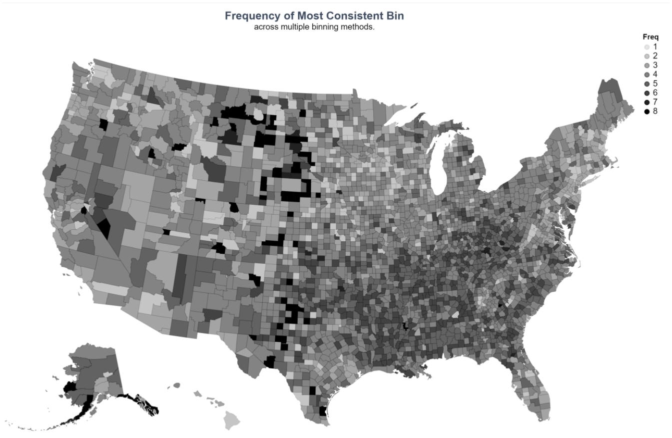
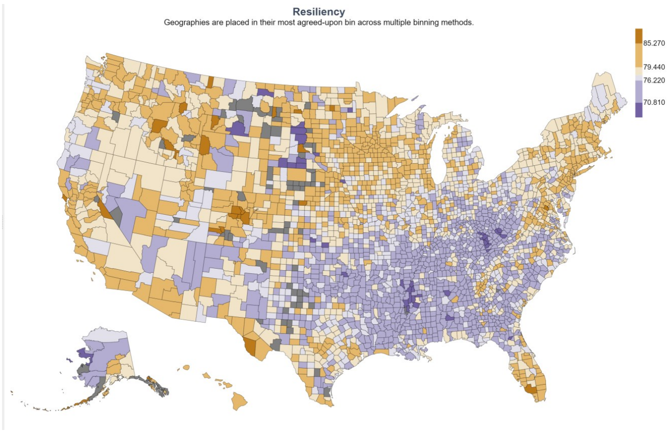
Resiliency: A Consensus Data Binning Method

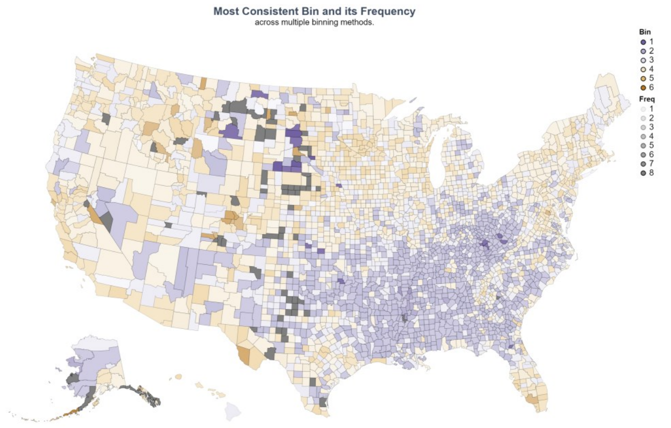
Arpit Narechania
Alex Endert
Clio Andris
Georgia Institute of Technology

[arpitnarechania; endert; clio]@gatech.edu
@GT_Vis @EndertAlex @arpitnarechania @friendly_cities
Corresponding Author (Arpit Narechania)

GIScience 2023







Resiliency: A Consensus Data Binning Method

Arpit Narechania
Alex Endert
Clio Andris
Georgia Institute of Technology

[arpitnarechania; endert; clio]@gatech.edu
@arpitnarechania @EndertAlex @friendly_cities
@GT_Vis
Corresponding Author (Arpit Narechania)

GIScience 2023

KEY POINTS

1) We share a method called 'resiliency', which is a consensus method for data binning for thematic / choropleth maps.

2) It helps users see multiple binning methods to 1) highlight consistent patterns and 2) detect where binning will matter the most.

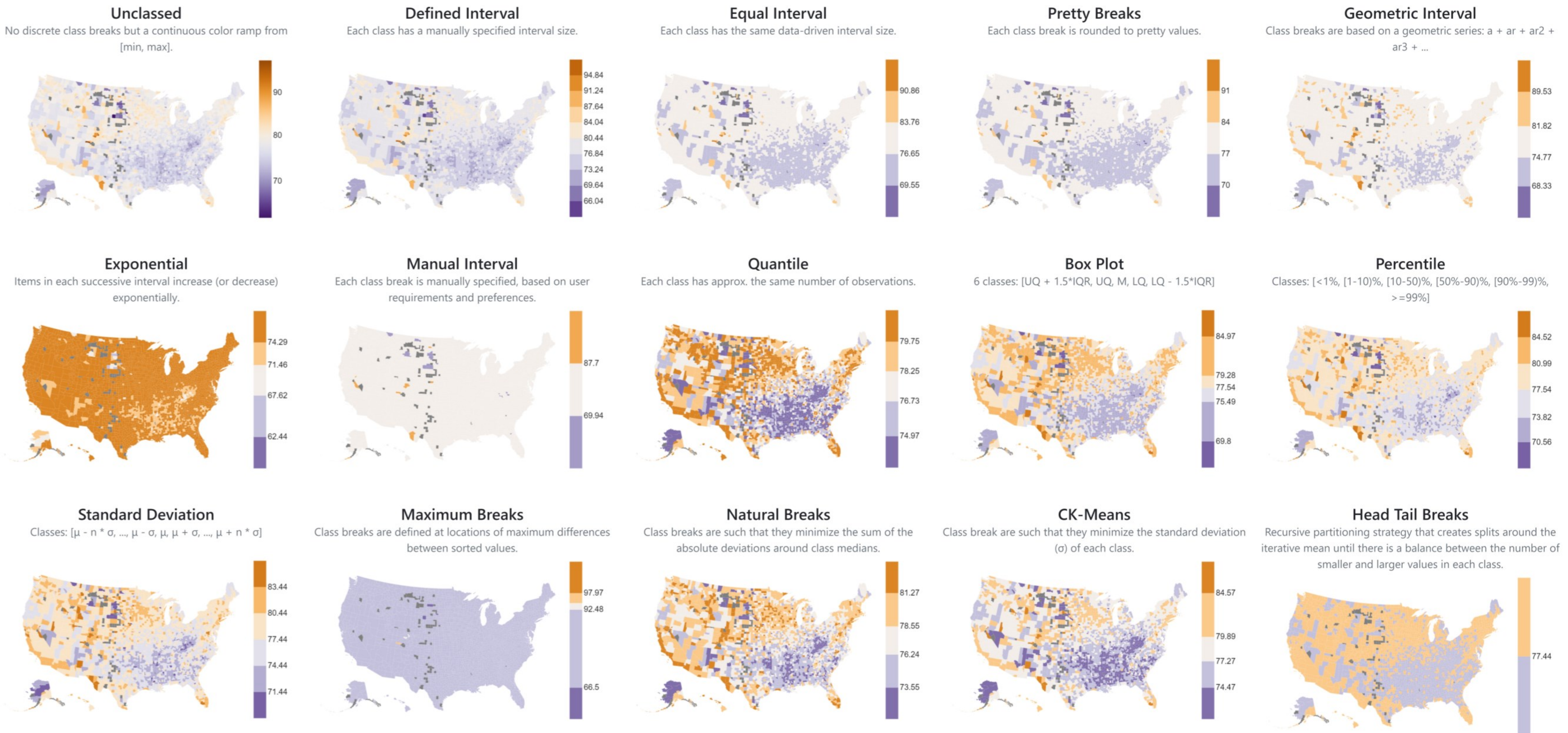
3) We implemented the method online in two different places.

AGENDA

- Binning Strategies for Thematic Choropleth Maps
- Resiliency: Ensemble Method
- Demonstration
- Limitations, Future Work + Conclusion

