

Linshu: An integrated end-cloud collaborative platform for invasive Brain-Computer Interface data analysis and sharing

Yiqin Zhang, Zhenda Zhang, Yingqi Jia, Chengqi Ke,
Qigang Rong, Dingbang Cao, Siyan Liu, Xiaoxing Zhang,
Chengyu Li[†], Jingpeng Wu[†]

Lingang Laboratory, Shanghai, China.

[†]Corresponding Authors: tonylicy@lglab.ac.cn; jingpeng.wu@lglab.ac.cn

Abstract

Invasive Brain-Computer Interface (iBCI) with high temporal and spatial resolution has witnessed vigorous growth and the community has accumulated large amount of data. Although there exists several handy software tools for data analysis, there is still a lack of an integrated data-sharing platform that is particularly valuable for the growing foundation models with billions of parameters. Here, we introduce Linshu, the first end-cloud collaborative BCI data processing and sharing system with straight-forward usage and guaranteed consistency. The Linshu system enables researchers to analyze BCI data locally, share data reproducibly, and access data interactively.

Keywords: BCI, Data standardization, End-cloud collaboration, Data sharing

1 Introduction

IBCI research has grown substantially over the past decade with emerging academic and clinical applications ranging from neurorehabilitation to communication assistance for patients with paralysis. Invasive approaches employ high-density electrode arrays [1], such as Neuropixels probes [2–4] and flexible micro-electrodes [5–7], with parallel recordings across thousands of channels and simultaneous monitoring of hundreds to thousands of neurons [8]. By densely implanting multiple probes with increased recording scale, researchers can capture neural population dynamics with

higher spatial resolution [9]. A single day of such recording with multiple sessions can yield terabyte-scale data [9].

Large-scale analysis of acquired data often relies on community-developed toolkits that collectively support data loading, signal processing, feature extraction, and behavioral analysis [10]. When research findings are published, the underlying data should be represented in standardized formats to promote semantic interoperability by enabling consistent interpretation across institutions and research teams [11]. Specialized conversion software can transform heterogeneous raw data into standardized files while incorporating intermediate analysis results [12–14]. Subsequently, data sharing can facilitate the exchange of high-quality data across teams, and thereby contribute to further advances in the field [15, 16].

We summarize the needs as follows:

1. **Unified data contract:** Enable structured data processing and analysis.
2. **Analysis engine:** Elastically perform computation at scale.
3. **Shared open-data platform:** Help researchers reproduce experiments and facilitate collaboration across iBCI community.

Prior efforts each address one or two of the three categories of problems, yet there is still a lack of integrated system with all the three features. Researchers must manually reconcile data contracts, processing paradigms, sharing workflows, and identify the subset best matching their needs from the multitude of available tools. Accurate identification depends directly on the researcher’s knowledge of the strengths and weaknesses of each tool, and frequently relies on computer-science expertise that is itself unrelated to the scientific question of interest.

Besides, foundation models with billions of parameters are increasingly capable of cross-study reasoning, yet their effectiveness in iBCI research is hampered by fragmented data representations across acquisition, analysis, and sharing stages. An integrated platform that unifies these stages under a common semantic framework would enable intelligent agents to navigate the entire research pipeline more reliably, reducing the manual reconciliation currently required across disparate tools.

Thus, the next generation of BCI data infrastructure should integrate three core capabilities, reducing the time researchers spend selecting and learning tools. As is shown in fig. 1, Linshu instantiates this approach and provides:

1. **The Linshu data contract:** the shared semantics of the system.
2. **The Linxi data analysis and processing engine:** a standalone software distribution providing reading, processing, and analysis functions for multimodal BCI data.
3. **The Linchou data sharing platform:** an online, publicly accessible platform that hosts data sharing and collaboration for iBCI research.

The Linxi engine and the Linchou platform natively support the Linshu data contract, provide capabilities for processing and sharing data instances, and offer publicly accessible end-cloud collaboration. Together, these capabilities enable iBCI data and associated research resources to be loaded and shared within seconds.

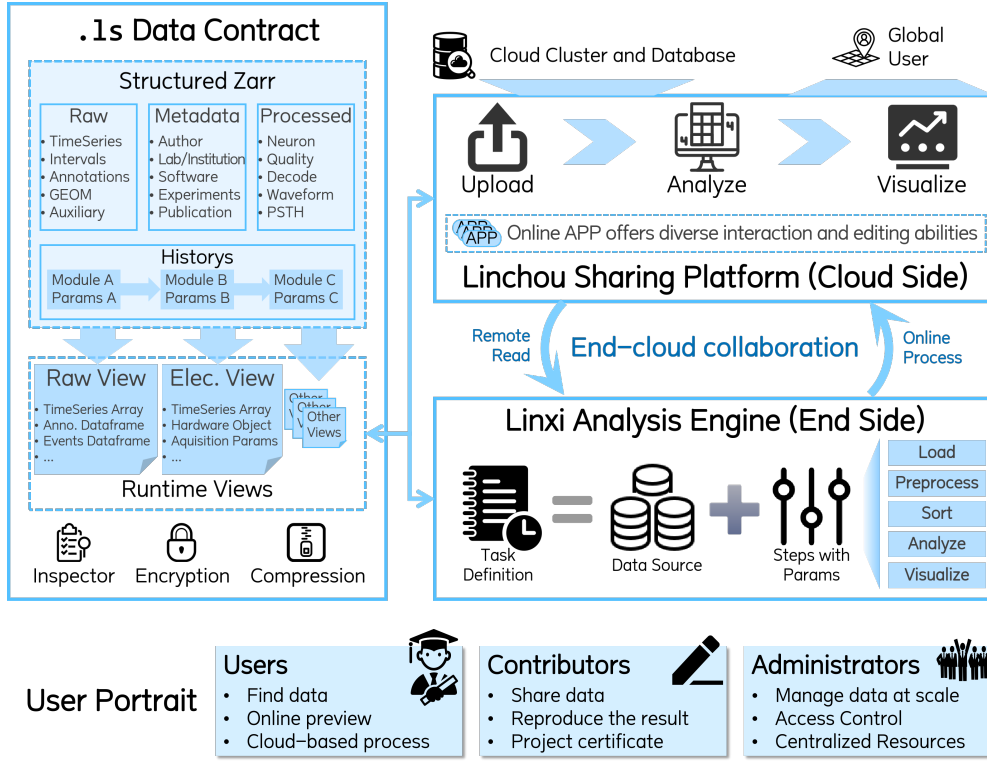


Fig. 1: Overview of Linshu system, which provides a data contract that bridges a local engine and an online platform through end-cloud collaboration, satisfying diverse user needs.

