RMT Class Documentation

This document describes the `RMT.m` class and the associated script `Fig_1_RMT_examples.m`, which implement Random Matrix Theory (RMT) examples from Harris et al. (2023).

Reference: Harris, I. D., Meffin, H., Burkitt, A. N. & Peterson, A. D. H. Effect of sparsity on network stability in random neural networks obeying Dale's law. *Phys. Rev. Res.* 5, 043132 (2023).

Overview

The `RMT` class provides a MATLAB implementation for constructing random connectivity matrices following Dale's Law with sparsity ($0 < \alpha \leq 1$). This framework generalizes classical results for fully connected networks to sparse networks where neurons function as either excitatory or inhibitory.

Key theoretical results implemented here include: 1. **Linear Scaling of Outlier**: The structural E-I imbalance (eigenvalue outlier λ_O) scales *linearly* with sparsity α. 2. **Non-linear Scaling of Radius**: The eigenspectral radius $\mathcal{R}$ and density depend on a *non-linear interaction* between sparsity α and the population means and variances. 3. **Local Outlier Control**: Control mechanisms (ZRS/SZRS) for "local" or "rogue" eigenvalues that escape the bulk spectrum due to low-rank structure that violates the i.i.d. assumption.

Key Equations from Harris (2023)

Network Model (Eq. 1-2)

The neural network dynamics are governed by a continuous rate model similar to those introduced by Hopfield (1982, 1984), found to be chaotic by Sompolinsky et al. (1988), and further analyzed by Harris et al. (2023) among others. Tau is assumed to be 1 in this class.

$$\frac{dx_i(t)}{dt} = -\frac{x_i(t)}{\tau} + \sum_{j=1}^{N} w_{ij}\phi(x_j(t)) \quad \text{(Eq. 1)}$$

Connectivity Matrix Structure (Eq. 6, 11-12)

The general sparse connectivity matrix with Dale's Law structure:

$$W = S \circ (AD + M) \quad \text{(Eq. 6)}$$

where $\circ$ denotes the Hadamard (element-wise) product. The components are:

Diagonal variance matrix (Eq. 11):

$$D = \text{diag}(\underbrace{\tilde{\sigma}_e, \ldots, \tilde{\sigma}_e}_{Nf \text{ times}}, \underbrace{\tilde{\sigma}_i, \ldots, \tilde{\sigma}_i}_{N(1-f) \text{ times}})$$

Low-rank mean structure (Eq. 12):

$$\mathbf{u} = (1, \ldots, 1)^\top, \quad \mathbf{v} = (\underbrace{\tilde{\mu}_e, \ldots, \tilde{\mu}_e}_{Nf \text{ times}}, \underbrace{\tilde{\mu}_i, \ldots, \tilde{\mu}_i}_{N(1-f) \text{ times}})^\top$$

$$M = \mathbf{u}\mathbf{v}^\top$$

Sparse Statistics (Eq. 15-16)

The sparsity-adjusted mean for each population $k \in \{e, i\}$:

$$\mu_{sk} = \alpha\tilde{\mu}_k \quad \text{(Eq. 15)}$$

The sparsity-adjusted variance:

$$\sigma_{sk}^2 = \alpha(1-\alpha)\tilde{\mu}_k^2 + \alpha\tilde{\sigma}_k^2 \quad \text{(Eq. 16)}$$

Note that for sparse networks, the effective variance depends on both the pre-sparsity mean $\tilde{\mu}_k$ and variance $\tilde{\sigma}_k^2$.

Outlier Eigenvalue and Spectral Radius (Eq. 17-18)

The outlier eigenvalue (when E-I imbalance exists):

$$\lambda_O = N[f\mu_{se} + (1-f)\mu_{si}] \quad \text{(Eq. 17)}$$

The spectral radius of the bulk eigenvalue disc:

$$\mathcal{R} = \sqrt{N[f\sigma_{se}^2 + (1-f)\sigma_{si}^2]} \quad \text{(Eq. 18)}$$

Zero Row Sum (ZRS) Conditions

A key contribution of Harris (2023) is the treatment of "local" eigenvalue outliers—eigenvalues that escape the bulk spectral disc boundaries despite the network being nominally balanced. These outliers arise from the low-rank structure imposed by Dale's Law which violates the independent and identically distributed (i.i.d.) assumption of random matrix theory. Note that the original Zero Row Sum (ZRS) condition was introduced by Rajan and Abbott (2006). These outliers are distinct from the "global" outlier λ_O (which results from structural E-I imbalance) and can destabilize dynamics by crossing the stability boundary prematurely.

