Visualization (Marks and Encoding)

Peter Ganong and Maggie Shi

January 14, 2026

Roadmap of the lecture

- Data for this lecture
- Fundamentals of data visualization
  - Data types: what information is in my data?
  - Encoding and marks: how do I want to convey this information?
- Applying this in `altair`

Global Health Data

Introducing global health data

- Throughout the rest of lecture, we will be visualizing global health and population data for a number of countries, over the time period of 1955 to 2005.
- The data was collected by the Gapminder Foundation and shared in Hans Rosling's fantastic TED talk.
- Roadmap: load data and review first five rows

Load data

Let's first load the dataset from the vega-datasets collection into a Pandas data frame.

```python
import altair as alt
from vega_datasets import data as vega_data
data = vega_data.gapminder()
data["cluster_name"] = data["cluster"].map({
    0: "South Asia",
    1: "Western Europe",
    2: "Sub-Saharan Africa",
    3: "Americas & Anglos",
    4: "East Asia",
    5: "Middle East & North Africa"
})
```

(Note: we deviate from Heer et al. do some data cleaning to fix a poorly defined variable, `cluster`)

Load data

```python
data.head(4)
```

	year	country	cluster	pop	life_expect	fe
0	1955	Afghanistan	0	8891209	30.332	7.
1	1960	Afghanistan	0	9829450	31.997	7.
2	1965	Afghanistan	0	10997885	34.020	7.
3	1970	Afghanistan	0	12430623	36.088	7.

data summary

For each `country` and `year` (in 5-year intervals), we have:

- `fertility`: fertility in terms of the number of children per woman
- `life_expect`: life expectancy in years
- `pop`: total population
- `cluster_name`: region

Fundamentals of visualization: Data Types

Data types: intro and roadmap

Core **data types**, as recognized by `altair`:

- `'N'`: *nominal* type
- `'O'`: *ordinal* type
- `'Q'`: *quantitative* type
- `'T'`: *temporal* type

Nominal (N)

- *Nominal* data consists of **unordered** category names.
  - Also called *categorical* data
- **Questions**: *Is value A the same or different from value B? (A = B)*?
- **Answers**: conclusion we should be able to try is whether the values are the same or different
- **In `gapminder` data**: the `country` field is Nominal

Ordinal (O)

- *Ordinal* data consist of values that have a specific **rank-ordering**.
  - Note: ordinal does not necessarily mean numerical. E.g., survey results: “Good”, “Ok”, “Bad”
- **Questions**: *Does value A come before or after value B? (A < B)*
- **Answers**: statements like “A is less than B” or “A is greater than B”.
- **In `gapminder` data**: `year` field can be treated as `O`rdinal.

Quantitative (Q)

- *Quantitative* data measures numerical differences among values. Two types: *interval* and *ratio*
    - *Interval* data
        - Questions: *what is the distance to value A from value B?*
        - Answers: “A is 12 units away from B”
    - *Ratio* data
        - Questions: *How many are there of value A?*, *Value A is what proportion of value B? (A / B)
        - Answers: “how many babies per parent?”, “A is 10% of B”

Quantitative (Q), continued

- Key difference between *interval* and *ratio*: 0 is essential for ratio, but not interval data
- *(Note: we are following Heer et al. in use of the term “ratio,” recognizing that they mean it as encompassing more than just ratios)*

Quantitative (Q), continued

- **In `gapminder` data:** `year` is a quantitative *interval* field
- Whereas `fertility` and `life_expect` are quantitative *ratio* fields – zero is meaningful for calculating proportions
- `altair` represents quantitative data, but does not make a distinction between interval and ratio types

Quantitative (Q), continued

Discussion questions

- Why is it so important to include zeros for ratio data?
- Can you give an example where omitting zeros on the plot would lead the reader to misleading conclusions?

Temporal (T)

- *Temporal* values measure time points or intervals.
- Special case of quantitative values (timestamps) with rich semantics and conventions (i.e., the Gregorian calendar).
- Example temporal values include date strings such as `“2019-01-04”` and `“Jan 04 2019”`
- Also standardized date-times such as the ISO date-time format: `“2019-01-04T17:50:35.643Z”`
- There are no temporal values in our global development dataset above, as the `year` field is simply encoded as an integer.

Discussion question I

What are examples of variables that are:

- Nominal
- Ordinal
- Quantitative

Let's try to come up with at least three examples of each. For each example, state the comparison in a sentence.

Discussion question II

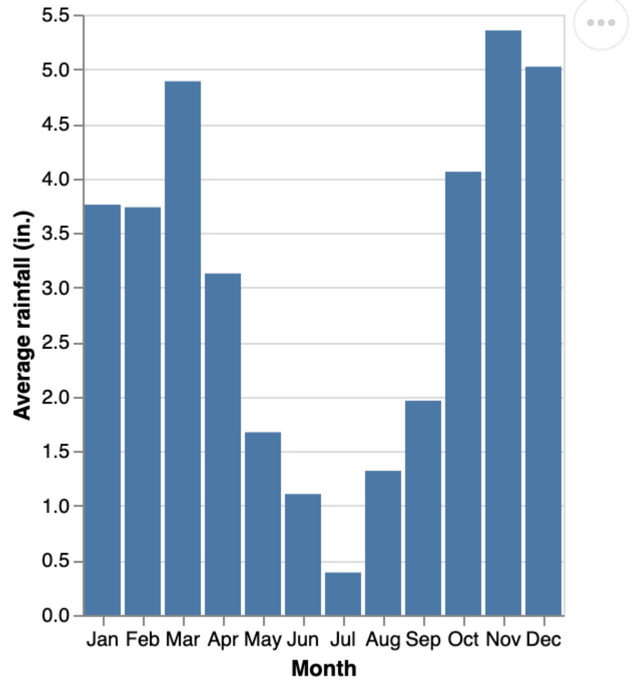
Suppose we have a dataset of ages (10 years old, 20 years old, 10 years old, 30 years old). What would it mean for these data to be:

- Nominal
- Ordinal
- Quantitative

What comparisons are feasible with each data type?

Add data types to last lecture's plot

```python
seattle = vega_data.seattle_weather()
alt.Chart(seattle).mark_bar().encode(
    alt.X('month(date):O', title = "Month"),
    alt.Y('average(precipitation):Q', title = "Average rainfall (in.)")
)
```



Do-pair-share

- What happens when you make `precipitation` `O`rdinal?
- What rank-ordering does `altair` assume when you declare the data ordinal?

Starter code: `viz_2_marksencoding/viz_2_marksencoding_dps.qm`

```python
seattle = vega_data.seattle_weather()
alt.Chart(seattle).mark_bar().encode(
    alt.X('month(date):O', title = "Month"),
    alt.Y('average(precipitation):Q', title = "Average rainfall (in.)")
)
```

Data types: summary

A single data series can have multiple meanings depending on data type

- `'N'`: a *nominal* type (unordered, categorical data),
- `'O'`: *ordinal* type (rank-ordered data),
- `'Q'`: *quantitative* type (numerical data with meaningful magnitudes), and
- `'T'`: *temporal* type (date/time data)

Explicitly specify the data type so that `altair` knows how to encode each variable

Fundamentals of visualization: Encodings

Visual encodings: roadmap

- Introduce types of visual encodings and rank them by their effectiveness
- More on color

Citing our sources

- Schwabish: “Better Data Visualizations” (link to purchase)
- Healy: “Data Visualization” (link to full text)
- Munzner. Visualization Analysis and Design. (slides and video)

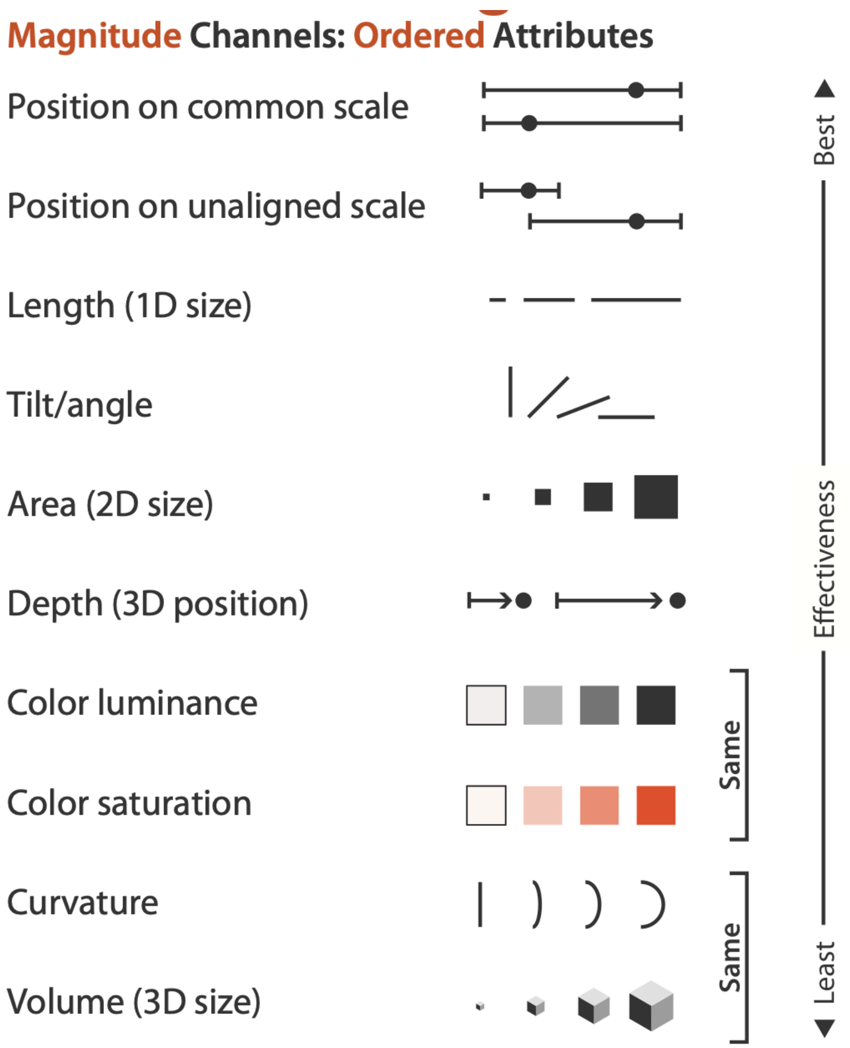
Recall: visualization guidelines

1. All axes and units are properly labeled and legible
2. No words or data points are cut off in your final output
3. **Encodings should be sensible/appropriate** – *what does this mean?*

