EigenEdge package for MATLAB

Edgar Dobriban*

April 26, 2015

Contents

1 Introduction **1**

2 Installation **2**

3 Example **2**

3.1 Background . . . 2

3.2 Quick example . . . 2

3.3 Longer example . . . 3

3.4 More examples . . . 5

4 Computational methods **6**

4.1 Atomic: computing the limit spectrum . . . 6

4.2 Moments and Quantiles of the ESD . . . 7

4.3 Utilities . . . 7

4.3.1 Arithmetic and geometric models for eigenvalues . . . 7

1 Introduction

The EigenEdge MATLAB package contains open source implementations of methods for working with eigenvalue distributions of large random matrices. In particular, it contains the Atomic method to compute the limit empirical spectrum of sample covariance matrices (proposed in Dobriban, 2015).

- Version: 0.0.1
- Requirements: Tested on MATLAB R2014b.
- Author: Edgar Dobriban
- License: `GPL-3`

In addition, this package contains the code to reproduce all simulation results from the paper Dobriban (2015). These are contained in the `\Experiments\Atomic` folder.

*Department of Statistics, Stanford University, `dobriban@stanford.edu`

2 Installation

Extract the archive in any folder, say to `<path>` . The main functions are in the `Code` directory, which needs to be on the Matlab path, along with all of its subfolders. This can be accomplished in at least three ways. First, you can add the following lines to your Matlab startup:

```
addpath('<path>/pvalue_weighting_matlab/Code')
addpath('<path>/pvalue_weighting_matlab/Code/Basic')
addpath('<path>/pvalue_weighting_matlab/Code/External Helper Code')
```

The second option is to add those line to scripts that call functions in this package. The third option is to execute the `setpaths.m` script every time you start a new session with the EigenEdge package. This will include the current folder in the Matlab search path.

An example computation is in the `\Experiments\Examples\example.m` file. This is described in Section 3.

This file is the main documentation for the package. To start, look at the example (Section 3) or at the methods implemented (Sections 4).

3 Example

Here we give a simple example of using the Atomic function for computing limit spectra of sample covariance matrices. We refer to the paper Dobriban (2015) for a full description of the method. The code for reproducing this example is available in the `\Experiments\Examplesexample.m` file in our package.

3.1 Background

We begin with the problem statement (borrowed from our paper). Consider a large $n \times p$ matrix $\mathbf{X}$, whose rows x^i are independent random vectors. Suppose that x^i are mean zero, and their covariance matrix is the $p \times p$ matrix $\Sigma = \mathbb{E} x^i x^{i\top}$. To estimate Σ, we form the sample covariance matrix

$$\widehat{\Sigma} = \frac{1}{n}\mathbf{X}^\top \mathbf{X}.$$

If n and p are of comparable size, then $\widehat{\Sigma}$ deviates substantially from the true covariance. The asymptotic theory of random matrices describes the behavior of the eigenvalues of $\widehat{\Sigma}$ as n, p grow large proportionally (e.g. Bai and Silverstein, 2009). It is well known that, if the distribution of the eigenvalues of Σ tends to a limit population spectral distribution (SD) H, as $n, p \to \infty$ and the aspect ratio $p/n \to \gamma$, then the random eigenvalue distribution of $\widehat{\Sigma}$ also tends to a deterministic limit empirical spectrum F (Marchenko and Pastur, 1967; Silverstein, 1995). This limit differs from the true spectrum.

3.2 Quick example

The Atomic method is designed to compute the limit sample spectrum F from the limit population spectrum H. It is easy to use: in its simplest form, we need to provide a vector of population eigenvalues `t` (in this case H is a uniform mixture of point masses at t_i), and an aspect ratio `gamma`, and call `atomic(t,gamma)`. Atomic will return a grid and pointwise density estimates of the limit spectrum F on this grid.

