# Extended CND Format — Data Formatting Manual

A single-page reference for formatting datasets into our extended CND format. Base CND fields follow the official specification; extension fields (marked **[EXT]**) add support for selective, absolute/global, and spatial attention decoding with arbitrary within-trial switches.

Two files per CND dataset (per subject where noted):

- `dataSub<N>.mat` → variable `neural`
- `dataStim.mat` (or `dataStim<N>.mat` if stimuli differ per subject) → variable `stim`

---

## Global conventions

- **Trials:** `neural.data` and `stim.data` must have the same number of trials and the same per-trial start sample. Time is always the first dimension `[time x ...]`.
- **Sampling rate:** `neural.fs` and `stim.fs` must match at load time. Reconcile (resample) upstream, before CND conversion — not in the loader.
- **Extension label fields** are all dense, per-sample, one cell per trial, at `neural.fs`.
- **Name arrays are positional:** `names{k}` describes label value `k`. Do **not** embed the index in the string (write `'attention'`, not `'1=attention'`).
- **Sentinels:** integer label fields use `0` = "undefined / none / not applicable"; azimuth uses `NaN`. Reserve these — real labels start at `1`.
- **Extension fields are categorical/label data**, not signals. Never pass them through filtering, downsampling, interpolation, or re-referencing. If resampling is unavoidable, use nearest-neighbor / hold.

---

## Data folder structure

Each dataset lives in its own top-level folder, organized as follows:

| Folder / file | Req/Opt | Contents |
|---|---|---|
| `neural/` | optional | Original, unconverted neural response files (e.g. `.bdf`, `.mat`, `.set`). Kept separate because it's often large; omit from shared/synced copies if space is a concern. |
| `stim/` | recommended | Original stimulus files (e.g. `.wav`). More important to keep than `neural/` — raw audio/video allows recomputing or adding new stimulus features later. |
| `toCND.m` | required | Conversion script that reads from `neural/` and `stim/` and writes out the `dataCND/` contents (base + extension fields as described above). |

| Folder / file | Req/Opt | Contents |
|---|---|---|
| `dataCND/` | required | Exact folder name. Holds the converted, CND-formatted dataset: `dataSub<N>.mat` (per-participant `neural`), `dataStim.mat` / `dataStim<N>.mat` (`stim`), and preprocessed versions (e.g. `pre_dataSub<N>.mat`). |
| other scripts and files | optional | Any additional preprocessing scripts or channel location files used to produce the `pre_dataSub<N>.mat` files. |

Example layout:

```
datasetName/
├── neural/             # optional, raw EEG/MEG/... (bdf, mat, ...)
├── stim/                # raw stimuli (wav, ...)
├── toCND.m              # neural/ + stim/ -> dataCND/
├── chanInfo.mat          # optional, shared channel locations
├── preprocessCND.m       # optional, other preprocessing scripts
└── dataCND/              # exact name
    ├── dataSub1.mat
    ├── dataSub2.mat
    ├── pre_dataSub1.mat
    ├── pre_dataSub2.mat
    └── dataStim.mat
```

---

## `neural` structure (`dataSub<N>.mat`)

Base fields (official CND)

| Field | Req/Opt | Type | Description |
|---|---|---|---|
| `neural.dataType` | required | string | Modality, e.g. `'EEG'`, `'MEG'`, `'ECoG'`. |
| `neural.deviceName` | required | string | Acquisition device, e.g. `'BioSemi ActiveTwo'`. |
| `neural.fs` | required | scalar (Hz) | Neural sampling rate. |
| `neural.data` | required | cell `{1 x nTrials}` | Each cell `[time x channels]` double. |
| `neural.chanlocs` | optional | struct array | Channel locations (EEGLAB-style: `.labels`, `.X`, `.Y`, `.Z`, ...). Strongly recommended; required for spatial algorithms and topoplots. |
| `neural.extChan` | optional | cell of structs | Non-scalp channels. Each `neural.extChan{i}` has `.data` (cell `{1 x nTrials}`, each `[time x nExt]`) and `.description` (string, e.g. `'Mastoids'`). |

| Field | Req/Opt | Type | Description |
|---|---|---|---|
| `neural.origTrialPosition` | optional | vector | Original index/order of each trial before any reordering. |
| `neural.deviceName` / other meta | optional | — | Any additional acquisition metadata may be stored as extra fields. |

### Extension fields — attention labels **[EXT]**

All three channels below are **optional and independent**: include only the ones a dataset supports; any combination is valid.