Visual encodings

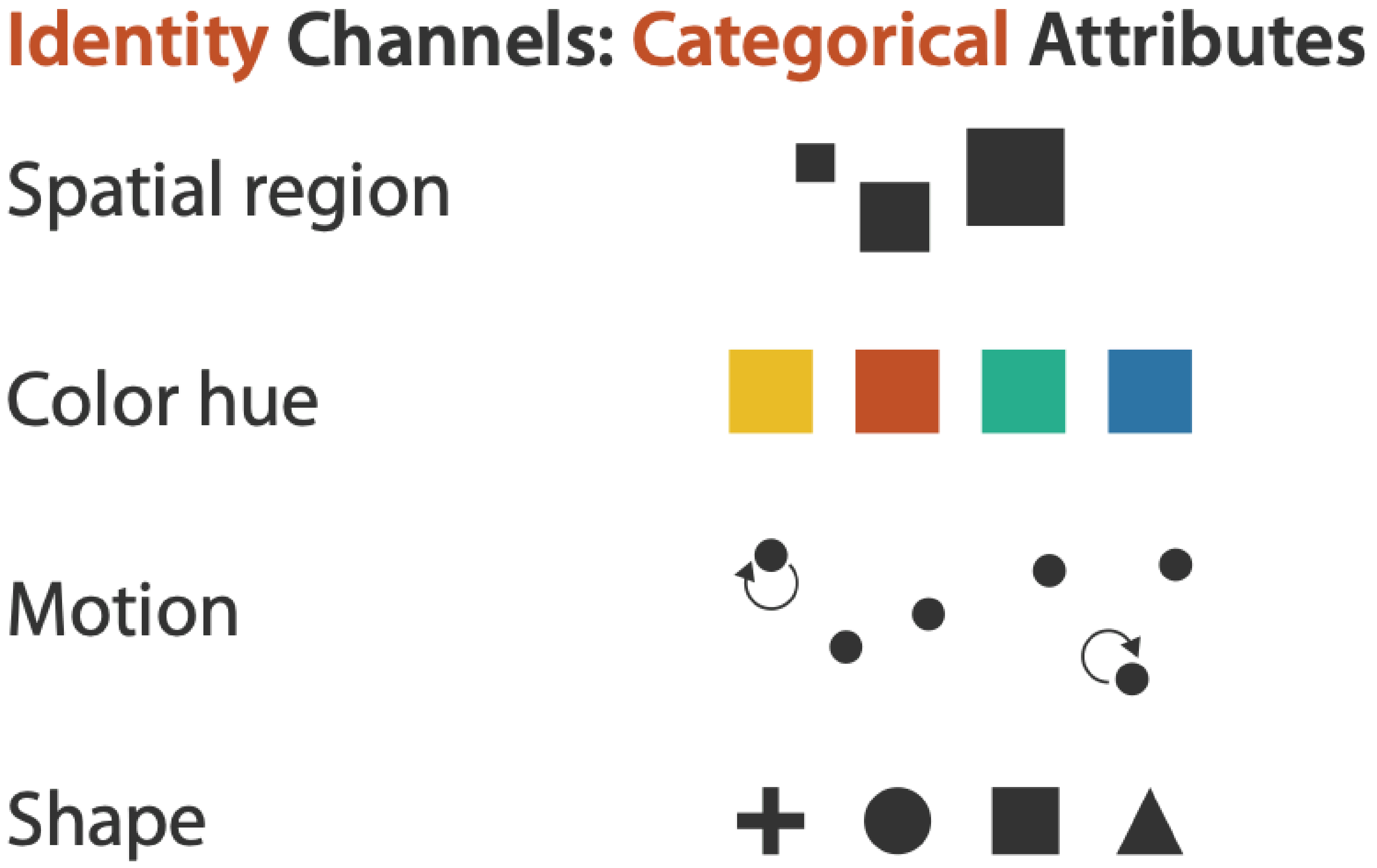
- **Visual encodings** map data variables to visual properties of a chart
- The encoding you choose should be appropriate for the data type and conclusions you want audience to draw
    - Good encodings reveal patterns and makes clear what comparison can be made
    - Bad encodings obfuscate and can be misleading

Encodings: ordered attributes



Source: Munzner (2014), Figure 5.6

Encodings: unordered attributes



Source: Munzner (2014), Figure 5.6

Choosing an encoding

1. What type of data do I have?
    - Nominal, ordinal, quantitative or temporal?
    - Are the variables ordered or unordered?
2. What do I want the viewer to conclude?
    - Are they comparing values? Estimating magnitudes? Spotting patterns?
    - How important is **perceptual accuracy**? E.g., is it enough to know *A* > *B*, or do I need to know *A == 3 x B*?

Encodings by data type: nominal

- **Questions**: *Is value A the same or different than value B? (A = B)*
- **Perceptual accuracy**: viewer should be able to easily differentiate between categories
- **Typical encodings**: position, color hue (blue, red, green), and shape
  - Importantly, encoding should *not* imply a rank-ordering
  - Size/length would be not be appropriate
  - *Position* sometimes implies a rank-ordering when we don't mean it

Encodings by data type: ordinal

- **Questions**: *Does value A come before or after value B? (A < B)*
- **Perceptual accuracy**: viewer should be able to detect rank-ordering
- **Typical encodings**: position, size, and color luminance/saturation (light vs. dark)

Encodings by data type: quantitative

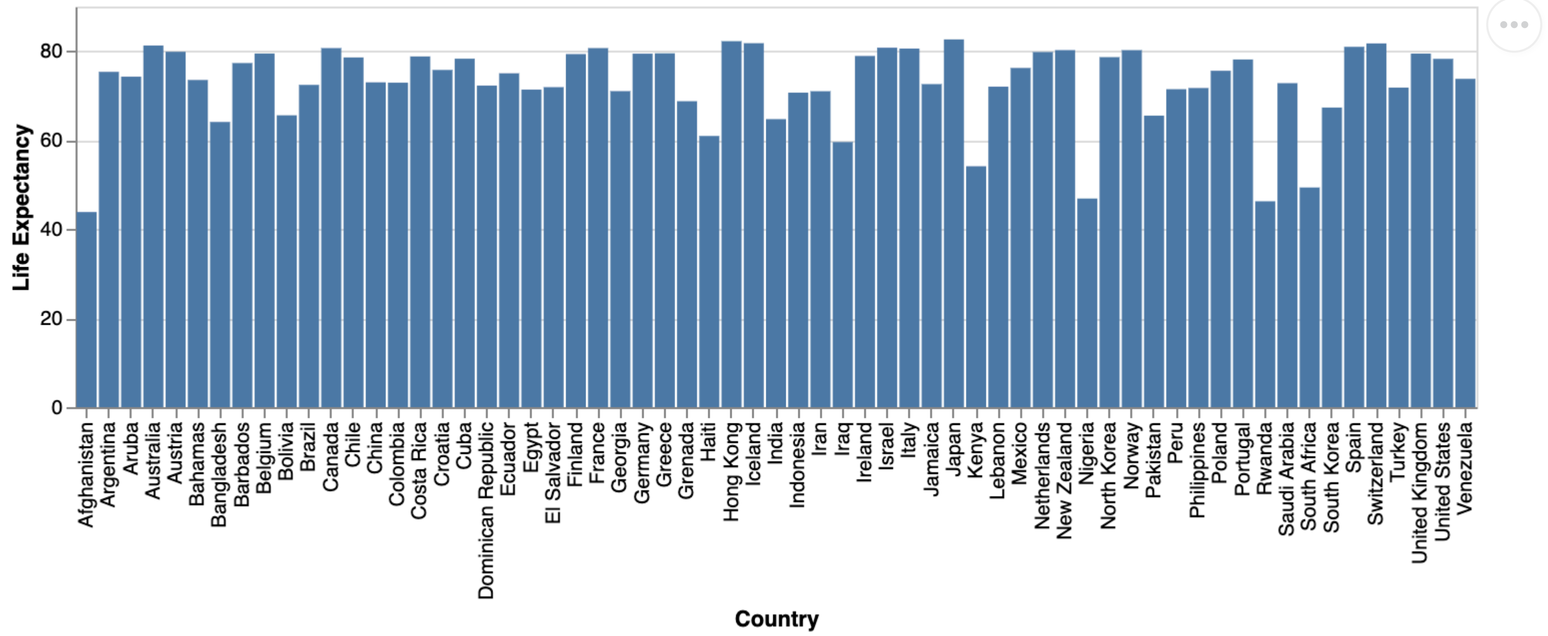
- **Questions**: *What is the distance to value A from value B? (A - B)? value A is what proportion of value B? (A / B)?*
- **Perceptual accuracy**: viewer should be able to detect relative magnitudes
- **Typical encodings**: position, length, size, and color luminance/saturation (light vs. dark)
  - Additionally, scale should go to 0 for ratio data

Can we rely on `altair`'s defaults?

