



Conversion Software

Instructions for Use

NRD2MED

NRD2MED is software to convert data from Neuralynx Raw Data format (".nrd") into the MED format.

NRD2MED is run from the command line.

The calling format is as follows:

```
NRD2MED first_session_file [output_directory] [channel_spec_file (cs)] [runtime_config_file (rc)] [name_1] [name_2] [name_3] [anonymized_ID] [ID_number]
```

Calling NRD2MED with no arguments will display the above and exit.

As is customary in command line utilities, arguments in square brackets are optional, and if they are not provided, default values or behavior is used.

NRD2MED Arguments:

`first_session_file`:

This is required, no default value is assumed. This will be the name of the MED session to be generated.

`output_directory`:

This is the directory in which the converted session will be stored. If empty (""), the value in the rc file will be used, otherwise this supersedes the rc value.

`channel_spec_file`:

This is the name of a channel specification (cs) file, which enumerates channel names, block sizes, and decimation frequencies for the channels to be recorded. This file will be described in more detail below. If no `channel_spec_file` is provided, NRD2MED will use the default channel specification file named "NRD2MED_cs.csv" in the current directory. If the default file does not exist, NRD2MED will use default channel names, and convert all channels at the native sampling frequency with no decimation.

`runtime_config_file`:

This is the name of a runtime configuration (rc) file, which specifies numerous details about how conversion should be performed. This file will be described in more detail below. If no `runtime_config_file` is provided, NRD2MED will use the default runtime configuration file named "NRD2MED_rc.txt" in the current directory. If the default file does not exist, NRD2MED will exit.

`name_1`, `name_2`, `name_3`:

These are subject names. They are typically used as first, middle, and last names, if the subject is a person. These names are stored in section 3 of the MED metadata files, along with other potentially subject identifying data, and can be encrypted separately from technical recording details. If not specified, these fields are simply empty.

anonymized_ID:

This is an anonymized subject identifying field. Anonymization is external, not a function of MED. An anonymized name or number are typical contents, as in a research protocol. It is stored as a string. This is stored in the universal header of every MED file, and *cannot be encrypted*. If not specified, this field is simply empty.

ID_number:

This is a subject ID number. If the subject is a human patient, it is typically used to store a medical record number. It is stored in section 3 of the MED metadata files, along with other potentially subject identifying data, and can be encrypted separately from technical recording details. If not specified, this field is simply empty.

The command line designed to minimize reentering recording details.

Notes

The runtime configuration (rc) file can be specified once for standard recording requirements, or a set of rc files can be kept for a various recording requirements which can then be specified on the command line.

The channel specification (cs) file specifies channel names, block samples, decimation frequencies, and channel descriptions. A command line utility to create cs files is provided with NRD2MED, called “create_channel_specs”. However, the cs file is formatted as csv (comma separated values) which can be created or edited by any spreadsheet program. Depending on typical usage, a new cs file may need to be created for each conversion, but if not, it can certainly be reused.

The remaining command line arguments (name_1, name_2, name_3, anonymized_ID, & ID_number) are things that are likely to vary between recordings, and so may be entered each time NRD2MED is launched.

DAT2MED

DAT2MED is software to convert data from the OpenEphys / Intan™ raw data format (".dat") or into the MED format. DAT2MED also converts ".bin" formatted files which are essentially the same except for the scaling factor.

DAT2MED is run from the command line.

The calling format is as follows:

```
DAT2MED dat_file [channel_count] [sampling frequency] [output_directory] [channel_spec_file (cs)] [runtime_config_file (rc)] [name_1] [name_2] [name_3] [anonymized_ID] [ID_number]
```

Calling DAT2MED with no arguments will display the above and exit.

As is customary in command line utilities, arguments in square brackets are optional, and if they are not provided, default values or behavior is used.

DAT2MED Arguments:

dat_file:

This is required, no default value is assumed. This will be the name of the MED session to be generated.

channel_count & sampling_frequency:

Dat files do not have a header that contains this information. These values can be entered on the command line. If they are not, DAT2MED looks for an associated xml file in the same directory that contains these values. If this file does not exist, or does not contain the values, it uses the values in the rc file. If the values are not specified there, the program exits.

output_directory:

Enclosing directory; the path does not include the session directory.

channel_spec_file:

This is the name of a channel specification (cs) file, which enumerates channel names, block sizes, and decimation frequencies for the channels to be recorded. This file will be described in more detail below. If no channel_spec_file is provided, DAT2MED will use the default channel specification file named "DAT2MED_cs.csv" in the current directory. If the default file does not exist, DAT2MED will use default channel names, and convert all channels at the native sampling frequency with no decimation.

runtime_config_file:

This is the name of a runtime configuration (rc) file, which specifies numerous details about how recording should be performed. This file will be described in more detail below. If no runtime_config_file is provided, DAT2MED will use the default runtime configuration file named "DAT2MED_rc.txt" in the current directory. If the default file does not exist, DAT2MED will exit.

name_1, name_2, name_3:

These are subject names. They are typically used as first, middle, and last names, if the subject is a person. These names are stored in section 3 of the MED metadata files, along with other potentially subject identifying data, and can be encrypted separately from technical recording details. If not specified, these fields are simply empty.

`anonymized_ID:`

This is an anonymized subject identifying field. Anonymization is external, not a function of MED. An anonymized name or number are typical contents, as in a research protocol. It is stored as a string. This is stored in the universal header of every MED file, and *cannot be encrypted*. If not specified, this field is simply empty.

