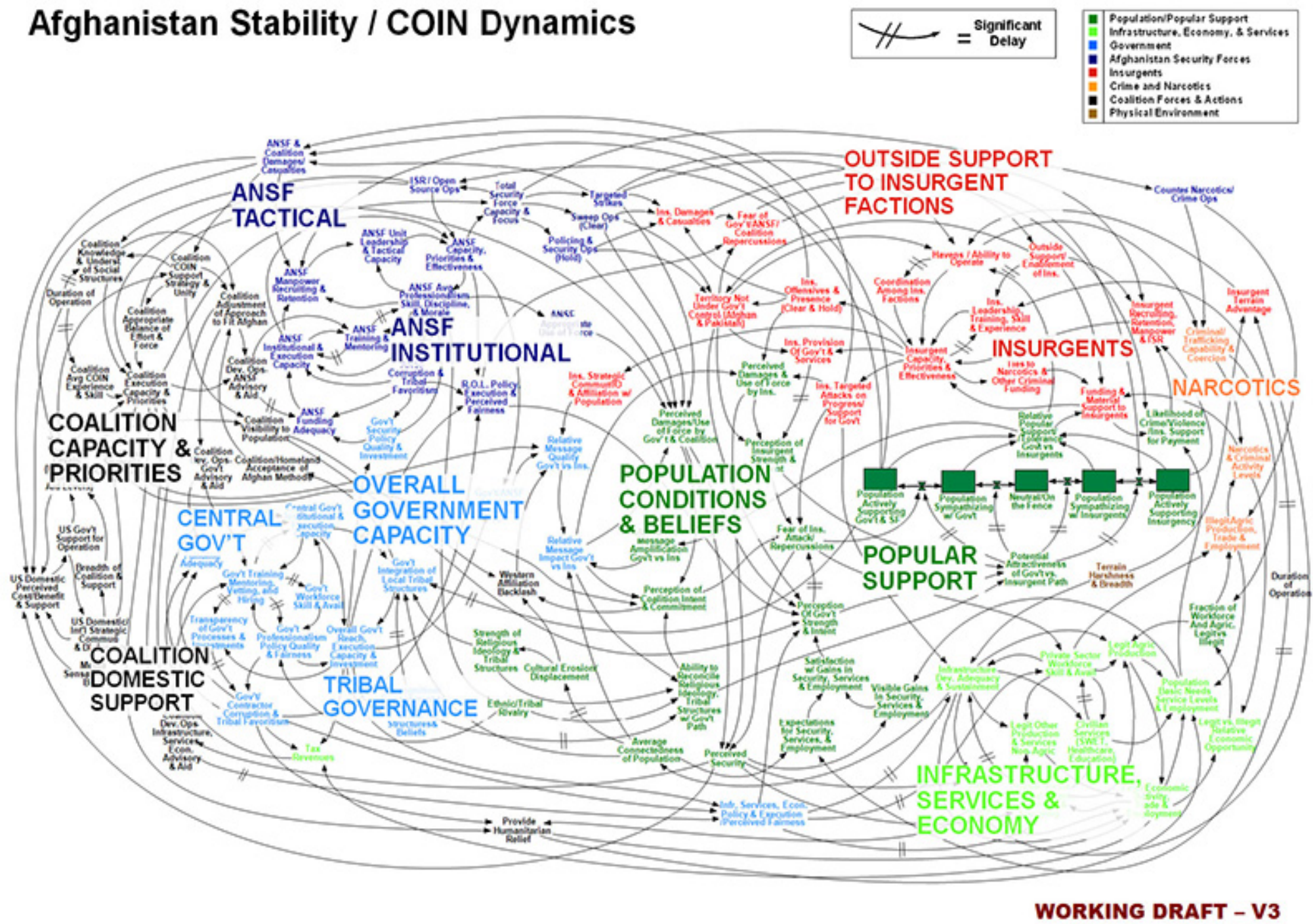
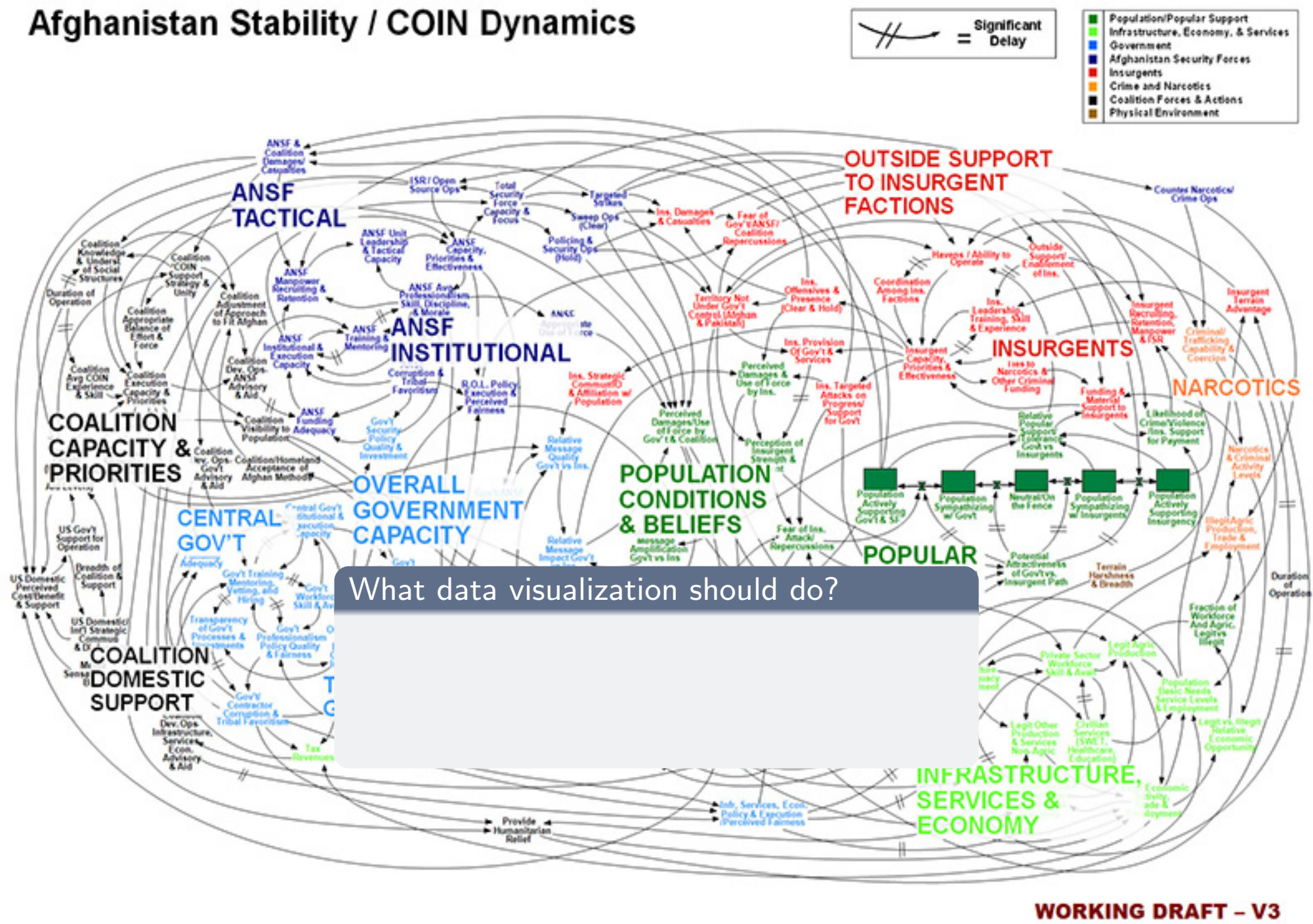
Working with Text Data
Bootcamp Section 2