- Even if you don't specify an encoding, `altair` may pick a default one
- Sometimes this is innocuous. E.g., it has to pick a default color to plot graphs in
- But sometimes the default encoding it chooses:
  - Implies order when there isn't one, or vice versa
  - Is a "wasted" opportunity to encode in a more informative way

Example: using `altair`’s defaults

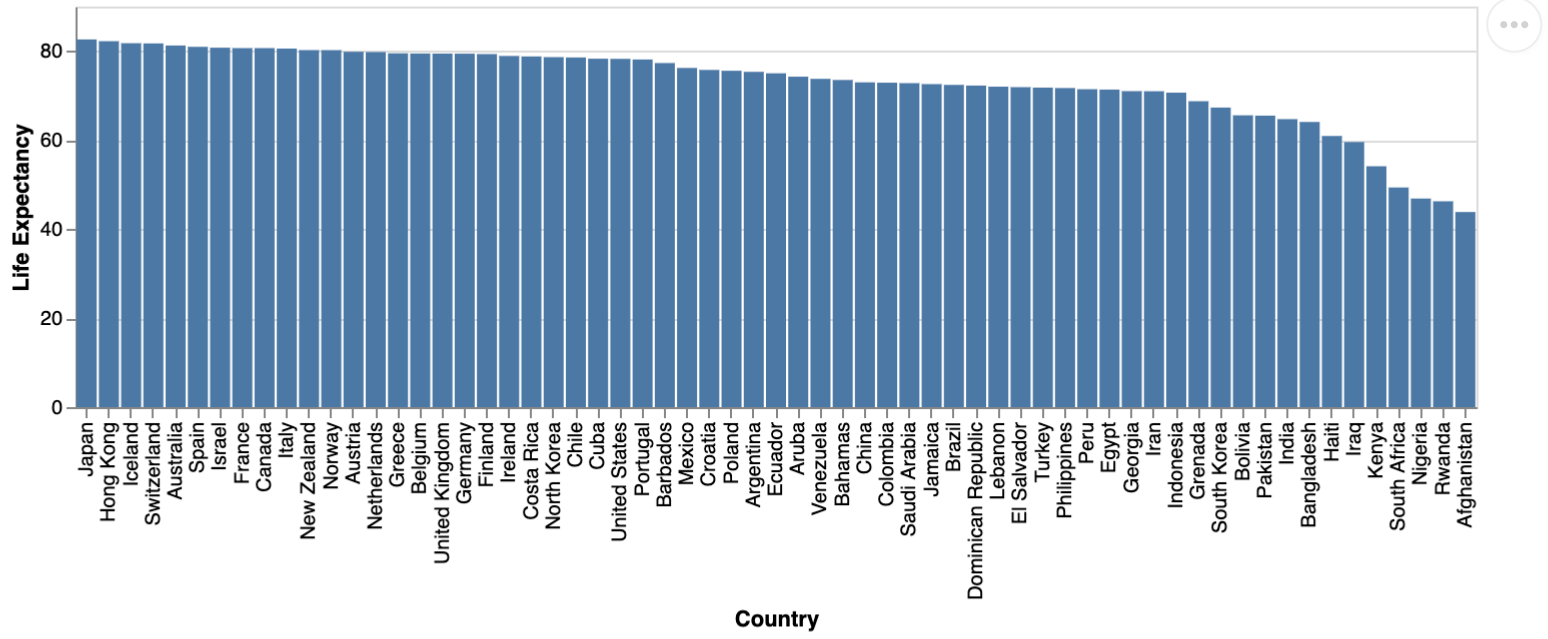
```python
df_latest = data[data['year'] == data['year'].max()]

alt.Chart(df_latest).mark_bar().encode(
    alt.X('country:N', title='Country'),
    alt.Y('life_expect:Q', title='Life Expectancy')
)
```



`altair` encodes positions for `country` *alphabetically*. For what kinds of questions would this be useful vs. not useful?

Example: using `altair`’s defaults

```python
alt.Chart(df_latest).mark_bar().encode(
    x=alt.X('country:N',
            =alt.EncodingSortField(field='life_expect', order='descending'sort),
            title='Country'),
    y=alt.Y('life_expect:Q', title='Life Expectancy')
)
```



To highlight differences or relative life expectancy, sorting by `life_expect` makes the x-axis encoding more useful

More depth on color

By `color`, we mean both **luminance/saturation** (light/dark) and **hue**

Why choose color deliberately?

- Using any software's default color palette is kind of like using comic sans font on a resume
- Choosing the "right" colors will make it easier for you to convey meaning
- Use colorbrewer2.org to choose your color palettes. Click through to site. Options include subsetting to colors that are colorblind safe and black and white printer safe

Color coordination… not just for clothing

- Within a project
  - You rarely produce a single plot in isolation. Usually it's part of an article, a website, etc. Use coordination as a communication tool
  - Use same color for a variable across multiple figures (e.g. green for income, blue for consumption)
  - If you are plotting data for the same groups across multiple figures, might use the same color for each group (e.g. UChicago always maroon, Northwestern as purple)

Color coordination… not just for clothing

- Across projects
  - Many organizations have official palettes and plot templates. UChicago’s is here. Good to ask if you are working for a big org if they have one.

Color palettes and their use cases

Altair has many pre-set color schemes:

Palette type	Use case
Categorical	`Nominal`
Sequential Single-Hue	`Ordinal` or `Quantitative`
Sequential Multi-Hue	Higher contrast, but harder to judge quantitative proximity
Diverging	Use if there is a midpoint (e.g. voting for `redblue`)

Visual encoding: summary

- Several ways to encode information visually
- How you encode should be informed by
  - Data type
  - If you want to convey order/ranking
  - What questions/answers you want plot to deliver
- Color is one of the easiest ways to convey meaning

Encoding channels in `altair`

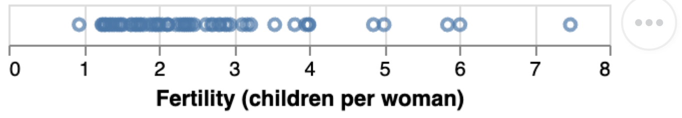
Encoding channels: roadmap

- `x`, `y`
  - Aside: whether to include 0
- `size`
- `color`
- `opacity`
- `shape`
- `column`, `row`

Throughout, we will highlight examples of “bad” uses of encodings and marks.

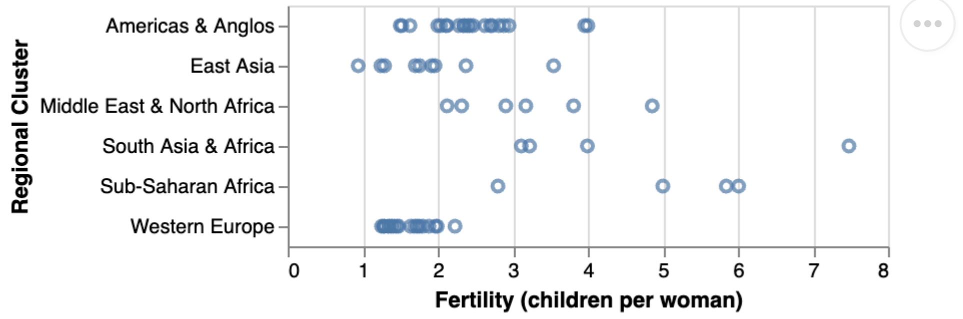
X

```python
data2000 = data.loc[data['year'] == 2000] #one year is more manageable

alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)")
)
```



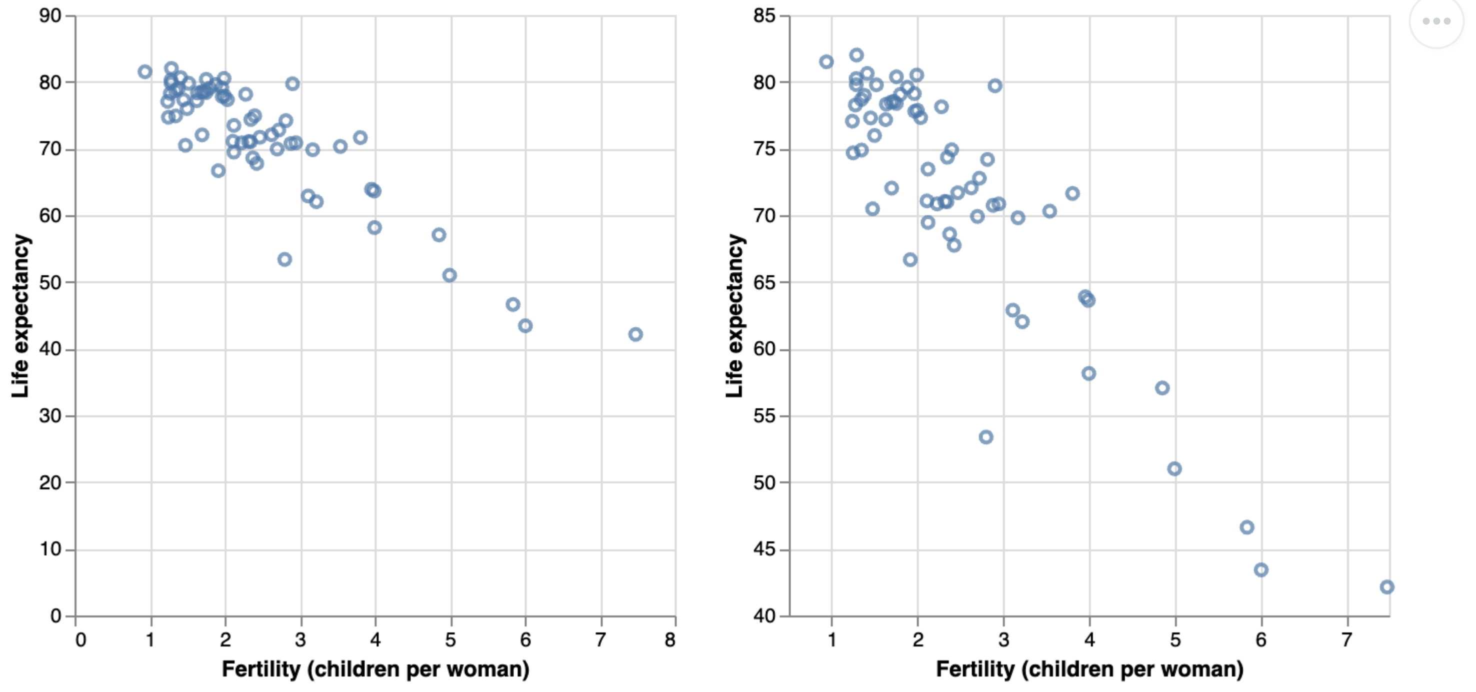
X + Y

```python
alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('cluster_name:N', title = "Regional Cluster")
)
```