`ID_number:`

This is a subject ID number. If the subject is a human patient, it is typically used to store a medical record number. It is stored in section 3 of the MED metadata files, along with other potentially subject identifying data, and can be encrypted separately from technical recording details. If not specified, this field is simply empty.

The command line designed to minimize reentering recording details.

Notes

The runtime configuration (rc) file can be specified once for standard recording requirements, or a set of rc files can be kept for a various recording requirements which can then be specified on the command line.

The channel specification (cs) file specifies channel names, block samples, decimation frequencies, and channel descriptions. A command line utility to create cs files is provided with DAT2MED, called “create_channel_specs”. However, the cs file is formatted as csv (comma separated values) which can be created or edited by any spreadsheet program. Depending on typical usage, a new cs file may need to be created for each conversion, but if not, it can certainly be reused.

The remaining command line arguments (name_1, name_2, name_3, anonymized_ID, & ID_number) are things that are likely to vary between recordings, and so may be entered each time DAT2MED is launched.

CSC2MED

CSC2MED is software to convert data from Neuralynx Continuously Sampled Channel format (“`.ncs`”) into the MED format.

CSC2MED is run from the command line.

The calling format is as follows:

```
CSC2MED CSC_directory [output_directory] [rc_file] [name_1] [name_2] [name_3] [ID_number] [anonymized_ID]
```

Calling CSC2MED with no arguments will display the above and exit.

As is customary in command line utilities, arguments in square brackets are optional, and if they are not provided, default values or behavior is used.

CSC2MED Arguments:

`CSC_directory`:

This is required, no default value is assumed. This will be the name of the MED session to be generated.

`output_directory`:

The path to where the converted MED session will be stored. If none is specified, it will use the “output directory” value from the rc file.

`runtime_config_file`:

This is the name of a runtime configuration (rc) file, which specifies numerous details about how the conversion should be performed. This file will be described in more detail below. If no `runtime_config_file` is provided, CSC2MED will use the default runtime configuration file named “CSC2MED_rc.txt” in the current directory. If the default file does not exist, CSC2MED will exit.

`name_1`, `name_2`, `name_3`:

These are subject names. They are typically used as first, middle, and last names, if the subject is a person. These names are stored in section 3 of the MED metadata files, along with other potentially subject identifying data, and can be encrypted separately from technical recording details. If not specified, these fields are simply empty.

`ID_number`:

This is a subject ID number. If the subject is a human patient, it is typically used to store a medical record number. It is stored in section 3 of the MED metadata files, along with other potentially subject identifying data, and can be encrypted separately from technical recording details. If not specified, this field is simply empty.

The command line designed to minimize reentering recording details.

`anonymized_ID`:

This is an anonymized subject identifying field. Anonymization is external, not a function of MED. An anonymized name or number are typical contents, as in a research protocol. It is

stored as a string. This is stored in the universal header of every MED file, and *cannot be encrypted*. If not specified, this field is simply empty.

Notes

The runtime configuration (rc) file can be specified once for standard conversion requirements, or a set of rc files can be kept for a various conversion requirements which can then be specified on the command line.

The remaining command line arguments (name_1, name_2, name_3, anonymized_ID, & ID_number) are things that are likely to vary between recordings, and so may be entered each time CSC2MED is launched.

MEF2MED

MEF2MED is software to convert data from MEF2 or MEF3 format files into the MED format.

MEF2MED is run from the command line.

The calling format is as follows:

```
MEF2MED MEF_session_directory output_directory [MEF_password] [RC_file]
```

Calling CSC2MED with no arguments will display the above and exit.

As is customary in command line utilities, arguments in square brackets are optional, and if they are not provided, default values or behavior is used.

MEF2MED Arguments:

MEF_session_directory:

This is required, no default value is assumed. This will be the name of the MED session to be generated.

output_directory:

The path to where the converted MED session will be stored. If none is specified, it will use the "output directory" value from the rc file.

MEF_password:

This is the password or the input MEF file. If the input is not encrypted, this parameter can be left out.

runtime_config_file:

This is the name of a runtime configuration (rc) file, which specifies numerous details about how the conversion should be performed. This file will be described in more detail below. If no runtime_config_file is provided, CSC2MED will use the default runtime configuration file named "CSC2MED_rc.txt" in the current directory. If the default file does not exist, CSC2MED will exit.

Runtime Configuration (rc) File

RC File Structure:

Any line not beginning with "%" is considered a comment and ignored by the function.

An Entry consists of "FIELD", "NOTES", "TYPE", "OPTIONS", "DEFAULT", and "VALUE" labels.

"TYPE", "OPTIONS", "DEFAULT", and "VALUE" labels must exist for each entry.

"NOTES" labels are optional. Any number of "NOTES" lines may appear in an entry. The other labels are restricted to one line.

This order is required in an entry:

```
%% FIELD:
%% NOTES: (optional)
%% TYPE:
%% OPTIONS:
%% DEFAULT:
%% VALUE:
```

Entry "TYPE"s are one of: string, float, integer, ternary.

The string "NO ENTRY" can be entered as a DEFAULT or VALUE where appropriate.

If "NO ENTRY" is entered as the value:

- a) string types will return zero-length strings
- b) float types will return 0.0
- c) integer types will return 0
- d) ternary types will return "UNKNOWN"

"DEFAULT" values are used when there is no text following in the value label or the word "DEFAULT" is specified as the value.

"PROMPT" will ask the user for input and present the OPTIONS and DEFAULT.