2 Results

2.1 Linshu data contract

To use the system proposed in this work, users can format their data following the proposed Linshu data contract first using the data importer. The internal formatter supports seven data acquisition systems, including commonly used SpikeGLX, SpikeGadgets and Plexon. Because every component of the Linshu system natively recognizes the Linshu data contract, no additional data conversion is required in our processing system.

Here, we position the Linshu data contract as a shared semantic layer across the other subsystems of the platform, with all operations on data realized through a "contract view". The data contract describes the data structure and constrains the operations that can be applied to the data, making it a superset of what traditional data formats cover.

When an external function module reads data, the data contract supplies a view, which is a mapping of the original data that carries the information required for

the upcoming operation. Modifications to the view can reflect back onto the original data, which are monitored by Linxi internal rules. This design confers the following advantages:

1. **Type safety:** All data modifications pass validation, curbing quality degradation from suboptimal practices.
2. **Minimal exposure:** The Linshu data contract embeds large structural definitions, yet users rarely need to grasp the entire schema. The contract view lets users focus on the slice of interest without handling the whole object, simultaneously reducing data-transfer overhead.
3. **Traceability:** The platform logs the view-generation method, operation parameters, module version, and materialized result, rendering the data-analysis process reproducible.
4. **Decoupling contract schema from operation logic:** As the data contract evolves, the underlying storage layout will likely change. Modifying only the view’s mapping rules enables peripheral functions to absorb these changes transparently, lowering long-term maintenance cost.
5. **Constrained operations:** Routing all operations through views curbs the violations that user code or custom tools introduce by acting directly on raw data.
6. **AI-Ready:** View definitions can be injected into context as Agent Prompt or Skill, enabling orderly automated analysis of complex data.

Most data formats are designed to be read, not to project themselves as other formats. Without this capability, maintainers of other toolkits must invest effort in authoring read-side interfaces. To reduce this friction, the contract view provides reverse adaptation by mapping the data contract into the key runtime objects of dominant toolkits in the existing ecosystem. Users declare only the target type they need, and the contract handles the mapping details. For example, a user familiar with the SpikeInterface toolkit obtains `BaseRecording` and `BaseSorting` objects directly from the Linshu data contract, whereas a user familiar with the MNE toolkit obtains `Raw` and `Epochs` objects with equal ease. These objects are already familiar to users, so the learning cost stays low and the contract view slots the data contract into existing workflows, while community maintainers need not author dedicated entry points for reading it.

The runtime of the data contract is implemented based on Pydantic, which is employed to structurally specify field types, field semantics, and other metadata. Data view mappings are provided through Python properties. For large-scale arrays, Linshu leverages Xarray and Dask to implement lazy loading and chunked loading capabilities, thereby enhancing program responsiveness during the ingestion of massive datasets. Instance persistence to storage infrastructure adopts the Zarr format, achieving favorable access performance in cloud-native environments.

2.2 Linxi data analysis and processing engine

IBCI research in dense-recording settings can generate data on the order of terabytes [9], and the corresponding processing pipelines may depend on multiple independent

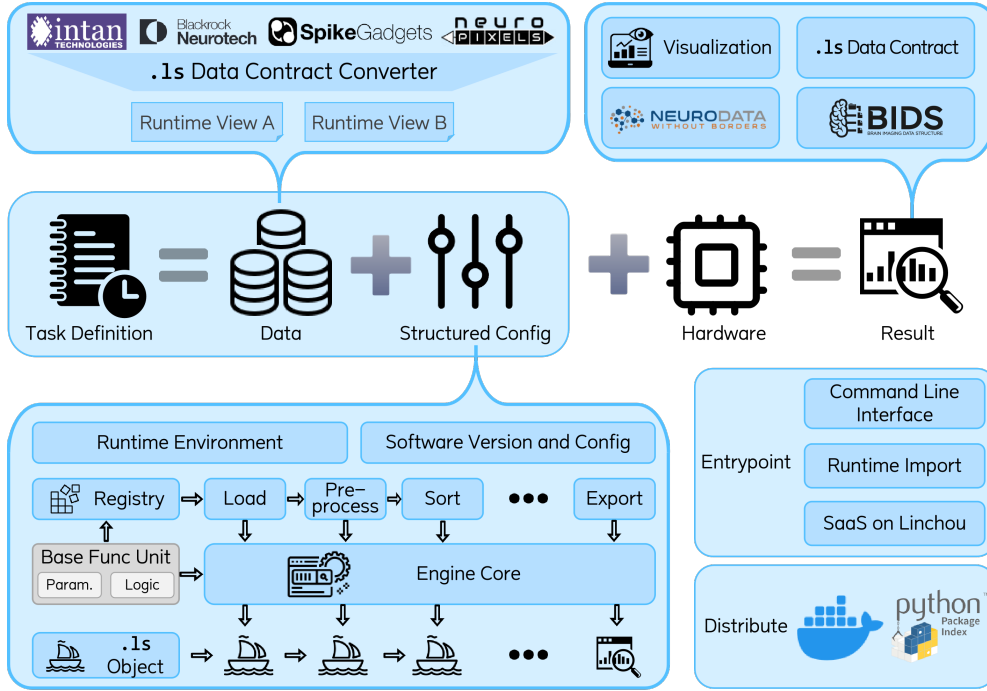


Fig. 2: Design of Linxi, which packages data and structured configurations into a task definition that its core executes. Benefiting from the Linshu data contract, the engine accepts data from mainstream sources, ships with functional modules common to iBCI research, and produces diverse outputs for further user investigation.

tool libraries. Researchers must assemble custom scripts tailored to each experimental paradigm. First, the form these scripts should take remains an open question. The field typically expresses them in Python or MATLAB, and cross-team sharing of data processing pipelines generally relies on semantic interpretation of the code rather than a structured exchange format. Second, the diversity of research objectives has caused the community’s tool libraries to grow ever larger. Before processing their data, researchers usually need to invest time comparing available libraries and learning how to use them.

Linxi draws on and integrates many established toolkits [13, 17–19] in the field, delivering a reproducible, modular data analysis and computation engine. The engine favors building compatibility bridges between existing tools, providing native implementations only when prior work on the relevant functionality is sparse or under-developed. fig. 2 illustrates its overall design. Data and Structured Configurations combine into a Task Definition, the engine’s basic processing unit. The Engine Core reads this definition, orchestrates computation across heterogeneous platforms, and yields outputs as a Linshu data contract instance or in other structured formats.

When data arrives at Linxi from the acquisition device, the engine first loads it into a Linshu data contract instance. Each subsequent stage then mutates this same

instance, with each modification feeding as input into the next stage. Intermediate states produced during processing persist within the instance, with explicit boundaries delimiting each step. To leverage the gains from this decoupling, the Linxi engine records the detailed processing history within the Linshu data contract. Combined with the Linchou platform described in detail later, this capability enables the Linshu system to share both the data itself and the methodology underlying its processing.