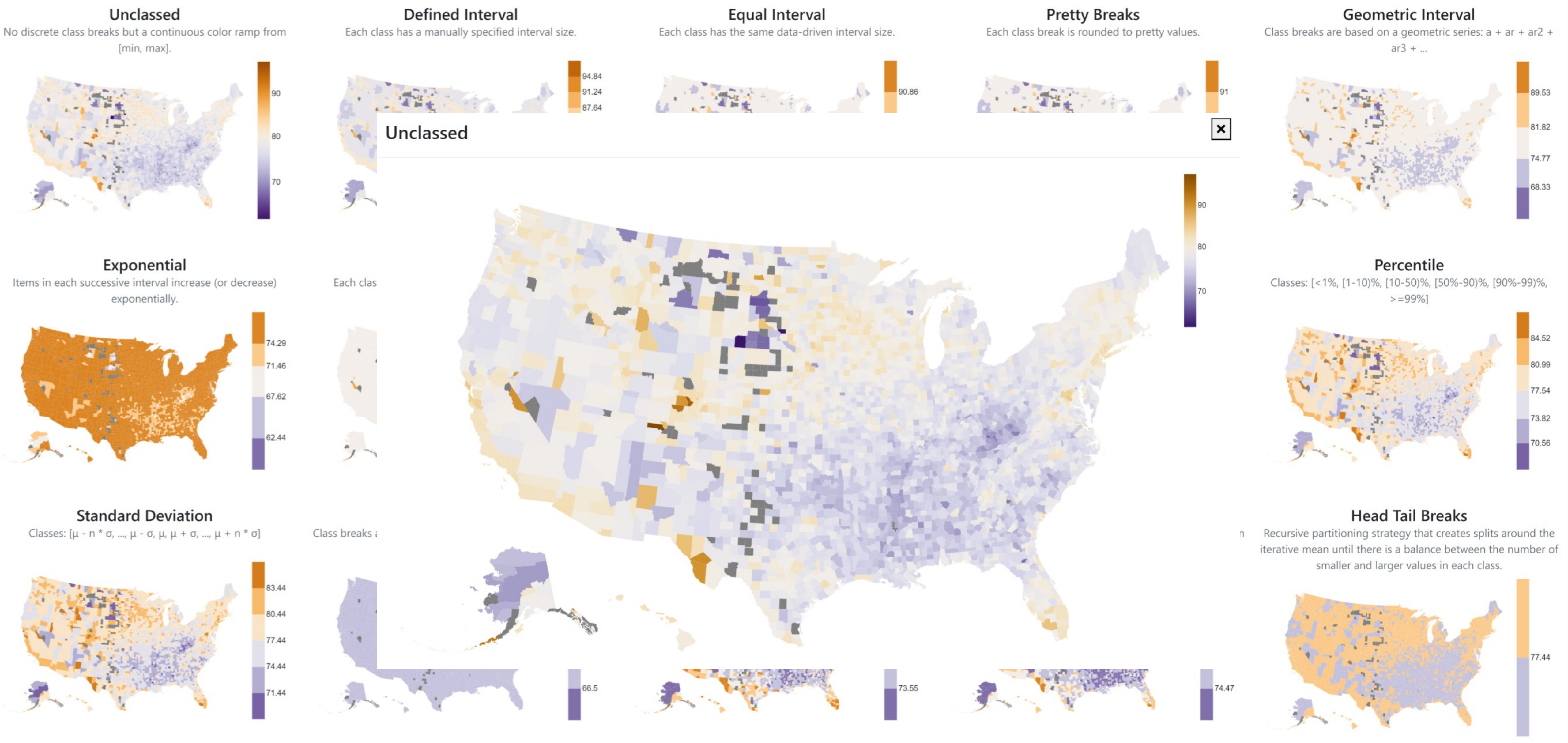
A DATASET WITH THE SAME VALUES CAN BE BINNED DIFFERENTLY

US LIFE EXPECTANCY BY COUNTY (2021, CDC)



SOURCE: AUTHORS

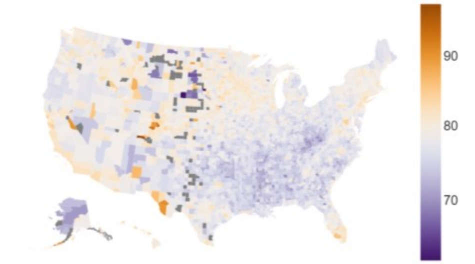
US LIFE EXPECTANCY BY COUNTY (2021, CDC)



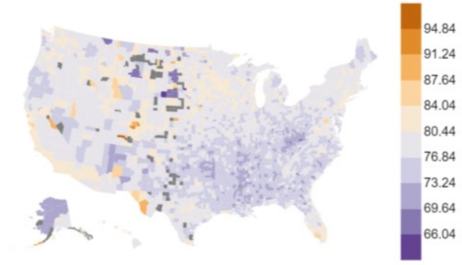
SOURCE: AUTHORS

US LIFE EXPECTANCY BY COUNTY (2021, CDC)

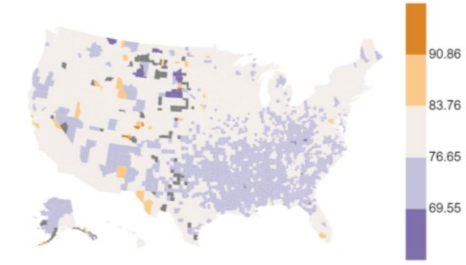
Unclassed
No discrete class breaks but a continuous color ramp from [min, max].



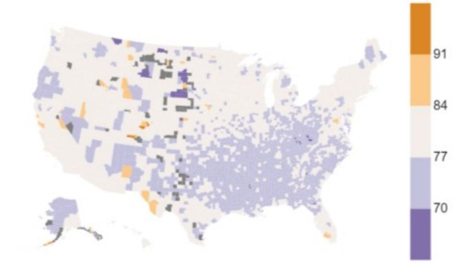
Defined Interval
Each class has a manually specified interval size.



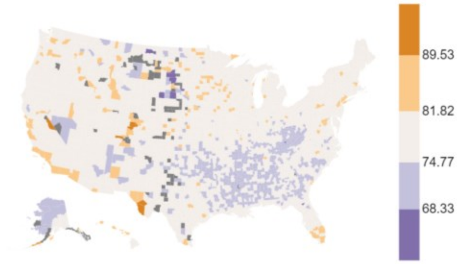
Equal Interval
Each class has the same data-driven interval size.



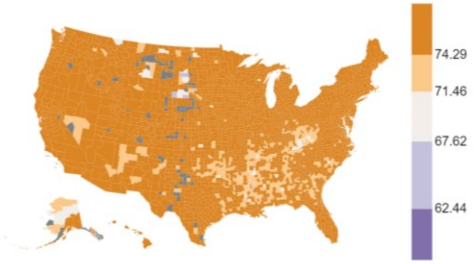
Pretty Breaks
Each class break is rounded to pretty values.



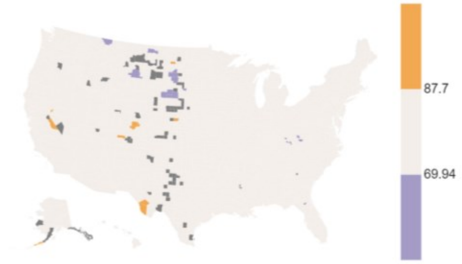
Geometric Interval
Class breaks are based on a geometric series: a + ar + ar2 + ar3 + ...



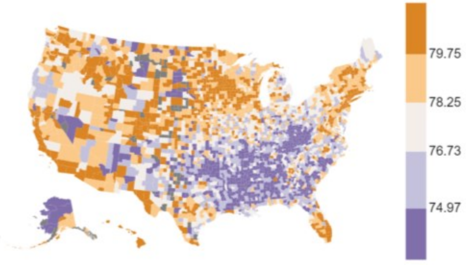
Exponential
Items in each successive interval increase (or decrease) exponentially.



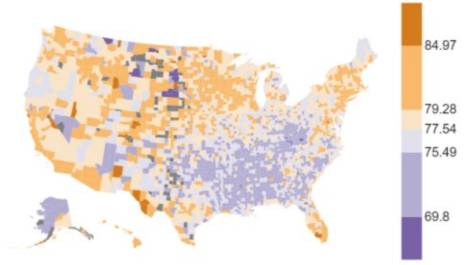
Manual Interval
Each class break is manually specified, based on user requirements and preferences.



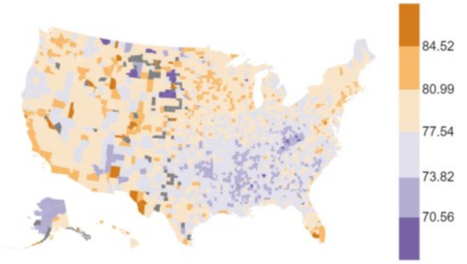
Quantile
Each class has approx. the same number of observations.



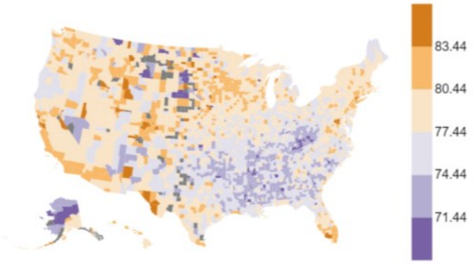
Box Plot
6 classes: [UQ + 1.5*IQR, UQ, M, LQ, LQ - 1.5*IQR]



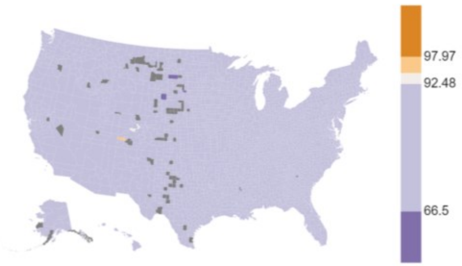
Percentile
Classes: [<1%, [1-10)%, [10-50)%, [50%-90)%, [90%-99)%, >=99%]



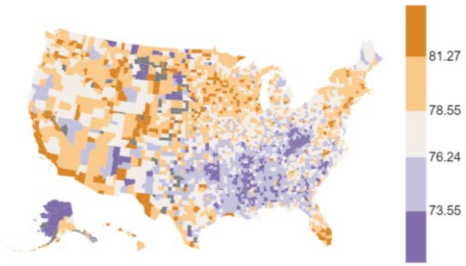
Standard Deviation
Classes: [μ - n * σ, ..., μ - σ, μ, μ + σ, ..., μ + n * σ]



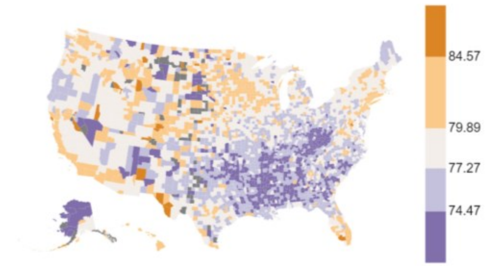
Maximum Breaks
Class breaks are defined at locations of maximum differences between sorted values.