The user can also enter the strings "DEFAULT", "NO ENTRY", or any of the defined OPTION strings, as well as free text if appropriate.

"PROMPT" and "NO ENTRY" are not considered OPTIONS, but may serve as values or defaults.

The "OPTIONS" label may be modified to "OPTIONS ONLY", in which case, only the listed options or "NO ENTRY", "DEFAULT", or "PROMPT" are permitted as values.

RC File Contents:

```
%% FIELD: Channel Specification File
%% NOTES: CSC2MED & MEF2MED do not use a CS file at this time, but probably will in
%% NOTES: the future. Currently, this field applies to NRD2MED & DAT2MED
%% NOTES: If not specified (NO ENTRY), all channels will be recorded at the incoming
%% NOTES: sampling frequency, with default filenames.
%% NOTES: Build file with "create_channel_specs" or in spreadsheet or
%% NOTES: text editor program.
%% NOTES: Default output file name from "create_channel_specs" is
%% NOTES: "<program_name>_cs.csv"
%% TYPE: string
%% OPTIONS:
%% DEFAULT: <program_name>_cs.csv
%% VALUE: DEFAULT

%% FIELD: Output Directory
%% NOTES: Path does not include session directory.
%% TYPE: string
%% OPTIONS: CURRENT WORKING DIRECTORY
%% DEFAULT: CURRENT WORKING DIRECTORY
%% VALUE: DEFAULT

%% FIELD: Packet Buffer Seconds
%% NOTES: This applies only to acquisition with segmenting allowed.
%% NOTES: If segmenting is done, the buffer size needed is dependent on the speed of
%% NOTES: the media.
%% NOTES: Points of reference (with 512 channels): slow external drive can require up
%% NOTES: to 90 sec, fast internal SSD requires about 3 sec.
%% NOTES: Just good practice to consume less OS resources, within reason.
%% NOTES: This field does not matter for any of the converters, although they do use a
%% NOTES: buffer. There is no consequence to falling behind, as in acquisition.
%% TYPE: float
%% OPTIONS:
%% DEFAULT: 10.0
%% VALUE: DEFAULT

%% FIELD: Session Exists Behavior
%% NOTES:
%% TYPE: string
%% OPTIONS ONLY: PROMPT IF EXISTS, OVERWRITE, CREATE NEW SEGMENT, APPEND LAST SEGMENT,
%% NOTES: QUIT
%% DEFAULT: PROMPT IF EXISTS
%% VALUE: DEFAULT

%% FIELD: Include Segmented Session Records
%% NOTES: Divide appropriate session-level records into separate files for each
%% NOTES: segment
%% NOTES: SyLg (System Log) and Sgmt (Segment) records will go into the standard
%% NOTES: session records file
%% NOTES: Note (Annotations) and NlxP (Neuralynx Port) records will be put into their
%% NOTES: respective segmented record files, unless this is not enabled
%% TYPE: ternary
```

%% OPTIONS ONLY: YES, NO
%% DEFAULT: YES
%% VALUE: DEFAULT

%% FIELD: Block Samples
%% NOTES: superseded by values in CS file, if specified
%% TYPE: integer
%% OPTIONS: USE BLOCK DURATION
%% DEFAULT: 50000
%% VALUE: DEFAULT

%% FIELD: Block Duration
%% NOTES: superseded by "Block Samples", if both specified
%% NOTES: units of microseconds
%% TYPE: float
%% OPTIONS: USE BLOCK SAMPLES
%% DEFAULT: USE BLOCK SAMPLES
%% VALUE: DEFAULT

%% FIELD: Session Description
%% NOTES:
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: DEFAULT

%% FIELD: Equipment Description
%% NOTES:
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: Neuralynx Atlas / Pegasus acquisition system

%% FIELD: Reference Description
%% NOTES:
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: DEFAULT

%% FIELD: Antialias Filter
%% NOTES: roll off starts at 4 samples per cycle
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: NO
%% VALUE: DEFAULT

%% FIELD: Noise Floor Filter
%% NOTES: reduces noise floor (resulting in better compression)
%% NOTES: do not use on channels with small units whose amplitudes may be near the
%% NOTES: noise floor, fine for EEG
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: NO

%% VALUE: DEFAULT

%% FIELD: Line Noise Filter

%% NOTES: reduces line noise without attenuating physiologic signal in the same band

%% NOTES: this option performs somewhat better with longer blocks - e.g. 10 seconds

%% TYPE: ternary

%% OPTIONS ONLY: YES, NO

%% DEFAULT: NO

%% VALUE: DEFAULT

%% FIELD: Low Frequency Filter Setting

%% NOTES: informational only from input signal, does not apply a highpass filter

%% NOTES: units in Hz

%% TYPE: float

%% OPTIONS:

%% DEFAULT: NO ENTRY

%% VALUE: 0.0

%% FIELD: High Frequency Filter Setting

%% NOTES: informational only from input signal, does not apply a lowpass filter

%% NOTES: units in Hz

%% NOTES: 9 kHz is analog antialiasing filter cutoff in Nlx hardware

%% TYPE: float

%% OPTIONS:

%% DEFAULT: NO ENTRY

%% VALUE: 9000.0

%% FIELD: Notch Filter Frequency Setting

%% NOTES: informational only from input signal, does not apply a notch filter

%% NOTES: units in Hz

%% TYPE: float

%% OPTIONS:

%% DEFAULT: NO ENTRY

%% VALUE: DEFAULT

%% FIELD: AC Line Frequency

%% NOTES: ensure this is set correctly if using Line Noise Filter

%% NOTES: Europe / Africa / Asia / Australia mostly 50 Hz Americas mostly 60 Hz

%% NOTES: units in Hz

%% TYPE: float

%% OPTIONS: 50.0, 60.0

%% DEFAULT: NO ENTRY

%% VALUE: 60.0

%% FIELD: Amplitude Units Conversion Factor

%% NOTES:

%% TYPE: float

%% OPTIONS:

%% DEFAULT: 1.0

%% VALUE: DEFAULT

%% FIELD: Amplitude Units Description

%% NOTES:

%% TYPE: string

%% OPTIONS:
%% DEFAULT: microvolts
%% VALUE: DEFAULT

%% FIELD: Time Base Units Conversion Factor
%% NOTES:
%% TYPE: float
%% OPTIONS:
%% DEFAULT: 1.0
%% VALUE: DEFAULT

%% FIELD: Time Base Units Description
%% NOTES:
%% TYPE: string
%% OPTIONS:
%% DEFAULT: microseconds
%% VALUE: DEFAULT

%% FIELD: Apply Recording Time Offset
%% NOTES:
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: YES
%% VALUE: DEFAULT

%% FIELD: Standard Timezone Acronym
%% NOTES: e.g "MST" for "Mountain Standard Time"
%% NOTES: NOT shorter synonym e.g. "MT" for "Mountain Time"
%% NOTES: NOT the Daylight Saving / Summer Time version e.g. "MDT" for
%% NOTES: "Mountain Daylight Time"
%% NOTES: this field is used define the correct timezone for territories containing
%% NOTES: multiple timezones
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: DEFAULT

%% FIELD: Recording Country
%% NOTES: e.g United States, France, Czech Republic
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: DEFAULT

%% FIELD: Recording Territory
%% NOTES: state, province
%% NOTES: e.g. Montana, Ontario, Queensland
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: DEFAULT

%% FIELD: Recording Locality
%% NOTES: e.g. city, town, village, hamlet

%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: DEFAULT

%% FIELD: Recording Institution
%% NOTES: or organization
%% NOTES: e.g. University of Colorado, Dark Horse Neuro, Inc.
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: DEFAULT

%% Encryption NOTES:
%% LEVEL 0 LEVEL 1 LEVEL 2
%% Level 0 encryption is equivalent to no encryption
%% Level 1 encryption requires a LEVEL 1 password
%% Level 2 encryption requires a LEVEL 2 password
%% Level 2 passwords provide access to both level 1 & 2 encrypted fields
%% Level 2 encryption requires specification of both LEVEL 1 & LEVEL 2 passwords even
%% if LEVEL 1 encryption is not used anywhere in the file

%% FIELD: Metadata Section 2 Encryption Level
%% NOTES: technical data about the recording
%% TYPE: string
%% OPTIONS ONLY: LEVEL 0, LEVEL 1, LEVEL 2
%% DEFAULT: LEVEL 1
%% VALUE: DEFAULT

%% FIELD: Metadata Section 3 Encryption Level
%% NOTES: potentially data-source identifying data
%% TYPE: string
%% OPTIONS ONLY: LEVEL 0, LEVEL 1, LEVEL 2
%% DEFAULT: LEVEL 2
%% VALUE: DEFAULT

%% FIELD: Time Series Data Encryption Level
%% NOTES:
%% TYPE: string
%% OPTIONS ONLY: LEVEL 0, LEVEL 1, LEVEL 2
%% DEFAULT: LEVEL 0
%% VALUE: DEFAULT

%% FIELD: Segment Record Encryption Level
%% NOTES: Typically the same as Metadata Section 2 Encryption Level
%% TYPE: string
%% OPTIONS ONLY: LEVEL 0, LEVEL 1, LEVEL 2
%% DEFAULT: LEVEL 1
%% VALUE: DEFAULT

%% FIELD: Neuralynx Port Record Encryption Level
%% NOTES: Neuralynx parallel port events
%% TYPE: string
%% OPTIONS ONLY: LEVEL 0, LEVEL 1, LEVEL 2

%% DEFAULT: LEVEL 0
%% VALUE: DEFAULT

%% FIELD: Annotation Record Encryption Level
%% NOTES: User entered notes during acquisition
%% TYPE: string
%% OPTIONS ONLY: LEVEL 0, LEVEL 1, LEVEL 2
%% DEFAULT: LEVEL 2
%% VALUE: DEFAULT

%% FIELD: System Log Record Encryption Level
%% NOTES: System entered data during acquisition
%% TYPE: string
%% OPTIONS ONLY: LEVEL 0, LEVEL 1, LEVEL 2
%% DEFAULT: LEVEL 0
%% VALUE: DEFAULT

%% FIELD: Level 1 Password
%% NOTES: NO ENTRY is equivalent to LEVEL 0 encryption (no encryption)
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: L1_password

%% FIELD: Level 1 Password Hint
%% NOTES: presented to user if incorrect password is entered
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: the Level 1 password is 'L1_password'

%% FIELD: Level 2 Password
%% NOTES: NO ENTRY does not prevent LEVEL 1 encryption
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: L2_password

%% FIELD: Level 2 Password Hint
%% NOTES: presented to user if incorrect password is entered
%% TYPE: string
%% OPTIONS:
%% DEFAULT: NO ENTRY
%% VALUE: the Level 2 password is 'L2_password'