Under this design, data processing is abstracted as a staged declarative pipeline, with typical stages spanning data loading, pre-process, device geom, sorting, post-process, quality control, analysis, and export. Linxi provides parameterized functional modules. Researchers can examine their function and parameters, select the appropriate ones, and compose a task definition describing the entire processing workflow. A well-formed task definition carries the complete semantics of a data-processing methodology and can be optionally embedded in the Linshu data contract, enabling other users to reproduce the workflow.

The AI agent is a rapidly emerging class of artificial intelligence technology that interprets users’ vague requirements, selects and invokes appropriate tools, and converts the task into executable steps. Configuring a pipeline still requires knowledge of each module’s API. Linxi exposes detailed descriptions of each functional module to the agent through the Model Context Protocol (MCP). Drawing on the comprehension capacity of a large language model (LLM), the agent searches for and composes Linxi’s functionalities to produce a configuration file that matches the user’s needs.

Linxi is an independently distributable engine that supports standalone deployment in local environments. Its stateless design also lets it deliver data-processing capabilities to other platforms in Software as a Service (SaaS) form. Linxi ships through PyPI, Docker Image, and Git repositories, giving the community multiple distribution channels to choose from. The Linchou data-sharing platform deeply integrates Linxi to deliver hybrid end-cloud data-analysis services.

Diverse research goals impose niche requirements on the engine, motivating customizable functional extensions for Linxi. We adopt a registry-based plugin system to implement every processor in the repository [fig. 3](#), and provide a plugin-repository template, serving advanced users as entry points for extending and modifying the engine. Open-source contributors can further package self-developed functional modules as plugins.

2.3 Linchou data sharing platform

Based on the formatted data using Linxi, we provide the Linchou data sharing platform to extend the scope of data collaboration across teams and regions, which is a publicly accessible cloud portal that offers data hosting, sharing, online analysis and visualization. The Linchou platform provides distributed storage and elastic computing capacity. [fig. 4](#) is the illustration of it.

First, Linchou serves as a data sharing platform, users can upload their datasets after filling a detailed metadata form along with correspondence information. The Linshu data contract instance is the recommended format as it can utilize advanced abilities provided by the platform.

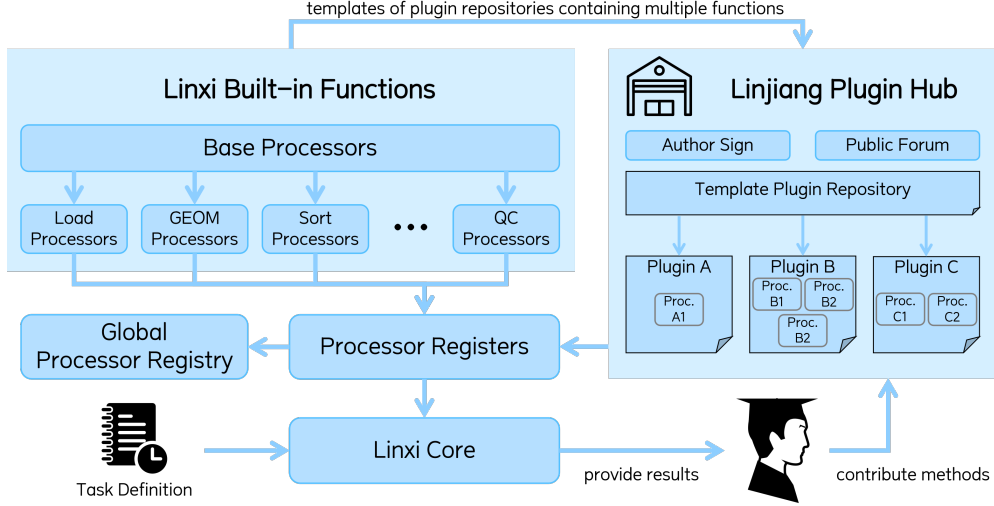


Fig. 3: Design of Linxi plugin system.

Second, Linchou integrates the Linxi engine with a user-friendly online interface for pipeline configuration. As is shown in Extended Data Fig. 1.b, users select available functional modules on the web page, drag and combine them, configure parameters, and assemble a directed acyclic graph. The system translates the graph into a Linxi configuration file, combines it with the selected dataset pointer, acquires computing resources from the elastic cloud cluster, submits it to the cluster scheduler for execution. Upon completion, the integrated Linxi engine writes standardized outputs, including Linshu files and analysis figures, back to the platform, where users can inspect results online and download the derived data without retrieving the raw recordings.

2.4 An end-to-end pass by a single declarative configuration

A representative Linshu workflow, illustrated in fig. 5, processes the data of a published working-memory study [20], in which brain-wide single-unit spiking and behavioral data were recorded from 40 mice performing an olfactory delayed paired-association task across 116 sessions. Analyzing these recordings spans format conversion, signal conditioning, spike sorting, quality assessment, anatomical annotation, and task-aligned firing-rate analysis.

The processing methodology is captured in a single declarative configuration file that names each processing step and its parameters without in-line scripts. The configuration declares a sequence of seven staged operations spanning data loading, preprocessing, sorting, postprocessing, quality assessment, analysis, and export. Through the processor registry as is shown in fig. 3, the engine resolves each step to a callable unit. Extension plugins supply the loaders, sorters, and visualization modules specific to the laboratory’s acquisition hardware. The same configuration file is portable across local execution and remote submission, reflecting the stateless engine

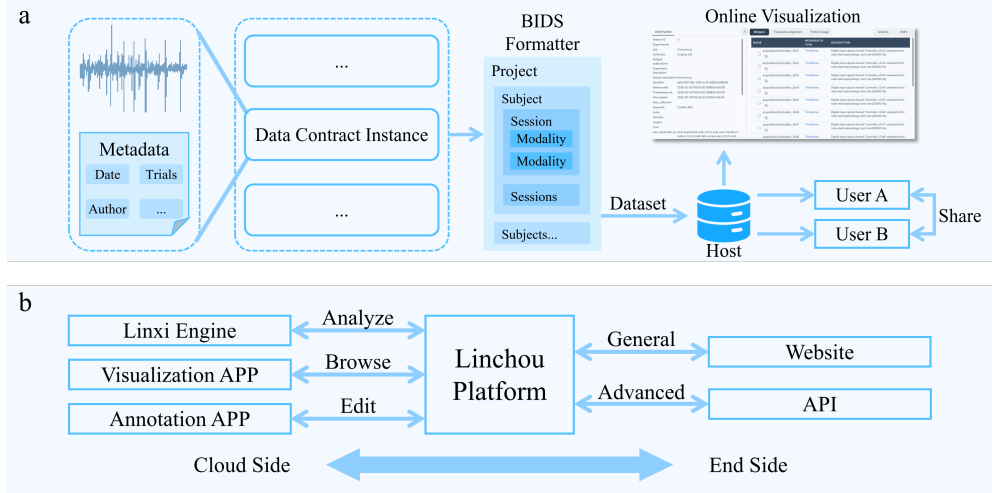


Fig. 4: Design of Linchou platform. **a**, Linchou employs a hierarchical data architecture that consolidates raw electrophysiological recordings and metadata into data instances, which are organized in a BIDS-compliant directory structure. Both visualization and cross-user sharing operate on this unified data structure. **b**, Linchou is an end-cloud collaborative system in which computationally intensive data parsing and transmission are offloaded to the cloud. On the end side, operations typically involve modest traffic requirements and graphical rendering workloads.

