BCILattice — User Manual

Complete guide to using the BCILattice research platform

BCINexus Platform · v1.0 · 2026

1. Introduction

BCILattice is a professional desktop application for Brain-Computer Interface (BCI) and neuroscience research. It provides every step of the research workflow — from raw signal import through preprocessing, paradigm design, machine learning, experiment tracking, and report generation — in a single integrated environment that runs entirely on your own machine.

This manual covers all modules of BCILattice and the BCINexus cloud platform. Whether you are running your first EEG study or managing a multi-site clinical trial, this document provides step-by-step guidance for every part of the workflow.

1.1 How This Manual is Organised

2. Installation & First Launch

2.1 System Requirements

Component	Minimum	Recommended
Operating System	Windows 10 / macOS 12 / Ubuntu 20.04	Windows 11 / macOS 14 / Ubuntu 22.04
CPU	4-core x86-64	8-core x86-64
RAM	8 GB	16 GB or more
GPU	Not required (CPU fallback)	NVIDIA CUDA GPU, 8 GB VRAM
Storage	10 GB free	100 GB+ SSD
Network	None (fully offline)	For cloud sync and community

2.2 Download

Go to bcinexus.io/download and select your operating system. The installer is approximately 800 MB and includes all Python dependencies and all required components — no separate installations required.

2.3 Windows Installation

- Run BCILattice-Setup.exe as a normal user (no administrator rights required).
- Accept the licence agreement and choose an install location.
- Allow all required components to install when prompted.
- Click Install and wait for completion (~3 minutes).
- Launch BCILattice from the Start menu or desktop shortcut.

2.4 macOS Installation

- Open BCILattice.dmg and drag BCILattice.app to Applications.
- On first launch: right-click the app → Open (required by macOS Gatekeeper).
- Allow BCILattice through the Firewall when prompted.

2.5 Linux Installation

- Make the AppImage executable: chmod +x BCILattice-installer.AppImage
- Run: ./BCILattice-installer.AppImage

2.6 First Launch

When BCILattice opens for the first time BCILattice initialises and performs a quick self-check. You will be invited to create a free BCINexus account. This is optional — all core features work offline — but an account enables cloud sync and community access.

> *i BCILattice never uploads data automatically. An account is only required for cloud features you explicitly enable.*

3. Data Manager

The Data Manager is your entry point for every research session. It handles importing, previewing, and organising raw signal files from any supported acquisition system.

3.1 Supported File Formats

Modality	Formats	Notes
EEG	EDF, BDF, GDF, FIF, VHDR, SET, CNT, CSV, XLSX	All major clinical and research EEG systems
fNIRS	SNIRF, FIF, TXT, CSV, XLSX	SNIRF is the international standard format
EMG	EDF, BDF, GDF, FIF, TXT, CSV	EMG from any clinical recorder

3.2 Importing a Single File

- Click Data Manager in the left sidebar.
- Click Import Files and select your EEG, fNIRS, or EMG file.
- BCILattice previews the file: raw signal on the left, file metadata on the right.
- Select the label column (if your file contains class labels for supervised learning).
- Click Add to Session.

3.3 Importing a Multi-Subject Folder

For studies with many subjects stored in a folder hierarchy:

- Click Import Folder and select the root directory.
- Set the folder depth (1–5 levels) matching your directory structure.
- BCILattice recursively scans and lists every compatible file.
- Configure a session name and label column, then click Import All.
- A live progress bar tracks the batch import.

3.4 Signal Preview

Each imported file offers a dual preview: raw signal on the left panel, and a processed preview on the right (after preprocessing is applied). Use the preview to verify that the correct channels are loaded and that the signal looks as expected before running any analysis.

3.5 NeuralFlow Label Sync

If you have designed a paradigm in NeuralFlow, set the label source to 'NeuralFlow Paradigm' in the label selector. BCILattice will use the paradigm timeline to generate labels automatically, aligned to the signal sampling rate.

4. Preprocessing

The Preprocessing module provides a visual, no-code interface for applying research-grade signal cleaning. Every parameter you configure is saved in the session file so that the exact pipeline can be reproduced months later.

