Project Tasks - Sinusoidal Peak Detection

Project: Peak Detection using Convolution
Author: Yair Levi
Last Updated: October 8, 2025

📊 Project Status Overview

Phase	Status	Completion	Priority
Phase 1: Core Implementation	✅ Complete	100%	High
Phase 2: Documentation	✅ Complete	100%	High
Phase 3: Testing & Validation	🔄 In Progress	60%	High
Phase 4: Enhancements	📋 Planned	0%	Medium
Phase 5: Deployment	📋 Planned	0%	Low

Legend: ✅ Complete | 🔄 In Progress | 📋 Planned | ⏸️ On Hold | ❌ Blocked

---

🎯 Sprint 1: Core Functionality (COMPLETE)

✅ Completed Tasks

- ☑ **TASK-001**: Implement sinusoidal signal generation
  - **Status**: ✅ Complete
  - **Assignee**: Yair Levi
  - **Completed**: Oct 8, 2025
  - **Details**: Created 10-cycle sine wave with 200 samples/cycle
  - **Validation**: Signal verified with correct amplitude and frequency
- ☑ **TASK-002**: Implement peak template extraction
  - **Status**: ✅ Complete
  - **Assignee**: Yair Levi
  - **Completed**: Oct 8, 2025
  - **Details**: Extract 30-sample window centered on first peak
  - **Validation**: Template contains correct peak shape
- ☑ **TASK-003**: Implement convolution algorithm
  - **Status**: ✅ Complete
  - **Assignee**: Yair Levi
  - **Completed**: Oct 8, 2025
  - **Details**: Used NumPy convolve with 'same' mode
  - **Validation**: Output length matches input signal
- ☑ **TASK-004**: Implement peak detection logic
  - **Status**: ✅ Complete
  - **Assignee**: Yair Levi
  - **Completed**: Oct 8, 2025
  - **Details**: Local maxima detection with threshold filtering
  - **Validation**: All 10 peaks detected correctly
- ☑ **TASK-005**: Create visualization plots

    - **Status**: ✅ Complete
    - **Assignee**: Yair Levi
    - **Completed**: Oct 8, 2025
    - **Details**: Three-panel plot with signal, template, and results
    - **Validation**: Plots render correctly with proper labels
- ☑ **TASK-006**: Add console output and statistics
    - **Status**: ✅ Complete
    - **Assignee**: Yair Levi
    - **Completed**: Oct 8, 2025
    - **Details**: Print peak counts, locations, and spacing
    - **Validation**: All metrics displayed correctly

📝 Sprint 2: Documentation (COMPLETE)

✅ Completed Tasks

- ☑ **TASK-007**: Create Product Requirements Document (PRD)
    - **Status**: ✅ Complete
    - **Assignee**: Yair Levi
    - **Completed**: Oct 8, 2025
    - **Deliverable**: PRD.md with full specifications
    - **Notes**: Includes functional requirements, use cases, and success metrics
- ☑ **TASK-008**: Create README.md documentation
    - **Status**: ✅ Complete
    - **Assignee**: Yair Levi
    - **Completed**: Oct 8, 2025
    - **Deliverable**: README.md with installation and usage instructions
    - **Notes**: Comprehensive guide for users and contributors
- ☑ **TASK-009**: Create TASKS.md file
    - **Status**: ✅ Complete
    - **Assignee**: Yair Levi
    - **Completed**: Oct 8, 2025
    - **Deliverable**: Task tracking document
    - **Notes**: Current document tracking all project tasks
- ☑ **TASK-010**: Add inline code comments
    - **Status**: ✅ Complete
    - **Assignee**: Yair Levi
    - **Completed**: Oct 8, 2025
    - **Details**: Document key algorithm steps and parameters
    - **Validation**: Code is well-commented and readable

🧪 Sprint 3: Testing & Validation (IN PROGRESS)

🔄 In Progress

- ☑ **TASK-011**: Manual testing with default parameters
    - **Status**: ✅ Complete
    - **Assignee**: Yair Levi
    - **Priority**: High

  - **Due Date**: Oct 9, 2025
  - **Details**: Verify correct peak detection with standard settings
  - **Test Cases**: 10 cycles, 200 samples/cycle, 30-sample template
- □ **TASK-012**: Edge case testing
  - **Status**: 🔄 In Progress
  - **Assignee**: Yair Levi
  - **Priority**: High
  - **Due Date**: Oct 12, 2025
  - **Test Cases**:
    - Single cycle signal
    - Very high sampling rate (1000 samples/cycle)
    - Very low sampling rate (50 samples/cycle)
    - Different template sizes (10, 50, 100 samples)