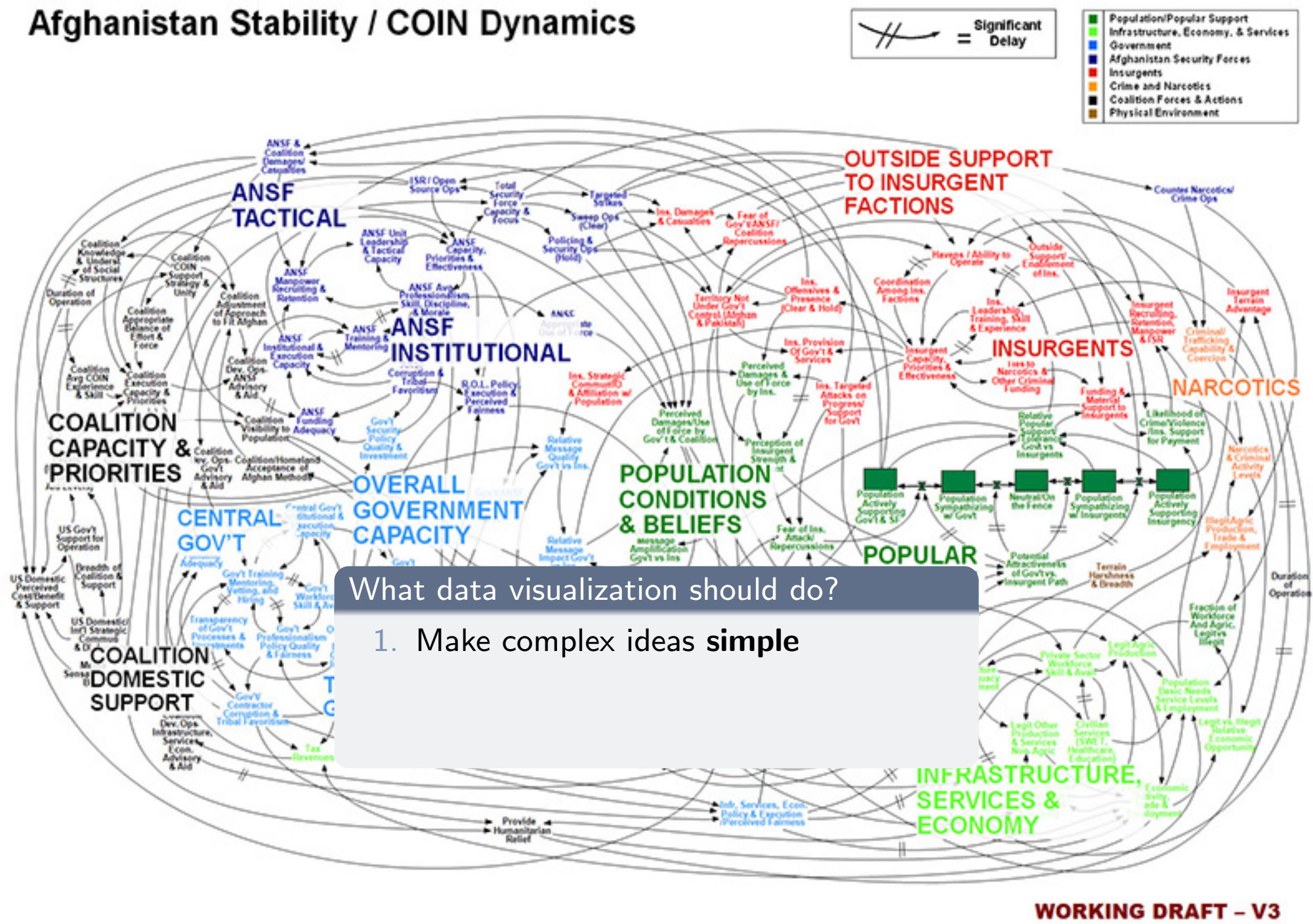
Joseph Adler, Drew Conway, Jake Hofman, Hilary Mason