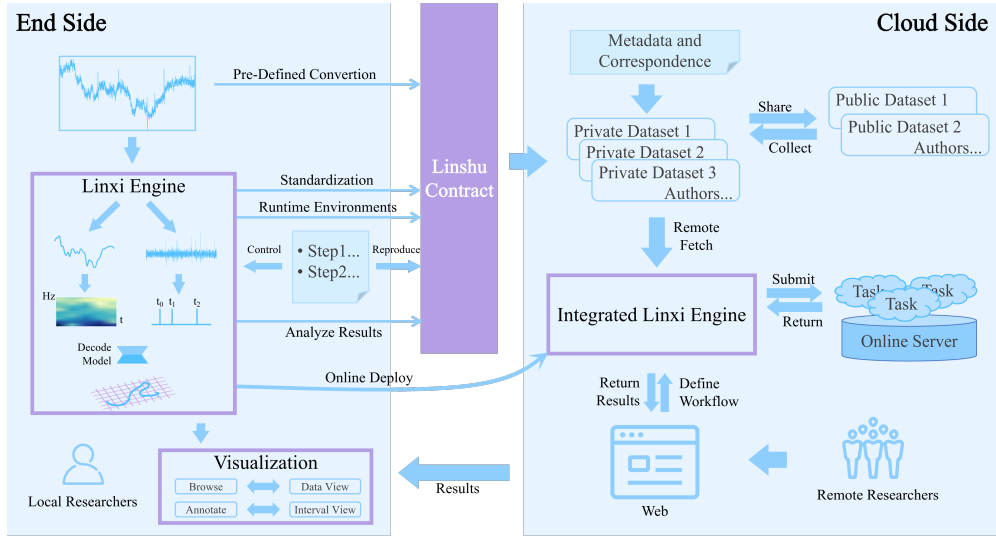


Fig. 5: System context of a representative workflow. The Linshu platform spans a local engine, a centralized data-contract layer, and a cloud platform. The workflow executes one such end-to-end pass through this architecture.

design that allows Linxi to deliver data-processing capabilities to other platforms in a service form.

As is shown in fig. 5, the first stage ingests the raw acquisition and turns it into an instance of the Linshu data contract, with the original time series, channel geometry, and acquisition metadata made available through a contract view. Pre-processing then applies a chain of signal-conditioning operations, including bandpass filtering, common-average referencing, normalization, and whitening. Each expressed as a parameter set to the Linxi engine. The conditioning operates on the recording object held by the Linshu data contract instance.

Spike sorting delegates to a Kilosort 4-based sorter. The sorter runs against the same contracted instance, so the unit identifiers it produces are attached to the same provenance record as the underlying time series. Automated curation assigns every detected unit to one of the following three classes, putative single unit, multi-unit activity, or noise. SpikeInterface [13] internal implementation and bombcell [19] method are integrated to achieve robust curation. Quality assessment follows at both the unit and the channel levels, producing a structured quality report.

Analysis stages populate the contract with the derivatives the laboratory consumes in its day-to-day work: a unit-level table exported in spreadsheet form, per-unit visualizations such as raster plots and waveform overlays, and aggregate dataset statistics. The export stage finally assembles the complete Linshu data contract instance. The instance contains the original recording, every intermediate state produced along the pipeline, the parameters of each operation, the engine and processor versions involved, and the derived products that the laboratory’s downstream tools read directly. Because the data contract records both the inputs and the operations that produced each derivative, dataset users with permission receiving the instance can re-execute or audit the processing without referring to external notes.

The scenario above makes concrete the role of each subsystem in a real workflow. The Linxi engine consumes the declarative definition and runs the pipeline on the contracted view. The contract accumulates the lineage and the intermediate results, and the resulting Linshu data instance is ready to be uploaded to Linchou, where collaborators can re-execute the same configuration against the standardized data or browse the analysis outputs using the platform’s visualization tools. The shared derivatives and the shared methodology thereby work smoothly together, supporting both reproduction of the laboratory’s own analyses and reuse by other teams working on similar multi-channel electrophysiology data.

3 Discussion

3.1 A shallow stack by design

Following the organizational structure of the Linshu subsystems of data contract, processing method and sharing platform, we categorize related work into such three major areas for comparison Table 1. Providing an integrated solution for data processing is the goal of most open-source efforts in this field. They focus on algorithmic or engineering problems whose definitions and requirements are relatively well specified, and can therefore be accomplished by individual developers or small research teams. The

Table 1: Comparison of capability coverage across related tools and platforms

Ecosystem	Analyze Pipeline	Sharing Platform	
		Managed Sharing	Hosted Execution
SpikeInterface[13]	✓		
MNE[17]	✓		
Brainstorm[14]	✓		
MetaBCI[21]	✓		
ControlIt[22]	✓		
NeuroPycon[23]	✓		
BrainFusion[24]	✓		
DANDI[15]		✓	✓
WeBrain[25]	✓	✓	✓
Linshu	✓	✓	✓

development of data formats demands interdisciplinary knowledge, requiring a solid background in computer science combined with a foundation in neuroscience, and thus presents a relatively high barrier to entry for developers. Data sharing platforms require still greater financial investment. It is difficult for individual researchers to sustain the stable operation of such platforms, and consequently the available options are the fewest of the three.

Linshu system covers all of these concerns and provides a data contract that every subsystem natively recognizes, allowing data to flow efficiently through the lifecycle and reducing the need to chain tools of disparate origin.

From an ecosystem perspective shown in Extended Data Table 1, Neurodata Without Borders (NWB) is currently widely adopted format for offline data archive, and is designed to be used in conjunction with the Hierarchical Data Modeling Framework (HDMF). Emphasizing flexibility in data representation, the framework provides the NDX extension mechanism, supports both Python and MATLAB, and has fostered a thriving ecosystem. A number of toolkits have been developed around this standard, and several third-party toolkits have also begun to offer native support for it.