📋 Planned

- □ **TASK-013**: Create unit tests
  - **Status**: 📋 Planned
  - **Priority**: Medium
  - **Due Date**: Oct 15, 2025
  - **Details**: Write pytest unit tests for all functions
  - **Coverage Goal**: >80%
  - **Tests Needed**:
    - Signal generation validation
    - Template extraction correctness
    - Peak detection accuracy
    - Edge case handling
- □ **TASK-014**: Performance benchmarking
  - **Status**: 📋 Planned
  - **Priority**: Medium
  - **Due Date**: Oct 16, 2025
  - **Metrics**:
    - Execution time for various signal lengths
    - Memory usage profiling
    - Scalability testing (up to 100k samples)
  - **Tools**: cProfile, memory_profiler
- □ **TASK-015**: Cross-platform testing
  - **Status**: 📋 Planned
  - **Priority**: Low
  - **Due Date**: Oct 18, 2025
  - **Platforms**:
    - Windows 10/11
    - macOS (Intel and Apple Silicon)
    - Linux (Ubuntu 20.04+)

🚀 Sprint 4: Enhancements (PLANNED)

📋 Feature Enhancements

- □ **TASK-016**: Add noise handling capability
  - **Status**: 📋 Planned

  - **Priority**: High
  - **Effort**: 8 hours
  - **Description**: Support signal detection with Gaussian noise
  - **Acceptance Criteria**:
    - Detect peaks with SNR down to 10dB
    - Add noise level parameter
    - Update visualization to show noisy signal
- ☐ **TASK-017**: Implement adaptive threshold
  - **Status**: 📋 Planned
  - **Priority**: Medium
  - **Effort**: 6 hours
  - **Description**: Auto-calculate optimal detection threshold
  - **Algorithm**: Use statistical methods (mean + N*std)
  - **Validation**: Compare with manual threshold results
- ☐ **TASK-018**: Add multi-frequency support
  - **Status**: 📋 Planned
  - **Priority**: Medium
  - **Effort**: 10 hours
  - **Description**: Detect peaks in signals with multiple frequencies
  - **Requirements**:
    - Support 2-5 simultaneous frequencies
    - Separate detection for each frequency
    - Color-coded visualization
- ☐ **TASK-019**: Export results functionality
  - **Status**: 📋 Planned
  - **Priority**: Low
  - **Effort**: 4 hours
  - **Formats**: CSV, JSON, Excel
  - **Data**: Peak locations, amplitudes, timestamps
  - **File Naming**: Automatic timestamp-based naming
- ☐ **TASK-020**: Create configuration file support
  - **Status**: 📋 Planned
  - **Priority**: Low
  - **Effort**: 5 hours
  - **Format**: YAML or JSON
  - **Parameters**: All configurable values
  - **Validation**: Schema validation for config files

📋 Code Quality Improvements

- ☐ **TASK-021**: Refactor code into modular functions
  - **Status**: 📋 Planned
  - **Priority**: Medium
  - **Effort**: 6 hours
  - **Modules**:
    - `signal_generator.py`: Signal generation functions
    - `template_extractor.py`: Template operations
    - `peak_detector.py`: Detection algorithms
    - `visualizer.py`: Plotting functions
    - `utils.py`: Helper functions
- ☐ **TASK-022**: Add type hints throughout codebase
  - **Status**: 📋 Planned
  - **Priority**: Low

    - **Effort**: 3 hours
    - **Tool**: mypy for static type checking
    - **Coverage**: All function signatures
- ☐ **TASK-023**: Set up linting and formatting
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 2 hours
    - **Tools**:
        - black (code formatting)
        - flake8 (linting)
        - isort (import sorting)
- ☐ **TASK-024**: Add logging system
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 4 hours
    - **Library**: Python logging module
    - **Levels**: DEBUG, INFO, WARNING, ERROR
    - **Output**: Console and optional file logging

🎨 Sprint 5: User Experience (PLANNED)

📋 UX Improvements

- ☐ **TASK-025**: Create command-line interface (CLI)
    - **Status**: 📋 Planned
    - **Priority**: Medium
    - **Effort**: 8 hours
    - **Library**: argparse or Click
    - **Arguments**:
        - `--cycles`: Number of cycles
        - `--samples`: Samples per cycle
        - `--template-size`: Template window size
        - `--output`: Output file path
        - `--no-plot`: Disable visualization
- ☐ **TASK-026**: Add interactive mode
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 10 hours
    - **Features**:
        - Adjust parameters with sliders
        - Real-time visualization updates
        - Export current view
    - **Library**: matplotlib widgets or Plotly Dash
- ☐ **TASK-027**: Create GUI application
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 20 hours
    - **Framework**: PyQt5 or Tkinter
    - **Features**:
        - Parameter input forms
        - Real-time plot display
        - File import/export