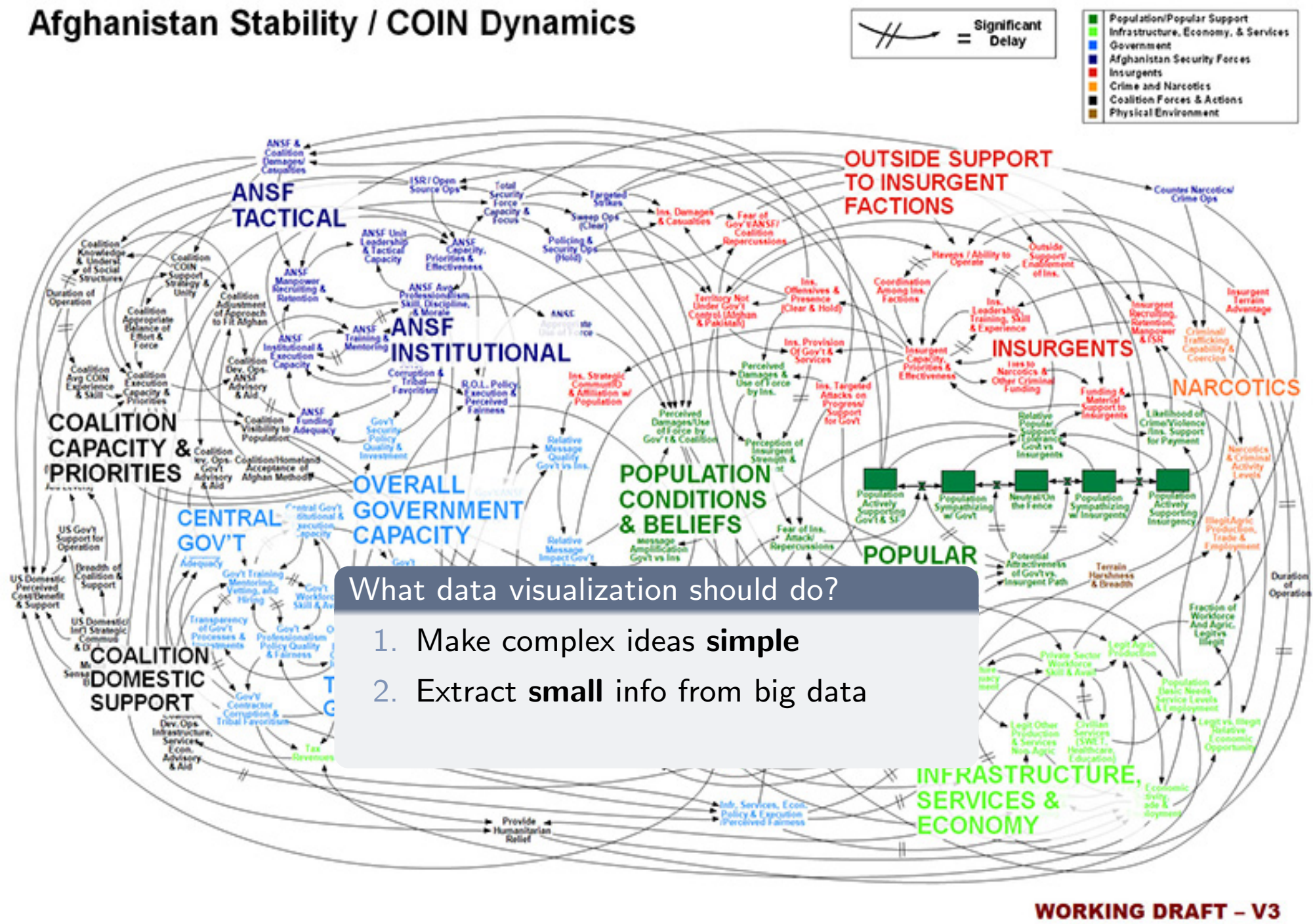
February 1, 2011

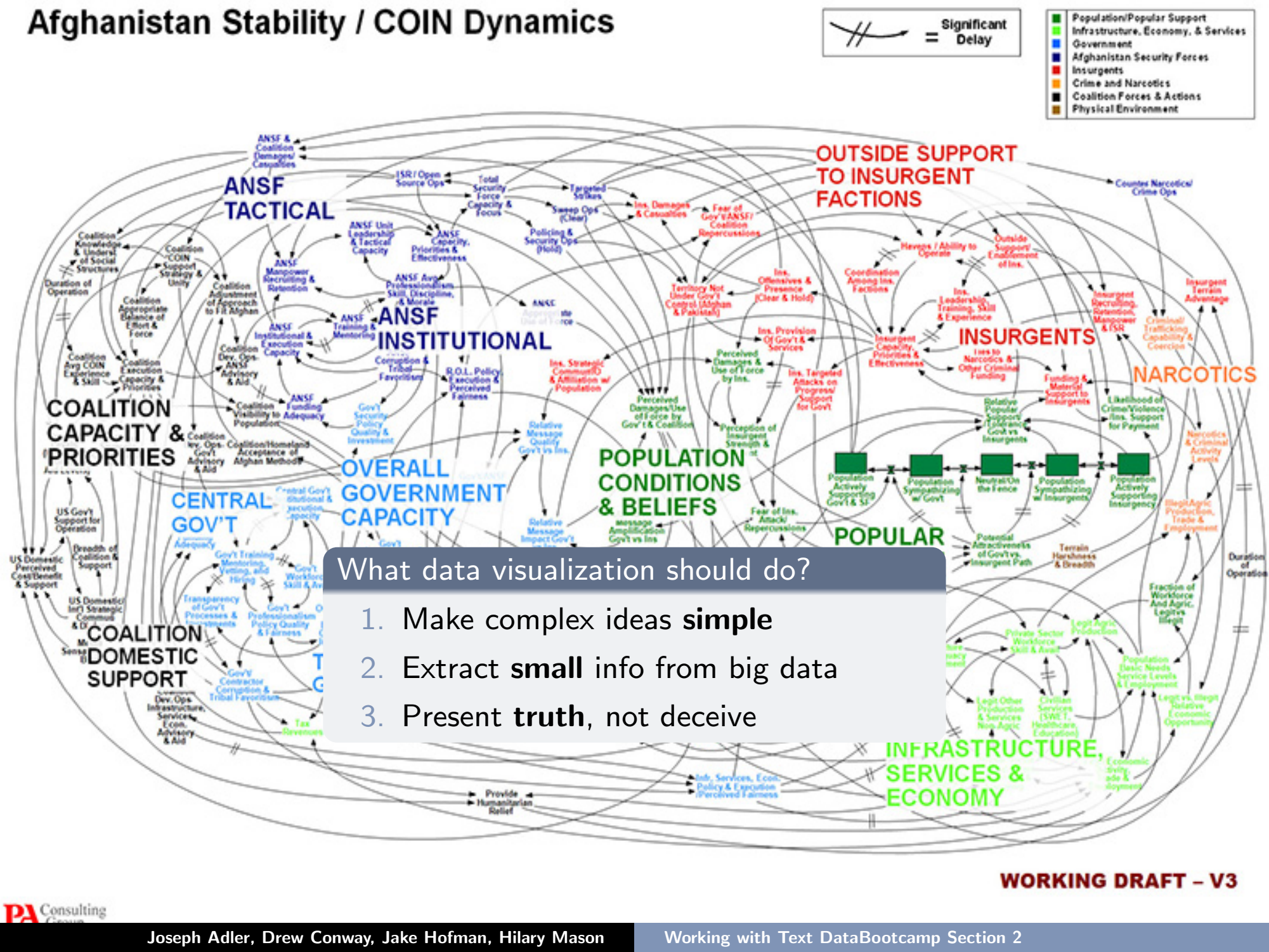
Afghanistan Stability / COIN Dynamics



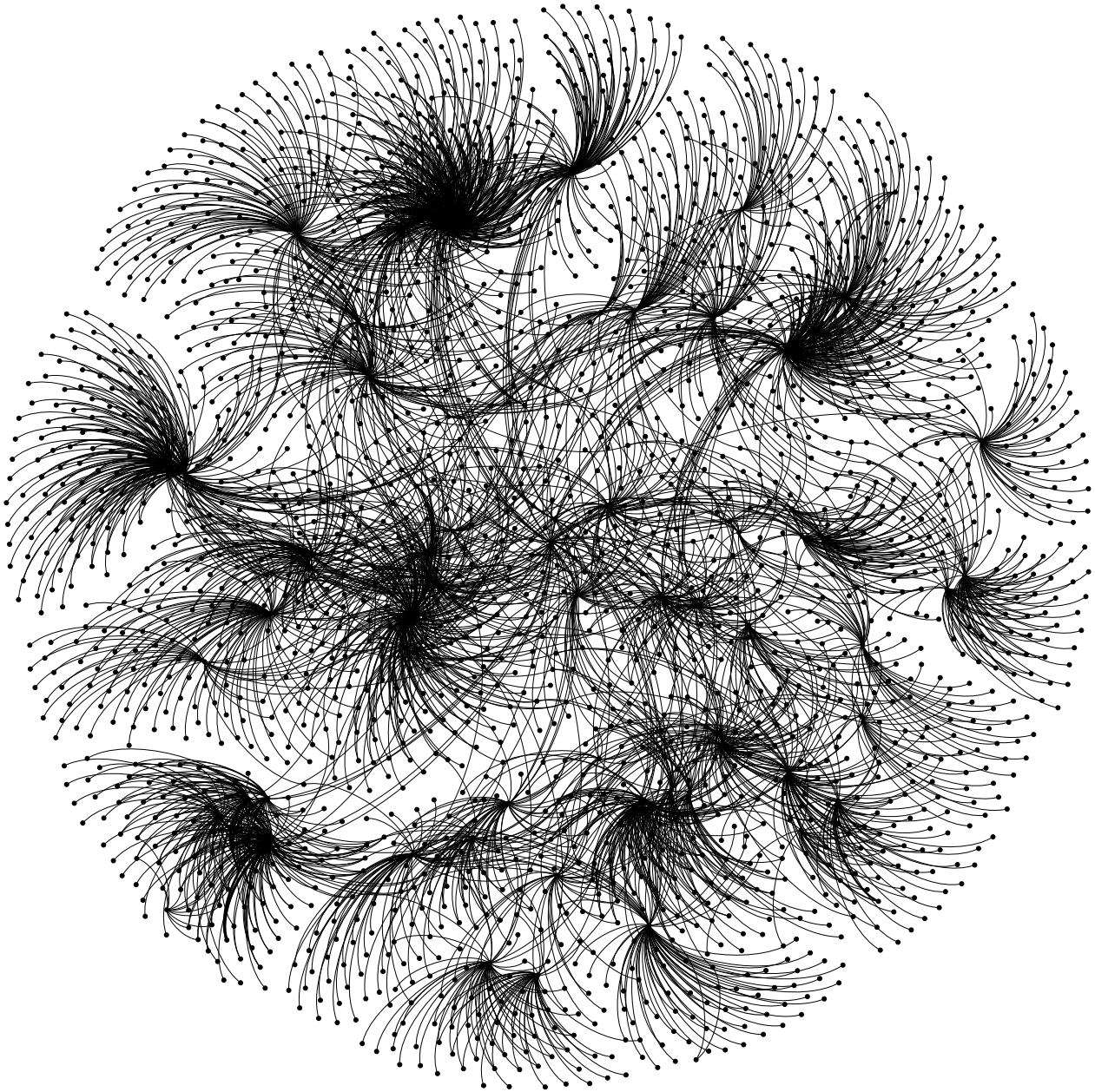

Afghanistan Stability / COIN Dynamics
= Significant Delay
Population/Popular Support
Infrastructure, Economy, & Services
Government
Afghanistan Security Forces
Insurgents
Crime and Narcotics
Coalition Forces & Actions
Physical Environment
ANSF TACTICAL
ANSF INSTITUTIONAL
OUTSIDE SUPPORT TO INSURGENT FACTIONS
INSURGENTS
NARCOTICS
COALITION CAPACITY & PRIORITIES
OVERALL GOVERNMENT CAPACITY
CENTRAL GOV'T
POPULATION CONDITIONS & BELIEFS
POPULAR
COALITION DOMESTIC SUPPORT
INFRASTRUCTURE, SERVICES & ECONOMY
What data visualization should do?
WORKING DRAFT – V3


Afghanistan Stability / COIN Dynamics
= Significant Delay
Population/Popular Support
Infrastructure, Economy, & Services
Government
Afghanistan Security Forces
Insurgents
Crime and Narcotics
Coalition Forces & Actions
Physical Environment
ANSF TACTICAL
ANSF INSTITUTIONAL
OUTSIDE SUPPORT TO INSURGENT FACTIONS
INSURGENTS
NARCOTICS
COALITION CAPACITY & PRIORITIES
OVERALL GOVERNMENT CAPACITY
CENTRAL GOV'T
POPULATION CONDITIONS & BELIEFS
POPULAR
COALITION DOMESTIC SUPPORT
INFRASTRUCTURE, SERVICES & ECONOMY
What data visualization should do?
1. Make complex ideas simple
WORKING DRAFT – V3