NWB remains the field’s most widely adopted standard, and its strengths are those of a well-governed archive, broad community support and long-term preservation. Its design assumptions, however, were shaped by the storage technologies and tooling available at its inception (2014), and adapting the format to cloud-native object storage and to a runtime role across the processing chain has required retrofitting additional backends and conversion layers [12, 26]. Linshu was designed around these needs from the outset, keeping the software stack shallow and easy to evolve. The two standards thus differ in purpose: NWB is built to preserve the results of completed science, whereas Linshu is built to further support the ongoing lifecycle that produces them.

3.2 Cloud-native from contract to platform

The Linshu data contract is built on Pydantic Models, enabling users to infer the hierarchical structure of files and the recommended types for each field directly from the Python code. Multichannel time-series recordings in iBCI research routinely reach tens of gigabytes per session, far beyond what a typical workstation can hold in memory, while conventional loaders require the entire recording to be read before any operation. To relieve this constraint, Linshu adopts xarray and Dask as its IO middleware, where arrays are loaded lazily and processed in chunks, so that only the data slice required by each operation enters memory, substantially reducing the hardware required at runtime. During persistence, Linshu data are materialized in the Zarr format. Zarr’s chunked array structure aligns naturally with the key-value model of object storage, which is the dominant storage paradigm in cloud computing. This alignment enables high-throughput, cost-efficient access to distributed datasets and reduces latency in distributed data access scenarios.

The Linchou data sharing platform integrates the cloud-friendly Linshu data contract and the Linxi engine lets users select data and request computation resources directly through the web interface and submit tasks for existing datasets in the cloud. Analysis results of modest size are delivered to the client side, enabling users to obtain the data their research requires without downloading the raw data to a local machine.

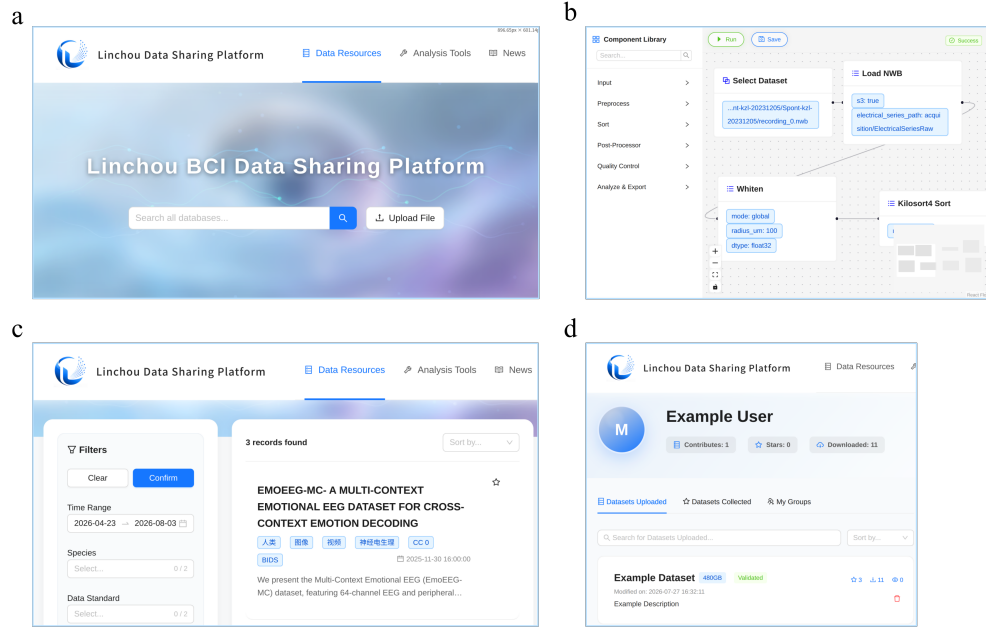
As analyses built on low-quality data rarely pay off, users often need first inspect or annotate the data before deciding whether to analyze it. Linchou therefore provides convenient online browsing and editing tools that let users quickly examine datasets hosted on the cloud platform. By detecting the data format, the platform surfaces those suited to browsing or editing each dataset, and users select as needed. These web-based applications are cloud-optimized, and their chunked access to Linshu data contracts enables start-up in seconds, even for datasets of several hundred gigabytes, improving the efficiency of large-scale data processing.

3.3 Outlook

Currently, the functionalities provided by the Linshu system focus on signal processing algorithms with high utilization rates in the invasive domain. Moving forward, we plan to offer more fine-grained downstream functional modules and extend coverage to additional acquisition sources such as ultrasound and fMRI. This concurrently necessitates agile yet stable iterations of our data contract, as data storage practices and their associated metadata vary considerably across modalities.

Linchou and Linxi will provide more online-available applications in future releases, alongside more stable APIs. We anticipate that these interfaces can serve as an Agent Harness, enabling more rapid validation of scientific hypotheses in the BCI domain.

Extended Data



Extended Data Fig. 1. **a**, Linchou Homepage, providing entry points for data retrieval and upload. **b**, Online Data Processing. Users select and assemble pre-prepared functional modules into a complete data processing pipeline, then submit it to the server for online execution. **c**, Data Retrieval Results. Users can filter based on meta-attributes. The right panel displays matching datasets along with their tags and descriptions. **d**, User Profile Page supports browsing bookmarked datasets, managing and editing self-uploaded datasets, and managing user groups.

Extended Data Table 1. Comparison across BCI ecosystems

Existing Work	Schema	Processing	Sharing
SpikeGadgets[27]	.rec [†]	Trodes	N/A
SpikeGLX[28]	.bin [†]	CatGT [‡]	N/A
Plexon[29]	.plx [†]	Offline Sorter [‡]	N/A
Intan[30]	.rhd [†]	RHX	N/A
Open Ephys[31]	.spikes [†]	O.E. GUI	N/A
Blackrock[32]	.ns1 [†]	Central [‡]	N/A
EDF[33]	.edf	⊗	N/A
NWB[11]	.nwb	⊗	⊗
Linshu	.ls	Linxi	Linchou

[†] Only one representative format is shown, the ecosystem actually offers more format extensions.

[‡] Only one representative software is listed, the vendor actually provides more specialized tools with finer-grained functionalities.