%% FIELD: Level 3 Password
%% NOTES: Specification of a Level 3 Password is a failsafe mechanism for retrieval of
%% NOTES: level 1 & level 2 passwords
%% NOTES: The "USE DHN L3 PW" option will encode using a proprietary Dark Horse Neuro
%% NOTES: password. DHN can retrieve lost passwords if this is option is selected.
%% TYPE: string
%% OPTIONS: USE DHN L3 PW
%% DEFAULT: USE DHN L3 PW
%% VALUE: DEFAULT

```

%% FIELD: Include Session-level Segment Records
%% NOTES: tiny footprint, significantly improves segment search performance
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: YES
%% VALUE: DEFAULT

%% FIELD: Include Channel-level Segment Records
%% NOTES: tiny footprint, significantly improves segment search performance
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: YES
%% VALUE: DEFAULT

%% FIELD: Include Neuralynx Port Records
%% NOTES:
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: YES
%% VALUE: DEFAULT

%% FIELD: Neuralynx Port Trigger Mode
%% NOTES:
%% TYPE: string
%% OPTIONS ONLY: NO TRIGGER, ANY CHANGE, HIGH BIT SET
%% DEFAULT: HIGH BIT SET
%% VALUE: DEFAULT

%% FIELD: Neuralynx Port Zero is Reset
%% NOTES: In NO TRIGGER and ANY CHANGE trigger modes, subport values of zero do not
%% NOTES: generate records.
%% NOTES: This allows NO TRIGGER mode to store values only when there the subport is
%% NOTES: non-zero.
%% NOTES: This allows ANY CHANGE mode to store onset value / time pairs only, when
%% NOTES: offset times are not required, reducing the number of records.
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: YES
%% VALUE: DEFAULT

%% FIELD: Number of Subports in Neuralynx Port
%% NOTES: no subports (full 32 bits) == 1 subport in this context
%% TYPE: integer
%% OPTIONS ONLY: 1, 2, 4
%% DEFAULT: 2
%% VALUE: DEFAULT

%% FIELD: Compression Algorithm
%% NOTES: RED: Range Encoded Derivatives => lossless
%% NOTES: PRED: Predictive Range Encoded Derivatives => lossless (slower than RED, but
%% NOTES: better compression)
%% NOTES: MBE: Minimal Bit Encoding => lossless (generally used as fall through
%% NOTES: algorithm for degenerate data)

```


%% NOTES: VDS: Vectorized Data Stream => lossy (highest compression level; see other
%% NOTES: VDS options below)
%% TYPE: integer
%% OPTIONS ONLY: RED, PRED, MBE, VDS
%% DEFAULT: PRED
%% VALUE: DEFAULT

%% FIELD: Compression Algorithm Fall Through
%% NOTES: Currently applies only to RED & PRED compression.
%% NOTES: If MBE would be smaller, block is compressed with MBE.
%% NOTES: Generally only occurs in blocks with degenerate data.
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: YES
%% VALUE: DEFAULT

%% FIELD: VDS Threshold
%% NOTES: Applies only to VDS compression.
%% NOTES: Range 0.0 to 10.0
%% NOTES: 0 is lossless (redirects to PRED), 10 is very lossy.
%% TYPE: float
%% OPTIONS:
%% DEFAULT: 5.0
%% VALUE: DEFAULT

%% FIELD: VDS LFP Filter Cutoff
%% NOTES: Applies only to VDS compression.
%% TYPE: float
%% OPTIONS: USE HIGH FREQUENCY FILTER SETTING, NO FILTER
%% DEFAULT: USE HIGH FREQUENCY FILTER SETTING
%% VALUE: DEFAULT

%% FIELD: VDS Scale by Baseline
%% NOTES: Applies only to VDS compression.
%% NOTES: Scale data by block baseline width. Increases compression by ~25%.
%% NOTES: Creates higher compression & decreased fidelity in proportion to the
%% NOTES: baseline width on a block by block basis.
%% NOTES: Do not use on channels with small units whose amplitudes may be near the
%% NOTES: noise floor.
%% TYPE: ternary
%% OPTIONS ONLY: YES, NO
%% DEFAULT: NO
%% VALUE: DEFAULT

Channel Specification (cs) File

CS File Structure:

The CS file is a plain text file, formatted as comma-separated values (csv). These files can be created or edited in any spreadsheet or word-processing program. The five columns are:

1. Physical Channel Number
2. Channel Name
3. Decimation Frequency
4. Block Samples
5. Channel Description

The column names are not included in the file.

Details:

1. Physical Channel Number
 - This is the input plug number.
 - MED numbering for these starts at 1.
2. Channel Name
 - This is the base name, no extension. The software will add MED extensions.
3. Block Samples
 - The number of samples in a compressed block for the channel.
 - 0 (zero) here will use the value specified in the rc file.
4. Decimation Frequency
 - The sampling frequency to which the channel should be decimated.
 - 0 (zero) here will disable decimation on the channel.
 - If decimation is enabled, a low-pass antialiasing filter will be applied with its roll-off starting at 1/4 the decimation frequency (minimum 4 samples/cycle).
5. Channel Description
 - No entry is required.
 - All punctuation is allowed (including commas).
 - Newlines are not allowed.
 - Maximum of 1023 ascii or 255 UTF-8 characters in length.

Create Channel Specs Utility:

As stated, the CS file is a plain text file, formatted as comma-separated values (csv). The command line utility “create_channel_specs” exists in the NRD2MED / DAT2MED directory to facilitate creation of these files if the user desires. It is launched as:

```
create_channel_specs <return>
```

The utility prompts the user for channel group names, and the other cs information, and builds the file according to that input.