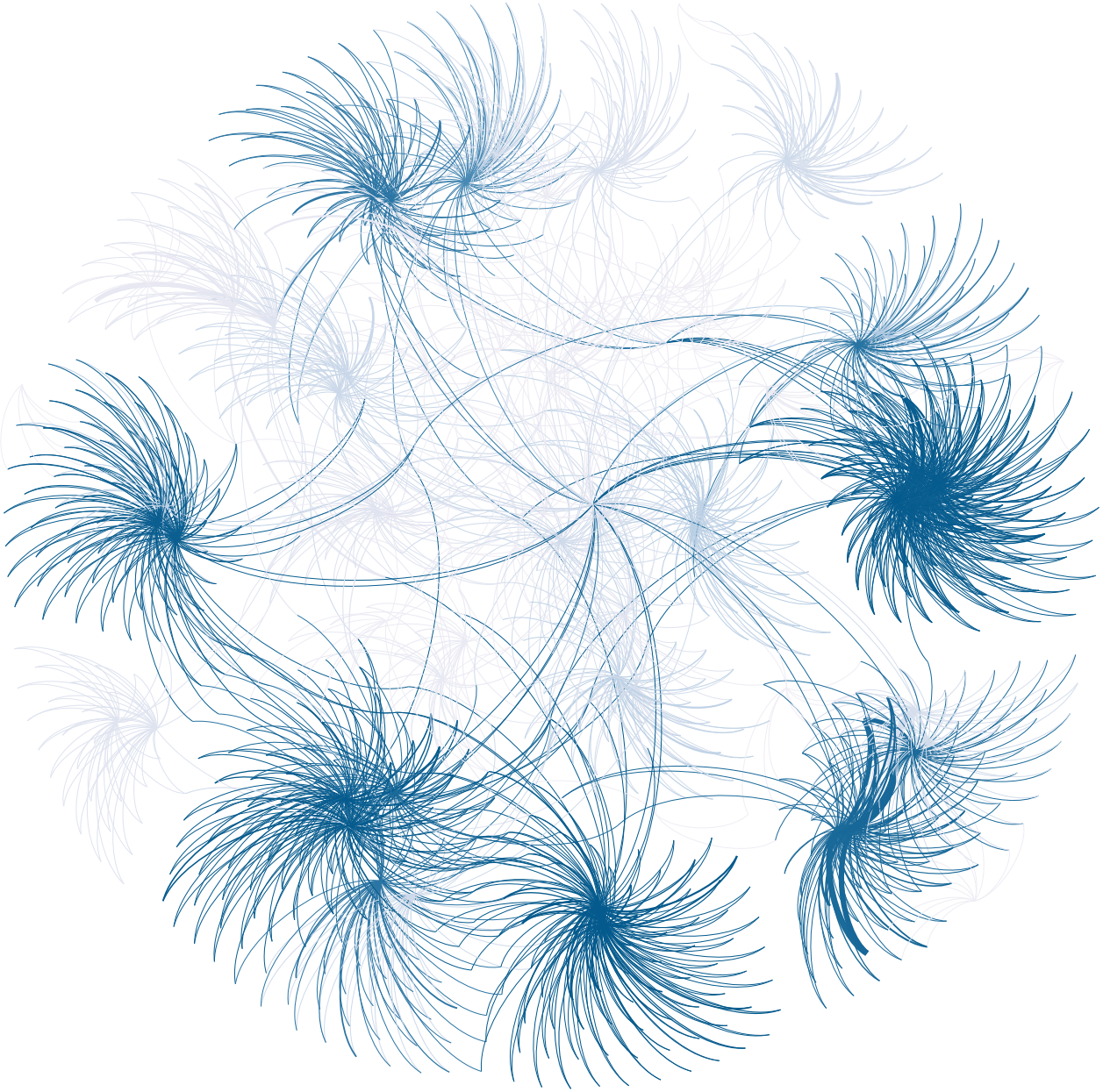

Afghanistan Stability / COIN Dynamics
= Significant Delay
Population/Popular Support
Infrastructure, Economy, & Services
Government
Afghanistan Security Forces
Insurgents
Crime and Narcotics
Coalition Forces & Actions
Physical Environment
ANSF TACTICAL
ANSF INSTITUTIONAL
OUTSIDE SUPPORT TO INSURGENT FACTIONS
INSURGENTS
NARCOTICS
COALITION CAPACITY & PRIORITIES
OVERALL GOVERNMENT CAPACITY
CENTRAL GOV'T
POPULATION CONDITIONS & BELIEFS
POPULAR
COALITION DOMESTIC SUPPORT
INFRASTRUCTURE, SERVICES & ECONOMY
What data visualization should do?
1. Make complex ideas **simple**
2. Extract **small** info from big data
WORKING DRAFT – V3


Afghanistan Stability / COIN Dynamics
= Significant Delay
Population/Popular Support
Infrastructure, Economy, & Services
Government
Afghanistan Security Forces
Insurgents
Crime and Narcotics
Coalition Forces & Actions
Physical Environment
ANSF TACTICAL
ANSF INSTITUTIONAL
OUTSIDE SUPPORT TO INSURGENT FACTIONS
INSURGENTS
NARCOTICS
COALITION CAPACITY & PRIORITIES
OVERALL GOVERNMENT CAPACITY
CENTRAL GOV'T
POPULATION CONDITIONS & BELIEFS
POPULAR
COALITION DOMESTIC SUPPORT
INFRASTRUCTURE, SERVICES & ECONOMY
What data visualization should do?
1. Make complex ideas simple
2. Extract small info from big data
3. Present truth, not deceive
WORKING DRAFT – V3