⊗: There are stable and reliable tools available, but they were developed through unofficial contributions and support.

a

```
> linxi --help
usage: linxi [-h] {convert,process,check,index,stat} ...

Linxi CLI

positional arguments:
  {convert,process,check,index,stat}
                                Available subcommands
  convert                  Run data conversion pipeline
  process                  Run data processing pipeline
  check                    Check for incomplete processing lockfiles
  index                    Count samples based on probe structure
  stat                     Generate statistics report
```

b

```
> linxi process --help
usage: linxi process [-h] [-i INPUT_PATH] [-o OUTPUT] -c CONFIG
                    [--cfg-option CFG_OPTION [CFG_OPTION ...]] [--skip-dry-run] [--raise-errors]
                    [--update-completed] [--incomplete-workdir-strategy {delete,raise,resume}]

options:
  -h, --help            show this help message and exit
  -i, --input-path INPUT_PATH
                        Optional input path, can also be specified in config file.
  -o, --output, --work-dir OUTPUT
                        Work directory for process output
  -c, --config CONFIG    Pipeline config YAML
  --cfg-option CFG_OPTION [CFG_OPTION ...]
                        Override pipeline configuration (e.g., ProcessorName.param.key=value)
  --skip-dry-run          Skip validation dry run
  --raise-errors          Raise errors instead of logging them
  --update-completed      Update completed output data with a refined pipeline.
  --incomplete-workdir-strategy {delete,raise,resume}
                        Strategy for incomplete workdir with lock file: delete|raise|resume
```

c

```
> linxi index --help
usage: linxi index [-h] {spikegadgets,intan,spikeglx} index_path

positional arguments:
  {spikegadgets,intan,spikeglx}
  index_path            Path to search for samples

options:
  -h, --help            show this help message and exit
```

d

```
> linxi stat --help
usage: linxi stat [-h] [-o OUTPUT] [-n N_JOBS] [--resume-parse] directory

positional arguments:
  directory            Directory to search NWB files recursively

options:
  -h, --help            show this help message and exit
  -o, --output OUTPUT    Output path for statistics Excel file
  -n, --n-jobs N_JOBS    Number of parallel jobs
  --resume-parse         Resume parsing from existing statistics file
```

Extended Data Fig. 2. The command line interface of Linxi. **a**, the main entry point, available after installing the Linxi package via pip, which provides five primary functions. **b**, the processing entry point, where users specify configurations and working directories to launch the Linxi core for iBCI data processing. **c**, the indexing tool for quantifying locally available data. **d**, the statistics entry point for aggregating results from multiple processing runs.

Code Availability

The Linshu data contract can be accessed by <https://gitee.com/lgl-tianqiong/Linshu-format.git>. The Linxi's git repository can be accessed by <https://gitee.com/lgl-tianqiong/Linxi.git>. The Linchou can be accessed by <https://dbci.lglab.ac.cn>.

Data Availability

The hosted data of this work can be accessed by Linchou data sharing platform: <https://dbci.lglab.ac.cn>.

Declaration of generative AI in scientific writing

AI was used solely for language polishing. The authors assume full responsibility for all scientific statements in the article.

Declaration of competing interests

The authors declare no competing interests.

Funding Statements

This research is supported by the Lingang Laboratory, Grant No. LGL-1630.

Acknowledgements

The authors thank Jiapeng Yin, and Tianmeng He, along with the teams they lead, for providing high-quality data that was used to validate the feasibility of the Linshu system in real-world research operations. The authors thank Yantao Tang and Yipeng Zhou for their contributions during the initial phase of this project.

References

- [1] Siegle, J.H., Steinmetz, N.A.: Large-scale electrophysiology at single-spike resolution. *Nature Reviews Neuroscience* **27**(6), 395–415 (2026) <https://doi.org/10.1038/s41583-026-01042-4>
- [2] Jun, J.J., Steinmetz, N.A., Siegle, J.H., Denman, D.J., Bauza, M., Barbarits, B., Lee, A.K., Anastassiou, C.A., Andrei, A., Aydın, ., Barbic, M., Blanche, T.J., Bonin, V., Couto, J., Dutta, B., Gratiy, S.L., Gutnisky, D.A., Häusser, M., Karsh, B., Ledochowitsch, P., Lopez, C.M., Mitelut, C., Musa, S., Okun, M., Pachitariu, M., Putzeys, J., Rich, P.D., Rossant, C., Sun, W.-l., Svoboda, K., Carandini, M., Harris, K.D., Koch, C., O’Keefe, J., Harris, T.D.: Fully integrated silicon probes for high-density recording of neural activity. *Nature* **551**(7679), 232–236 (2017) <https://doi.org/10.1038/nature24636>
- [3] Steinmetz, N.A., Aydın, C., Lebedeva, A., Okun, M., Pachitariu, M., Bauza, M., Beau, M., Bhagat, J., Böhm, C., Broux, M., Chen, S., Colonell, J., Gardner, R.J., Karsh, B., Kloosterman, F., Kostadinov, D., Mora-Lopez, C., O’Callaghan, J., Park, J., Putzeys, J., Sauerbrei, B., Daal, R.J.J., Vollan, A.Z., Wang, S., Welkenhuysen, M., Ye, Z., Dudman, J.T., Dutta, B., Hantman, A.W., Harris, K.D., Lee, A.K., Moser, E.I., O’Keefe, J., Renart, A., Svoboda, K., Häusser, M., Haesler, S., Carandini, M., Harris, T.D.: Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science* **372**(6539), 4588 (2021) <https://doi.org/10.1126/science.abf4588> <https://www.science.org/doi/pdf/10.1126/science.abf4588>
- [4] Lakunina, A.A., Socha, K.Z., Ladd, A.E., Bowen, A.J., Chen, S., Colonell, J., Doshi, A., Karsh, B., Krumin, M., Kulik, P., Li, A.J., Neutens, P., O’Callaghan, J., Olsen, M., Putzeys, J., Bai Reddy, C., Tilmans, H.A.C., Vargas, S., Welkenhuysen, M., Ye, Z., Häusser, M., Koch, C., Ting, J.T., Dutta, B., Harris, T.D., Steinmetz, N.A., Svoboda, K., Siegle, J.H., Carandini, M.: Neuropixels opto: combining high-resolution electrophysiology and optogenetics. *Nature Methods* **23**(6), 1207–1216 (2026) <https://doi.org/10.1038/s41592-026-03076-z>
- [5] Wang, J., Wang, T., Liu, H., Wang, K., Moses, K., Feng, Z., Li, P., Huang, W.: Flexible electrodes for brain–computer interface system. *Advanced Materials* **35**(47) (2023) <https://doi.org/10.1002/adma.202211012> . Cited by: 126
- [6] Liu, X., Gong, Y., Jiang, Z., Stevens, T., Li, W.: Flexible high-density micro-electrode arrays for closed-loop brain–machine interfaces: a review. *Frontiers in Neuroscience* **18** (2024) <https://doi.org/10.3389/fnins.2024.1348434> . Cited by: 32; All Open Access, Gold Open Access, Green Open Access
- [7] Ban, S., Chong, D., Kwon, J., Lee, S., Huang, Y., Yoo, S., Yeo, W.-H.: Advances in flexible high-density microelectrode arrays for brain-computer interfaces. *Biosensors and Bioelectronics* **292**, 118102 (2026) <https://doi.org/10.1016/j.bios.2025.118102>