Requiring 0 on axis range vs. not

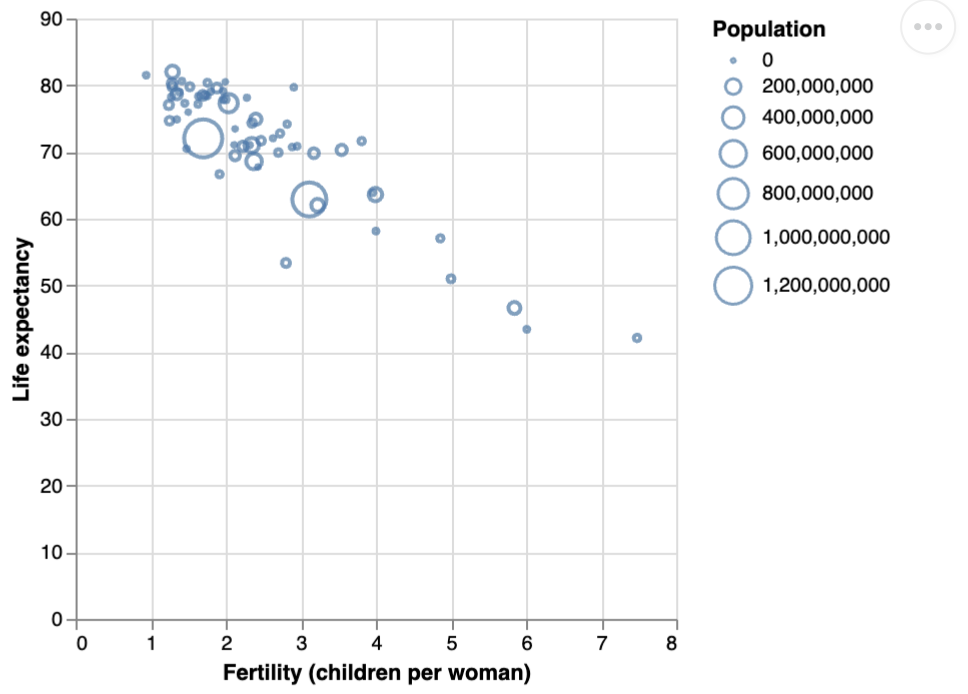
```
default_with_zero = alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q',title = "Life expectancy")
)
zero_excluded = alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q',  scale=alt.Scale(zero=False), title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q', scale=alt.Scale(zero=False), title = "Life expectancy")
)
default_with_zero | zero_excluded
```



Discussion question: which plot do you prefer (and why?)

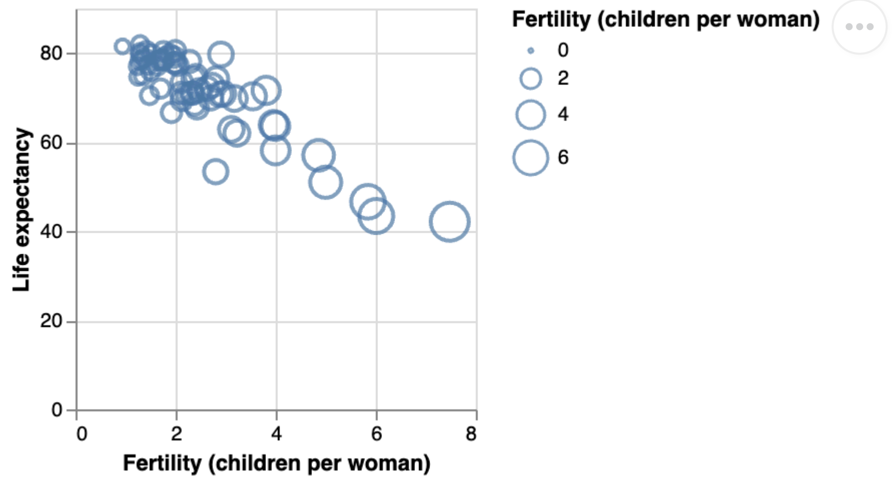
size

```python
alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q',title = "Life expectancy"),
    alt.Size('pop:Q', title = "Population")
)
```



Bad use of `size`

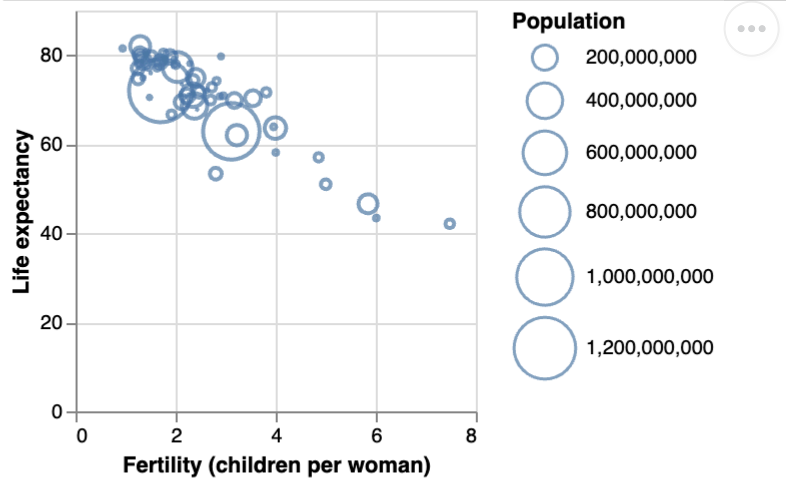
```python
alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q',title = "Life expectancy"),
    alt.Size('fertility:Q', title = "Fertility (children per woman)")
)
```



“Bad” use of encodings: redundant encodings for `fertility`: X and `size`. `altair`’s grammar of graphics makes this very obvious.

`size` with 1000 pixels for largest dot

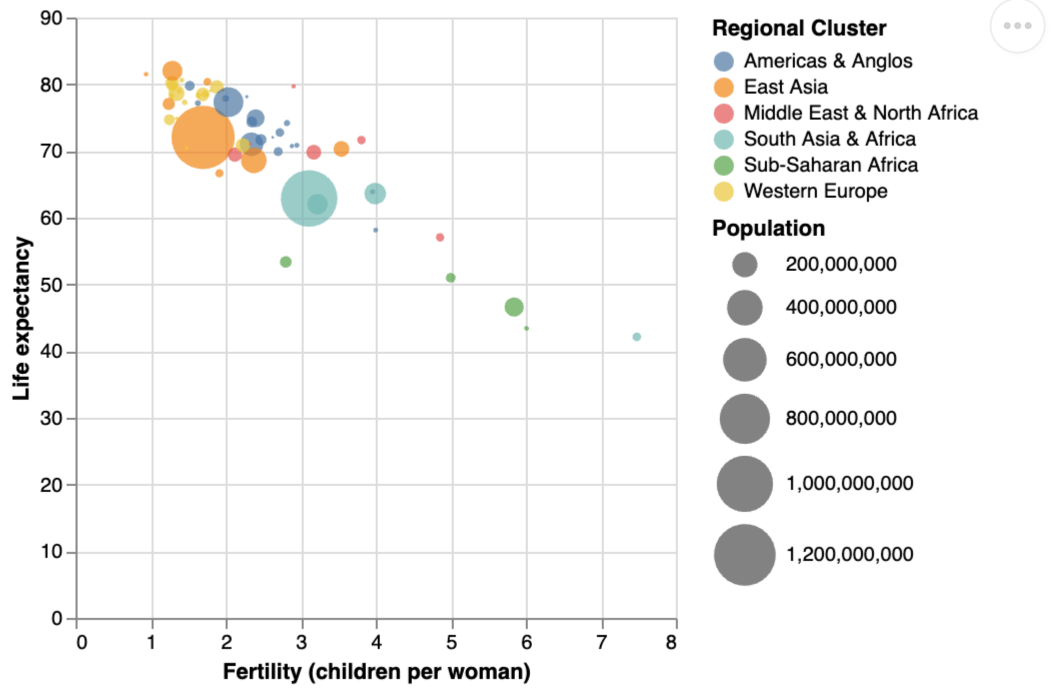
```
alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q',title = "Life expectancy"),
    alt.Size('pop:Q', scale=alt.Scale(range=[0,1000]))
)
```



Note: `alt.Scale(range=[0,1000])` indicates the *visual* size of the marks (in pixels), and is not in reference to values in the underlying data

add `color`

```python
alt.Chart(data2000).mark_point(filled=True).encode(
   alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q',title = "Life expectancy"),
    alt.Size('pop:Q', scale=alt.Scale(range=[0,1000]), title = "Population"
    alt.Color('cluster_name:N', title = "Regional Cluster")
)
```



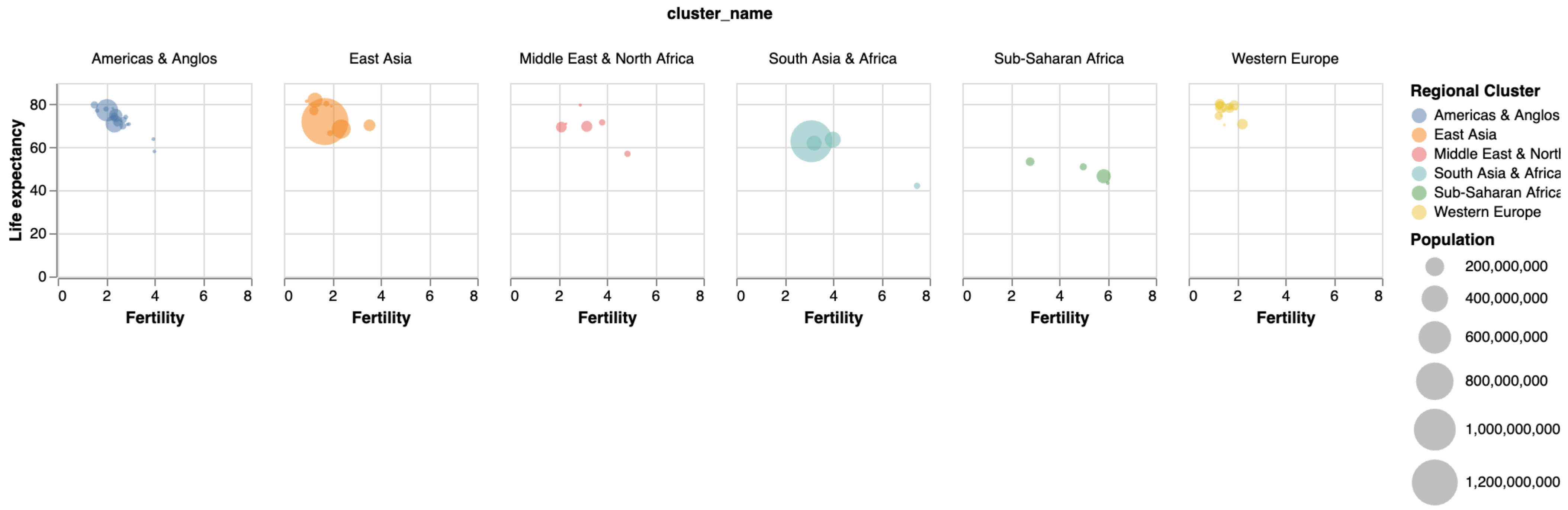
opacity

```
alt.Chart(data2000).mark_point(filled=True).encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q', title = "Life expectancy"),
    alt.Size('pop:Q', scale=alt.Scale(range=[0,1000]), title = "Population"
    alt.Color('cluster_name:N', title = "Regional Cluster"),
    alt.OpacityValue(0.2)
)
```