Make complex ideas **simple**

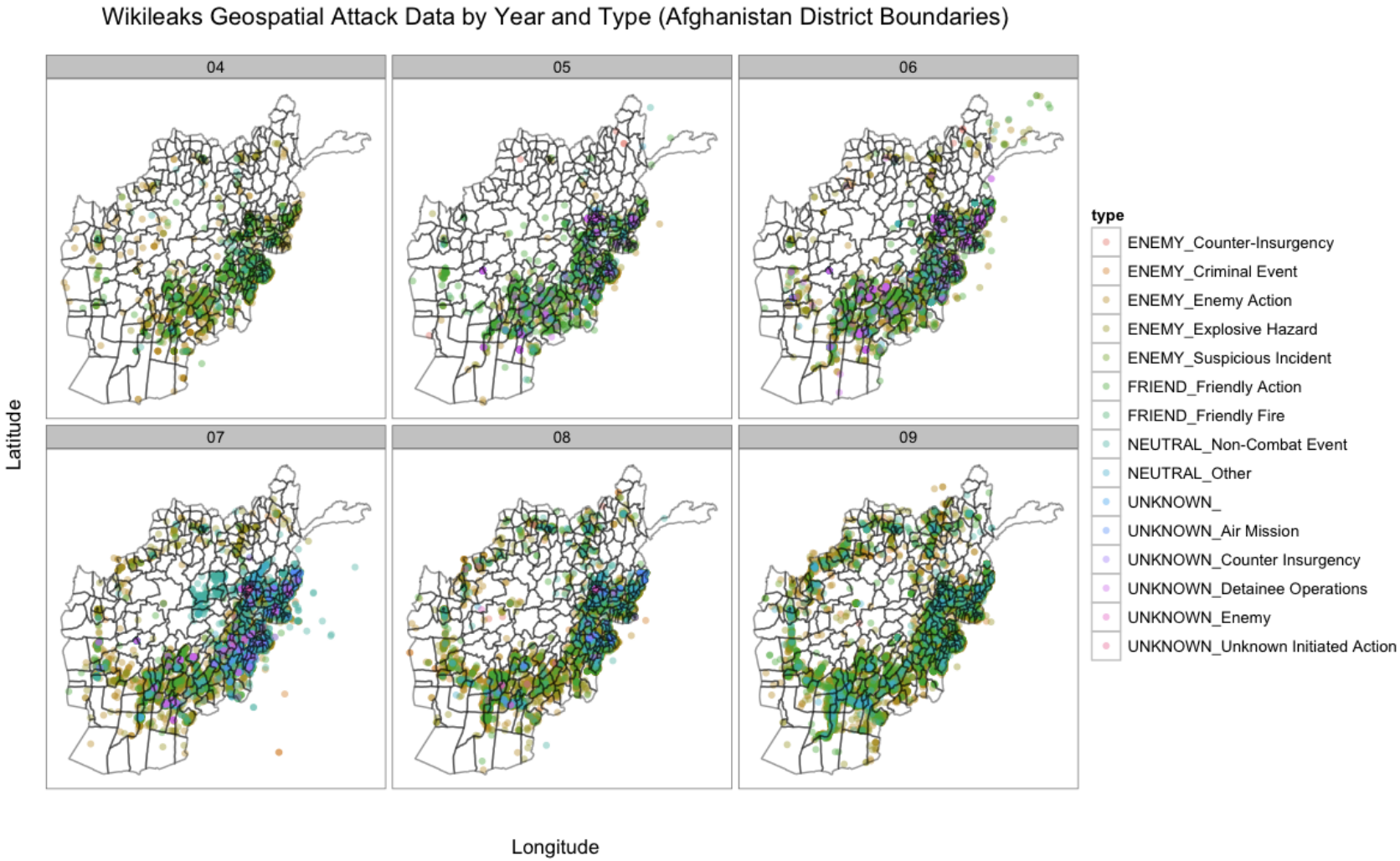


Make complex ideas **simple**

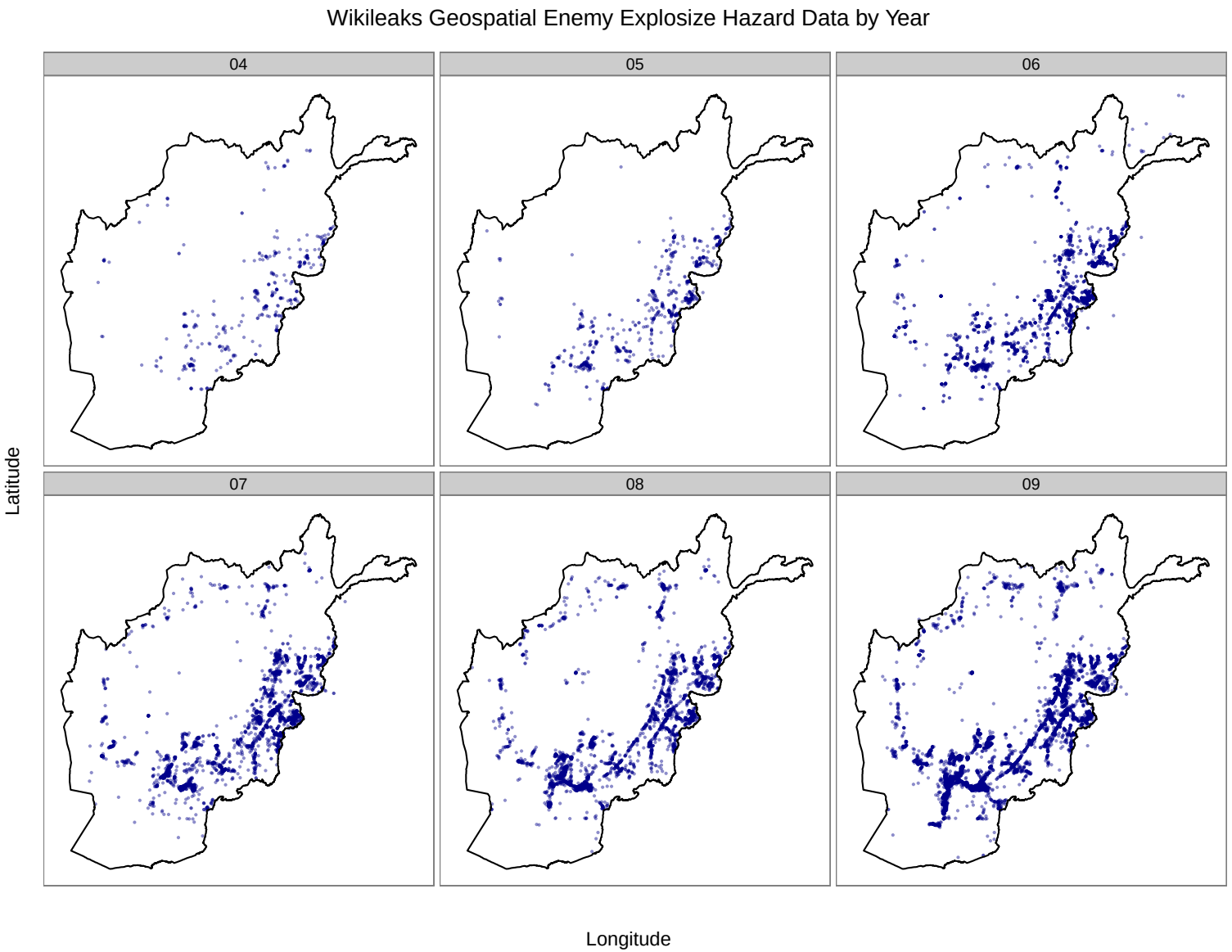


Good example of complexity reduction

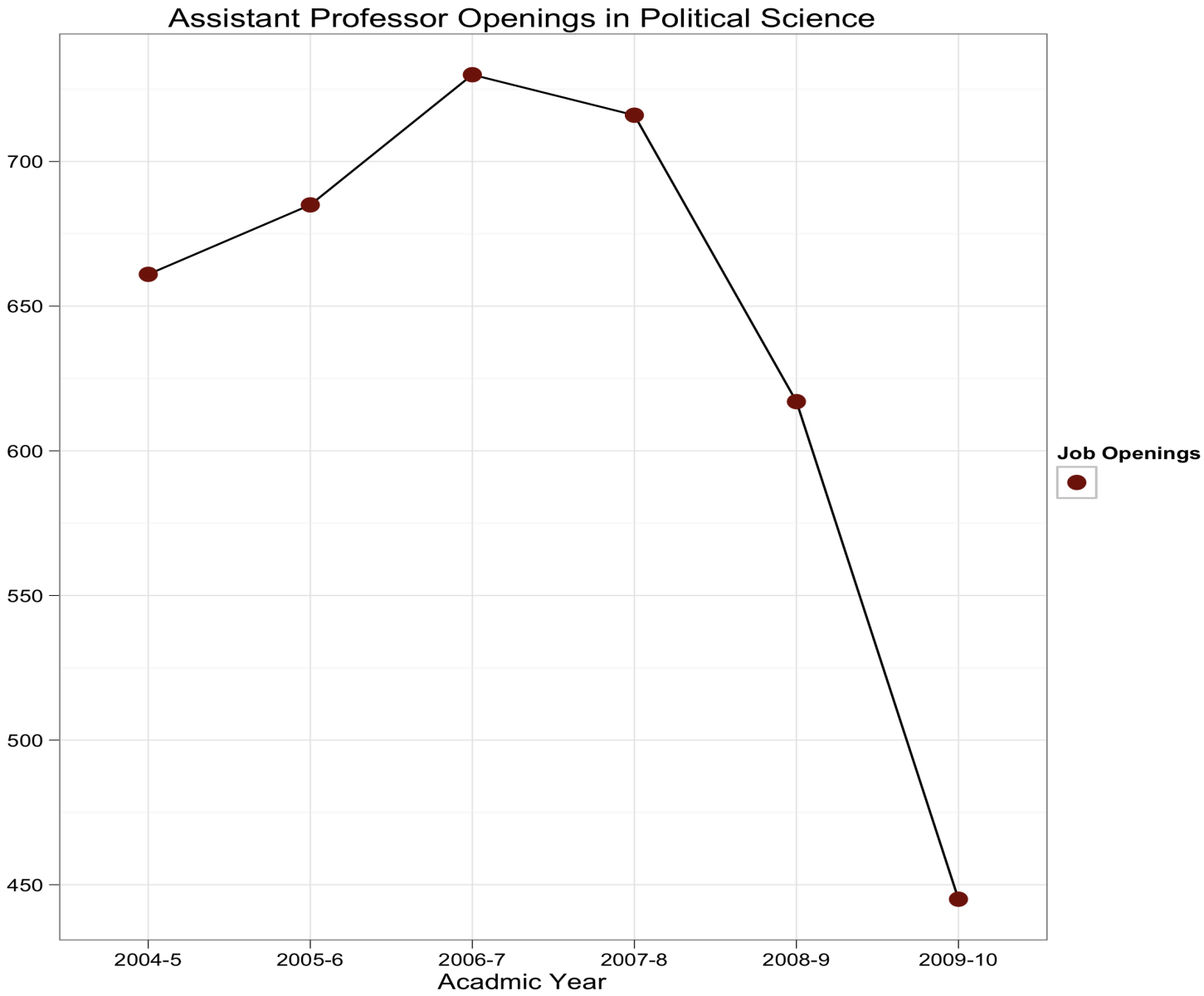
Extract **small** info from big data



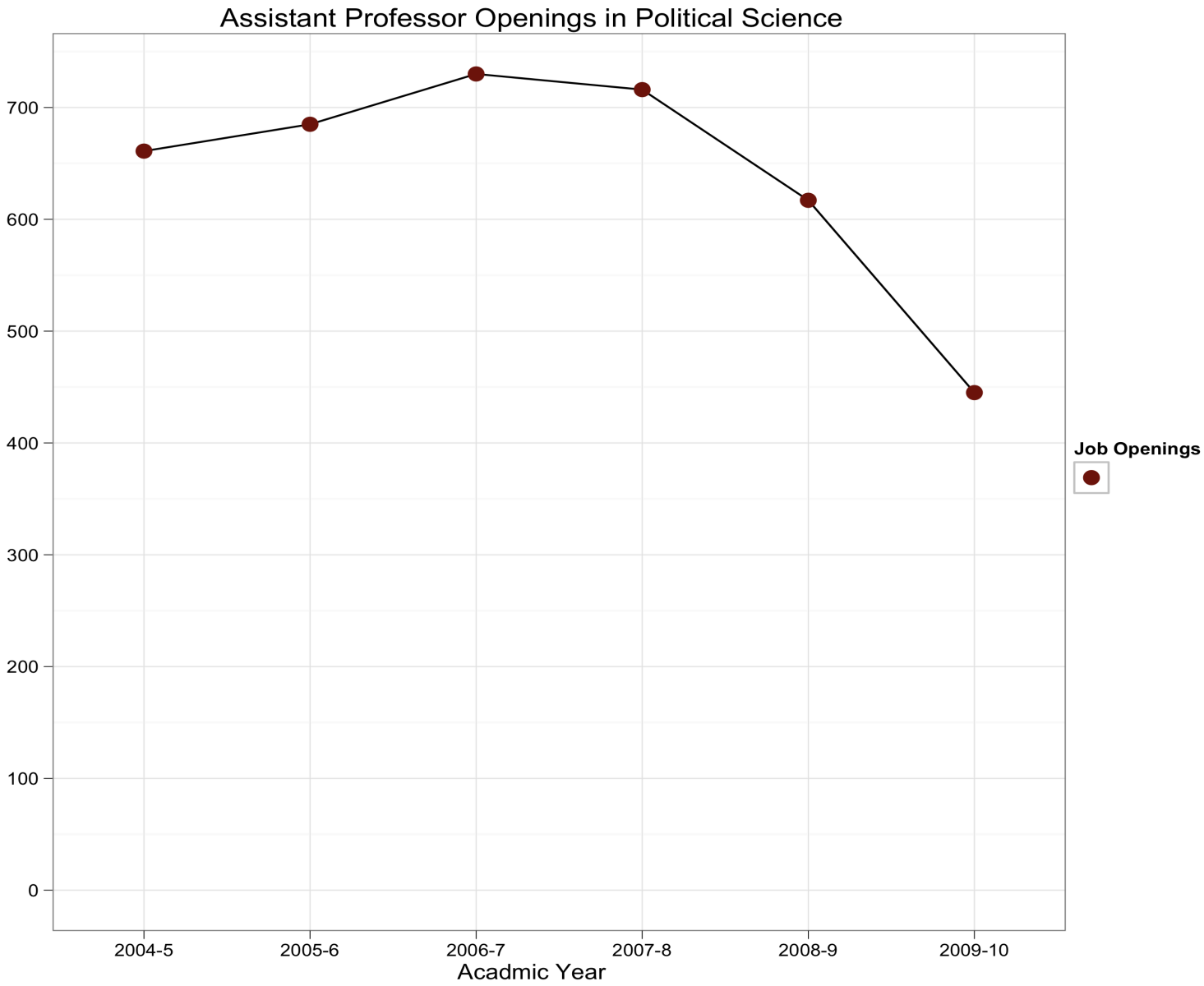
Extract **small** info from big data



Present **truth**, do not deceive



Present **truth**, do not deceive



Commercial data visualization tools

Data visualization is very popular...

Commercial data visualization tools

Data visualization is very popular...

Strata ♡'s visualization!

```
$ wget -O strata.htm http://strataconf.com/strata2011/public/schedule/full
$ tr -cs 'A-Za-z' '\n' < strata.html | grep -c ''visual''
24
```

Commercial data visualization tools

Data visualization is very popular...