Vectorized Data Stream (VDS) Compression

VDS is a lossy compression scheme that converts time series data into a series of scalable anchors. This data representation can be interpolated at arbitrary resolution. VDS compression is computationally expensive, but decompression is very efficient.

VDS has four main parameters:

1. Threshold
2. Local Field Potential Lowpass Filter Cutoff (LFP High FC)
3. Scale by Baseline
4. Block Samples

Threshold:

This parameter ranges from 0.0 to 10.0 and governs the lossiness of the compression, and is inversely related to the resultant compression ratio (CR):

$$CR = \text{compressed_size} / \text{uncompressed_size} \text{ (usually expressed as a percentage)}$$

A threshold of 0.0 is lossless, and under the hood is redirected to PRED compression. A threshold of 10 is very lossy, and very close to the maximum compression that the algorithm can achieve. Fraction threshold within the range are permitted for finer control (e.g. thresh = 5.5).

Local Field Potential Lowpass Filter Cutoff (LFP High FC):

VDS can optionally perform the following:

1. Excise transients
2. Low pass filter the resultant trace with the user specified LFP high fc
3. Restore the transients to the filtered signal

The result of this process is to reduce the baseline variance between transients, and thus provide higher compression. LFP filtering is optional.

Scale by Baseline:

This is a boolean parameter that scales the data by a measure of the baseline amplitude for each MED block. The baseline width is a good surrogate for the margin of error in the amplitude of the signal at each point in time. Scaling by baseline reduces signal fidelity by this margin or error resulting in better compression. In practice the compression increases by about 30% with only minor reductions in signal fidelity. The computational cost is negligible.

Block Samples:

This parameter should typically be much larger than for lossless compression because the number of stored samples in VDS is dramatically reduced compared to the lossless compression algorithms. Lower stored samples results in poorer statistical models, and a proportionately greater volume of model data. This number can be tailored to your needs, but 10 second VDS blocks are a good rule of thumb.

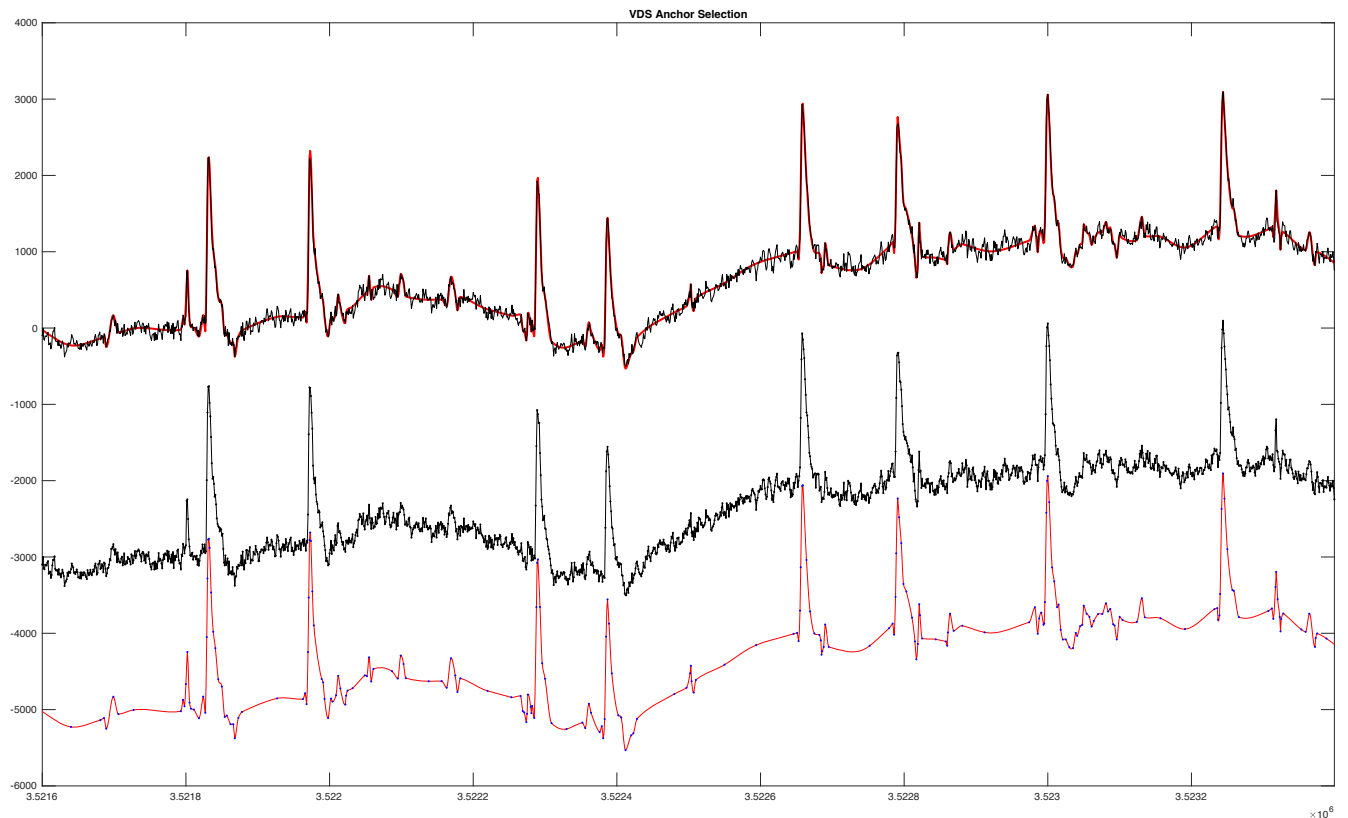
VDS Compression Table:

(compression ratios based on 16-bit input data)

Compression Ratio Comparison (384 channels, 30 KHz, 10 sec blocks, run on 8 Core 3 GHz Intel Xeon with 32 GB RAM & 2 TB internal SSD)

VDS Threshold (lossiness)	LFP Cutoff	Compression Speed (x faster than real time)	Compression Ratio	Compression Ratio (blocks scaled by baseline)
Lossless (-AA, -NFF)	n/a	49.18	37.01%	
Lossless (+AA, -NFF)	n/a	20.34	32.01%	
Lossless (-AA, +NFF)	n/a	18.97	23.65%	
Lossless (+AA, +NFF)	n/a	18.94	22.92%	
1	No Filter	1.40	27.26%	18.30%
2	No Filter	1.61	24.38%	17.04%
3	No Filter	1.80	21.60%	15.47%
4	No Filter	1.99	18.80%	13.68%
5	No Filter	2.24	16.00%	11.78%
6	No Filter	2.63	13.20%	9.94%
7	No Filter	2.63	10.40%	8.03%
8	No Filter	2.32	7.60%	6.01%
9	No Filter	1.98	4.80%	3.89%
10	No Filter	1.54	2.00%	1.65%
1	500	2.44	10.02%	7.31%
2	500	2.47	9.12%	6.70%
3	500	2.56	7.95%	5.89%
4	500	2.64	7.08%	5.28%
5	500	2.72	6.03%	4.53%
6	500	2.80	5.02%	3.80%
7	500	2.87	4.00%	3.04%
8	500	2.88	3.00%	2.30%
9	500	2.69	2.00%	1.54%
10	500	1.94	1.00%	0.81%
1	200	1.23	8.53%	6.11%
2	200	2.51	7.75%	5.59%
3	200	2.56	6.82%	4.96%
4	200	2.64	6.09%	4.46%
5	200	2.72	5.27%	3.89%
6	200	2.80	4.40%	3.28%
7	200	2.85	3.56%	2.67%
8	200	2.87	2.70%	2.04%
9	200	2.74	1.78%	1.35%
10	200	2.31	0.94%	0.76%