Question: are we encoding anything here?

column

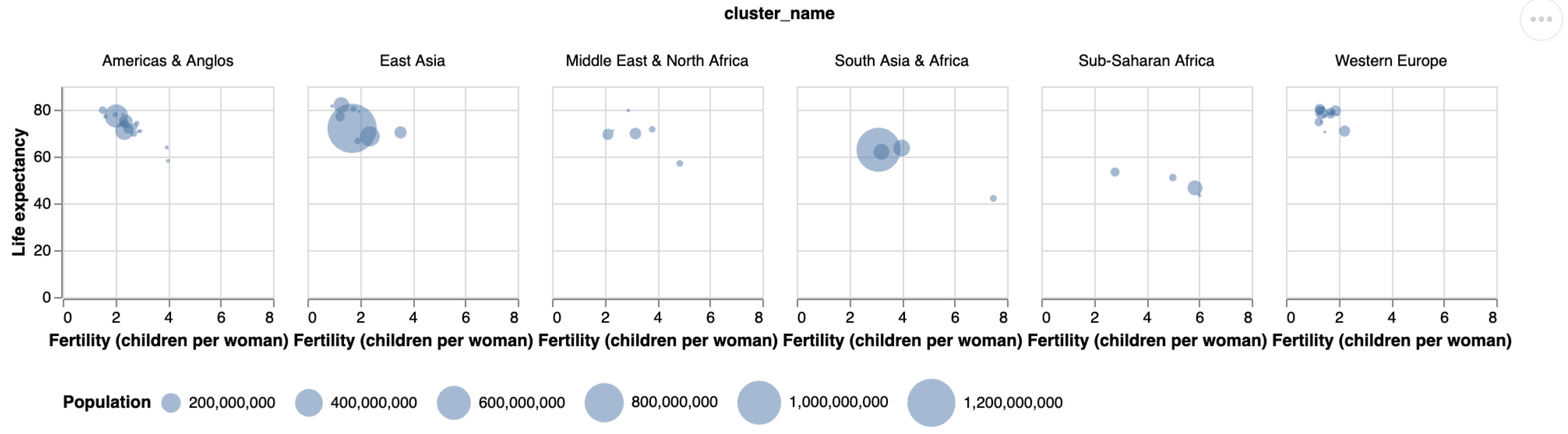
```
alt.Chart(data2000).mark_point(filled=True).encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q', title = "Life expectancy"),
    alt.Size('pop:Q', scale=alt.Scale(range=[0,1000]), title = "Population")
    alt.Color('cluster_name:N', title = "Regional Cluster"),
    alt.OpacityValue(0.5),
    alt.Column('cluster_name:N')
)
```



“Bad” use of encodings: now the `Color` and `Column` encodings are redundant. It’s pretty, but could be confusing!

Cleaning up the graph

```python
alt.Chart(data2000).mark_point(filled=True).encode(
    alt.X('fertility:Q', title = "Fertility"),
    alt.Y('life_expect:Q', title = "Life expectancy"),
    alt.Size('pop:Q', scale=alt.Scale(range=[0,1000]),
             legend=alt.Legend(orient='bottom', titleOrient='left'),
             title = "Population"),
    alt.OpacityValue(0.5),
    alt.Column('cluster_name:N'))
```



We can clean it up by eliminating superfluous encodings and moving legend

Encoding channels: summary

- `x`: Horizontal (x-axis) position of the mark.
- `y`: Vertical (y-axis) position of the mark.
- `size`: Size of the mark. May correspond to area or length, depending on the mark type.
- `color`: Mark color, specified as a legal CSS color.
- `opacity`: Mark opacity, ranging from 0 (fully transparent) to 1 (fully opaque).
- `shape`: Plotting symbol shape for `point` marks.
- `column`: Facet the data into horizontally-aligned subplots.
- `row`: Facet the data into vertically-aligned subplots.

Graphical marks in `altair`

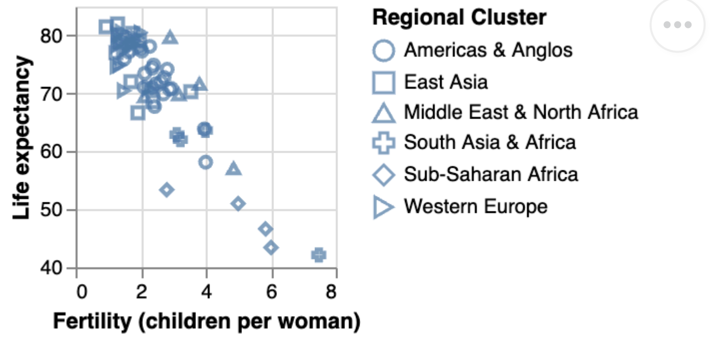
Graphical marks: roadmap

Prior section used only `mark_point()`. Now will cover

- `mark_point()`
  - `mark_circle()`
  - `mark_tick()`
- `mark_bar()`
- `mark_line()`
- `mark_area()`

`mark_point()`: add information using `alt.Shape()`

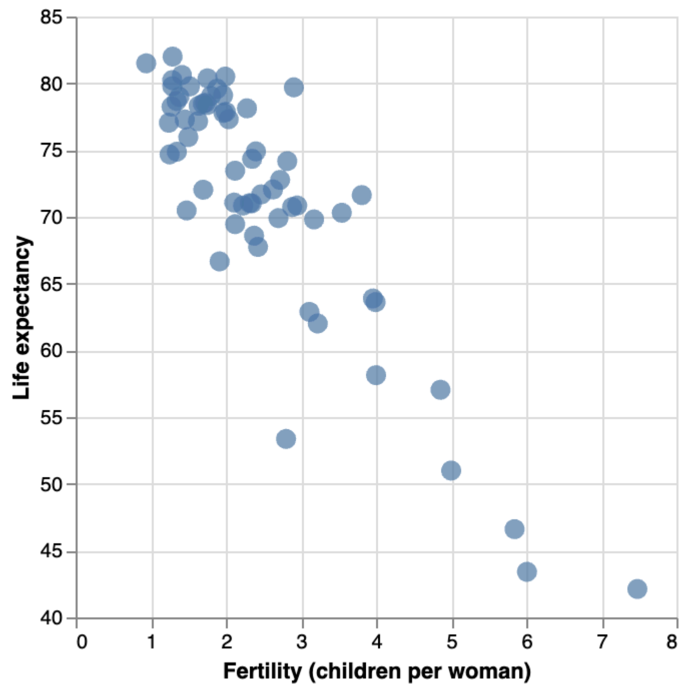
```
alt.Chart(data2000).mark_point().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q', scale=alt.Scale(zero=False), title = "Life expec
    alt.Shape('cluster_name:N', title = "Regional Cluster")
)
```



Discussion: thoughts on how well `alt.Shape` communicates `cluster_name`?

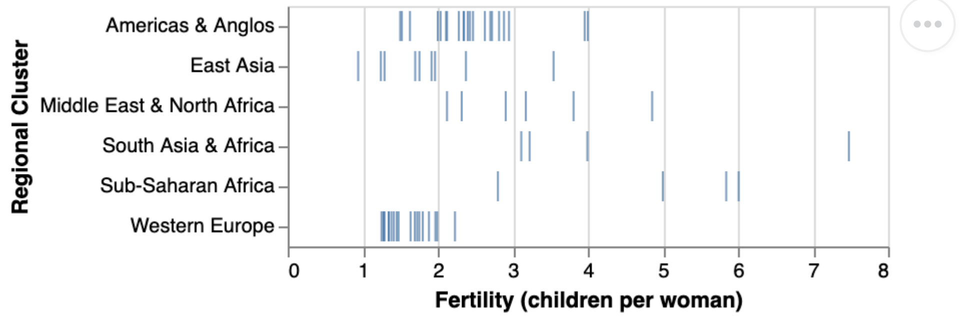
`mark_circle()` wrapper for `mark_point(filled=True)`

```python
alt.Chart(data2000).mark_circle(size=100).encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('life_expect:Q', scale=alt.Scale(zero=False), title = "Life expec
)
```



mark_tick()

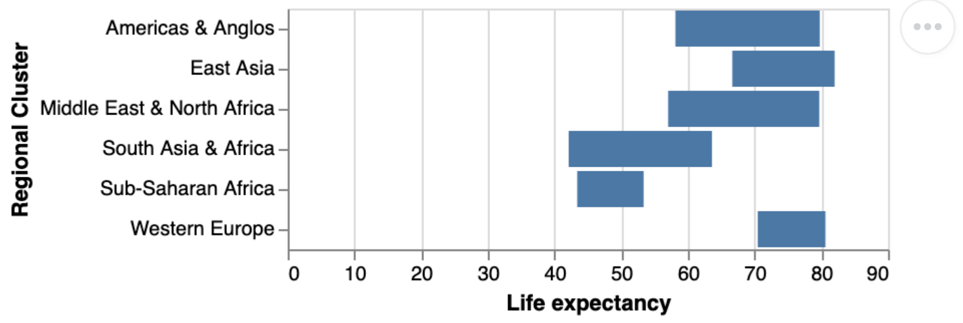
```python
alt.Chart(data2000).mark_tick().encode(
    alt.X('fertility:Q', title = "Fertility (children per woman)"),
    alt.Y('cluster_name:N', title = "Regional Cluster")
)
```



Useful for comparing values along a single dimension with minimal overlap.