- [8] Vries, S.E., Siegle, J.H., Koch, C.: Sharing neurophysiology data from the allen brain observatory. *eLife* **12**, 85550 (2023) <https://doi.org/10.7554/eLife.85550>
- [9] Meshulam, L., Angelaki, D., Benson, J., McRoberts, I., Noel, J.-P., Arlandis, J., Bonacchi, N., Bougrova, K., Catarino, J.A., Cazettes, F., Crombie, D., DeWitt, E.E., Freitas-Silva, L., Laranjeira, I.C., Mainen, Z.F., Meijer, G.T., Rai, P., Raiser, G., Rau, F., Schartner, M.M., Winter, O., Urai, A.E., Aguillon-Rodriguez, V., Soitu, C., Zador, A.M., Krasniak, C.S., Dan, Y., Hu, F., Benson, B., Ganguli, S., Acerbi, L., Chapuis, G.A., Findling, C., Gerçek, B., Huber, F., Pouget, A., Barrell, H., Birman, D., Miller, K., Nylund, K., Roth, N., Steinmetz, N.A., Tucker, M., Yang, K., Fiete, I.R., Liu, A., Schaeffer, R., Churchland, A.K., Davatolagh, F., Khanal, A., Melin, M., Murakami, M., Denève, S., Gordeliy, I., Ahmadi, M., Amjad, J., Hiratani, N., Krishnagopal, S., Latham, P., Pezzotta, A., Xu, Z., Banga, K., Bhagat, J., Faulkner, M., Harris, K.D., Krumin, M., Picard, S., Quadrado, C., Rossant, C., Wells, M.J., Wool, L.E., Carandini, M., Landemard, A., Socha, K.Z., Bruijns, S.A., Dayan, P., Huntenburg, J.M., Kundu, D., Oloomi, F., Tessereau, C., Ashwood, Z.C., Engel, T., Fetcho, R., Haetzel, L.M., Langdon, C., McMannon, B., Mohammadi, Z., Vazquez, A.P., Pillow, J.W., Roy, N.A., Shi, Y., Witten, I.B., Campbell, R., Ghani, N., Hofer, S.B., Martinez-Vergara, H., Miska, N.J., Mscic-Flogel, T., West, S.J., Yang, Y., Svoboda, K., Taheri, M., Häusser, M., Lau, P.Y.P., Makri-Cottingham, A., Perrenoud, S., Abbot, L., Bayer, H.M., Boussard, J., Buchanan, E.K., Fabbri, M., Hurwitz, C., Langfield, C., Lee, H.D., Mitelut, C., Paninski, L., Sanjeev, K., Varol, E., Wang, S., White-way, M.R., Windolf, C., Yu, H., Zhang, Y., Birman, D., Bougrova, K., Engel, T.A., Gerçek, B., Huber, F., Mscic-Flogel, T.D., Pan-Vazquez, A., Schartner, M., Laboratory, I.B.: A brain-wide map of neural activity during complex behaviour. *Nature* **645**(8079), 177–191 (2025) <https://doi.org/10.1038/s41586-025-09235-0>
- [10] Acerbi, L., Aguillon-Rodriguez, V., Ahmadi, M., Amjad, J., Angelaki, D., Arlandis, J., Ashwood, Z.C., Banga, K., Barrell, H., Bayer, H.M., Benson, J., Benson, B., Bhagat, J., Birman, D., Bougrova, K., Boussard, J., Bruijns, S.A., Carandini, M., Catarino, J., Cazettes, F., Dan, Y., Davatolagh, F., Dayan, P., Denève, S., Dong, L.L., Engel, T., Fabbri, M., Fiete, I., Findling, C., Freitas-Silva, L., Ganguli, S., Gerçek, B., Ghani, N., Gordeliy, I., Haetzel, L.M., Häusser, M., Hiratani, N., Hofer, S., Hu, F., Huber, F., Hurwitz, C., Khanal, A., Krasniak, C.S., Krishnagopal, S., Krumin, M., Langdon, C., Latham, P., Lau, P., Lee, H., Liu, A., Mainen, Z.F., Vergara, H.M., Mcgrory, C., McMannon, B., Meijer, G.T., Melin, M., Meshulam, L., Miska, N.J., Mitelut, C., Mohammadi, Z., Mscic-Flogel, T., Murakami, M., Noel, J.-P., Nylund, K., Vazquez, A.P., Paninski, L., Pezzotta, A., Picard, S., Pillow, J.W., Pouget, A., Roth, N., Roy, N.A., Sanjeev, K., Schaeffer, R., Shi, Y., Socha, K.Z., Soitu, C., Svoboda, K., Taheri, M., Tessereau, C., Urai, A.E., Varol, E., Wells, M.J., Whiteway, M.R., Windolf, C., Witten, I., Wool, L.E., Zador, A.M., Bonacchi, N., Chapuis, G.A., Churchland, A.K., DeWitt, E.E.J., Faulkner, M., Harris, K.D., Huntenburg, J.M., Hunter, M., Laranjeira, I.C., Rossant, C., Sasaki, M., Schartner, M.M., Shen, S., Steinmetz, N.A., Walker, E.Y., West, S.J., Winter, O., Laboratory, T.I.B.: A modular