4.1 EEG Preprocessing

Setting	Options	Typical Value
Bandpass filter	Low cutoff (Hz) and High cutoff (Hz)	0.5 – 40 Hz
Notch filter	50 Hz or 60 Hz + harmonics (up to 2)	50 Hz (Europe) / 60 Hz (USA)
Re-referencing	CAR, Linked Mastoids, Cz, None	CAR
Resampling	Target sampling rate (Hz)	250 Hz

4.2 fNIRS Preprocessing Pipeline

The fNIRS pipeline runs automatically in sequence when enabled:

- TDDR — Temporal Derivative Distribution Repair for motion artefact correction
- Optical Density conversion
- Z-score detrending (baseline drift removal)
- Short-Channel Regression (SSR) for superficial signal removal
- Beer-Lambert Law conversion to oxyhaemoglobin (HbO) and deoxyhaemoglobin (HbR)

4.3 EMG Preprocessing

- Full-wave rectification — converts negative signal values to positive
- Envelope extraction — smoothed amplitude envelope for event detection

4.4 Custom Python Code Override

Every preprocessing step can be overridden with a custom Python function. Click the 'Custom Code' toggle in any preprocessing step and write your function in the built-in code editor. The function receives the MNE Raw object and must return it:

```
def custom_process(raw):
    # raw is an MNE Raw object
    raw.filter(1.0, 45.0)
    return raw
```

4.5 Before/After Preview

Click Preview after configuring the pipeline to see the raw and processed signals side by side. The preview runs in a background worker so it does not block the UI.

5. NeuralFlow — Visual Paradigm Designer

NeuralFlow is a drag-and-drop block-and-port canvas for designing BCI paradigms without writing code. Blocks represent signal sources, processing steps, and output targets. Typed ports prevent incompatible connections at design time.

5.1 Opening NeuralFlow

- Click Neural Flow in the left sidebar.
- The canvas opens with the block palette on the left and the property inspector on the right.

5.2 Adding Blocks

- Drag a block from the palette onto the canvas, or double-click the canvas to open the block search.
- Available block types include: EEG Source, Bandpass, CAR Reference, FBCSP, LDA, Output, and any blocks from installed community paradigms.

5.3 Connecting Blocks

- Click an output port (right side of a block) and drag to an input port (left side of another block).
- Ports are colour-coded by data type. Incompatible types cannot be connected — BCILattice prevents the connection and shows an error in the inspector.
- Delete a connection by clicking it and pressing Delete.

5.4 Block Properties

Click any block to open it in the Property Inspector panel on the right. Every configurable parameter appears here with a type-appropriate input field (text field, number slider, dropdown). Changes take effect immediately.

5.5 Compiling a Paradigm

- Click Compile in the top toolbar.
- BCILattice validates all port connections and required parameters.
- Errors are listed in the inspector — fix each one and re-compile.
- A successful compile produces a validated paradigm ready for export or use in the ML pipeline.

5.6 Exporting

- Binary .nflow — compact, version-controlled format for local storage and team sharing.
- JSON — human-readable format for debugging or external tool integration.

5.7 Importing a Community Paradigm

Browse community paradigms at bcinexus.io/pipelines. Click Import into BCILattice to open a bcilattice:// deep link. BCILattice opens and loads the paradigm directly onto the NeuralFlow canvas. You can then customise it for your own study.

5.8 Timeline and Label Generation

The Timeline view (below the canvas) shows the temporal structure of the paradigm. Set the label generation rate (samples per second) to align labels with your signal sampling rate. The generated label column is automatically synchronised with the Data Manager.

6. ML Suite — Machine Learning Pipeline

The ML Suite provides a visual canvas for building and running machine learning pipelines on your locally stored data. All training runs on your own machine — no data is sent to the cloud during training.

6.1 ML Pipeline Canvas

The ML Pipeline canvas works similarly to NeuralFlow: drag blocks from the palette, connect them with typed ports, configure parameters in the inspector. The ML Suite palette contains scikit-learn and PyTorch blocks organised by category.