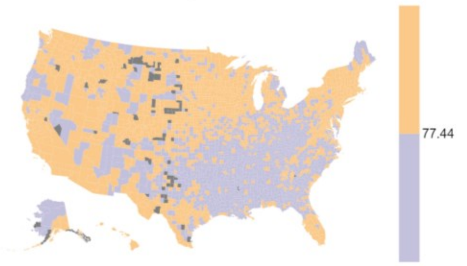
Natural Breaks
Class breaks are such that they minimize the sum of the absolute deviations around class medians.



CK-Means
Class break are such that they minimize the standard deviation (σ) of each class.

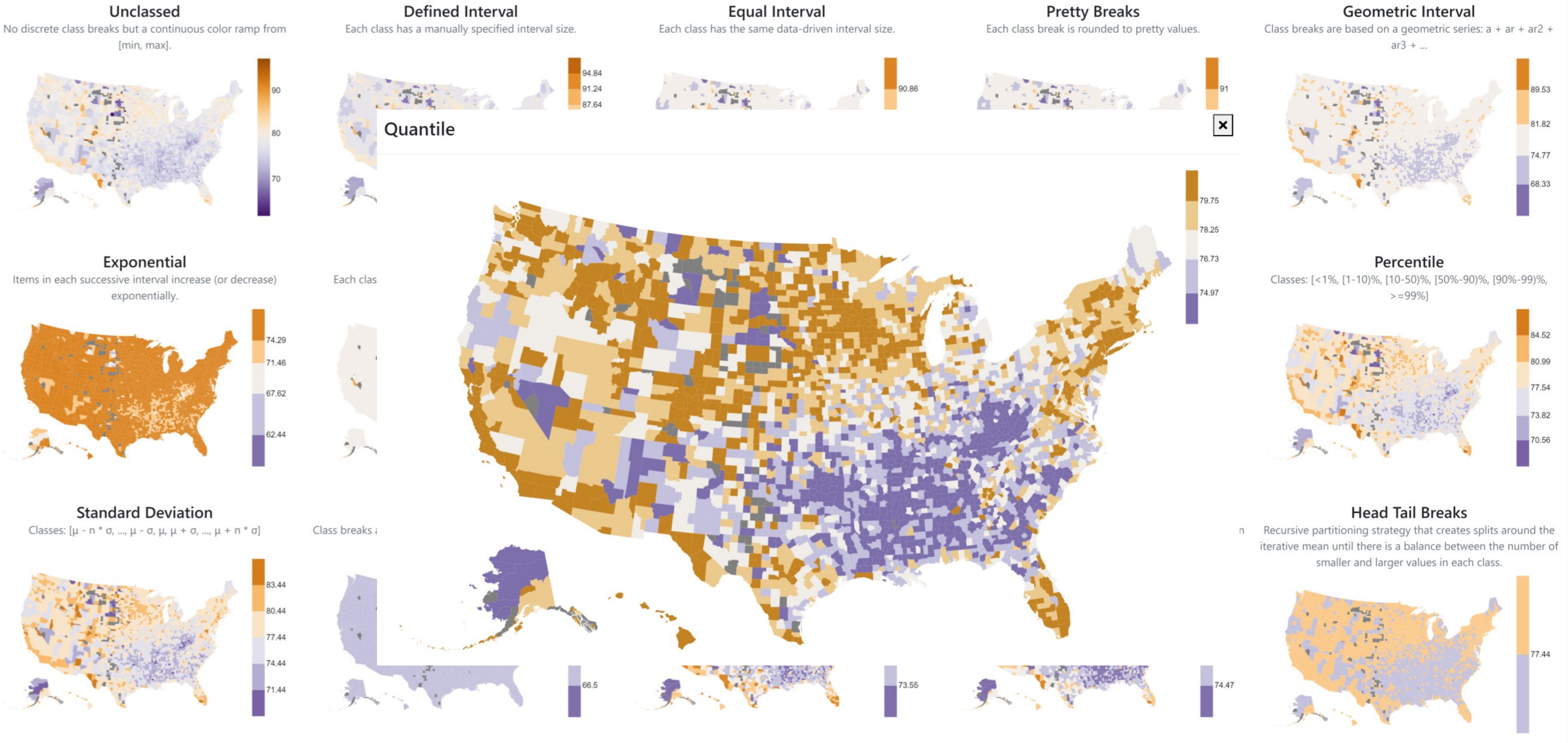


Head Tail Breaks
Recursive partitioning strategy that creates splits around the iterative mean until there is a balance between the number of smaller and larger values in each class.



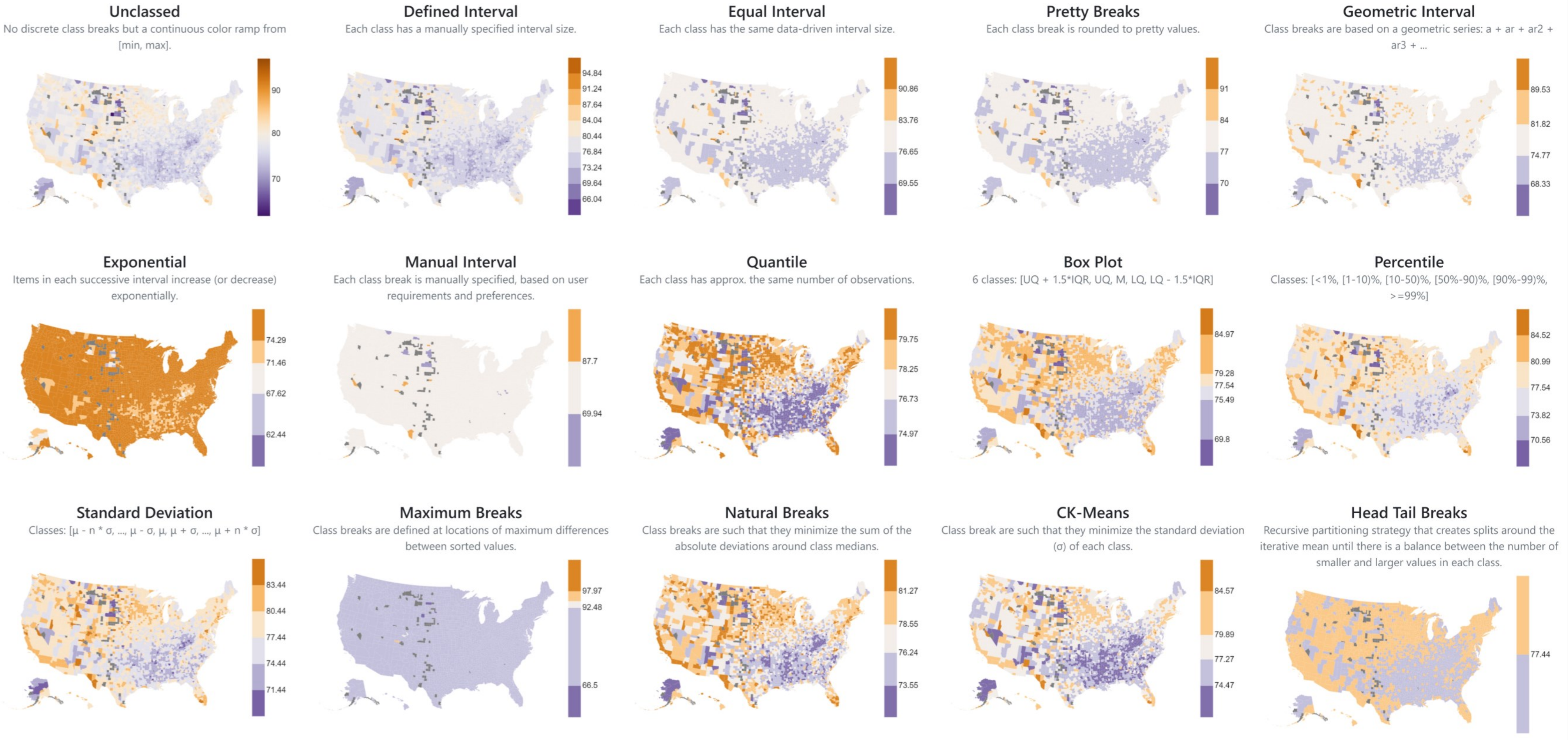
SOURCE: AUTHORS

US LIFE EXPECTANCY BY COUNTY (2021, CDC)