The class implements three specific conditions to manage these outliers:

1. ZRS (Dense Zero Row Sum, Eq. 24-25)

For **dense, balanced** matrices ($\alpha = 1$), a projection operator P enforces a strict zero row sum on the random component AD.

$$P = I_N - \frac{\mathbf{u}\mathbf{u}^\top}{N} \quad \text{(Eq. 24)}$$

The connectivity matrix becomes $W = ADP + \mathbf{u}\mathbf{v}^\top$ (Eq. 25). This algebraic projection ensures $ADPu = \mathbf{0}$, strictly confining all random eigenvalues to the bulk. However, this operation "fills in" zero entries, making it unsuitable for sparse matrices as it **destroys sparsity**.

2. SZRS (Sparse Zero Row Sum, Eq. 30-31)

For **sparse, balanced** matrices, an algebraic projection is impossible without destroying sparsity. Instead, a numerical **Sparse Zero Row Sum (SZRS)** condition is applied. This method enforces a zero row sum by subtracting the row-average from *only* the nonzero elements, preserving the sparsity pattern S.

$$W = S \circ (AD + \mathbf{u}\mathbf{v}^\top) - B \quad \text{(Eq. 30)}$$

where $B_{ij} = S_{ij}\bar{W}_i$ is the correction term derived from the row average $\bar{W}_i$ of nonzero elements (Eq. 31). This effectively acts as a "tight" dynamic balance condition.

3. Partial SZRS (Eq. 32)

For **sparse, unbalanced** matrices, applying the full SZRS would force all row sums to zero, inadvertently removing the global structural imbalance (and thus the global outlier λ_O) that defines the network's excitatory/inhibitory dominance.

To solve this, the **Partial SZRS** applies the correction *only* to the random component $J = S \circ AD$, leaving the structural mean component $S \circ M$ (where $M = \mathbf{u}\mathbf{v}^\top$) intact.

$$\bar{J}_i = \frac{\sum_j J_{ij}}{\sum_j S_{ij}} \quad \text{(Eq. 32)}$$

The final matrix is $W = (S \circ AD - B_{partial}) + (S \circ M)$. This critical distinction allows the user to control the radius of the bulk spectrum (removing local outliers) while preserving the global mean structure responsible for λ_O.

RMT Class Properties

Property Table

Property	Symbol	Type	Description
`N`	N	Public	System size (number of neurons)
`alpha`	α	Public	Sparsity/connection probability ($0 < \alpha \leq 1$)
`f`	f	Public	Fraction of excitatory neurons
`mu_tilde_e`	$\tilde{\mu}_e$	Public	Normalized mean of excitatory population ($\tilde{\mu}_e = \mu_e/\sqrt{N}$)
`mu_tilde_i`	$\tilde{\mu}_i$	Public	Normalized mean of inhibitory population ($\tilde{\mu}_i = \mu_i/\sqrt{N}$)
`sigma_tilde_e`	$\tilde{\sigma}_e$	Public	Normalized std dev of excitatory population ($\tilde{\sigma}_e = \sigma_e/\sqrt{N}$)
`sigma_tilde_i`	$\tilde{\sigma}_i$	Public	Normalized std dev of inhibitory population ($\tilde{\sigma}_i = \sigma_i/\sqrt{N}$)
`A`	A	Public	Base random matrix (Gaussian, mean 0, variance 1)
`S`	S	Public	Sparsity mask (logical matrix)
`zrs_mode`	-	Public	ZRS mode: `'none'`, `'ZRS'`, `'SZRS'`, or `'Partial_SZRS'`
`shift`	-	Public	Scalar diagonal shift for eigenvalues
`description`	-	Public	Text description for labeling
`outlier_threshold`	-	Public	Multiplier for $\mathcal{R}$ to determine outlier eigenvalues (default 1.04)
`E`	$\mathcal{E}$	Dependent	Logical index for excitatory neurons
`I`	$\mathcal{I}$	Dependent	Logical index for inhibitory neurons
`u`	$\mathbf{u}$	Dependent	Left vector for low-rank structure: $\mathbf{u} = \text{ones}(N, 1)$ (Eq. 12)
`v`	$\mathbf{v}$	Dependent	Right vector for low-rank structure (Eq. 12)
`D`	D	Dependent	Diagonal variance matrix (Eq. 11)
`W`	W	Dependent	Connectivity/weight matrix (Eq. 6, 25, 30)
`mu_se`	μ_{se}	Dependent	Sparse excitatory mean: $\mu_{se} = \alpha\tilde{\mu}_e$ (Eq. 15)
`mu_si`	μ_{si}	Dependent	Sparse inhibitory mean: $\mu_{si} = \alpha\tilde{\mu}_i$ (Eq. 15)
`sigma_se_sq`	σ^2_{se}	Dependent	Sparse excitatory variance (Eq. 16)

Property	Symbol	Type	Description
`sigma_si_sq`	σ_{si}^2	Dependent	Sparse inhibitory variance (Eq. 16)
`lambda_O`	λ_O	Dependent	Outlier eigenvalue prediction (Eq. 17)
`R`	$\mathcal{R}$	Dependent	Spectral radius prediction (Eq. 18)

Property Notes

- **Tilde notation**: Properties with `_tilde` suffix follow Harris (2023) convention where $\tilde{\mu} = \mu/\sqrt{N}$ and $\tilde{\sigma} = \sigma/\sqrt{N}$. These are the "pre-sparsity" parameters.
- **Dependent properties**: These are computed on-the-fly from the stored properties. Accessing `W`, for example, constructs the full weight matrix based on the current `zrs_mode`.
- **Eigenvalue caching**: The class maintains a private cache (`eigenvalues_cache`, `eigenvalues_valid`) to avoid recomputing eigenvalues unless parameters change.