Typical VDS Lossy Waveform Results

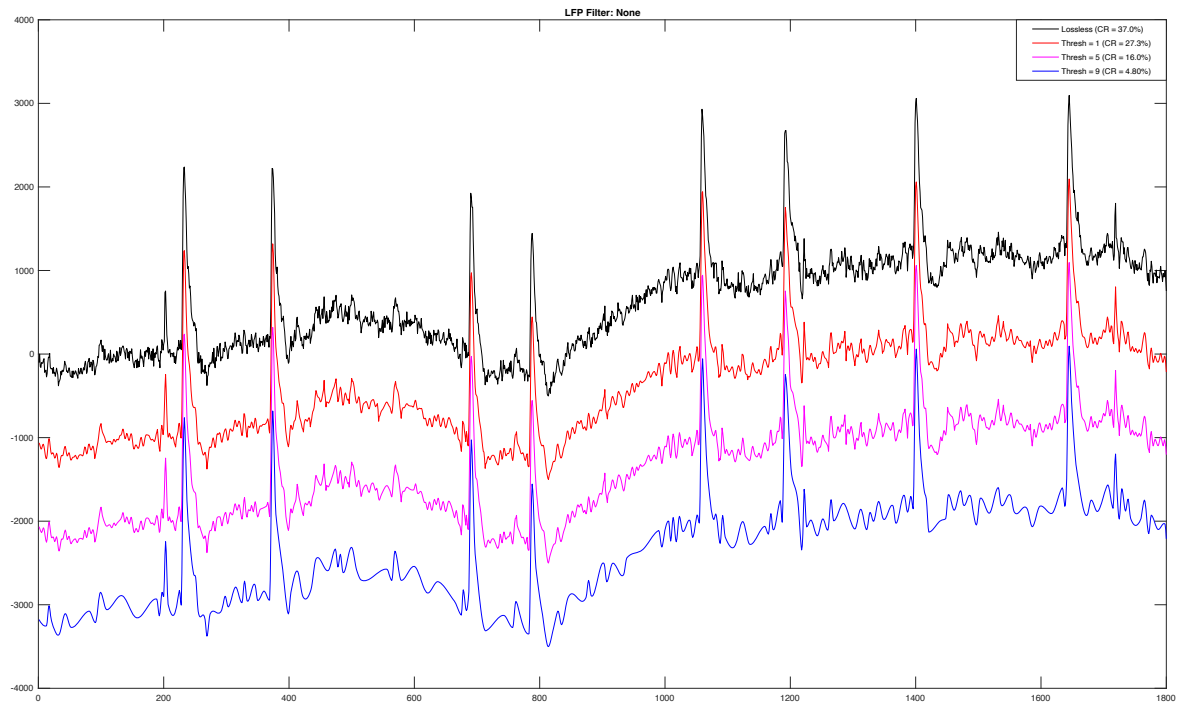


VDS Algorithm: Anchor Selection

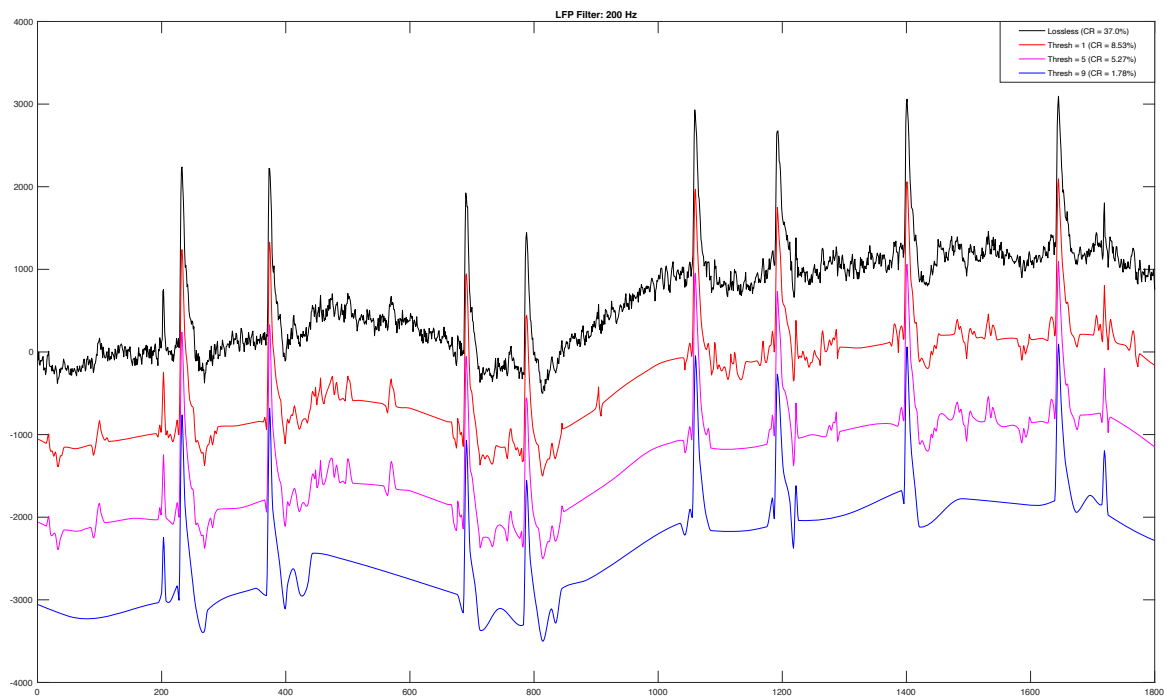
(Compressed with thresh = 5, LFP high fc = 500 Hz)

The top traces are the raw data (black), & the VDS data (red) overlaid on it. The middle trace (black line) is the raw data with the sample points displayed (black dots). The bottom trace is the interpolated VDS data (red line) with the anchor points displayed (blue dots).

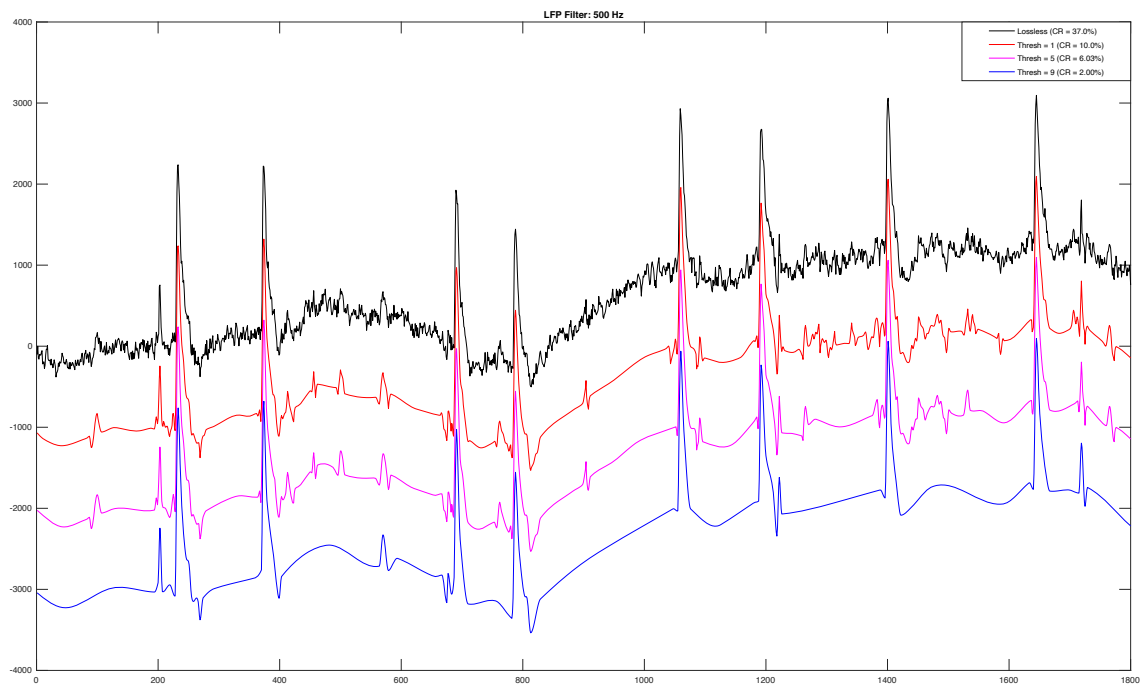
VDS Algorithm: No LFP Filter (thresholds = 1, 5, & 9)



VDS Algorithm: LFP Filter @ 200 Hz (thresholds = 1, 5, & 9)



VDS Algorithm: LFP Filter @ 500 Hz (*thresholds = 1, 5, & 9*)



VDS Algorithm: Scale by Baseline (*raw data*)