SOURCE: AUTHORS

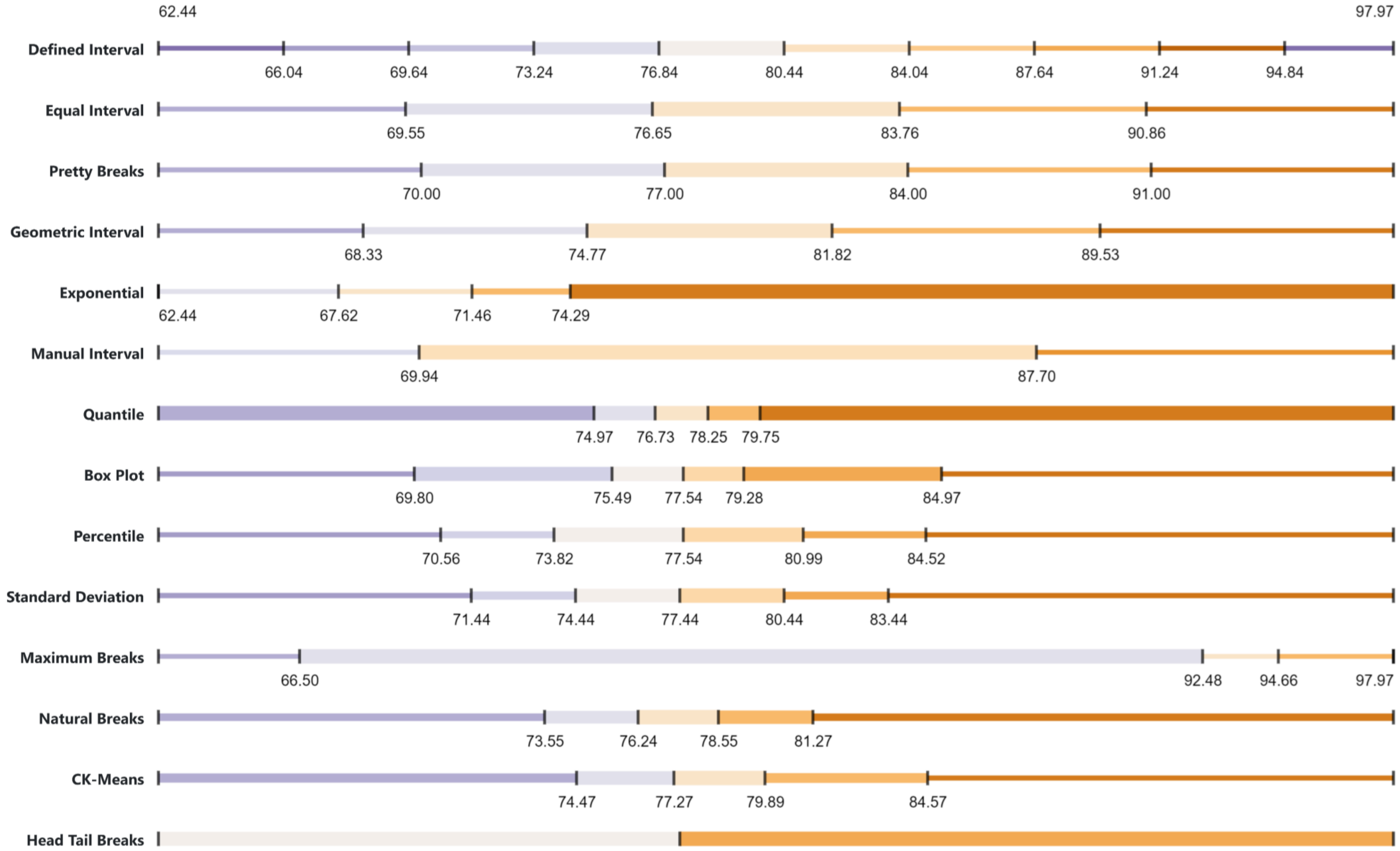
US LIFE EXPECTANCY BY COUNTY (2021, CDC)



SOURCE: AUTHORS

CHOOSING A BINNING METHOD CAN BE DIFFICULT

US LIFE EXPECTANCY BY COUNTY (2021, CDC)



SOURCE: AUTHORS

RELATED LITERATURE

Quantile and minimum boundary error are suited for general reading of epidemiological rate maps (Brewer and Pickle 2002).

Quartile, equal interval, standard deviation, and natural breaks are accurate for data sets with specific distributional characteristics, but none of them accurately bin all types of distributions (Smith 1986).

Equal interval, natural breaks, standard deviation, quantiles, & pretty breaks are particularly common (Brewer and Pickle 2002).

Round-number bin breaks, which are easy to read and remember, can constrain the outputs of optimization algorithms that have more significant digits than the map user would prefer or the data warrants (Monmonier 1982).

Genetic algorithms (Armstrong et al. 2003) and proximity-based (Monmonier 1973) binning methods, which promote spatially compact and homogeneous regionalization on map, are less common but also important. The head/tail break system by Jiang (2013) is a relatively new, helpful method.

--Marc Armstrong, Ningchuan Xiao, and David Bennett. Using genetic algorithms to create multicriteria class intervals for choropleth maps. Annals of the Association of American Geographers, 93:595 – 623, 09 2003.
--Cynthia A. Brewer and Linda Pickle. Evaluation of methods for classifying epidemiological data on choropleth maps in series. Annals of the Association of American Geographers, 92(4):662–681, 2002.
--Richard M Smith. Comparing traditional methods for selecting class intervals on choropleth maps. The Professional Geographer, 38(1):62–67, 1986.
--Mark Monmonier. Maximum-difference barriers: An alternative numerical regionalization method. Geographical Analysis, 5(3):245–261, 1973.
--Mark Monmonier. Flat laxity, optimization, and rounding in the selection of class intervals. Cartographica: The International Journal for Geographic Information and Geovisualization, 19(1):16–27, 1982.
--Bin Jiang. Head/tail breaks: A new classification scheme for data with a heavy-tailed distribution. The Professional Geographer, 65(3), 482-494. 2013.

A SIMPLE ALGORITHM TO HELP HIGHLIGHT **COMMON RESULTS** ACROSS METHODS + DETECT DISCREPANCIES

Algorithm 1 Resiliency

```
input  : data values V, binning methods M, binning options O
output : resiliency bin breaks RB
// Compute bin breaks for all M
bin breaks B ← { }
for method m in M do
    B[m] = ComputeBins(V, O, m)
// Determine bins for all V across all M
bin ids ID ← { }
for value v in V do
    for method m in M do
        ID[v][m] = AssignBin(v, B[m])
// Compute the frequency of each bin among all M
bin frequencies BF ← { }
for value v in V do
    BF[v] = ComputeBinFrequency(ID[v])
// Place values in their most frequent bins
most frequent bins MFB ← { }
for value v in V do
    MFB[v] = ComputeMostFrequentBin(BF[v])
// Compute Resiliency
resiliency bin breaks RB ← { }
working bin assignments WFB ← MFB
while ValidateBins(RB) do
    RB, WFB = ResolveConflicts(WFB, RB)
return RB
```

A SIMPLE ALGORITHM TO HELP HIGHLIGHT **COMMON RESULTS** ACROSS METHODS + DETECT DISCREPANCIES

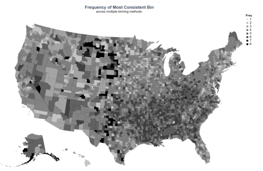
Algorithm 1 Resiliency

input : data values **V**, binning methods **M**, binning options **O**
output : resiliency bin breaks **RB**
```
// Compute bin breaks for all M
```
bin breaks **B** ← { }
for *method m in* ***M*** **do**
    B[m] = ComputeBins($\boldsymbol{V}$, $\boldsymbol{O}$, m)
```
// Determine bins for all V across all M
```
bin ids **ID** ← { }
for *value v in* ***V*** **do**
    for *method m in* ***M*** **do**
        ID[v][m] = AssignBin(v, $\boldsymbol{B}[m]$)
```
// Compute the frequency of each bin among all M
```
bin frequencies **BF** ← { }
for *value v in* ***V*** **do**
    BF[v] = ComputeBinFrequency($\boldsymbol{ID}[v]$)
```
// Place values in their most frequent bins
```
most frequent bins **MFB** ← { }
for *value v in* ***V*** **do**
    MFB[v] = ComputeMostFrequentBin($\boldsymbol{BF}[v]$)
```
// Compute Resiliency
```
resiliency bin breaks **RB** ← { }
working bin assignments **WFB** ← **MFB**
while ValidateBins($\boldsymbol{RB}$) **do**
    RB, WFB = ResolveConflicts($\boldsymbol{WFB}$, $\boldsymbol{RB}$)
return RB