The code below (excerpted from `\Experiments\Examplesexample.m`) shows an example, in which the eigenvalues are placed at `t = [1; 5]`, and the aspect ratio `gamma = 1/2`:

```
t = [1; 5]; %location of population eigenvalues: H = 1/n\sum delta_{t_i}
gamma = 1/2; %aspect ratio gamma = p/n
[grid, density] =  atomic(t, gamma); %compute limit spectrum
```

The density can be plotted with

```
figure, plot(grid, density,'r','LineWidth',4) %plot
xlabel('Eigenvalue')
ylabel('Density');
```

The results are shown below:

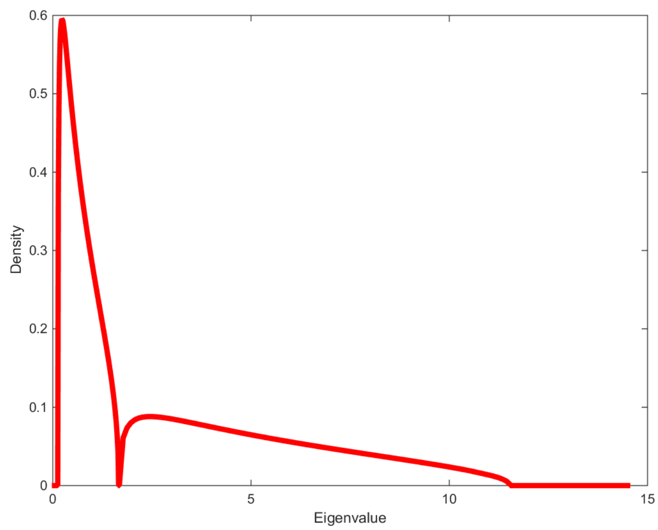

Figure 1: Density of the limit spectrum

In this case the limit spectrum F has two components that barely separate. Using ATOMIC, we could perform experiments by changing the input parameters `t,gamma`, to examine precisley when the two clusters separate. This is, however, beyond the scope of the current example.

3.3 Longer example

In this example, we compute the limit sample spectrum of a more complicated population spectrum, which is a mixture of point masses and a continuous uniform distribution. This example was also presented in our paper Dobriban (2015) as an illustration of the power of our method ATOMIC. We are not aware of any other algorithm that can compute this quantity with the same speed and accuracy.

Specifically, we set H a mixture

$$H = \sum_{i=1}^{J} w_i \delta_{t_i} + \sum_{t=1}^{T} w_t^* U_{a_t,b_t},$$

with $J = 10$ population point masses at $t_1 = 2$,$t_2 = 3, \ldots, t_{10} = 11$ with weights forming an arithmetic progression with step $r = 0.005$: $w_1 = 0.0275, w_2 = 0.0325, \ldots, w_{10} = 0.0725$. The second component, with weight $w_1^* = 1/2$, is a uniform distribution - or a 'boxcar' - on $[a_1, b_1] = [0.5, 1.5]$. We also set $\gamma = 0.01$.

The density of the limit empirical spectrum is shown in Figure 2. The computation takes a few seconds on a desktop computer. *A priori* it is not obvious how many clusters there are in the SD and what their shape and geometric properties are. ATOMIC provides a precise approximation of the density of the limit spectrum. From this several insights can be derived: there are 11 clusters in total, and the two rightmost ones just barely separate; the height of the clusters decreases while the width of the point-mass clusters increases.

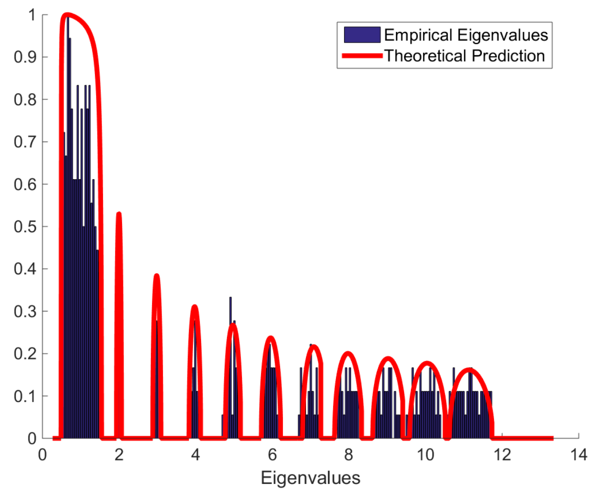

Figure 2: Boxcar + point mass mixture example: ATOMIC computes the density of the limit spectrum (normalized to have maximum equal to 1 for display purposes).

We will now explain how to compute this example with our software. This code is also available in the `\Experiments\Examples\example.m` file.

