Mesoscale Individualized NeuroDynamic (MINDy) Modeling

Matthew F. Singh*, Todd S. Braver, ShiNung Ching
Washington University in St. Louis
*f.singh@wustl.edu

Abstract

This document provides instructions for performing MINDy model-fitting in MATLAB as described in the paper "Individualized Dynamic Brain Models: Estimation and Validation with Resting-State fMRI". Note that the provided code generally makes heavy use of binary implicit expansion (R2016b or later).

1 Using MINDy with non-HCP Scan Sequences

Most of the included files are general purpose. However, the function "MINDy_Simple" calls a script ("Chosen_ParSTR) which generates the hard-coded structures "ParStr" and "Pre". These structures contain hyperparameters used in the original MINDy paper, including the scanning TR (720ms in Pre.TR). If a different scanner TR is used, the field Pre.TR should be modified accordingly (measuring TR in unit sesconds) and used as an input argument to the function "MINDy_FULL" (see below). Depending upon the amount and quality of data, it may also be necessary for users to modify the regularization hyperparameters. In the case of less data or much larger parcellations (e.g. over one-thousand regions) it may be necessary to increase the regularization parameters: ParStr.SpScale and ParStr.SpPls1/ParStr.SpPls2 (it is recommended to always use ParStr.SpPls1=ParStr.SpPls2). Doing so will prevent over-fitting. By contrast, when more/higher quality data is used than HCP or when a much smaller parcellation is chosen, it may be possible to decrease the regularization parameters which will decrease regularization bias leading to potentially more accurate models.

2 WARNING: MINDy_Simple uses HCP parameters

The incuded files include both low-level functions that perform the actual model estimation and two high-level functions. The high-level function MINDy_Simple is hard-coded to use the parameters used in the paper which originally introduced MINDy: "Individualized Dynamic Brain Models: Estimation and Validation with Resting-State fMRI" which uses single-session (1 day=2400 TRs) HCP data with a 419-parcel atlas. If you are using a different scanning acquisition sequence than HCP you will need to

3 Data and Formats

For the highest-level MINDy functions (e.g. "MINDy_Simple") only one argument needs to be supplied: the data time series (see below). For lower-level functions, there are three main types of inputs (excluding options): data cells/matrices (two arguments), a structure containing model-fitting hyper-parameters ("ParStr') and a structure containing information for performing the deconvolution ("Pre").

3.1 Data

All data is assumed to be formatted as **region** $\times$ **time**. Data can be input as a single matrix for a single continuous recording or as a cell containing multiple matrices (e.g. multiple recording segments). In general, we assume that data for each channel has already been z-scored–otherwise the hyper-parameters may need to be adjusted to compensate for the new data range. For the highest-level MINDy functions (e.g. MINDy_Simple) data is input from the original time series as a single argument (one matrix or one cell) prior to hemodynamic deconvolution. The function performs hemodynamic deconvolution and derivative estimation automatically. These functions then feed this information into lower-level functions (e.g. MINDy_Base) which use the deconvolved and differentiated data to perform model-fitting. Alternatively these lower-level functions can be accessed directly with data that has already been appropriately processed. In this case there are two input data arguments: a deconvolved time series and the temporal derivative of the deconvolved time series. Both of these time-series should have the same length. Accessing these functions directly allows the user to customize the method of differentiation/deconvolution as opposed to the default method of Weiner-deconvolution with finite-difference derivative estimation.

Don't Worry! Hyperparameters only need to be passed when using Lower-Level MINDy functions (e.g. for customization).

Even then, you won't need to change most of them.

3.2 Hyperparameters: ParStr

The hyperparameters to the model-fitting are passed to all MINDy functions as a structure ("ParStr"). This structure can be generated using the script: "ChosenParSTR" which will also populate the fields with the default values used for fitting HCP models as in the original publication. There are two exceptions: "ChosenParSTR" does not automatically include the values for the size of each training batch ("BatchSz") or the number of batches ("NBatch"). Within the high-level MINDy functions (e.g. MINDy_Simple) these are automatically set as in the paper, whereas users will need to set them manually to use lower-level functions–this omission is intended as a reminder to set the training length appropriately as the parameter most likely to need adjustment between use-cases (data quality, amount of data, etc.).