A SIMPLE ALGORITHM TO HELP HIGHLIGHT **COMMON RESULTS** ACROSS METHODS + DETECT DISCREPANCIES

Algorithm 1 Resiliency

```
input  : data values V, binning methods M, binning options O
output : resiliency bin breaks RB
// Compute bin breaks for all M
bin breaks B ← { }
for method m in M do
    B[m] = ComputeBins(V, O, m)
// Determine bins for all V across all M
bin ids ID ← { }
for value v in V do
    for method m in M do
        ID[v][m] = AssignBin(v, B[m])
// Compute the frequency of each bin among all M
bin frequencies BF ← { }
for value v in V do
    BF[v] = ComputeBinFrequency(ID[v])
// Place values in their most frequent bins
most frequent bins MFB ← { }
for value v in V do
    MFB[v] = ComputeMostFrequentBin(BF[v])
// Compute Resiliency
resiliency bin breaks RB ← { }
working bin assignments WFB ← MFB
while ValidateBins(RB) do
    RB, WFB = ResolveConflicts(WFB, RB)
return RB
```



A SIMPLE ALGORITHM TO HELP HIGHLIGHT **COMMON RESULTS** ACROSS METHODS + DETECT DISCREPANCIES

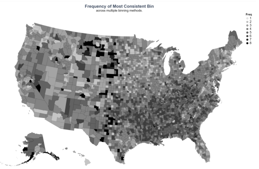
Algorithm 1 Resiliency

input : data values **V**, binning methods **M**, binning options **O**
output : resiliency bin breaks **RB**

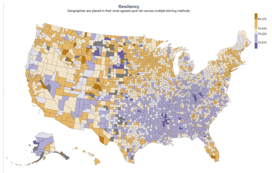
```
// Compute bin breaks for all M
```
bin breaks **B** $\leftarrow \{\}$
for *method m in* ***M*** **do**
  B$[m]$ = ComputeBins($\boldsymbol{V}$, $\boldsymbol{O}$, m)

```
// Determine bins for all V across all M
```
bin ids **ID** $\leftarrow \{\}$
for *value v in* ***V*** **do**
  for *method m in* ***M*** **do**
    ID$[v][m]$ = AssignBin(v, $\boldsymbol{B}[m]$)

```
// Compute the frequency of each bin among all M
```
bin frequencies **BF** $\leftarrow \{\}$
for *value v in* ***V*** **do**
  BF$[v]$ = ComputeBinFrequency($\boldsymbol{ID}[v]$)



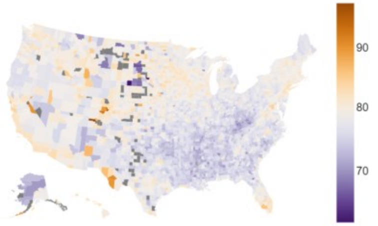
```
// Place values in their most frequent bins
```
most frequent bins **MFB** $\leftarrow \{\}$
for *value v in* ***V*** **do**
  MFB$[v]$ = ComputeMostFrequentBin($\boldsymbol{BF}[v]$)



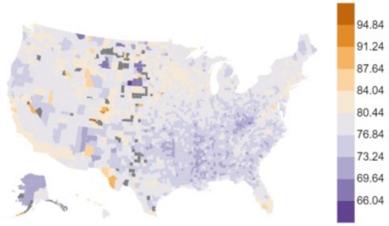
```
// Compute Resiliency
```
resiliency bin breaks **RB** $\leftarrow \{\}$
working bin assignments **WFB** $\leftarrow$ **MFB**
while ValidateBins($\boldsymbol{RB}$) **do**
  RB, **WFB** = ResolveConflicts($\boldsymbol{WFB}$, $\boldsymbol{RB}$)
return RB

US LIFE EXPECTANCY BY COUNTY (2021, CDC)

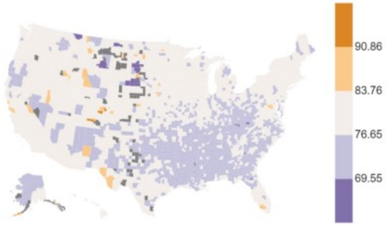
Unclassed
No discrete class breaks but a continuous color ramp from [min, max].



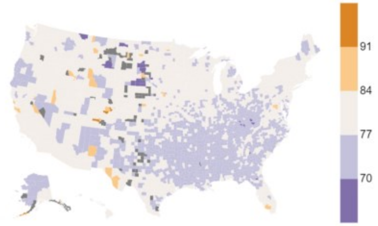
Defined Interval
Each class has a manually specified interval size.



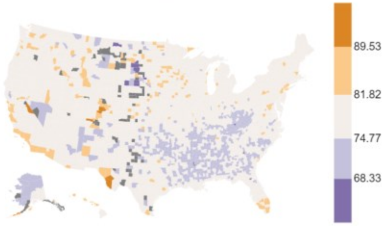
Equal Interval
Each class has the same data-driven interval size.



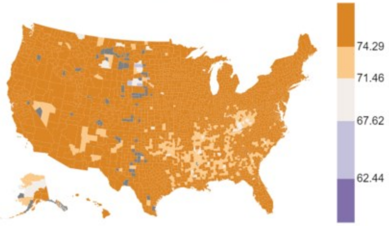
Pretty Breaks
Each class break is rounded to pretty values.



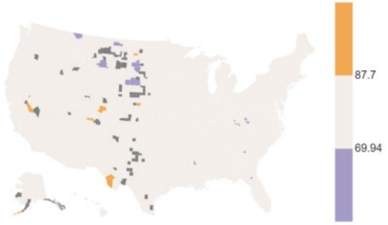
Geometric Interval
Class breaks are based on a geometric series: a + ar + ar2 + ar3 + ...



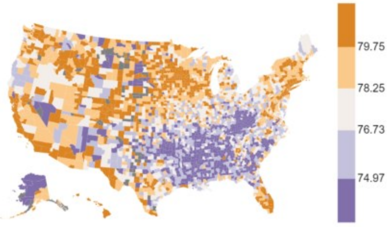
Exponential
Items in each successive interval increase (or decrease) exponentially.



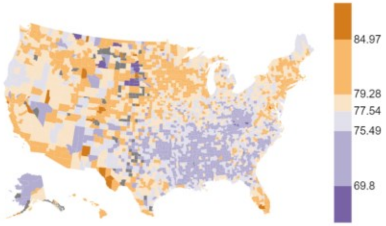
Manual Interval
Each class break is manually specified, based on user requirements and preferences.



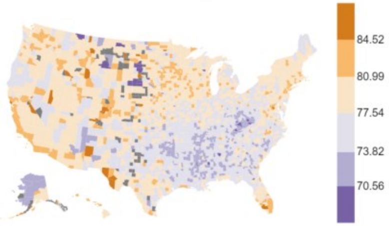
Quantile
Each class has approx. the same number of observations.



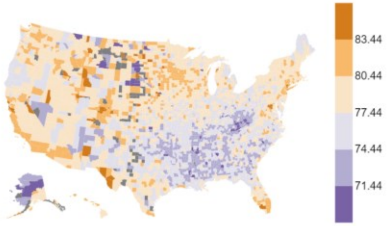
Box Plot
6 classes: [UQ + 1.5*IQR, UQ, M, LQ, LQ - 1.5*IQR]



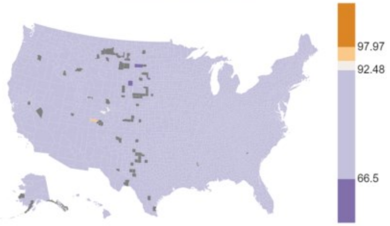
Percentile
Classes: [<1%, [1-10)%, [10-50)%, [50%-90)%, [90%-99)%, >=99%]



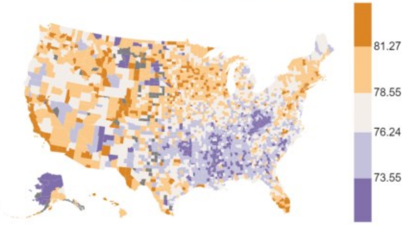
Standard Deviation
Classes: [μ - n * σ, ..., μ - σ, μ, μ + σ, ..., μ + n * σ]



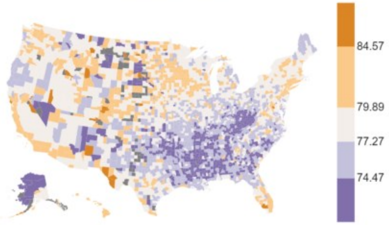
Maximum Breaks
Class breaks are defined at locations of maximum differences between sorted values.