First we should generate the parameters of the spectrum. We represent atomic distributions by two vectors: one for the locations of the eigenvalues, and one for the weights of the corresponding point masses. These will be denoted by `w,t`, respectively. In our paper, we introduced an arithmetic model for eigenvalues

$$H_{\text{atomic}} = \sum_{j=0}^{J-1} (a+jb)\delta_{c+jd}.$$

The eigenvalues are placed at $c + jd$, for some $c > 0$ and $d \in \mathbb{R}$ such that $c + jd > 0$ for all j. They have weights $a + jb$ for some $a, b > 0$. The constants a, b are constrained so that the sum of the weights is 1, thus only one of them, say b, is a free parameter. This is a flexible model giverned by only three parameters.

We provide a specialized function `arithmetic_model()` to easily generate the corresponding vectors t, w. This function is parametrized in the following way: the arithmetic sequence t is specified by its smallest and largest elements `t_min,t_max`, and the length of the sequence `J`. The weights are specified by the *increment* between the weights, called `gap` in the example below. This parameter is enough to determine them uniquely, because they have the additional constraint that they sum to 1. In our case, it turns out that the gap was initially 0.01. The syntax to accomplish this is:

```
gap = 0.01;
t_min = 2;
t_max = 11;
K = 10;
[t,w]  = arithmetic_model(K, t_min,t_max,gap);
```

Further, we should remember that the atoms will account for only 1/2 of the entire mass, therefore we must multiply the weights by 1/2: `w = 1/2*w;`

We now move to the second step: representing the uniform distribution. This is represented as a $2 \times T$ matrix of lower and upper endpoints of the uniform distributions, and by a vector of weights. In our case there is only one uniform distribution, located at [0.51.5], and with weight 1/2:

```
r = [0.5 1.5];
```

```
w_int = 1/2;
```

Next, we set `\gamma = 0.01`. Finally, we set the accuracy parameter `\varepsilon = 1e-5;`. We can now call the `atomic()` function to compute the density of the limit spectrum. In addition the parameters `grid, density` that we already saw in the first example, and which are the main results of the computation, `atomic()` also returns `K_hat, l_hat, u_hat`, which are numerical approximations to the number of clusters in the support, and the lower/upper endpoints:

```
[grid, density,~, ~, ~, K_hat, l_hat, u_hat] =  atomic(t, gamma, w, r,w_int,epsi);
```

In this example `K_hat = 11`, so the clusters are apparently disjoint.

Finally, we should generate a large random matrix to verify, at least qualitatively, that the results are correct. While this is quite simple to do, we provide some helper functions to make it even easier. We provide a function to take random draws from distributions H that are mixtures of point masses and uniforms, of the form 3.3. This function `random_draw_disc_unif()` takes as parameters the same representation t, w, r, w_{int} that was used to call Atomic. However, it also takes a parameter `p` to specify the number of draws from the distribution. We choose first `n`, the number of samples, and then `p` is determined using $\gamma = p/n$. The code to generate a random draw from H is

```
rng(0);
n = 5*1e4;
p = floor(gamma*n);
pop_eigs = random_draw_disc_unif(t,w,r,w_int, p);
```

The we generate a Gaussian random matrix with diagonal population covariance matrix having the eigenvalues `pop_eigs`, and compute its eigenvalues, as well as their histogram:

```
[X] = 1/sqrt(n)*randn(n,p)*diag(sqrt(pop_eigs));
D = svd(X,'econ').^2;
[heights,locations] = hist(D,10*floor(sqrt(p)));
width = locations(2)-locations(1);
heights = heights / max(heights);;
```

Finally, Figure 2 is generated using:

```
figure, hold on
bar(locations,heights);
plot(grid, density/max(density),'r','LineWidth',4)
set(gca,'fontsize',14)
xlabel('Eigenvalues');
h = legend( 'Empirical Eigenvalues','Theoretical Prediction','Location','best');
set(h,'FontSize',14);
```

This finishes the advanced example use of the `Atomic` function.

3.4 More examples

Several more reproducible examples are provided with the software package. These include the computational results presented in the paper Dobriban (2015), which are found in the folder `\Experiments\Atomic`.

The examples include (and their locations are the following folders):

1. Support identification of the sample spectrum: `\Support identification`
2. Accuracy of computing the density with various algorithms: `\Test correctness`
3. Timing comparisons of various algorithms: `\Timing`
4. Newton's method: `\Newton method`

4 Computational methods

4.1 Atomic: computing the limit spectrum

The ATOMIC method computes the limit sample spectrum of covariance matrices from the limit population spectrum. The main user interface for this computation is `atomic()`. This function takes as input the population spectral distribution H, aspect ratio γ, and other parameters, and computes the density of the limit spectrum on a grid.