Strata ♡'s visualization!

```
$ wget -O strata.htm http://strataconf.com/strata2011/public/schedule/full
$ tr -cs 'A-Za-z' '\n' < strata.html | grep -c ''visual''
24
```

Open-source visualization tools

For this tutorial we will not be using any commercial tools

- Instead utilizing **only open-source tools**

Open-source visualization tools

For this tutorial we will not be using any commercial tools

- Instead utilizing **only open-source tools**

There are many tools at our disposal

- Here we will use **two premier scientific computing environments**

Python's scientific computing holy trinity

Python's scientific computing holy trinity

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

Python's scientific computing holy trinity

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

The primary data type in `SciPy` is an array

- Data manipulation is similar to that of MATLAB

Python's scientific computing holy trinity

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

The primary data type in `SciPy` is an array

- Data manipulation is similar to that of MATLAB

`NumPy` is an extension of the `SciPy` data type to include **multidimensional arrays and matrices**

- Provides many functions for working on arrays and matrices
- Very useful for representing relational data

Python's scientific computing holy trinity

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

The primary data type in `SciPy` is an array

- Data manipulation is similar to that of MATLAB

`NumPy` is an extension of the `SciPy` data type to include **multidimensional arrays and matrices**

- Provides many functions for working on arrays and matrices
- Very useful for representing relational data

Both `SciPy` and `NumPy` rely on the C library `LAPACK` for very fast implementation

Python's scientific computing holy trinity

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

The primary data type in `SciPy` is an array

- Data manipulation is similar to that of MATLAB

`NumPy` is an extension of the `SciPy` data type to include **multidimensional arrays and matrices**

- Provides many functions for working on arrays and matrices
- Very useful for representing relational data

Both `SciPy` and `NumPy` rely on the C library `LAPACK` for very fast implementation

`matplotlib` is **primary plotting library in Python**

- Supports 2- and 3-D plotting
- API allows embedding in apps

Python's scientific computing holy trinity

SciPy.org Sponsored By ENTHOUGHT

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

The primary data type in `SciPy` is an array

- Data manipulation is similar to that of MATLAB

NumPy

`NumPy` is an extension of the `SciPy` data type to include **multidimensional arrays and matrices**

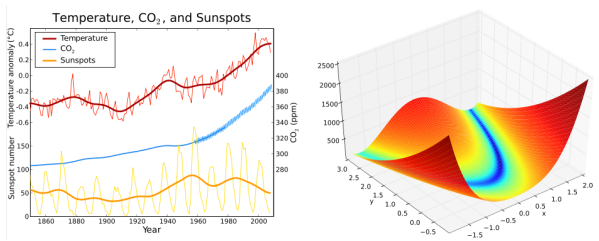
- Provides many functions for working on arrays and matrices
- Very useful for representing relational data

Both `SciPy` and `NumPy` rely on the C library `LAPACK` for very fast implementation

matplotlib

`matplotlib` is **primary plotting library in Python**

- Supports 2- and 3-D plotting
- API allows embedding in apps



Python's scientific computing holy trinity

SciPy.org Sponsored By ENTHOUGHT

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

The primary data type in `SciPy` is an array

- Data manipulation is similar to that of MATLAB

NumPy

`NumPy` is an extension of the `SciPy` data type to include **multidimensional arrays and matrices**

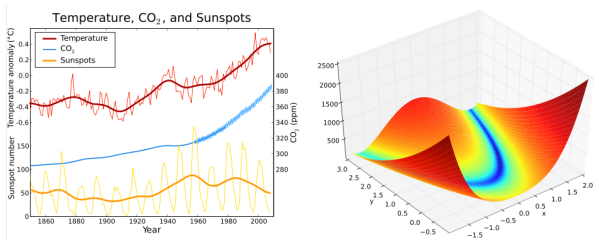
- Provides many functions for working on arrays and matrices
- Very useful for representing relational data

Both `SciPy` and `NumPy` rely on the C library `LAPACK` for very fast implementation

matplotlib

`matplotlib` is **primary plotting library in Python**

- Supports 2- and 3-D plotting
- API allows embedding in apps



All graphics are highly customizable and professional publication ready

Python's scientific computing holy trinity

SciPy.org Sponsored By ENTHOUGHT

Python's primary library for **mathematical and statistical** computing. Containing sub-libs for

- Numeric optimization
- Clustering
- Linear algebra
- ..and many others

The primary data type in `SciPy` is an array

- Data manipulation is similar to that of MATLAB

NumPy

`NumPy` is an extension of the `SciPy` data type to include **multidimensional arrays and matrices**

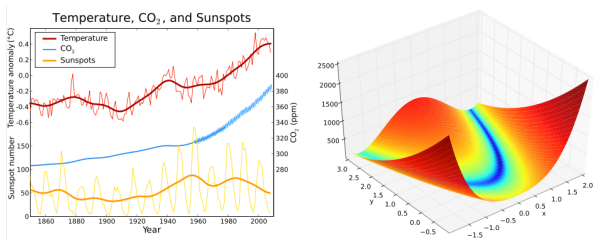
- Provides many functions for working on arrays and matrices
- Very useful for representing relational data

Both `SciPy` and `NumPy` rely on the C library `LAPACK` for very fast implementation

matplotlib

`matplotlib` is **primary plotting library in Python**

- Supports 2- and 3-D plotting
- API allows embedding in apps



All graphics are highly customizable and professional publication ready

Data visualization in in R

The R Language

lattice

ggplot2

Data visualization in in R

The R Language

"freely available language and environment for statistical computing and graphics..."

lattice

ggplot2

Data visualization in in R

The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

lattice

ggplot2

Data visualization in in R

The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

R Task Views

- 28 development areas
- Bayesian, ML, NLP, **Graphics**

lattice

ggplot2

Data visualization in in R

The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

R Task Views

- 28 development areas
- Bayesian, ML, NLP, **Graphics**

Two popular visualization packages

- `lattice`
- `ggplot2`

`lattice`

`ggplot2`

The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

R Task Views

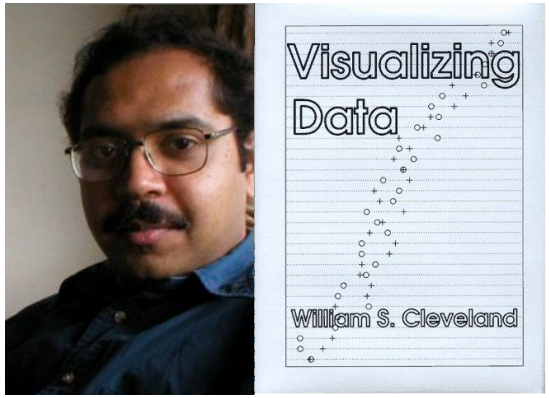
- 28 development areas
- Bayesian, ML, NLP, **Graphics**

Two popular visualization packages

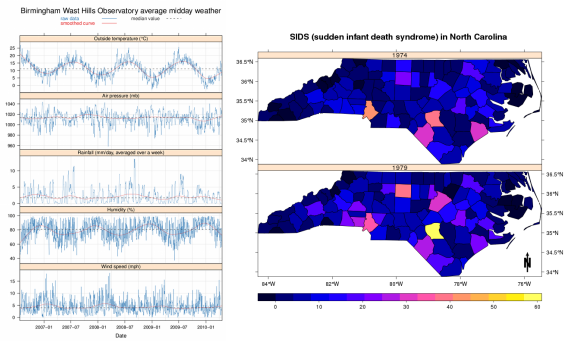
- `lattice`
- `ggplot2`

`lattice`

Developed by Deepayan Sarkar



- Implementation of Trellis graphics



`ggplot2`

The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

R Task Views

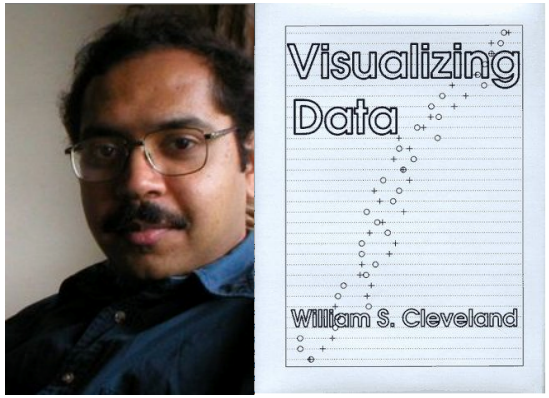
- 28 development areas
- Bayesian, ML, NLP, **Graphics**

Two popular visualization packages

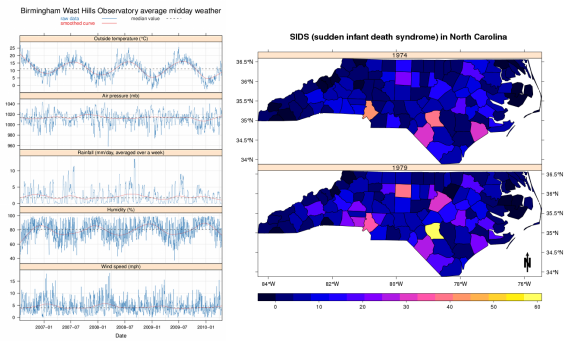
- `lattice`
- `ggplot2`

`lattice`

Developed by Deepayan Sarkar



- Implementation of Trellis graphics



`ggplot2`

The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

R Task Views

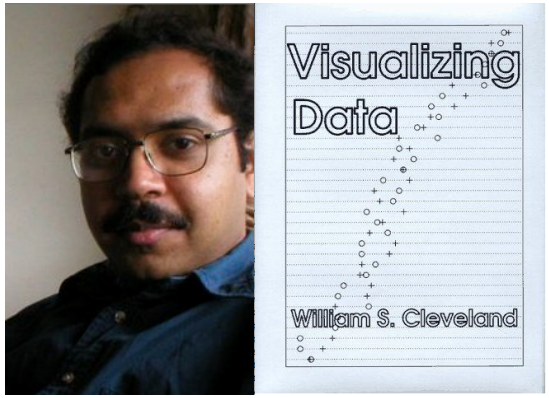
- 28 development areas
- Bayesian, ML, NLP, **Graphics**

