Reducing the Number of Data Points

MATLAB Implementation

Tamas Kis | kis@stanford.edu

TAMAS KIS
https://github.com/tamaskis

Copyright © 2021 Tamas Kis

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

LICENSE

Contents

1 Download and Installation

1.1 Download from MATLAB Central's File Exchange

The `shrink_data_set` function is available for download on MATLAB® Central's File Exchange at https://www.mathworks.com/matlabcentral/fileexchange/86218-reduce-number-of-data-points-shrink_data_set.

1.2 Download from GitHub

The `shrink_data_set` function is available for download on GitHub® at https://github.com/tamaskis/shrink_data_set-MATLAB.

1.3 Files Included With Download

There are **five** files included in the downloaded `zip` file:

1. `EXAMPLE.M` – *example for using the* `shrink_data_set` *function*
2. `LICENSE` – *license for the* `shrink_data_set` *function*
3. `README.md` – *markdown file for GitHub documentation*
4. `Reducing the Number of Data Points - MATLAB Implementation.pdf` – *this PDF*
5. **`shrink_data_set.m` – *MATLAB function to reduce the number of points in a data set***

1.4 Accessing the `shrink_data_set` Function in a MATLAB Script

There are **four** options for accessing the `shrink_data_set` function in a MATLAB script:

1. Copy the `shrink_data_set` function to the *end* of your MATLAB script.
2. Place the `shrink_data_set.m` file in the same folder as the MATLAB script.
3. Place the `shrink_data_set.m` file into whatever folder you want, and then use the `addpath(folderName)` command[1] where the `folderName` parameter is a string that stores the filepath of the folder that `shrink_data_set.m` is in *relative to* the folder that your script is in.
4. Make a toolbox by first opening `shrink_data_set.m`, then going to the `HOME` tab in MATLAB, and finally selecting `Package Toolbox` in the drop-down menu under `Add-Ons`. Once you package the `shrink_data_set` function as a toolbox, you can use it in any script.

[1] https://www.mathworks.com/help/matlab/ref/addpath.html

2 shrink_data_set

Reduces the number of data points in a data set to a specified number.

Syntax

```
X_shrunk = shrink_data_set(X,N)
X_shrunk = shrink_data_set(X,N,'columns')
X_shrunk = shrink_data_set(X,N,'rows')
```

Description

`X_shrunk = shrink_data_set(X,N)` shrinks a matrix `X` (storing a data set) so that it is left with only `N` rows. This syntax assumes the various variables are organized into separate columns. Therefore, `X_shrunk` will be the original matrix `X` shrunk down to `N` data points.

`X_shrunk = shrink_data_set(X,N,'columns')` shrinks a matrix `X` (storing a data set) so that it is left with only `N` rows. The input `'columns'` indicates that the variables are organized into separate columns, which implies that to remove data points, we need to remove rows. Therefore, `X_shrunk` will be the original matrix (i.e. data set) `X` shrunk down to `N` data points.

`X_shrunk = shrink_data_set(X,N,'rows')` shrinks a matrix `X` (storing a data set) so that it is left with only `N` columns. The input 'rows' indicates that the variables are organized into separate rows, which implies that to remove data points, we need to remove columns. Therefore, `X_shrunk` will be the original matrix (i.e. data set) `X` shrunk down to `N` data points.

NOTE: Sometimes, the function will not be able to return exactly `N` points (due to rounding issues). However, the purpose of this function is mainly to reduce the size of a data set when not all the points are needed. For example, plotting $y = x^2$ with 100 points rather than 1000 points will (to the naked eye) not be visually any worse, but will be a lot faster for the computer to perform.

Examples

Example 2.1

Define a data set $\{(x_i, y_i)\}_{i=1}^{1001}$ such that $y_i = x_i + \sin(x_i)\ \forall\ i \in [0, 1001]$ and for x in the domain $x \in [0, 10]$. Then, shrink this data set so it is left with only 5 data points.

■ *SOLUTION*

First, we define the data set. Let's define it as row vectors.

```
% defines data set
x = 0:0.01:10;
y = x+sin(x);
```

To use the `shrink_data_set` function, we must first compile this data set into a matrix. Since `x` and `y` are defined as row vectors,

```
% compiles data set into matrix
X = [x,y];
```

Now, we can use the `shrink_data_set` function with the specification `'rows'` to indicate that the variables (i.e. x and y) are separated into rows (in this case, x occupies the first row while y occupies the second row).

```
% shrinks data set to have only 5 points
X_shrunk = shrink_data_set(X,5,'rows');
```

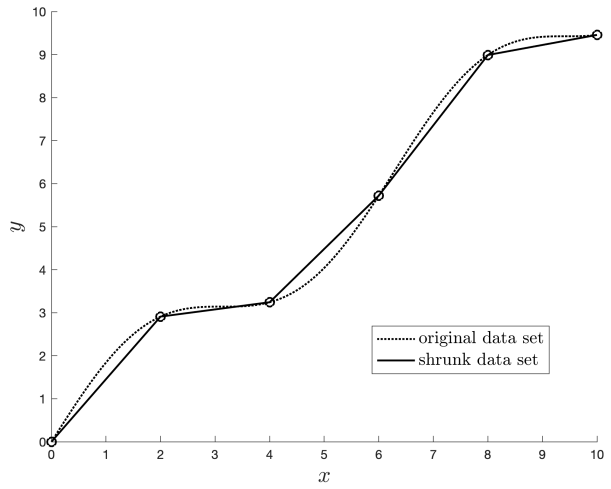
Since we started off with two individual vectors (`x` and `y`), it is usually useful to separate `X_shrunk` into the vectors `x_shrunk` and `y_shrunk`.

```
% extracts x_shrunk and y_shrunk from X_shrunk
x_shrunk = X_shrunk(1,:);
y_shrunk = X_shrunk(2,:);
```

Finally, we can create a plot to visualize the original and shrunk data sets.

```
% creates a plot
figure;
hold on;
plot(x,y,'k:','linewidth',1.5);
plot(x_shrunk,y_shrunk,'color','k','linewidth',1.5);
plot(x_shrunk,y_shrunk,'o','linewidth',1.5,'markersize',7,'color','k',...
    'handlevisibility','off');
hold off;
xlabel('$x$','interpreter','latex','fontsize',18);
ylabel('$y$','interpreter','latex','fontsize',18);
legend('original data set','shrunk data set','interpreter','latex',...
    'fontsize',14,'location','best');
```

The resulting plot is shown below.



Clearly, the shrunk data set has 6 data points, whereas we requested only 5. However, there is no direct way to resolve this; we do not wish to remove the endpoints from the data set, we want to select the points as uniformly as possible, and we can only select integer indices.