X and X2

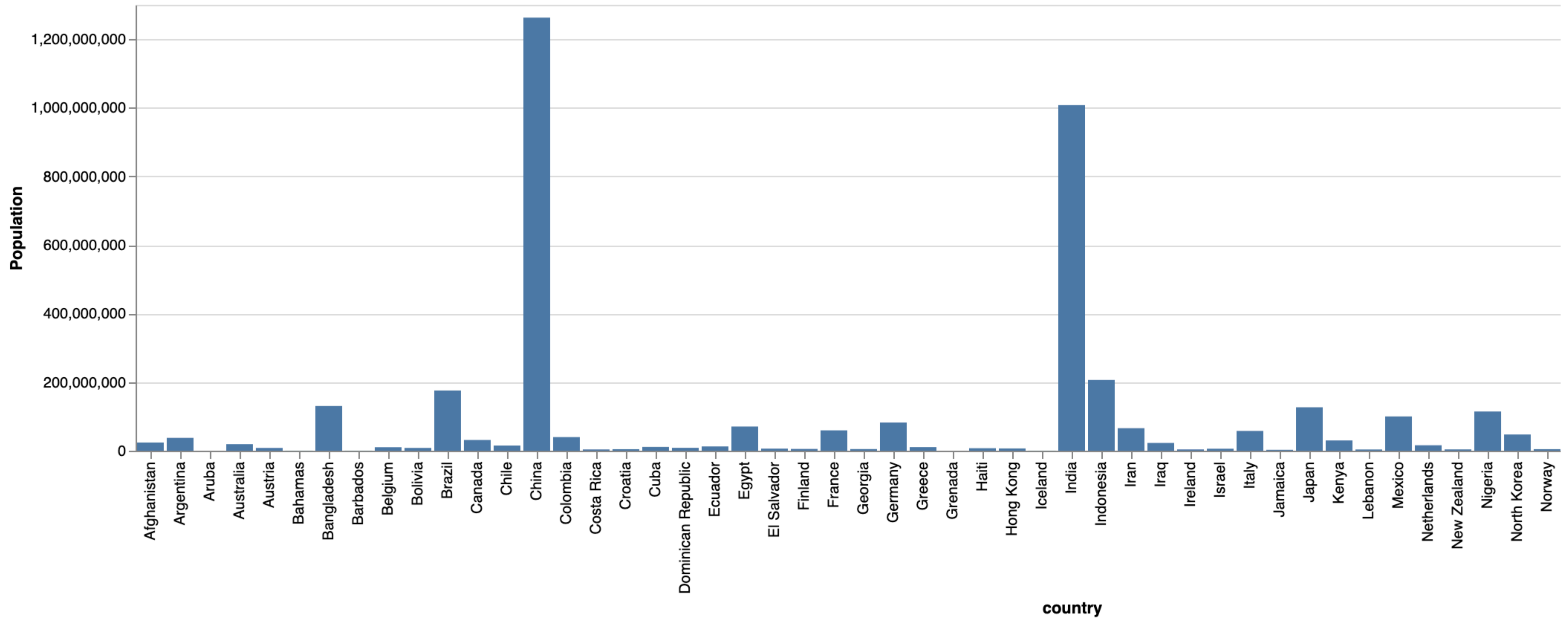
```python
alt.Chart(data2000).mark_bar().encode(
    alt.X('min(life_expect):Q', title = "Life expectancy"),
    alt.X2('max(life_expect):Q'),
    alt.Y('cluster_name:N', title = 'Regional Cluster')
)
```



A *dot plot* drawn with tick marks is sometimes referred to as a *strip plot*.

mark_bar()

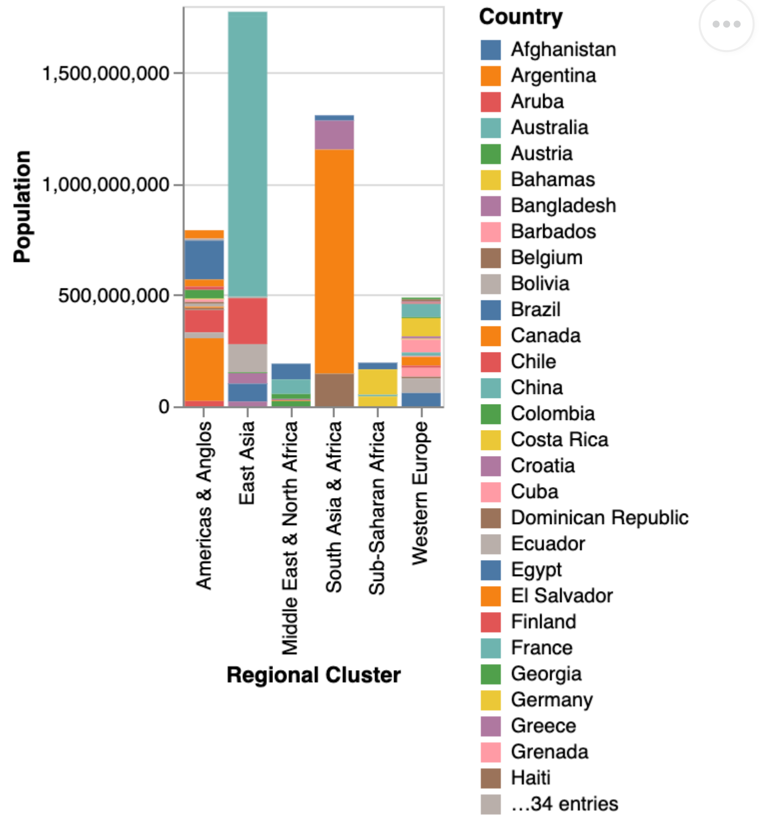
```python
alt.Chart(data2000).mark_bar().encode(
    alt.X('country:N', title = "Country"),
    alt.Y('pop:Q', title = "Population")
)
```



"Bad" use of encoding: here is an instance of a "wasted" opportunity to encode something useful on the x-axis

alt.Color() for a stacked bar plot

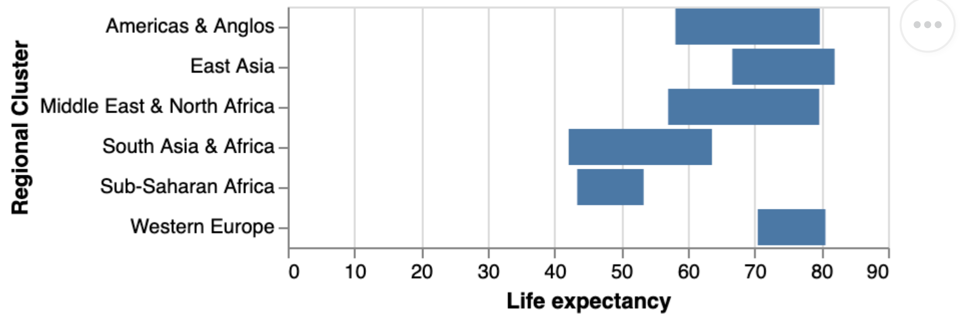
```python
alt.Chart(data2000).mark_bar().encode(
    alt.X('cluster_name:N', title = "Regional Cluster"),
    alt.Y('pop:Q', title = "Population"),
    alt.Color('country:N', title = "Country")
)
```



"Bad" use of color – way too many categories! Requires reader to move back and forth between graph and legend to parse.

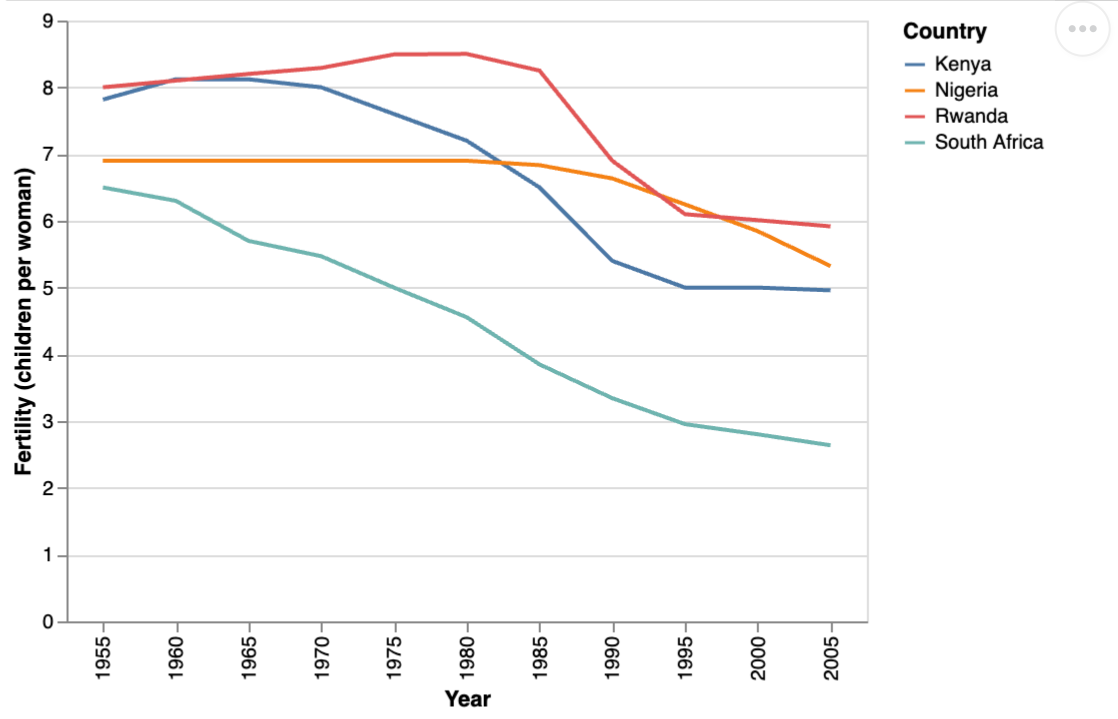
X2( ) to show intervals

```python
alt.Chart(data2000).mark_bar().encode(
    alt.X('min(life_expect):Q', title = "Life expectancy"),
    alt.X2('max(life_expect):Q'),
    alt.Y('cluster_name:N', title = "Regional Cluster")
)
```