        - Settings persistence
- ☐ **TASK-028**: Add progress indicators
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 2 hours
    - **Library**: tqdm
    - **Use Cases**: Large signal processing, batch operations

---

📦 Sprint 6: Deployment & Distribution (PLANNED)

📋 Deployment Tasks

- ☐ **TASK-029**: Create requirements.txt
    - **Status**: 📋 Planned
    - **Priority**: High
    - **Effort**: 1 hour
    - **Include**: All dependencies with version constraints
- ☐ **TASK-030**: Set up virtual environment documentation
    - **Status**: 📋 Planned
    - **Priority**: Medium
    - **Effort**: 2 hours
    - **Guides**: venv, conda, and poetry setups
- ☐ **TASK-031**: Create setup.py for pip installation
    - **Status**: 📋 Planned
    - **Priority**: Medium
    - **Effort**: 3 hours
    - **Goal**: Enable `pip install peak-detection`
    - **Metadata**: Complete package information
- ☐ **TASK-032**: Publish to PyPI (test and production)
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 4 hours
    - **Steps**:
        1. Test on TestPyPI
        2. Create distribution packages
        3. Upload to PyPI
        4. Verify installation
- ☐ **TASK-033**: Create Docker container
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 5 hours
    - **Base Image**: python:3.9-slim
    - **Include**: All dependencies and example data
- ☐ **TASK-034**: Set up CI/CD pipeline
    - **Status**: 📋 Planned
    - **Priority**: Low
    - **Effort**: 8 hours
    - **Platform**: GitHub Actions
    - **Stages**:
        - Automated testing
        - Code quality checks

    - Build verification
    - Deployment automation

🐛 Bug Fixes & Issues

Open Issues

- ☐ **ISSUE-001**: Template positioning edge cases
  - **Status**: 📋 Open
  - **Priority**: Medium
  - **Reported**: Oct 8, 2025
  - **Description**: Template may not work for very short signals (<1 cycle)
  - **Workaround**: Validate minimum signal length
  - **Assigned**: TBD

Closed Issues

None yet.

📅 Milestones

Milestone	Target Date	Status	Progress
M1: Core Implementation	Oct 8, 2025	✅ Complete	100%
M2: Documentation	Oct 8, 2025	✅ Complete	100%
M3: Testing & Validation	Oct 18, 2025	🔄 In Progress	60%
M4: Feature Enhancements	Nov 1, 2025	📋 Planned	0%
M5: Version 1.0 Release	Nov 15, 2025	📋 Planned	0%
M6: PyPI Publication	Nov 30, 2025	📋 Planned	0%

🎯 Future Roadmap

Version 2.0 (Q1 2026)

- Real-time signal processing
- Machine learning-based peak detection
- GPU acceleration for large datasets
- Web-based interface
- Multi-channel signal support

Version 3.0 (Q3 2026)

- Hardware integration (DAQ devices)
- Cloud processing capabilities
- Mobile app (iOS/Android)
- Advanced analytics dashboard

📊 Metrics & KPIs

Code Metrics

- **Lines of Code**: ~150 (current)
- **Test Coverage**: 0% (target: 80%)
- **Code Quality Score**: TBD (target: A)
- **Documentation Coverage**: 100%

Performance Metrics

- **Execution Time**: <2s for 2000 samples
- **Memory Usage**: <100MB
- **Peak Detection Accuracy**: 100% (clean signal)

Project Metrics

- **Open Tasks**: 29
- **Completed Tasks**: 11
- **Overall Progress**: 27.5%
- **Issues Open**: 1
- **Contributors**: 1

💡 Ideas & Suggestions

Community Suggestions

Add community-submitted ideas here

Technical Debt

- ☐ Refactor monolithic script into modules
- ☐ Add comprehensive error handling
- ☐ Improve variable naming consistency
- ☐ Add input validation

Research Topics

- Wavelet-based peak detection comparison
- FFT-based frequency analysis integration
- Adaptive template selection algorithms
- Performance optimization with Cython/Numba

📝 Notes

Development Environment

- **Python Version**: 3.9+

- **IDE**: VS Code / PyCharm
- **Version Control**: Git
- **Repository**: GitHub

Team Contacts

- **Project Lead**: Yair Levi
- **Contributors**: Open for contributions
- **Issue Tracking**: GitHub Issues

Meeting Schedule

- **Stand-up**: N/A (solo project)
- **Sprint Review**: End of each milestone
- **Retrospective**: After major releases

Last Review: October 8, 2025
Next Review: October 15, 2025
Document Owner: Yair Levi