6.2 Available Block Categories

Category	Example Blocks
Preprocessing	StandardScaler, MinMaxScaler, PCA
Feature Extraction	Channel Selection, Feature Selection, Signal Projection
Dimensionality Reduction	PCA, NCA, t-SNE, UMAP
Classification	LDA, SVM, Random Forest, Gradient Boosting, EEGNet (PyTorch)
Cross Validation	KFold, StratifiedKFold, LeaveOneOut
Pipeline Blocks	Input Block, Output Block, Loop Block, Custom Code
Analysis	Channel Selection, Custom Labels, Signal Projection

6.3 Workflow Designer

The Workflow Designer tab (next to the pipeline canvas) maps your pipeline to subjects and sessions. Configure which subjects, which sessions, and which data source (clean data, channel-selected data, features, or dimensionality-reduced data) the pipeline should use for each training run.

6.4 Compiling and Running a Pipeline

- Click Compile in the toolbar to validate all block connections and parameter types.
- A successful compile auto-registers the pipeline to the BCILattice pipeline engine.
- Click Train to start the batch training run.
- The Batch Training Dashboard opens with dual live progress bars: overall progress and per-subject progress.
- Per-subject metric tiles (Accuracy, F1 score, AUC) appear as each subject completes.

6.5 Parallel Workflow Tabs (Researcher+ plan)

Open multiple pipeline tabs side by side to compare different architectures or hyperparameter configurations on the same dataset. Each tab has its own pipeline canvas, workflow designer, and training run.

6.6 Custom Code Block

Drag a Custom Code block onto the canvas to inject arbitrary Python processing at any point in the pipeline. The block receives the data output from the previous block and must return the processed data:

```
def custom_process(data):
    # data is a NumPy array
    return data
```

7. Analysis Suite

The Analysis Suite is a four-step analytical dashboard for understanding your signal data before committing to a training run. Run it after preprocessing to inform your channel selection and feature engineering decisions.

7.1 Step 1: Channel Selection

Channel Selection scores every channel in your dataset for its discriminative power between classes and retains only the top-N percent. This reduces dimensionality and often significantly improves classifier accuracy.

- Method: choose from available statistical methods (e.g. t-Value, ANOVA).
- Percentile: the top percentage of channels to keep (e.g. 20 means keep the top 20%).
- Ignore Classes: exclude specific class labels from the scoring computation (useful for a rest class).
- Output: a ranked table of channels with their scores, plus a channel-selected CSV file saved alongside the original.

7.2 Step 2: Dimensionality Reduction

Apply a dimensionality reduction method to the channel data or selected channels.

- Available methods: PCA, NCA (Neighbourhood Components Analysis), t-SNE, UMAP (if installed).
- Set the number of target components (e.g. 2 for 2D visualisation).
- Select channel scope: Full Data or Selected Channels (from Step 1).

7.3 Step 3: Feature Extraction

Extract statistical and spectral features from channels or reduced dimensions. The server computes each selected feature per-channel per-sample.

Feature	Description
Mean	Average amplitude per channel
Variance	Signal variance per channel
Kurtosis	Tail weight of the distribution (Fisher definition, unbiased)
Skewness	Asymmetry of the distribution
Maximum / Minimum	Peak and trough values
Range	Max minus Min
Standard Deviation	Square root of variance
Entropy	Shannon entropy of the absolute amplitude distribution
Median	50th-percentile value
Zero Crossing Rate	Number of sign changes in the signal
Signal Energy	Sum of squared samples
Root Mean Square	Square root of mean squared amplitude
Hjorth Activity	Signal variance (first Hjorth parameter)
Hjorth Mobility	Mean frequency proxy (sqrt of var of first derivative / var of signal)
Hjorth Complexity	Signal shape complexity (ratio of mobility of

	derivatives)
Power Spectral Density	Mean squared FFT magnitude — overall signal power

7.4 Step 4: Visualisation

Each analysis step produces up to three matplotlib figures. Adjust figure font sizes using the per-figure slider controls. Figures are generated server-side and displayed in the GUI panel.

8. Experiment Hub

The Experiment Hub automatically records every training run as a versioned experiment entry. You never lose a previous result — every run is tracked, comparable, and fully restorable.