The relation between ParStr fields and variables in the original paper is as follows:

Field	Symbol in Paper	Meaning
SpScale	λ_1	Sparse Regularization
SpDiag	λ_2	Recurrent Regularization
SpPls1	λ_3	Low-Rank Regularization (Pre)
SpPls2	λ_3	Low-Rank Regularization (Post)
L2SpPlsEN	λ_4	L2-Regularization of Low-Rank
Dmin	Dmin	Minimal value of D
Bmin	b^{-1}	Fixed b-parameter of transfer fct.
wPC	k	Maximum rank of W_L
Dec	μ	NADAM AR-1
Dec2	ν	NADAM AR-2
Reg	$\varepsilon(W)$	NADAM Regularization for W (all W's)
AReg	$\varepsilon(\alpha)$	NADAM Regularization for α
DRegScale	$\varepsilon(D_2)/\varepsilon(W)$	NADAM Reg. for D_2 rel. to W
Rate	Rate(W_S)	NADAM update rate for W
ARate	Rate(α)	NADAM update rate for α
wk1Rate	Rate(W_1)/Rate(W_S)	Relative Rate for W_1
wk2Rate	Rate(W_2)/Rate(W_S)	Relative Rate for W_2
DRate	Rate(D_2)/(Rate(W_S)×DRegScale)	Relative Rate for D_2
Dstart	min($D_2 seed$)	Minimal initial seeding for D_2
RandScale	SD($W seed$)	STD of initial seeding for all $W's$

3.3 Hyperparameters: Pre

The "Pre" structure, like "ParStr" is generated by evaluating the script "ChosenParSTR". Pre contains information used for performing the deconvolution step which is automatically performed by "MINDy_PreProc"

Field	Meaning
FiltAmp	Cut-off threshold in SD's for censoring spikes (linear interpolation)
TR	Temporal Resolution in seconds
ConvLevel	SNR filtering parameter for Weiner-deconvolution

4 Functions

The functions included with MINDy include the functions for fitting models, processing/deconvolving data, simulating fit models, and analyzing the quality of fits.

4.1 High-Level Fitting Functions (Where to start)

MINDy contains two main high-level-functions: MINDy_Simple and MINDy_FULL. MINDy_Simple only needs a single input argument: The parcellated time-series data formatted as $region \times time$. For a single recording-session data can be input as a matrix or a cell containing a single matrix or as a cell containing one matrix per recording segment. MINDy_Simple also allows an optional input to specify whether preprocessing (deconvolution/filtering) should be performed with admissable values 'y' (default) or 'n'.

The function MINDy_FULL allows greater customization. Its first three input arguments correspond to the data (same format as MINDy_Simple) and the ParStr, Pre fields (which can be generated for the published parameters using "ChosenParSTR" and modified from there). The remaining arguments in MINDy_FULL correspond to additional pre/post-processing options. These arguments and their admissable values are as follows:

Argument	Values	Effect
doSplit	'n'	Define $dx_t = x_{t+1} - x_t$ (advised for large TR/low noise)
	'y'	Define $dx_t = (x_{t+2} - x_t)/2$ (advised for small TR/high noise)
doPreProc	'n'	Don't perform deconvolution/censoring
	'y'	Do perform deconvolution/censoring
doInflate	'n'	Don't readjust for regularization bias
	'y'	Do readjust (advised)
doSmooth	'n'	Don't smooth timeseries
	'y'	Convolve time-series with [1 1] (after preprocessing)
doZ	'n'	Don't z-score
	'y'	Z-score each parcel (advised)

4.2 MINDy Outputs

All fitting functions produce a single structure for output with similar formatting. In all cases, this structure contains the field "Param" which contains parameter values in the cell ordering: $\{W_S, \alpha, b, c, W, D\}$. Here, W_S denotes the sparse components of the weights and the low-rank component can be reconstructed as $W_L = W - W_S$. The c element refers to a constant-drive term which is currently fixed at zero. The b term contains two elements: the slope variable denoted b in the paper (left column) and a shift-term for the transfer function (right column) which is currently fixed at zero.

In terms of their phenomenological significance, the parameters are as follows:

Weights (W): Param$\{5\}$

Curvature (α):Param$\{2\}$

Decay (D): Param$\{6\}$

All MINDy fitting functions also output the field: BadDerivs which gives the time points of each frame censored based upon derivative amplitudes and the field: GoodFrames which contains the indices of the remaining (uncensored) time points. BadDerivs gives indices relative to the concatenated time-series in its first cell, whereas GoodFrames has separate cells of indices for each recording session (e.g. scanning run). The field "oLength" gives the original length of each time-series before censoring. The field "Mat" contains the cell: $\{W_L, W_1, W_2^T\}$ in which W_L is the low-rank component of the weights with $W_L = W_1 W_2^T$. The field "GLMweights" gives the global scaling factor for the W and D terms for the regularization-compensating regression. These factors will be already factored into the Param field, but not the Mat field. Values before global scaling are contained in the fields "oldDecay" and "oldW1".