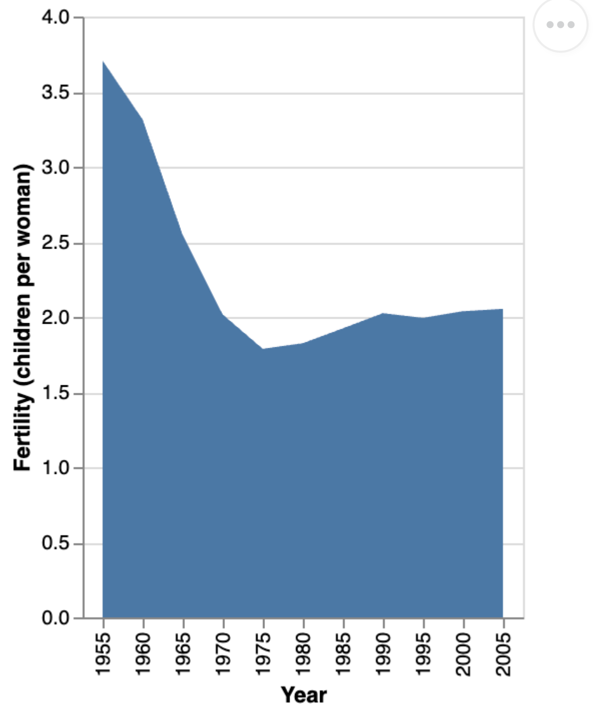
mark_line()

```python
data_cluster2 = data.loc[data['cluster'] == 2] #one cluster is more managea
alt.Chart(data_cluster2).mark_line().encode(
    alt.X('year:O', title = "Year"),
    alt.Y('fertility:Q', title = "Fertility (children per woman)"),
    alt.Color('country:N', title = "Country")
)
```



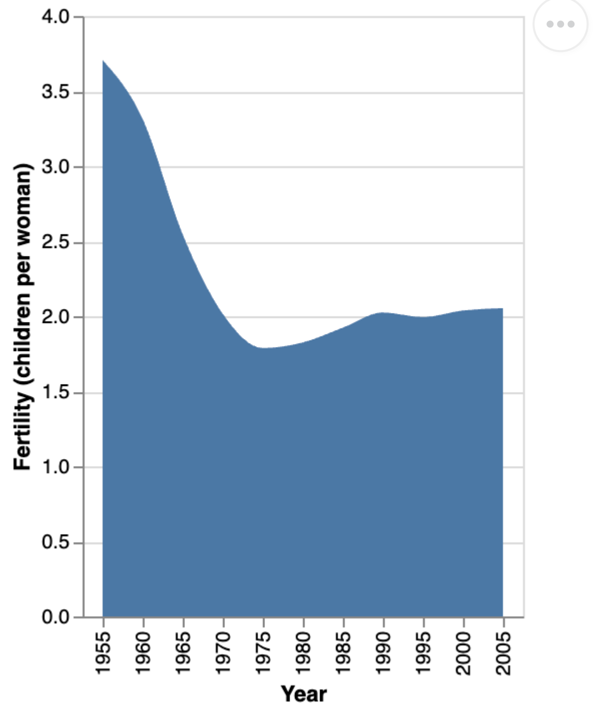
mark_area()

```python
dataUS = data.loc[data['country'] == 'United States']
alt.Chart(dataUS).mark_area().encode(
    alt.X('year:O', title = "Year"),
    alt.Y('fertility:Q', title = "Fertility (children per woman)")
)
```



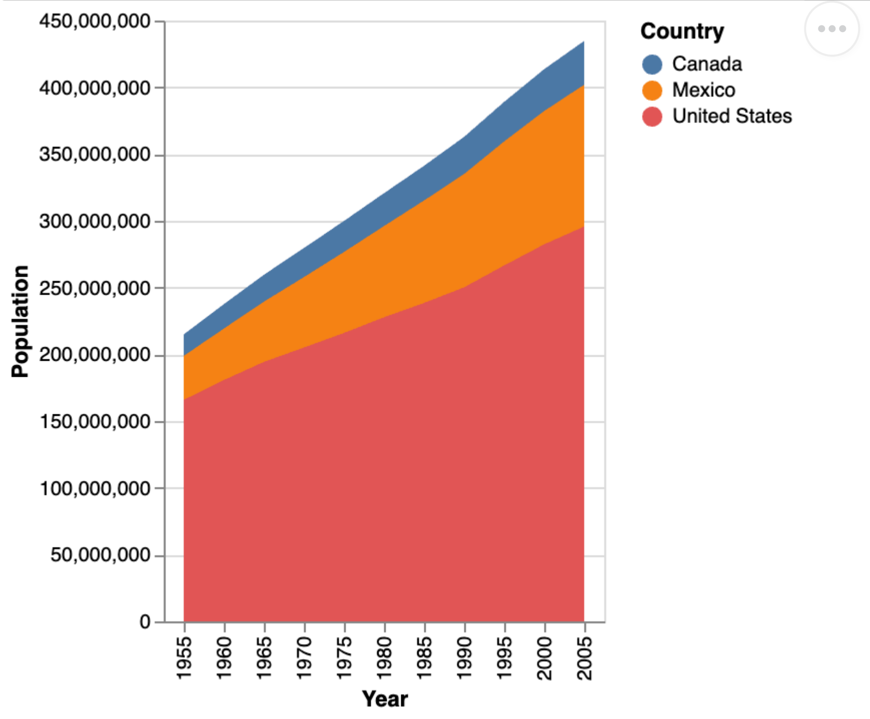
+ interpolate='monotone'

```
alt.Chart(dataUS).mark_area(interpolate='monotone').encode(
    alt.X('year:O', title = "Year"),
    alt.Y('fertility:Q', title = "Fertility (children per woman)")
)
```



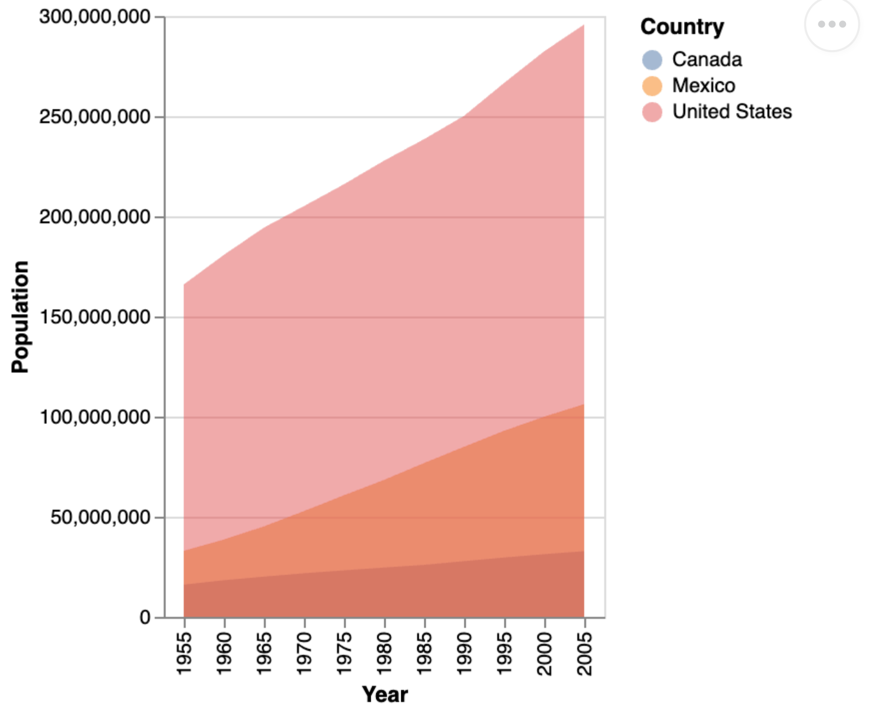
mark_area() with stacking

```python
dataNA = data[data['country'].isin(['United States', 'Mexico', 'Canada'])]
alt.Chart(dataNA).mark_area().encode(
    alt.X('year:O', title = "Year"),
    alt.Y('pop:Q', title = "Population"),
    alt.Color('country:N', title = "Country")
)
```



`mark_area()` with no stacking and opacity

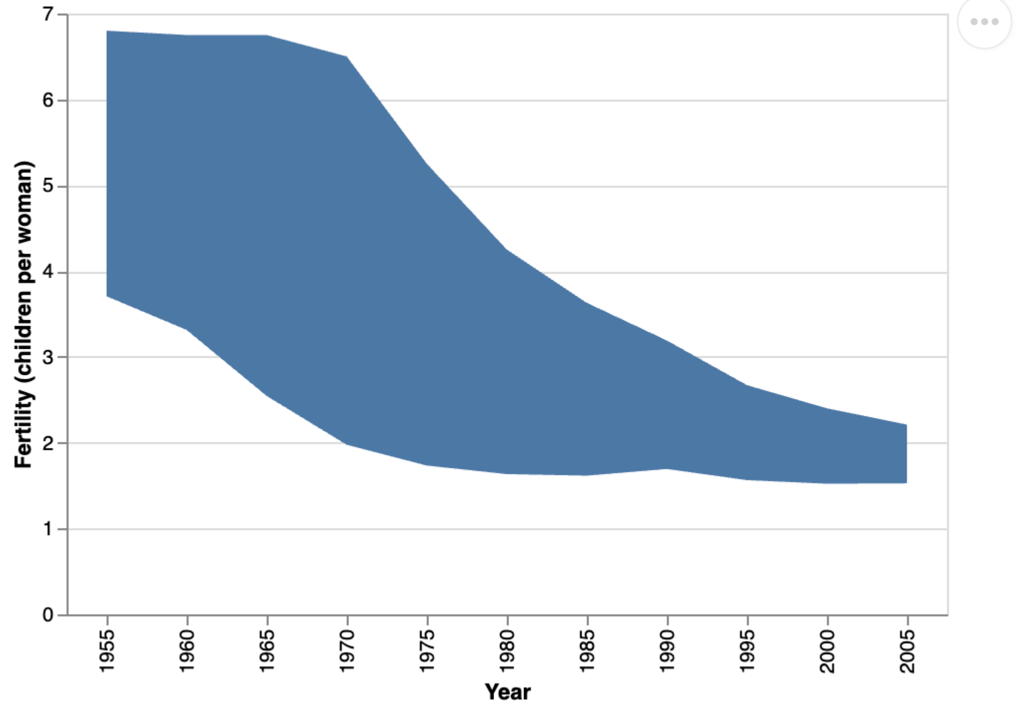
```python
alt.Chart(dataNA).mark_area(opacity=0.5).encode(
    alt.X('year:O', title = "Year"),
    alt.Y('pop:Q', stack=None, title = "Population"),
    alt.Color('country:N', title = "Country")
)
```



Discussion: is this a good use of color?