8.1 Experiment Table

The hub displays a table of all experiments with these columns:

- Title — experiment name
- Version — automatically incremented (v1, v2, v3…)
- Workflow — which workflow configuration was used
- Best Score — highest accuracy across all subjects
- Runs — number of subjects trained
- Last Updated — timestamp of last modification

8.2 Restoring a Previous Experiment

Right-click any experiment row and select Restore. A dialog lets you choose which components to restore:

- NeuralFlow Paradigm
- ML Pipeline (MLFlow)
- Workflow Configuration
- Preprocessing Configuration
- Training Results
- Analysis Results

Select any combination and click Restore. Changes take effect immediately without restarting.

8.3 Exporting Trained Models (Researcher+ plan)

- Right-click an experiment → Export Model.
- Choose format: PyTorch (.pt), ONNX (.onnx), TFLite (.tflite), or MATLAB (.mat).
- The exported file is saved to the location you specify.

9. Reports

The Reports module generates a complete, standalone session summary in one click. Reports are self-contained files (inline CSS, no external dependencies) suitable for lab notebooks, paper supplements, and institutional review.

9.1 Generating a Report

- Click Reports in the left sidebar.
- Click Generate Report.
- Choose HTML (web page, viewable in any browser) or PDF.
- Choose a save location.
- BCILattice compiles the report from the current session state in under 5 seconds.

9.2 Report Sections

Section	Content
Session Metadata	Session name, creation date, BCINexus version, export timestamp
Data Manager	Per-modality table: file names, subjects, sampling rate, duration, channels
Preprocessing	Complete parameter set: bandpass, notch, reference, resampling, custom code
NeuralFlow Paradigm	Block list, connection map, compiled paradigm filename, timeline summary
Training Results	Per-subject accuracy, F1 score, AUC; aggregate statistics across all subjects

10. Teams & Cloud

BCILattice's cloud layer provides storage synchronisation and team collaboration. All training and computation remains on your local machine. You control exactly what is uploaded, when, and to whom.

10.1 Creating an Account and Signing In

- Register at bcinexus.io/register or via the Sign In button in BCILattice.
- Verify your email address.
- Sign in from BCILattice — your JWT token is cached at ~/.bcilattice/auth.json.
- The token refreshes automatically. You only need to sign in once per device.

10.2 Creating or Joining a Team (Lab+ plan)

- Click Teams in the left sidebar.
- Click Create Team or accept an invitation link sent by your team owner.
- Assign roles to team members: Owner, Admin, Member, or Viewer.

10.3 Sharing a Dataset with Your Team

- In Data Manager, right-click a session → Upload to Team.
- BCILattice computes the SHA-256 checksum and uploads the file to the team's shared library.
- Team members can download the dataset directly into their own local Data Manager.
- All uploads and downloads appear in the timestamped Team Activity Feed.

10.4 Role Permissions

Action	Owner	Admin	Member	Viewer
Upload datasets/projects	✓	✓	✓	—
Download datasets/projects	✓	✓	✓	✓
Delete team files	✓	✓	—	—
Invite/remove members	✓	✓	—	—
Change member roles	✓	—	—	—
Manage billing	✓	—	—	—

10.5 Publishing a Session to BCINexus Cloud

- Open the session panel → Publish to BCINexus.
- Choose visibility: Private (only you), Team, or Public (community library).
- Choose which components to publish: NeuralFlow, MLFlow, Workflow, Preprocessing, Results.
- Click Publish. A bcilattice:// share link is created.

10.6 Cloud Storage Quotas

Plan	Storage	Uploads/Month	Max File Size
Free	500 MB	1	100 MB
Researcher	10 GB	5	1 GB
Lab	50 GB	50	5 GB
Institution	Custom	Unlimited	25 GB
Enterprise	Unlimited	Unlimited	Unlimited

11. AI Research Assistant

The AI Chat module lets you ask research questions about your loaded datasets and pipeline results using large language models (LLMs). BCILattice supports GPT-4o (OpenAI), Gemini (Google), and Claude (Anthropic). All models are accessed using your own API key — no costs are passed through BCINexus.