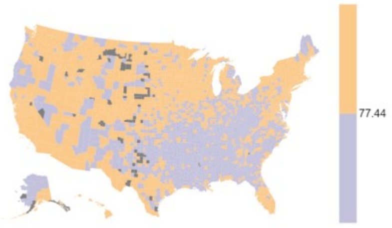
Natural Breaks
Class breaks are such that they minimize the sum of the absolute deviations around class medians.



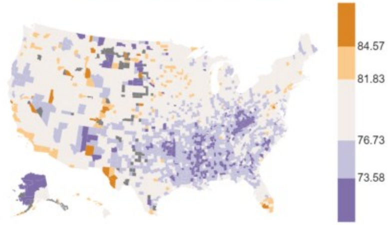
CK-Means
Class break are such that they minimize the standard deviation (σ) of each class.

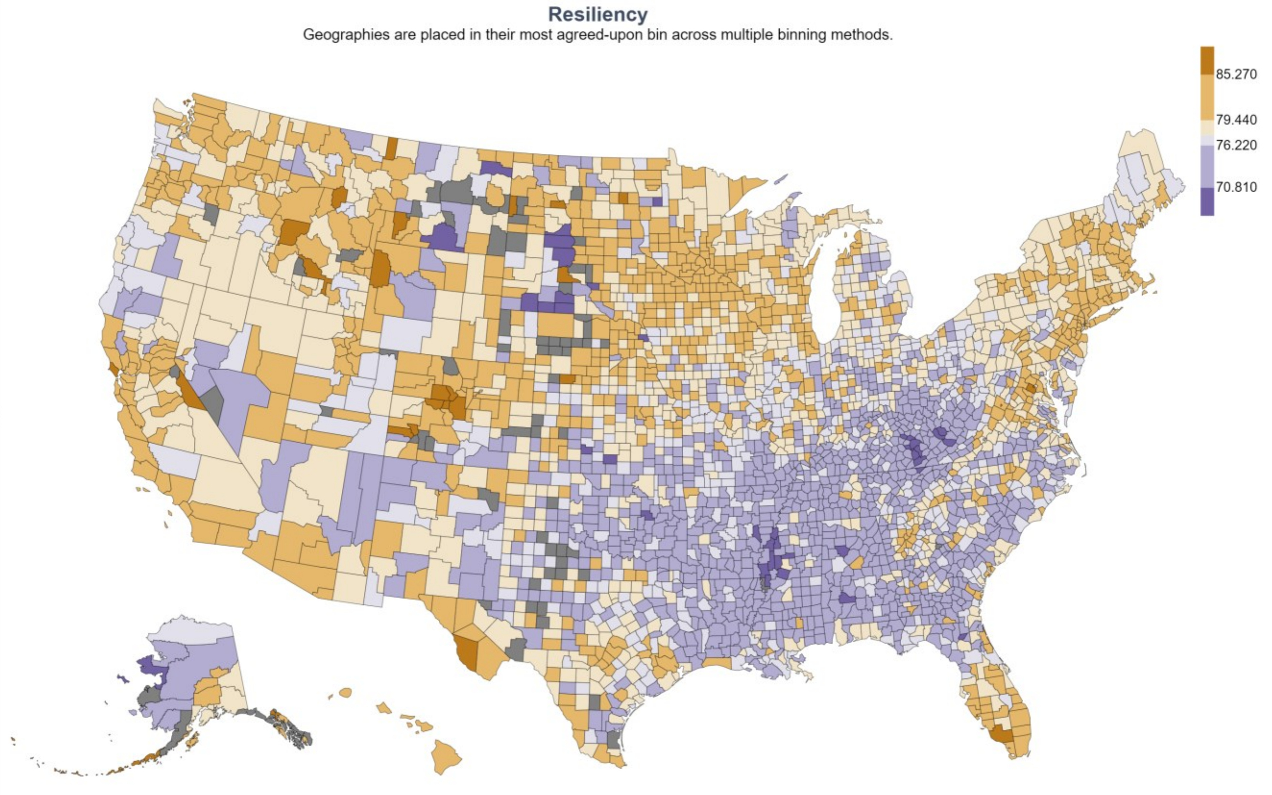


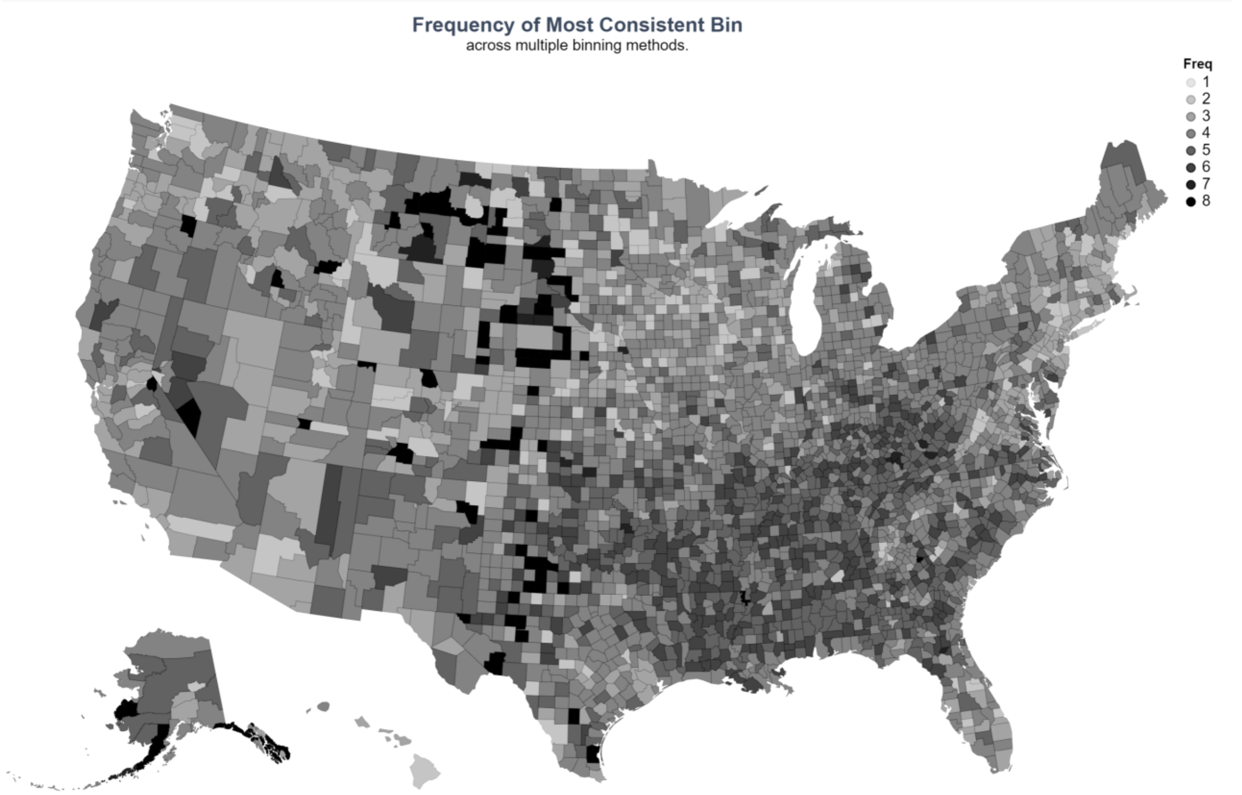
Head Tail Breaks
Recursive partitioning strategy that creates splits around the iterative mean until there is a balance between the number of smaller and larger values in each class.

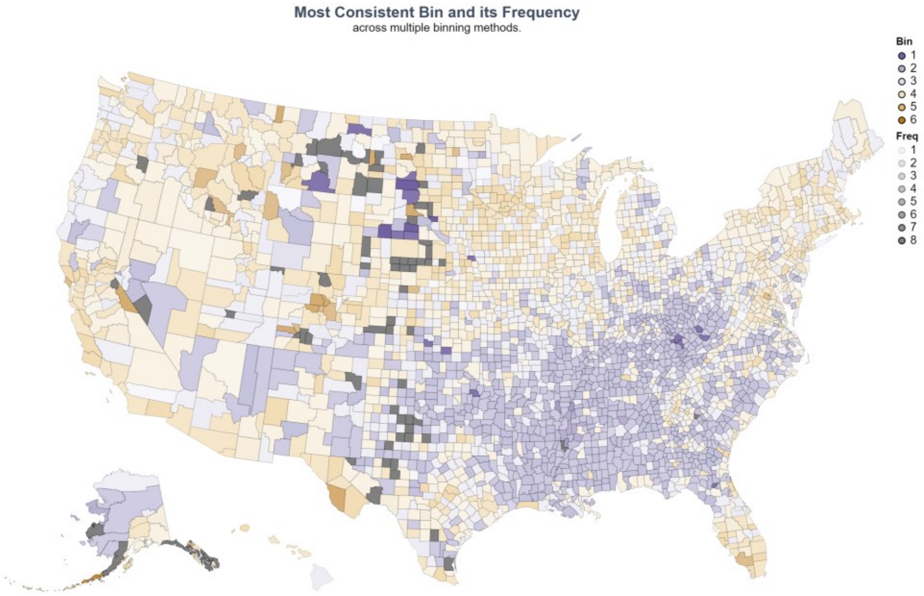


Resiliency
Class breaks are such that the underlying geographies are most consistent in those classes.