Two popular visualization packages

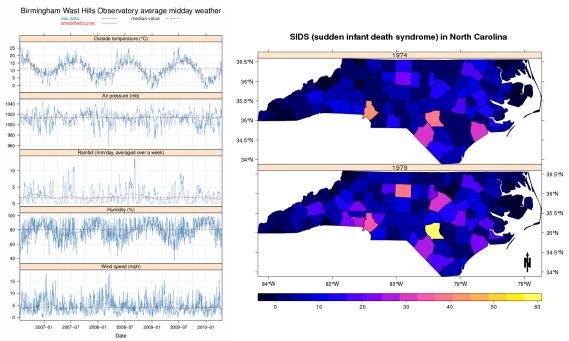
- `lattice`
- `ggplot2`

`lattice`

Developed by Deepayan Sarkar

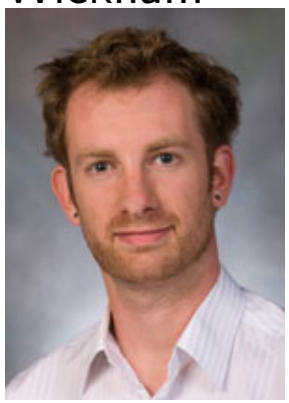


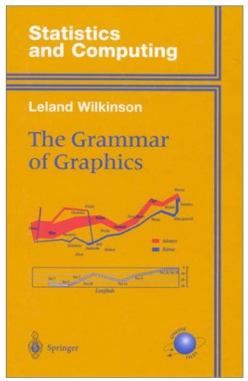
- Implementation of Trellis graphics



`ggplot2`

Developed by Hadley Wickham





The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

R Task Views

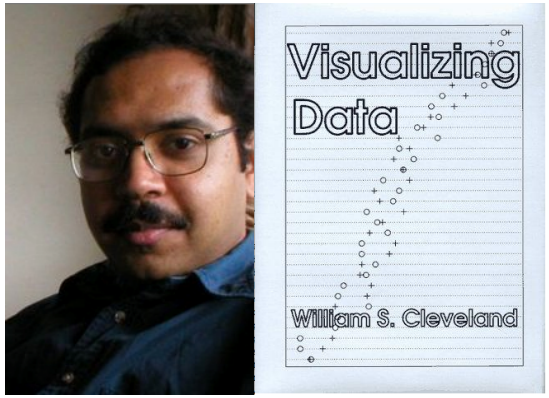
- 28 development areas
- Bayesian, ML, NLP, **Graphics**

Two popular visualization packages

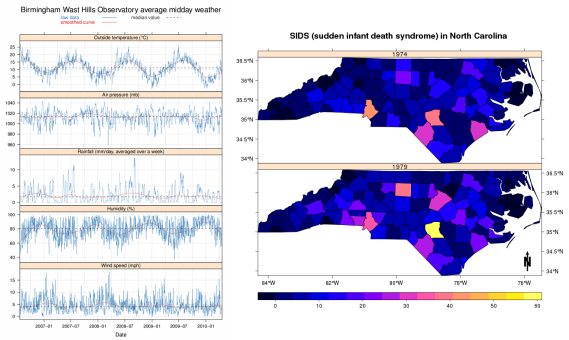
- `lattice`
- `ggplot2`

`lattice`

Developed by Deepayan Sarkar

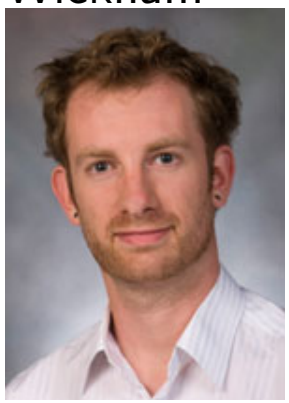


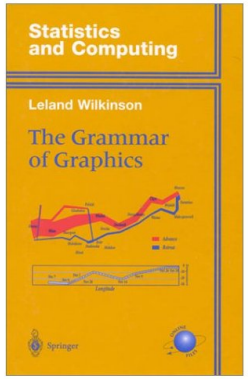
- Implementation of Trellis graphics



`ggplot2`

Developed by Hadley Wickham





- Visualizations are "grammatical layers"

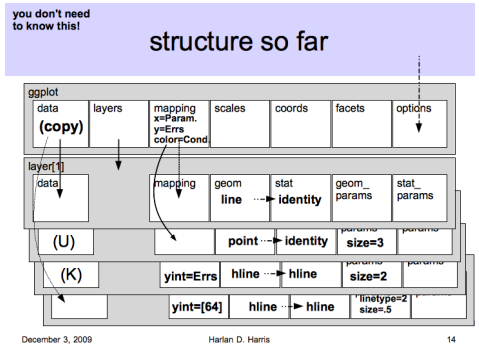


Image source: Harlan Harris

The R Language

"freely available language and environment for statistical computing and graphics..."

CRAN

- Massive library of specialized packages
- 2,775 available

R Task Views

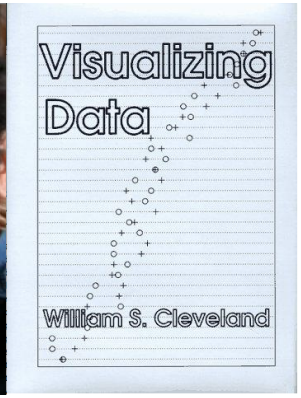
- 28 development areas
- Bayesian, ML, NLP, **Graphics**

Two popular visualization packages

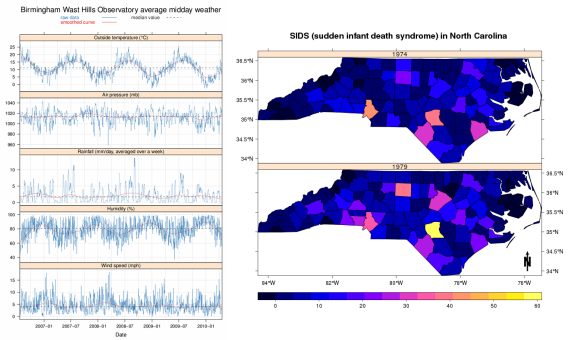
- `lattice`
- `ggplot2`

`lattice`

Developed by Deepayan Sarkar

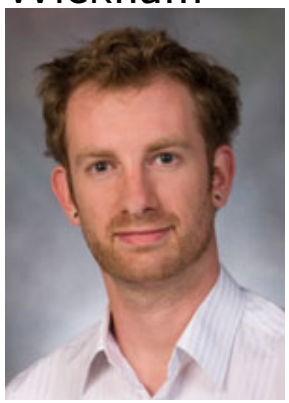


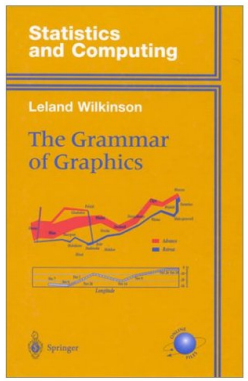
- Implementation of Trellis graphics



`ggplot2`

Developed by Hadley Wickham





- Visualizations are "grammatical layers"

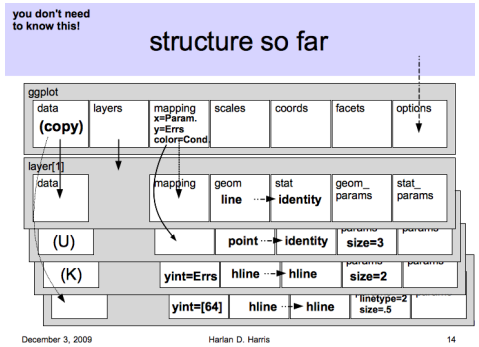


Image source: Harlan Harris

Creating a simple visualization

As an introduction to each environment we will make the same plot in both `matplotlib` and `ggplot2`

[1] Image source: `http://mathworld.wolfram.com/NormalDistribution.html`

Creating a simple visualization

As an introduction to each environment we will make the same plot in both `matplotlib` and `ggplot2`

To begin, we'll generate canonical data and visualize it

- Histogram of 10,000 randomly generated numbers from a **standard Normal distribution**
- $\mu = 0$, $\sigma = 1$
- Then, overlay Normal density function to observe "fit" [1]