The basic syntax of this function is :

```
[grid, density]= atomic(t,gamma)
```

In the basic setup, it requires only two inputs:

- `t`: a vector of positive entries, the locations of the population eigenvalues (length J)
- `gamma`: aspect ratio $\gamma = p/n$

In the basic case the population spectrum is defined as an equal mixture of point masses at t_i: $H = J^{-1}\sum_{i=1}^{J}\delta_{t_i}$.

The outputs are:

- `grid`: a positive grid where the density is computed
- `density`: pointwise numerical approximations to the density on the `grid`

The complete syntax of the function is:

```
[grid, density,m, v, mass_at_0, K_hat, l_hat, u_hat, x, f_hat]= atomic(t,gamma,varargin)
```

Here, in addition to the above two arguments `t,gamma`, a much larger set of input arguments can be specified within `varargin`. A complete call, including all parameters, looks like:

```
atomic(t, gamma, w, r, w_int, epsilon,M)
```

The extra arguments `w, r, w_int, epsilon,M` are optional. You can specify any subset of them, however, you must always use them in this order. If you wish to specify some later arguments, and leave some earlier ones default, then place an empty array `[]` in their place. For instance, to specify `epsilon = 1e-6` without specifying `w` etc, call `atomic(t,gamma,[],[],[],1e-6)`.

The inputs specify the population spectrum H as a mixture of point masses and uniform distributions:

$$H = \sum_{i=1}^{J} w_i\delta_{t_i} + \sum_{t=1}^{T} w_t^* U(a_t, b_t).$$

Here δ_t is the point mass at t and $U(a,b)$ is the uniform distribution on $[a,b]$. The parameters of the mixture can be set using the parameters of `atomic()`, which are, in order:

- `w`: mixture weights of the components of t in the input spectrum. This must be a positive vector of size J. The default is a uniform vector of length J: $w = (1, 1, \dots, 1)/J$. If a continuous part of the component is also specified with `r`, then this parameter must not be empty. The reason is that these weights and `w_int` implictly depend on each other, as they must sum to 1.
- `r`: intervals where the continuous part of the spectrum is supported. A real matrix of size $J \times 2$, the t-th row encodes the endpoints $[a_t, b_t]$ of the i-th continuous component of the spectrum, a uniform distribution on $[a_t, b_t]$.

- `w_int`: weights of the continuous part of the spectrum. This must be a non-negative vector. The entry `w_int[t]` specifies the weights of the i-th continuous component of the spectrum, a uniform distribution on $[a_t, b_t]$. If this is empty, then the defualt value is a uniform distribution subject to the constraint `sum(w)`sum(w_int) =1+.

- `epsilon`: accuracy parameter. As the accuracy parameter $\varepsilon \to 0$, the density approximation becomes more accurate. However, only the convergence of the procedure is guaranteed, not the actual accuracy.

- `M`: number of grid points in each interval of the support. This is an additional parameter which allows the user to specify how many grid points should be used in each interval in the support of the sample spectrum. This is available for additonal flexibility.

- `alg`: The algorithm used to compute the density. The available choices are `{'atomic','fp','Newton'};`. These stand for: `'atomic'` is the ATOMIC algorithm proposed in Dobriban (2015), `'fp'` is the fixed point method described in Couillet et al. (2011), and `'Newton'` is Newton's method, which the author learned from Jack W. Silverstein, and which is described in Dobriban (2015). Please see that paper for more information on each method.

4.2 Moments and Quantiles of the ESD

4.3 Utilities

4.3.1 Arithmetic and geometric models for eigenvalues

References

Bai, Z. and Silverstein, J. W. (2009). *Spectral analysis of large dimensional random matrices*. Springer Series in Statistics. Springer.

Couillet, R., Debbah, M., and Silverstein, J. W. (2011). A deterministic equivalent for the analysis of correlated mimo multiple access channels. *Information Theory, IEEE Transactions on*, 57(6):3493–3514.

Dobriban, E. (2015). Precise computation of limit spectra of sample covariance matrices.

Marchenko, V. A. and Pastur, L. A. (1967). Distribution of eigenvalues for some sets of random matrices. *Matematicheskii Sbornik*, 114(4):507–536.

Silverstein, J. W. (1995). Strong convergence of the empirical distribution of eigenvalues of large dimensional random matrices. *Journal of Multivariate Analysis*, 55(2):331–339.