Resiliency
Geographies are placed in their most agreed-upon bin across multiple binning methods.
85.270
79.440
76.220
70.810

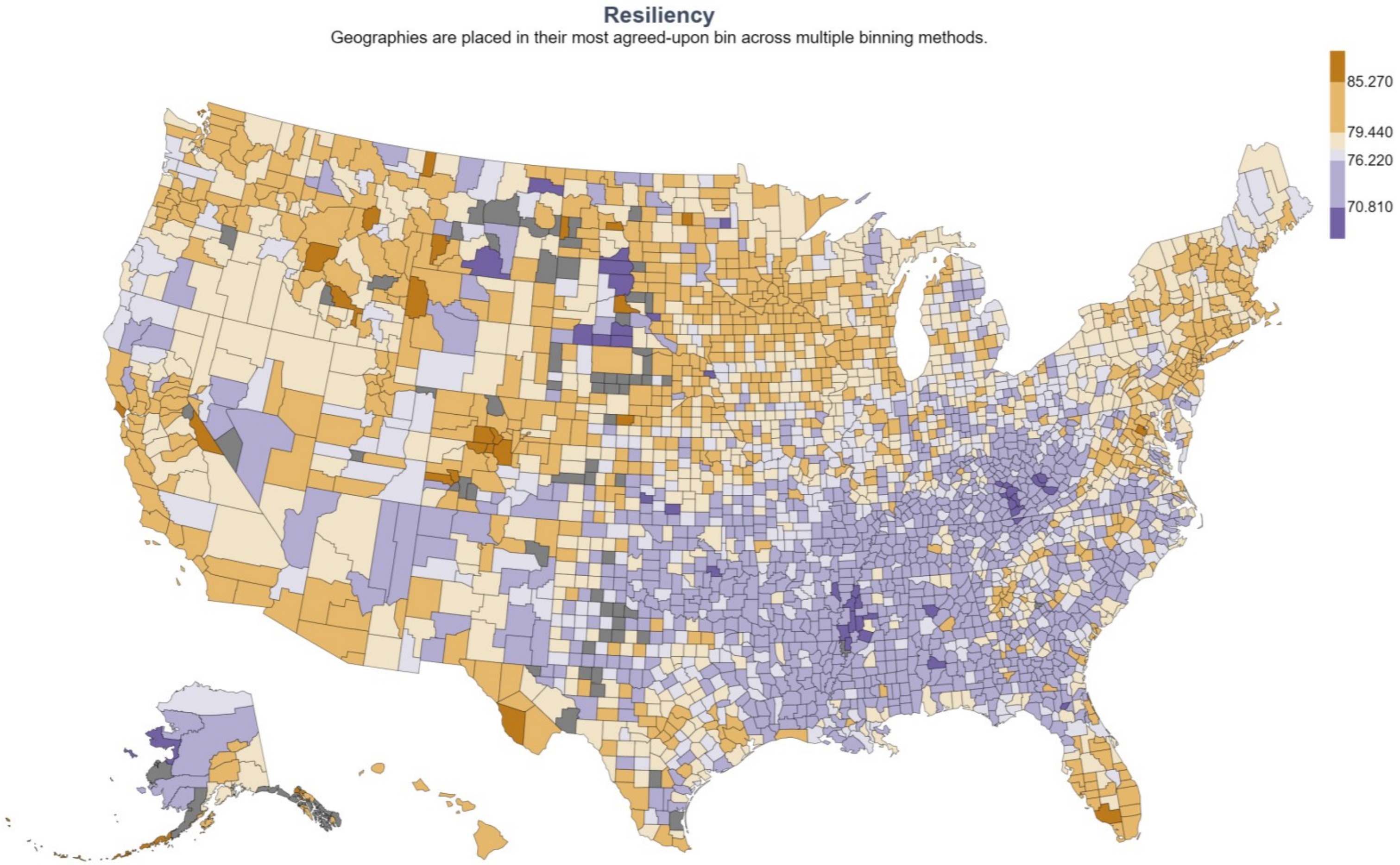

Frequency of Most Consistent Bin
across multiple binning methods.
Freq
1
2
3
4
5
6
7
8


Most Consistent Bin and its Frequency
across multiple binning methods.
Bin
1
2
3
4
5
6
Freq
1
2
3
4
5
6
7
8

RESILIENCY WEB INTERFACE

We'll examine US counties by different indicators

https://ocular.cc.gatech.edu/resiliency-app/#/app



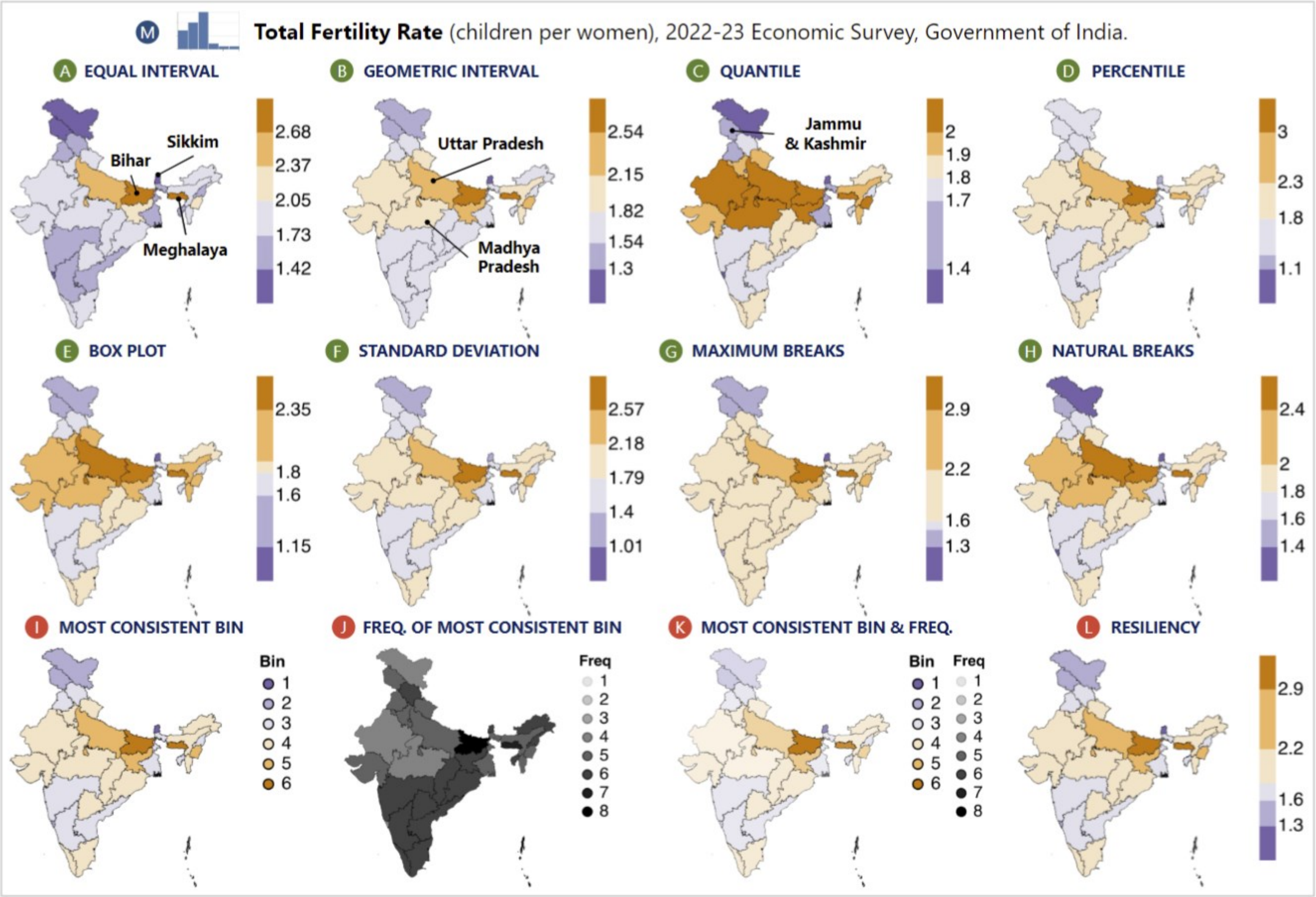


Figure 1 Small multiples of choropleth maps showing "Total Fertility Rate (children per women)" (**M**) in India [8] using established binning methods (**A-H**) and *resiliency* (**I-L**).