| Field | Req/Opt | Type | Description |
|---|---|---|---|
| `neural.attendedStream` | optional **[EXT]** | cell `{1 x nTrials}`, each `[time x 1]` int | *Selective attention.* Per-sample index `k` of the attended stream, matching the `_stream<k>` suffix in `stim`. `0` = no/undefined stream. Omit for non-competing-stream paradigms. |
| `neural.attentionState` | optional **[EXT]** | cell `{1 x nTrials}`, each `[time x 1]` int | *Absolute/global attention.* Per-sample attentional-state label. `0` = undefined. |
| `neural.attentionStateNames` | optional **[EXT]** | cell of strings | Positional names for `attentionState` values, e.g. `{'attention','no attention'}`. Required if `attentionState` is present. |
| `neural.attendedAzimuth` | optional **[EXT]** | cell `{1 x nTrials}`, each `[time x 1]` double | *Spatial attention.* Per-sample attended azimuth in **degrees, signed**. Convention: **0° = front, negative = left, positive = right.** `NaN` = undefined / non-spatial. For coarse "left/right-only" datasets, use representative values (e.g. ∓90) and record the coarse origin in `stim`/dataset meta. |

Derived on demand (do **not** store): stream-switch events via `diff()` on `attendedStream`; `attendedEar = sign(attendedAzimuth)`; categorical direction bins via `discretize(attendedAzimuth, edges)`.

---

## `stim` structure (`dataStim.mat` / `dataStim<N>.mat`)

### Base fields (official CND)

| Field | Req/Opt | Type | Description |
|---|---|---|---|
| `stim.data` | required | cell `{nFeatures x nTrials}` | Each cell a `[time x d]` feature matrix: `d = 1` univariate (e.g. envelope), `d > 1` multivariate (e.g. spectrogram bands). |
| `stim.names` | required | cell of strings `{nFeatures x 1}` | Name of each feature (row of `stim.data`). Aim to uniformize namings as much as possible, especially for cross-dataset combinations (e.g., use 'envelope', 'acousticEdge', 'wordOnset', ...) |
| `stim.fs` | required | scalar (Hz) | Stimulus-feature sampling rate; must equal `neural.fs` at load. |
| `stim.condIdxs` | optional | vector | Per-trial condition index. |
| `stim.condNames` | optional | cell of strings | Positional names for `condIdxs` values. |
| `stim.streamWav` | optional **[EXT]** | cell `{nbStreams x nTrials}` | Normalized wav **basename** per stream per trial (e.g. `'story3_part2.wav'`), aligned to the same stream index `k` as `env_stream<k>`. Identical string == same underlying wav, which enables repetition detection (`unique()` over the flattened cell) and leakage-safe cross-validation (group folds by wav so a repeat never straddles train/test). Use `''` for no file (silence / non-auditory feature). Keep basenames canonical and stable across the whole dataset — identity here is filename-based, so two different names holding identical audio would be under-counted as distinct. |

### Feature-naming convention **[EXT]**

Competing streams are stored as ordinary features under a fixed, **attention-independent** index `k` (stream identity / presentation channel / recording order — never assigned by which stream was attended):

| Feature name pattern | Meaning |
|---|---|
| `envelope_stream<k>` | Envelope of stream `k` (e.g. `envelope_stream1`, `envelope_stream2`). |
| `<feature>_stream<k>` | Any other per-stream feature, same convention. |
| `wordOnset`, `opticalFlow`, ... | Single-stream / non-competing features carry **no** `_stream<k>` suffix. |

`nbStreams` is inferred by counting features matching `_stream<k>`; there is no separate count field. Record coarse-direction origin (if any) in an optional dataset-level meta field.

---

## Minimum viable dataset (quick start)

To get a dataset loading and running a basic neural-tracking / TRF analysis, you need only:

- `neural.dataType`, `neural.deviceName`, `neural.fs`, `neural.data` (+ `neural.chanlocs` for anything spatial)
- `stim.data`, `stim.names`, `stim.fs`

Then add the attention channel(s) relevant to your paradigm:

| Paradigm | Add these fields |
|---|---|
| Selective attention (competing talkers) | `stim`: `envelope_stream1`, `envelope_stream2`, … • `neural`: `attendedStream` |
| Absolute / global attention | `neural`: `attentionState` + `attentionStateNames` |
| Spatial attention decoding | `neural`: `attendedAzimuth` (+ `chanlocs`) |
| Combined (e.g. selective + spatial) | any combination of the above — all channels are independent |

---

## Formatting checklist (per dataset)

1. ☐ `neural.data` and `stim.data` have equal trial counts, matching start samples, time on dim 1.
2. ☐ `neural.fs == stim.fs` (resampled upstream if needed).
3. ☐ `chanlocs` present if any spatial algorithm or topoplot is intended.
4. ☐ Competing streams named `<feature>_stream<k>` with **attention-independent** `k`.
5. ☐ Attention channel(s) present for the paradigm; each `[time x 1]` per trial, dense, at `neural.fs`.
6. ☐ Name arrays positional (no `'k='` prefixes); sentinels (`0` / `NaN`) reserved and not reused for real labels.
   6b. [ ] If `streamWav` is used: basenames normalized/canonical, aligned to `_stream<k>`, `''` for no-file; distinct-filename count matches the number of distinct stimuli expected from the experiment design.
7. ☐ Attention/label fields never filtered or resampled through the signal pipeline.