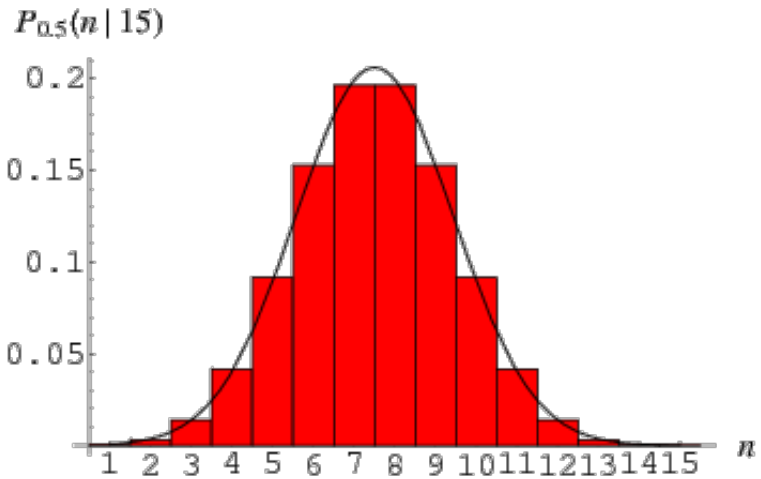
[1] Image source: `http://mathworld.wolfram.com/NormalDistribution.html`

Creating a simple visualization

As an introduction to each environment we will make the same plot in both `matplotlib` and `ggplot2`

To begin, we'll generate canonical data and visualize it

- Histogram of 10,000 randomly generated numbers from a **standard Normal distribution**
- $\mu = 0$, $\sigma = 1$
- Then, overlay Normal density function to observe "fit" [1]



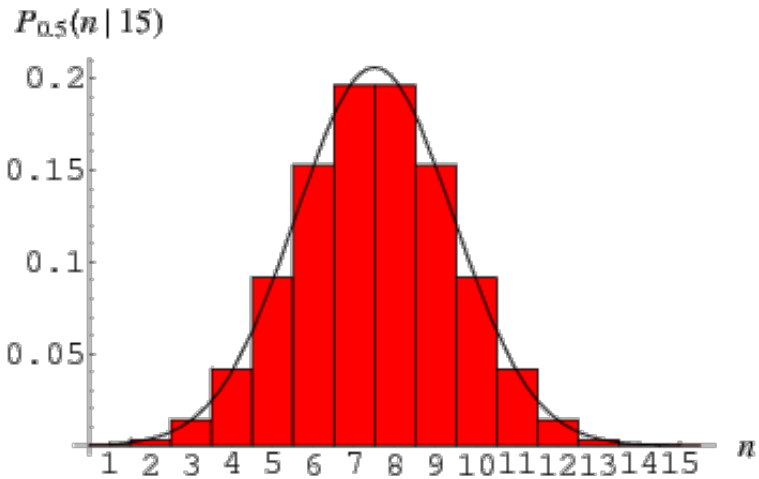
[1] Image source: `http://mathworld.wolfram.com/NormalDistribution.html`

Creating a simple visualization

As an introduction to each environment we will make the same plot in both `matplotlib` and `ggplot2`

To begin, we'll generate canonical data and visualize it

- Histogram of 10,000 randomly generated numbers from a **standard Normal distribution**
- $\mu = 0$, $\sigma = 1$
- Then, overlay Normal density function to observe "fit" [1]



We'll start by working in Python with `matplotlib`...

[1] Image source: `http://mathworld.wolfram.com/NormalDistribution.html`

My first matplotlib visualization: 1/3

Out first steps are to load the libraries and generate data

matplotlib and the normal distribution

```
>>> import matplotlib.pylab as plt
>>> from scipy.stats import norm
```

Generate 10,000 random draws from a normal

```
>>> random_normal = norm.rvs(0,1,size=10000)
```

Create a figure to draw to

```
>>> fig=plt.figure(figsize = (8,6))
```

My first matplotlib visualization: 2/3

Next, we draw the draw to the figure

Add the histogram and Normal PDF

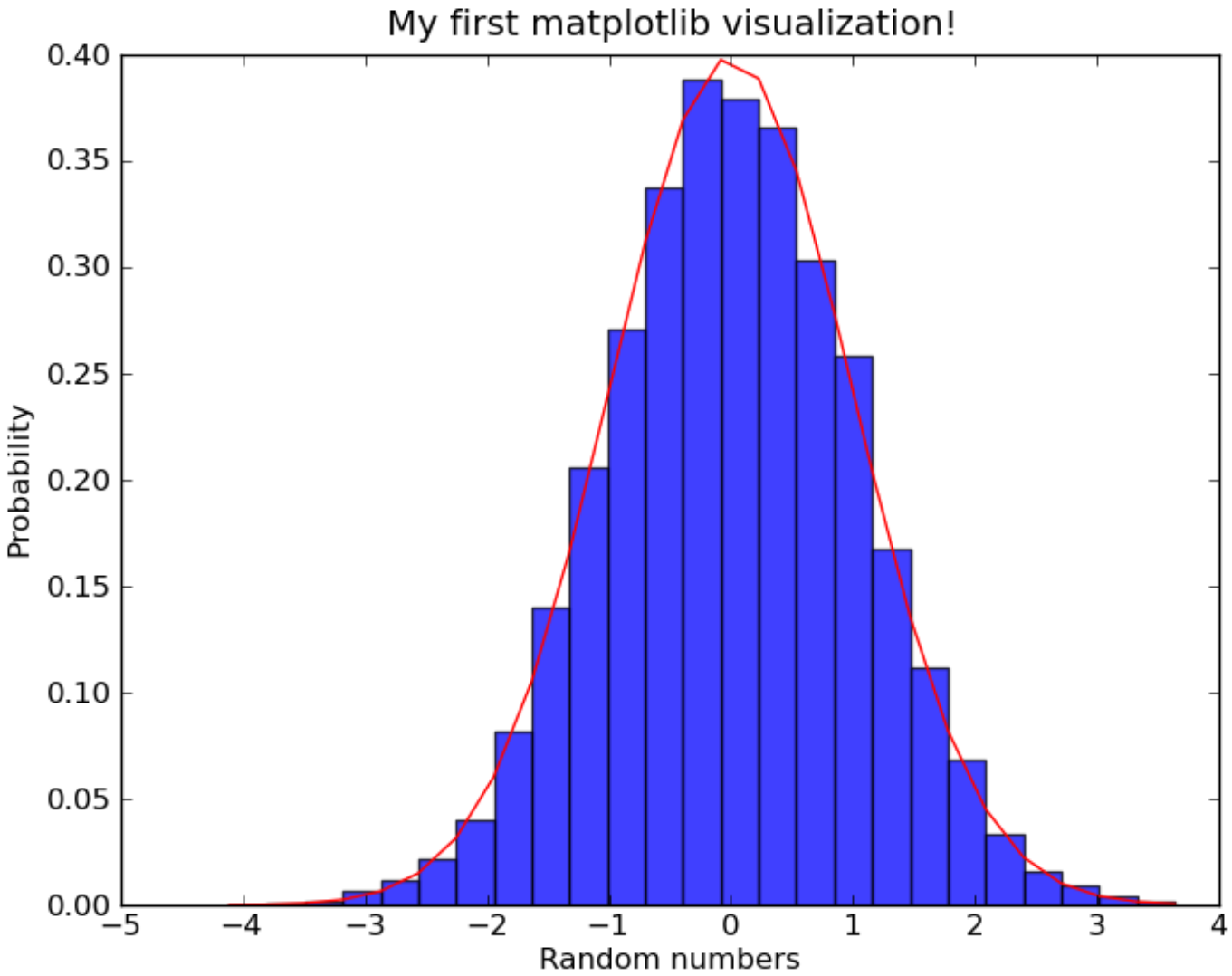
```
>>> n, bins, patches =
    plt.hist(random_numbers,normed=True,bins=25,
    alpha=0.75)
>>> y = norm.pdf(bins)
>>> plt.plot(bins,y,"r-")
```

Add plot labels, and save

```
>>> plt.xlabel("Random numbers")
>>> plt.ylabel("Density")
>>> plt.title("My first matplotlib visualization!")
>>> plt.savefig("matplotlib_first.png")
```

My first `matplotlib` visualization: 3/3

Now, bask in the glory of your data visualization!



My first ggplot2 visualization 1/4

Again, first load library and create data

Load the ggplot2 package

```
> library(ggplot2)
```

Create our first data.frame

```
> random.numbers<-rnorm(10000,0,1)
> norm.dframe<-as.data.frame(list(Norm=random.numbers))
```

Create base ggplot2 layer

```
> norm.plt<-ggplot(norm.dframe,aes(Norm)) +
    geom_histogram(aes(y = ..density.., fill="blue",
    colour="black",alpha=0.75))
```

My first ggplot2 visualization 2/4

Next, we build up layers from the base

Add Normal PDF

```
> norm.plt<-norm.plt+stat_function(fun = dnorm,
        colour = "red")
```

Deal with colors and legends

```
> norm.plt<-norm.plt+scale_colour_manual(values =
        c("black"="black","red"="red"), legend = FALSE)
> norm.plt<-norm.plt+scale_fill_manual(values =
    c("blue"="blue"), legend = FALSE)
> norm.plt<-norm.plt+scale_alpha(legend = FALSE)+
```

My first ggplot2 visualization 3/4

Finally, add add labels and save

This time, we'll make a PDF

```
> norm.plt<-norm.plt+xlab("Random numbers")
> norm.plt<-norm.plt+ylab("Density")
> norm.plt<-norm.plt+opts(title=
        "My first ggplot2 visualization!")
> ggsave(plot = norm.plt, filename =
    "ggplot2_first.pdf", height = 6,
    width = 8)
```

My first ggplot2 visualization 4/4

Who's the baddest data visualizer?! You are!