RMT Class Methods

Constructor

- `RMT(N)`: Creates an RMT object with system size `N`. Initializes with default values: `alpha=1` (dense), `f=0.5`, zero means, and unit variance scaled by $1/\sqrt{N}$.

Parameter Setters

- `set_params(mu_tilde_e, mu_tilde_i, sigma_tilde_e, sigma_tilde_i, f, alpha)`: Set all population parameters at once.
- `set_alpha(alpha)`: Set sparsity independently. Automatically regenerates the sparsity mask `S`.
- `set_zrs_mode(mode)`: Set the ZRS mode. Valid options: `'none'`, `'ZRS'`, `'SZRS'`, `'Partial_SZRS'`.

Computation Methods

- `compute_sigma_tilde_i_for_target_variance(target_variance)`: Computes the value of `sigma_tilde_i` needed to achieve a desired total variance $\mathrm{Var}(W)$, given the current `sigma_tilde_e` and `f`. Only supports dense matrices ($\alpha = 1$). Uses Eq. 14 and 16.
- `update_sparsity()`: Regenerates the sparsity mask `S` based on current `alpha`.
- `get_eigenvalues()`: Returns eigenvalues of `W`, using cached values if valid.
- `compute_eigenvalues()`: Forces recomputation and updates the cache.
- `eigenvalues()`: Property-like accessor for eigenvalues.

Display and Visualization

- `display_parameters()`: Prints a formatted summary comparing set parameters, theoretical predictions (Eq. 15-18), and measured statistics from the constructed matrix.
- `plot_spectrum(ax)`: Plots the eigenvalue spectrum on the specified axes. The theoretical radius circle (Eq. 18) is shown. Eigenvalues are colored by their relationship to the bulk:
  - Black circles: interior eigenvalues (within $\mathcal{R}$)
  - Black X marks: near-outliers (between $\mathcal{R}$ and `outlier_threshold * R`)
  - Green filled circles: far outliers (beyond `outlier_threshold * R`)

Utility Methods

- `copy()`: Creates a deep copy of the RMT object, including the random matrices `A` and `S`.

Example Script: Fig_1_RMT_examples.m

The script `Fig_1_RMT_examples.m` demonstrates various configurations of the `RMT` class, illustrating the theoretical concepts from Harris (2023).

Examples Created

1. **(a) Dense random matrix, unbalanced with global outlier**: 100% excitatory ($f = 1$), fully connected ($\alpha = 1$), with a small positive mean offset creating a global outlier eigenvalue.
2. **(b) Dense balanced shifted**: Removes the mean offset (balanced) and applies a diagonal shift of $-\mathcal{R}$ to move the spectral disc to the left, ensuring all eigenvalues have negative real parts (stable).
3. **(c) Dense balanced Dale's law**: Introduces the Dale's law structure with positive excitatory means ($\tilde{\mu}_e = 1/\sqrt{N}$) and negative inhibitory means ($\tilde{\mu}_i = -1/\sqrt{N}$), with $f = 0.5$.
4. **(d) Dense balanced Dale's ZRS**: Applies the projection-based ZRS (Eq. 24-25) to control local outliers while maintaining the balanced network structure.
5. **(e) Dense unbalanced Dale's different sigmas ZRS**: Demonstrates using `compute_sigma_tilde_i_for_target_varian` to set different standard deviations for E and I populations while maintaining a target total variance.
6. **(f) Sparse unbalanced with Partial SZRS**: Introduces sparsity ($\alpha = 0.5$) and applies Partial SZRS (Eq. 32) to control local outliers while preserving the E-I imbalance.

Script Workflow

1. Creates a structure array `G` to hold multiple RMT examples
2. Each example copies from the previous (using `copy()`) and modifies specific parameters
3. Computes eigenvalues for all examples
4. Plots all spectra in a single figure with consistent axis scaling
5. Formats axes with custom styling (removed default axes, added Re/Im labels)
6. Adds letter labels (a), (b), etc. to each subplot
7. Optionally saves figures in multiple formats

Related Files

- **Source Code**: `RandomMatrixTheory/RMT.m`
- **Example Script**: `RandomMatrixTheory/Fig_1_RMT_examples.m`