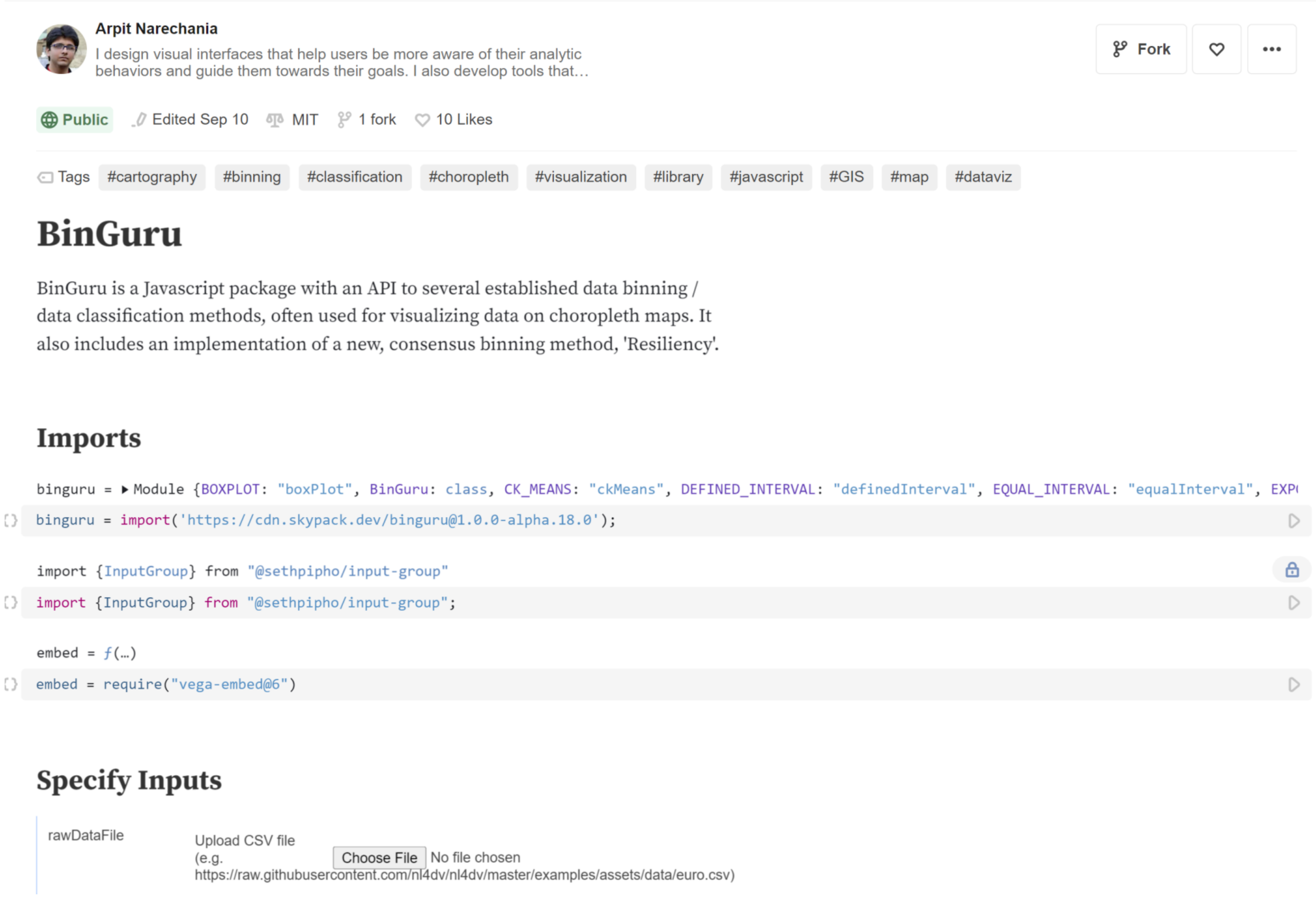
BINGURU: A JAVASCRIPT PACKAGE

Observable Notebook for users to fork and use with their own data for **exploration + education**:

https://observablehq.com/@arpitnarechania/binguru-demo

- via

https://github.com/arpitnarechania/binguru



LIMITATIONS

1) A mishmash of binning methods is not statistically-motivated.

2) The results are difficult to explain to a broad audience.

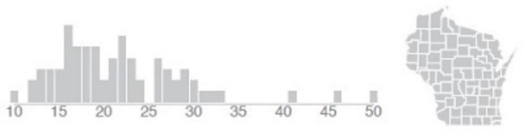
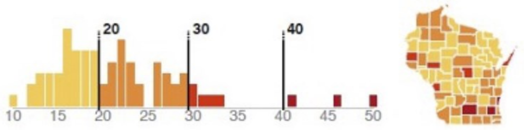
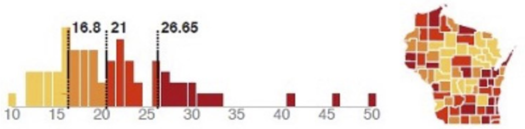
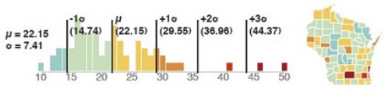
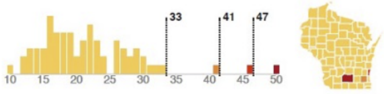
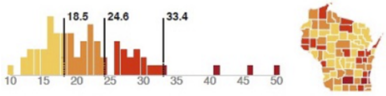
3) Inputs can be subjectively chosen by the user.

4) Even when 'all methods' are used as inputs, "objective" does not mean "correct".

FUTURE WORK

1) Improved **matching** of input data distribution to a suggested binning method.

Table 1. Classification Methods

Method	Description	Example	Suggested Use
			
Equal Interval	Class breaks at regular intervals along the number line at a set equivalent range. These breaks might be 20, 30, 40, etc, where each class is used to represent an equivalent range of measured data values. Classes are chosen regardless of the data. Equal interval is easy to read and understand; but it can be misleading in that no information is given on the distribution of the data within each distinct class. Method is calculated by taking the highest data value minus the lowest data value, and dividing by the number of classes desired to get class breaks at equivalent intervals. In this case, subtract 10 from 50, then divide by 4 to get intervals of 10.		Uniformly distributed data with familiar data ranges.
Quantiles (Equal Count)	Equal numbers of data observations are placed into each category. Data is classified into groups like Top 20%, Upper-Middle 20%, Middle 20%, Lower-Middle 20%, and Bottom 20%. This method is easy for the map reader to understand. Because there are equal numbers of observations in each class, the map will always produce distinguishable patterns. It can be misleading in that equal numbers of data values are in each class, so outliers are lost.		Evenly distributed data and ordinal data
Mean-Standard Deviation	Groups according to the distance to the mean standard deviation of the dataset. Using this method, the mean and standard deviation are taken from the dataset holistically, and the standard deviation from the mean is used to determine which class each data value falls in. This method is useful for normally distributed datasets in which classifying data as "above average" or "below average" makes a meaningful break in the data. This method does not work well with heavily skewed or non-normally distributed data. Mean-Standard Deviation classification is implemented by calculating the mean value of the dataset and the standard deviation, placing class breaks at the mean value and each standard deviation value. In our example, calculate the mean and standard deviation of the 72 data values, place a break at the mean and place additional breaks at the standard deviations. The following class breaks were created using QGIS (2017).		Normally distributed data
Maximum Breaks	Breaks are placed at the largest intervals between adjacent data values.This is an easy to understand method that works best with piecewise datasets with gaps. This method does not work well with skewed data. To implement, the data values are ordered from low to high and the difference between sequential data values are calculated. Breaks are placed where the differences are the largest, and the number of breaks is based on the number of classes desired. In our example, the largest breaks fall between 33 and 41 (8), 41 and 47, (6), and 47 and 50 (3), so we place our breaks at these points.		Piecewise and clustered data
Jenks-Caspall & Fisher-Jenks	Algorithmically optimal breaks are placed in data based on sums of deviations of means between individual classes. Initial breaks can be arbitrary and the algorithm is approached iteratively by moving values between classes until the smallest sum values are received (Slocum et al., 2005; Jenks, 1957). This minimizes variance within each class and maximizes variance between classes (Jiang, 2013).		Clustered and skewed data

Foster, M. (2019). Statistical Mapping (Enumeration, Normalization, Classification). *The Geographic Information Science & Technology Body of Knowledge* (2nd Quarter 2019 Edition), John P. Wilson (Ed.). DOI: 10.22224/gistbok/2019.2.2(link is external).

FUTURE WORK

1) Improved **matching** of input data distribution to a suggested binning method.

2) More intuitive **metrics** and refinement of a global output statistic.

FUTURE WORK

1) Improved **matching** of input data distribution to a suggested binning method.

2) More intuitive **metrics** and refinement of a global output statistic.

3) Documenting **real world** use cases.

CONCLUSIONS

1) We created an algorithm called 'resiliency', which is a consensus method for data binning.

2) It helps users see a merged version of binning methods to highlight consistent patterns and detect where binning will matter the most.

3) We implemented the method as a javascript package and a web tool.