`mark_area()` + `Y2` to show range

```python
alt.Chart(dataNA).mark_area().encode(
    alt.X('year:O', title = "Year"),
    alt.Y('min(fertility):Q', title = "Fertility (children per woman)"),
    alt.Y2('max(fertility):Q')
)
```



Conclusion: over time, both the overall fertility values and the variability have declined.

Graphical marks: summary

- `mark_point()` - Scatter plot points with configurable shapes.
    - `mark_circle()` - Scatter plot points as filled circles.
    - `mark_tick()` - Vertical or horizontal tick marks.
- `mark_bar()` - Rectangular bars.
- `mark_line()` - Connected line segments.
- `mark_area()` - Filled areas defined by a top-line and a baseline.

“Bad” marks & encoding practices to avoid

- Redundant encodings
- “Wasted opportunities” to encode useful information
- Encodings that require a lot of mental effort for audience
  - Audience has to look back at legend frequently
  - Or keep a lot in their working memory (e.g., shapes)

Labels

Roadmap

- Overarching principle: **minimize audience's mental effort**
- 2 rules of thumb

Why Labels matter

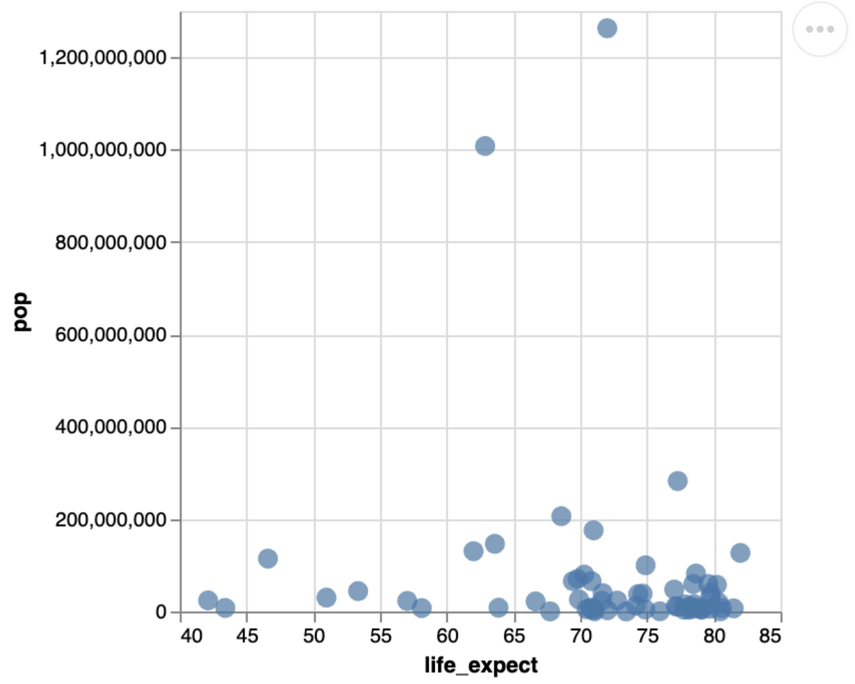
- Marks and encodings tell us *how* data vary
- The text tells us *what* varies
- **Well-chosen text reduces the amount of thinking the reader has to do.**
  - “What does this stand for?”
  - “What is the scale?”

Recall: visualization guidelines

1. **All axes and units are properly labeled and legible**
2. No words or data points are cut off in your final output
3. Encodings should be sensible/appropriate

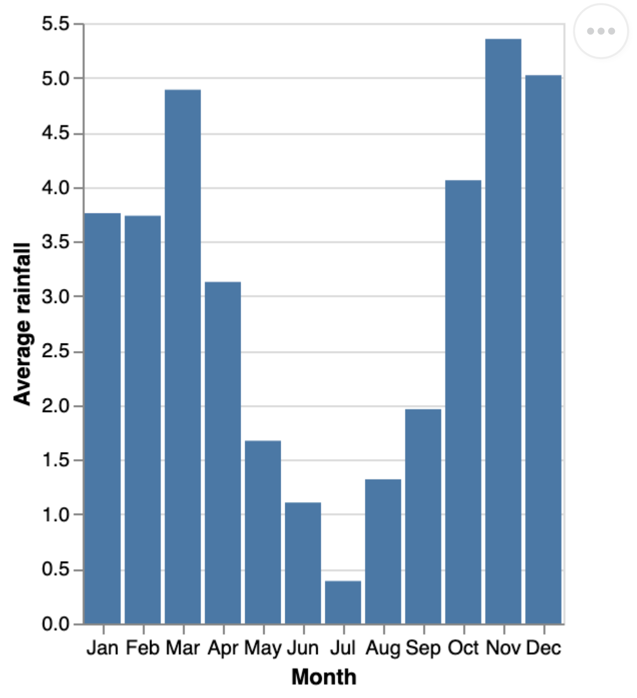
Suggestion 1A: label every axis

- No dataset variable names!
- Looks unprofessional and also slows reader down
- Here, they have to pause and decode what `pop` and `life_expect` stand for



Suggestion 1B: include units where appropriate

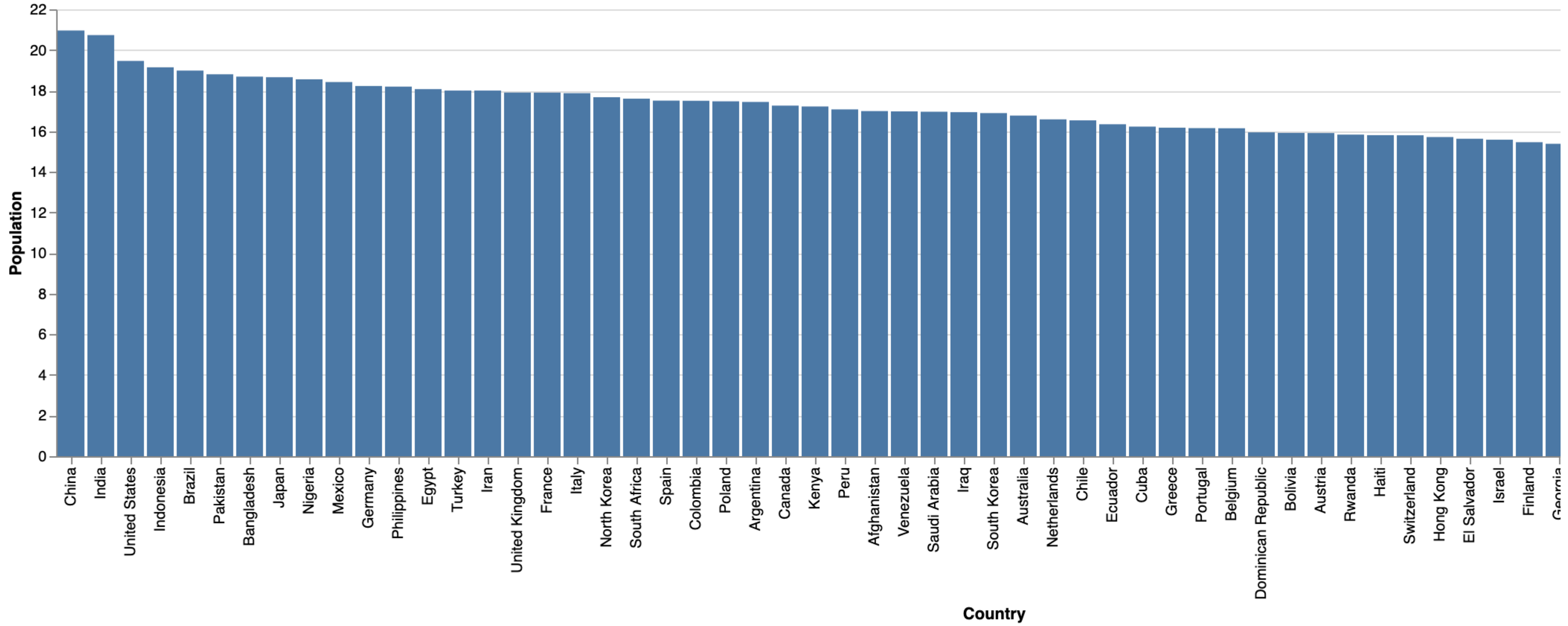
- Numbers without units are ambiguous
- “Average rainfall” is meaningless without unit (inches)



- Caveat: for some common measures, units are unnecessary – no need for “Life expectancy (years)”

Suggestion 1C: label your scale

- Never hide any transformations or scaling
- Here we are plotting log of population, but you wouldn’t know that from the label



- Percent (%) is also a scale you should make note of

Marks and Encodings: Summary

- Building blocks of data visualization are:
  - Data type
  - Encodings
- Also important: labels can make or break a graph
- No “right” way to visualize something – it depends on your audience/message
  - But there are a series of “bad” practices to avoid
  - Key idea: **minimize your reader’s mental effort**