- architecture for organizing, processing and sharing neurophysiology data. *Nature Methods* **20**(3), 403–407 (2023) <https://doi.org/10.1038/s41592-022-01742-6>
- [11] Rübél, O., Tritt, A., Dichter, B., Braun, T., Cain, N., Clack, N., Davidson, T.J., Dougherty, M., Fillion-Robin, J.-C., Graddis, N., Grauer, M., Kiggins, J.T., Niu, L., Ozturk, D., Schroeder, W., Soltesz, I., Sommer, F.T., Svoboda, K., Ng, L., Frank, L.M., Bouchard, K.: NWB:N 2.0: An accessible data standard for neurophysiology. *bioRxiv* (2019) <https://doi.org/10.1101/523035> . preprint
 - [12] Mayorquin, H., Baker, C., Adkisson-Floro, P., Weigl, S., Trappani, A., Taufer, L., Rübél, O., Dichter, B.: Neuroconv: Streamlining neurophysiology data conversion to the nwb standard. *Python in Science Conference, 2025* (2025) <https://doi.org/10.25080/cehj4257>
 - [13] Buccino, A.P., Hurwitz, C.L., Garcia, S., Magland, J., Siegle, J.H., Hurwitz, R., Hennig, M.H.: Spikeinterface, a unified framework for spike sorting. *eLife* **9**, 61834 (2020) <https://doi.org/10.7554/eLife.61834>
 - [14] Tadel, F., Baillet, S., Mosher, J.C., Pantazis, D., Leahy, R.M.: Brainstorm: A user-friendly application for meg/eeg analysis. *Computational Intelligence and Neuroscience* **2011**(1), 879716 (2011) <https://doi.org/10.1155/2011/879716> <https://onlinelibrary.wiley.com/doi/pdf/10.1155/2011/879716>
 - [15] Magland, J.F., Ly, R., Rübél, O., Dichter, B.: Facilitating analysis of open neurophysiology data on the dandi archive using large language model tools. *Scientific Data* **12**(1), 1988 (2025) <https://doi.org/10.1038/s41597-025-06285-x>
 - [16] Markiewicz, C.J., Gorgolewski, K.J., Feingold, F., Blair, R., Halchenko, Y.O., Miller, E., Hardcastle, N., Wexler, J., Esteban, O., Goncalves, M., Jwa, A., Poldrack, R.: The openneuro resource for sharing of neuroscience data. *eLife* **10**, 71774 (2021) <https://doi.org/10.7554/eLife.71774>
 - [17] Gramfort, A., Luessi, M., Larson, E., Engemann, D.A., Strohmeier, D., Brodbeck, C., Goj, R., Jas, M., Brooks, T., Parkkonen, L., Hämäläinen, M.S.: MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience* **7**(267), 1–13 (2013) <https://doi.org/10.3389/fnins.2013.00267>
 - [18] Hoyer, S., Hamman, J.: xarray: N-d labeled arrays and datasets in python. *Journal of Open Research Software* **5**(1), 10–10 (2017) <https://doi.org/10.5334/jors.148>
 - [19] Fabre, J.M.J., Beest, E.H., Peters, A.J., Carandini, M., Harris, K.D.: Bombcell: Automated Curation and Cell Classification of Spike-sorted Electrophysiology Data. <https://doi.org/10.5281/zenodo.8172821> . <https://doi.org/10.5281/zenodo.8172821>
 - [20] Huang, E., Xu, D., Zhu, H., *et al.*: Hierarchical replay of multi-regional sequential

- spiking associated with working memory. *Neuroscience Bulletin* (2025) <https://doi.org/10.1007/s12264-025-01524-y>
- [21] Mei, J., Luo, R., Xu, L., Zhao, W., Wen, S., Wang, K., Xiao, X., Meng, J., Huang, Y., Tang, J., Cheng, L., Xu, M., Ming, D.: Metabci: An open-source platform for brain-computer interfaces. *Computers in Biology and Medicine* **168**, 107806 (2024) <https://doi.org/10.1016/j.compbimed.2023.107806>
 - [22] Yang, W., Li, C., Xu, X., Wang, X., Guo, Y., Chai, X., Cao, T., Song, J., Wang, N., Zhang, X., Yang, Y., Zheng, C., Cui, H.: Controlit: A universal framework for translational, adaptive, and online brain-computer interfaces. *Advanced Intelligent Systems* **8**(4), 202501148 (2026) <https://doi.org/10.1002/aisy.202501148> <https://advanced.onlinelibrary.wiley.com/doi/pdf/10.1002/aisy.202501148>
 - [23] Meunier, D., Pascarella, A., Altukhov, D., Jas, M., Combrisson, E., Lajnef, T., Bertrand-Dubois, D., Hadid, V., Alamian, G., Alves, J., Barlaam, F., Saive, A.-L., Dehgan, A., Jerbi, K.: Neuropycon: An open-source python toolbox for fast multimodal and reproducible brain connectivity pipelines. *NeuroImage* **219**, 117020 (2020) <https://doi.org/10.1016/j.neuroimage.2020.117020>
 - [24] Li, W., Gao, C., Li, Z., Diao, Y., Li, J., Zhou, J., Zhou, J., Peng, Y., Chen, G., Wu, X., Wu, K.: Brainfusion: a low-code, reproducible, and deployable software framework for multimodal brain computer interface and brain body interaction research. *Advanced Science* **12**(32), 17408 (2025) <https://doi.org/10.1002/advs.202417408> <https://advanced.onlinelibrary.wiley.com/doi/pdf/10.1002/advs.202417408>
 - [25] Dong, L., Li, J., Zou, Q., Zhang, Y., Zhao, L., Wen, X., Gong, J., Li, F., Liu, T., Evans, A.C., Valdes-Sosa, P.A., Yao, D.: Webrain: A web-based braininformatics platform of computational ecosystem for eeg big data analysis. *NeuroImage* **245**, 118713 (2021) <https://doi.org/10.1016/j.neuroimage.2021.118713>
 - [26] Rübél, O., Tritt, A., Ly, R., Dichter, B.K., Ghosh, S., Niu, L., Baker, P., Soltesz, I., Ng, L., Svoboda, K., Frank, L., Bouchard, K.E.: The Neurodata Without Borders ecosystem for neurophysiological data science. *eLife* **11**, 78362 (2022) <https://doi.org/10.7554/eLife.78362>
 - [27] SpikeGadgets: Trodes Data Acquisition System. <https://spikegadgets.com/>. Accessed: 2026-08-13 (n.d.)
 - [28] Bill Karsh: SpikeGLX. <https://github.com/billkarsh/SpikeGLX>. Accessed: 2026-08-13 (n.d.)
 - [29] Plexon Inc.: OmniPlex Neural Data Acquisition System. <https://plexon.com/data-acquisition-with-the-omniplex-system/>. Accessed: 2026-08-13 (n.d.)

- [30] Intan Technologies: Intan Technologies: Low-Power, High-Performance Bioamplifiers and Data Acquisition Systems. <https://www.intantech.com/index.html>. Accessed: 2026-08-13 (n.d.)
- [31] Siegle, J.H., López, A.C., Patel, Y.A., Abramov, K., Ohayon, S., Voigts, J.: Open ephys: an open-source, plugin-based platform for multichannel electrophysiology. *Journal of Neural Engineering* **14**(4), 045003 (2017) <https://doi.org/10.1088/1741-2552/aa5eea>
- [32] Blackrock Neurotech: NeuroPort Neural Data Acquisition System. <https://blackrockneurotech.com/>. Accessed: 2026-08-13 (n.d.)
- [33] Kemp, B., Värri, A., Rosa, A.C., Nielsen, K.D., Gade, J.: A simple format for exchange of digitized polygraphic recordings. *Electroencephalography and Clinical Neurophysiology* **82**(5), 391–393 (1992) [https://doi.org/10.1016/0013-4694\(92\)90009-7](https://doi.org/10.1016/0013-4694(92)90009-7)