore
3. …

File\_utils.py dependencies and visualization/plots.py

**6. Repository verknüpfen und hochladen**

bash

*# GitHub Repository als remote hinzufügen*

git remote add origin https://github.com/YOURUSERNAME/ice-experiment-analyzer.git

*# Branch umbenennen (falls nötig)*

git branch -M main

*# Zum GitHub Repository hochladen*

git push -u origin main

**Nach dem Upload zu erledigen:**

**7. GitHub Repository konfigurieren**

**Repository Settings:**

* Gehen Sie zu Settings → General
* Features: Issues, Wiki aktivieren
* Branches: Branch protection für main aktivieren

**Secrets für CI/CD:**

* Settings → Secrets and variables → Actions
* Fügen Sie PYPI\_API\_TOKEN hinzu (wenn Sie später auf PyPI veröffentlichen wollen)

**8. Documentation verbessern**

bash

*# Weitere Dokumentation erstellen*

echo "# Installation Guide" > docs/installation.md

echo "# Usage Examples" > docs/usage.md

echo "# API Reference" > docs/api\_reference.md

**9. Tests hinzufügen**

**tests/test\_data\_processing.py:**

python

import pytest

from src.ice\_analyzer.data\_processing.sbe\_reader import read\_sbe\_data

def test\_read\_sbe\_data():

*# Ihre Tests hier*

pass

**10. Beispiel-Scripts erstellen**

**examples/run\_analysis.py:**

python

*#!/usr/bin/env python3*

"""

Example script showing how to use the Ice Experiment Analyzer

"""

from src.ice\_analyzer.main import IceExperimentAnalyzer

def main():

*# Example usage*

analyzer = IceExperimentAnalyzer("/path/to/your/data")

analyzer.run\_analysis(skip\_existing=True)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Langfristige Verbesserungen:**

**11. Package auf PyPI veröffentlichen**

bash

*# Package bauen*

python -m build

*# Auf Test-PyPI hochladen (zuerst testen)*

twine upload --repository testpypi dist/\*

*# Auf echtes PyPI hochladen*

twine upload dist/\*

**12. Continuous Integration verbessern**

* Code coverage Berichte
* Automatische Dokumentation mit Sphinx
* Performance Benchmarks

**13. Community Features**

* CONTRIBUTING.md erstellen
* Issue templates
* Pull request templates
* Code of conduct

**Wartung und Updates:**

**Regelmäßige Aufgaben:**

1. **Dependencies aktualisieren:**

bash

pip list --outdated

pip install --upgrade package\_name

1. **Tests laufen lassen:**

bash

pytest tests/ --cov=src/ice\_analyzer

1. **Code formatieren:**

bash

black src/ tests/

flake8 src/ tests/

1. **Neue Releases erstellen:**

bash

git tag v1.1.0

git push origin v1.1.0

**Checklist vor GitHub Upload:**

* Alle Ihre Funktionen in separate Module aufgeteilt
* Import-Statements in main.py angepasst
* Pfade in config.yaml angepasst
* README.md mit Ihren Daten personalisiert
* setup.py mit Ihren Informationen aktualisiert
* .gitignore überprüft
* Erste Tests geschrieben
* Git Repository initialisiert

**Tipp:** Beginnen Sie mit einer minimalen Version und erweitern Sie schrittweise. Das wichtigste ist, dass Ihr Code funktioniert und gut dokumentiert ist!

The key fingerprint is:

SHA256:IxKKkoEFVynJGGeqhfaW9S9JA1VHVoD9st+xu5jE22c lea@felea.de

The key's randomart image is:

+--[ED25519 256]--+

|o==o.. ...+=o. |

|oB+ . . .o. |

|=....o . |

|o=..o.o . . |

|= .+. .+S o |

|. . ...+. .. . |

| o . .o. o|

| . ..=oE|

| + =+|

+----[SHA256]-----+