11.1 Setting Up (BYOK — Bring Your Own Key)

- Go to Settings → AI Chat.
- Enter your API key for the provider(s) you want to use.
- Keys are stored locally on your machine and never transmitted to BCINexus.

11.2 Using the AI Assistant

- Click AI Chat in the left sidebar.
- Type your question in the chat input. Examples:
  - 'Which channels showed the highest t-value scores in the last channel selection run?'
  - 'Suggest preprocessing parameters for a 64-channel fNIRS working memory study.'
  - 'Explain the FBCSP algorithm in plain language.'
- BCILattice passes a summary of your current session data as context (RAG) with each message.
- Chat history is saved between sessions.

11.3 Privacy Note

> ***i*** *The message you type — including the session context summary — is sent to the LLM provider (OpenAI, Google, or Anthropic) using your API key. Raw dataset contents are never included. Review your LLM provider's privacy policy for their data handling terms.*

12. BCINexus Pipeline Library

The BCINexus community library contains 340+ published BCI research pipelines contributed by researchers worldwide. All pipelines are free to browse and import.

12.1 Browsing the Library

- Go to bcinexus.io/pipelines in a web browser.
- Filter by: Modality (EEG / fNIRS / EMG), Task type, Institution, Tags.
- Sort by: Download count, Reported accuracy, Publication date.
- Each pipeline card shows: author, institution, modality, best accuracy, download count.

12.2 Importing a Pipeline into BCILattice

- Click Import into BCILattice on any pipeline page.
- A bcilattice:// deep link opens BCILattice automatically.
- A dialog shows which components are available to import: NeuralFlow, MLFlow, Workflow, Preprocessing.
- Select the components you want and click Import.
- The imported components appear in their respective modules, ready to use or customise.

12.3 Publishing Your Own Pipeline

- Open the session panel → Publish to BCINexus → set visibility to Public.
- Add a description, accuracy results, and tags.
- Click Publish. After a brief review, your pipeline appears in the community library.
- Published pipelines are licensed under Creative Commons Attribution 4.0 (CC BY 4.0) by default.

13. Keyboard Shortcuts

Action	Windows / Linux	macOS
New session	Ctrl + N	Cmd + N
Open session	Ctrl + O	Cmd + O
Save session	Ctrl + S	Cmd + S
Export session	Ctrl + E	Cmd + E
NeuralFlow — Compile	Ctrl + B	Cmd + B
ML Suite — Compile	Ctrl + Shift + B	Cmd + Shift + B
Generate Report	Ctrl + R	Cmd + R
Search blocks	Space (on canvas)	Space (on canvas)
Delete selected block	Delete	Delete
Undo	Ctrl + Z	Cmd + Z
Redo	Ctrl + Y	Cmd + Shift + Z
Zoom in (canvas)	Ctrl + =	Cmd + =
Zoom out (canvas)	Ctrl + -	Cmd + -
Reset zoom (canvas)	Ctrl + 0	Cmd + 0

14. Troubleshooting

'Server: Not Running' in the status bar

- BCILattice failed to start.
- Restart BCILattice. If the issue persists, check application logs.
- Restart BCILattice. Check logs at ~/.bcilattice/logs/ for the specific error.

File import fails with 'unsupported format'

- Verify the file is not corrupted by opening it in another tool.
- Ensure the file extension matches the actual format.
- Try converting to EDF using your acquisition software.

Training is very slow

- GPU is not detected. Check View → System Info for CUDA status.
- Install the NVIDIA CUDA toolkit matching your driver version.
- BCILattice automatically falls back to CPU — CPU training is supported but slower.

NeuralFlow compile error: 'incompatible port type'

- A block output type does not match the connected input type.
- Check the port colour indicator on each block.
- See the block documentation in the Property Inspector panel.

Channel selection returns 0 selected channels

- Ensure preprocessing has been applied and a clean/filtered file exists.
- Verify the label column is correctly set in Data Manager.
- Try increasing the percentile value (e.g. from 5% to 20%).

Contact Support

Email: support@bcinexus.io

Documentation: bcinexus.io/docs

Community forum: community.bcinexus.io

Status page: status.bcinexus.io