Despliegue : Datos Tabulares

Python Fundamentals

PENSAMIENTO COMPUTACIONAL Y
PROGRAMACIÓN
COLEGIO LA GIROUETTE

\begin{document}

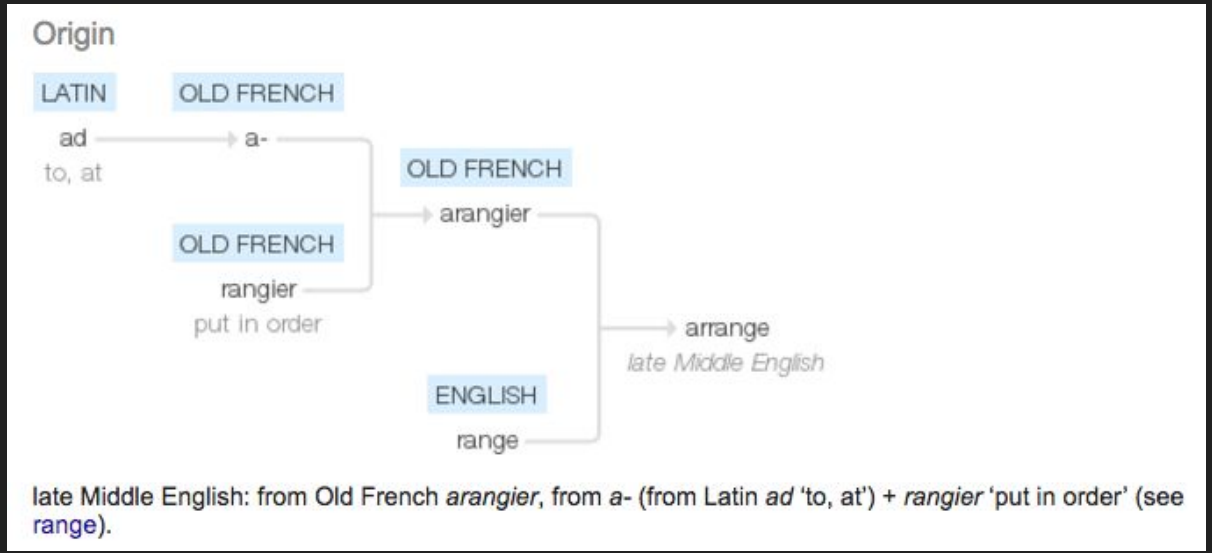
Darío Creado Figueroa

Fuente : Karim Pichara. / IIC2026

Clase de hoy: Arrange tables

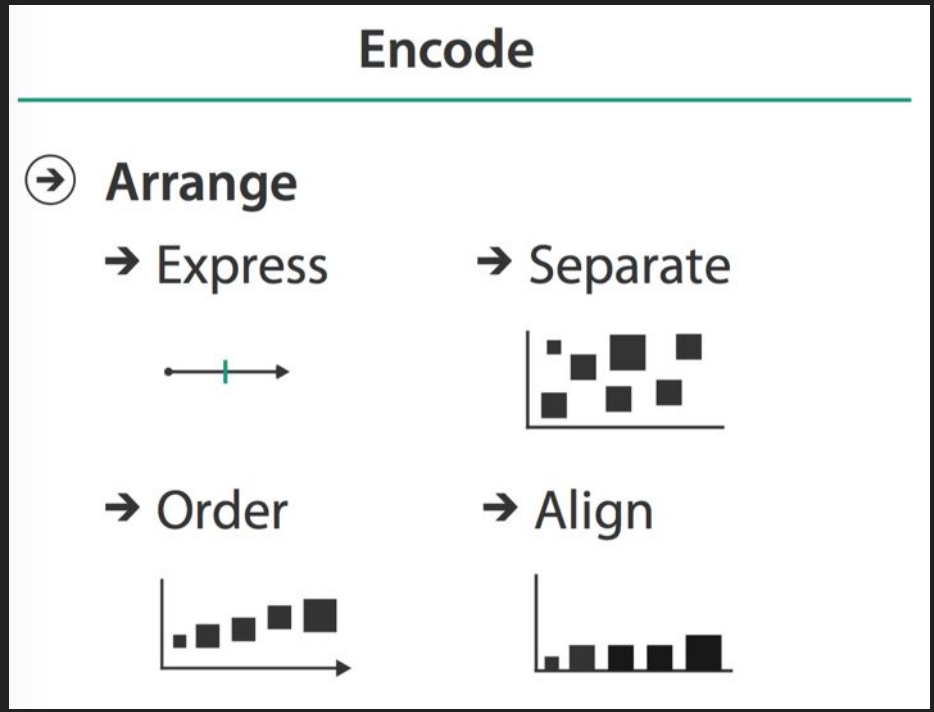
Arrange:

1. Organizar
2. Ordenar
3. Arreglar
4. Disponer
5. Concertar



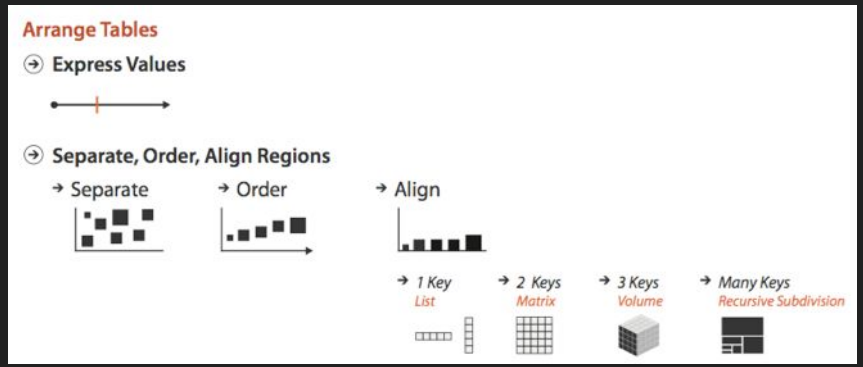
How: Encoding

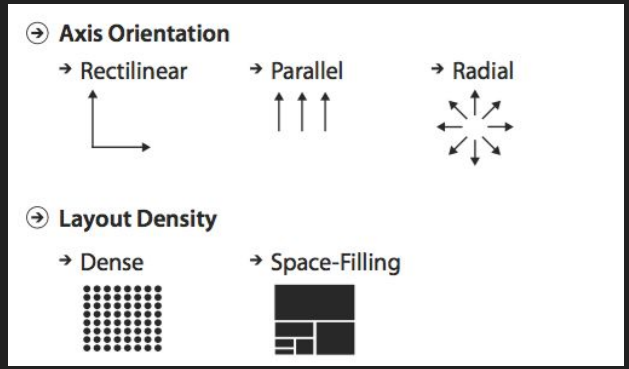
- Algunas clases atrás habíamos visto de forma general “How”: Cómo diseñar idioms de visualización.
- Una de las familias de codificación visual era Arrange.



Datos tabulares (*arrange table*)

- En esta clase, veremos los *encodings* para el diseño de depliegue de tablas.
- La idea es analizar los aspectos de los canales espaciales, que son los más importantes, ya que el uso del espacio domina el modelo mental que tiene el usuario acerca del *dataset*.





Datos tabulares (*arrange table*)

- Los tres canales más efectivos son:
  - posición en una escala en común
  - posición en una escala no alineada
  - largo
- Sin embargo, también hay canales no-espaciales efectivos.

Datos tabulares (*arrange table*)

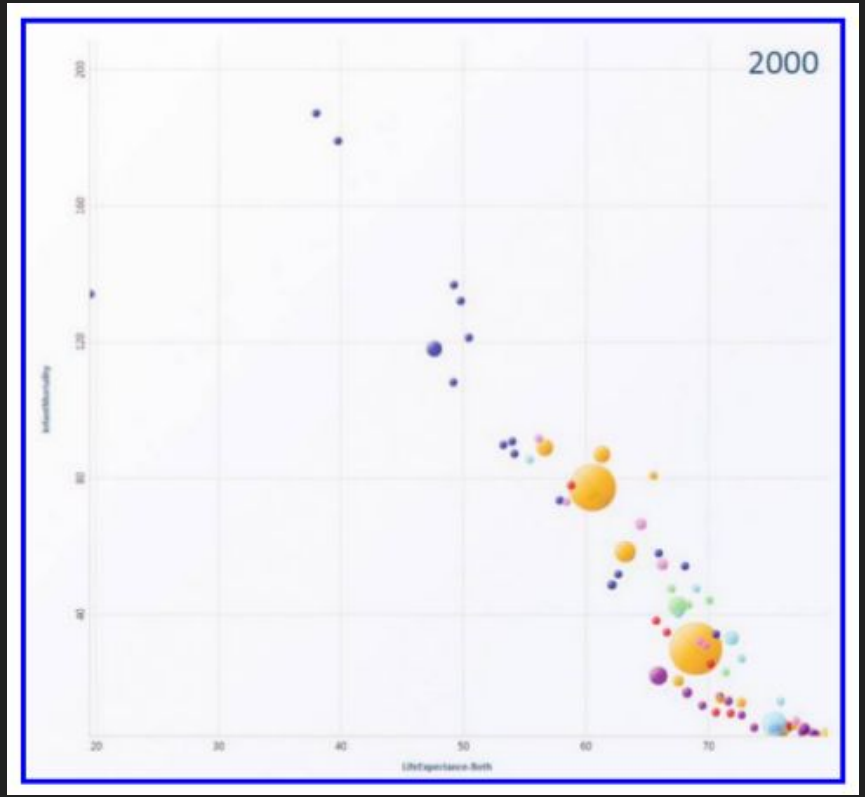
Este *framework* expone cuatro decisiones de diseño que debemos tomar al momento de mostrar nuestros datos tabulares.
Las decisiones que veremos son:

1. Uno es cómo expresar los valores
2. Otros tres son: separar, ordenar y alinear regiones
3. La orientación de los ejes puede ser: rectilínea, paralela o radial
4. El *layout* espacial puede ser denso o también podría ser *space-filling*.

1) Expresar los valores

- En el caso del *encoding* de un único atributo, cada ítem tendrá una marca en alguna posición a lo largo del eje.
- Los atributos adicionales pueden ser expresados en la misma marca usando canales no-espaciales, como el tamaño o el color.
- En un caso más complejo un objeto GLYPH puede ser usado, con una estructura interna que emerge a partir del uso de múltiples canales. Volveremos más adelante en el curso a hablar de los glyphs.

1) Expresar los valores (*scatterplot*)



1) Expresar los valores (*scatterplot*)

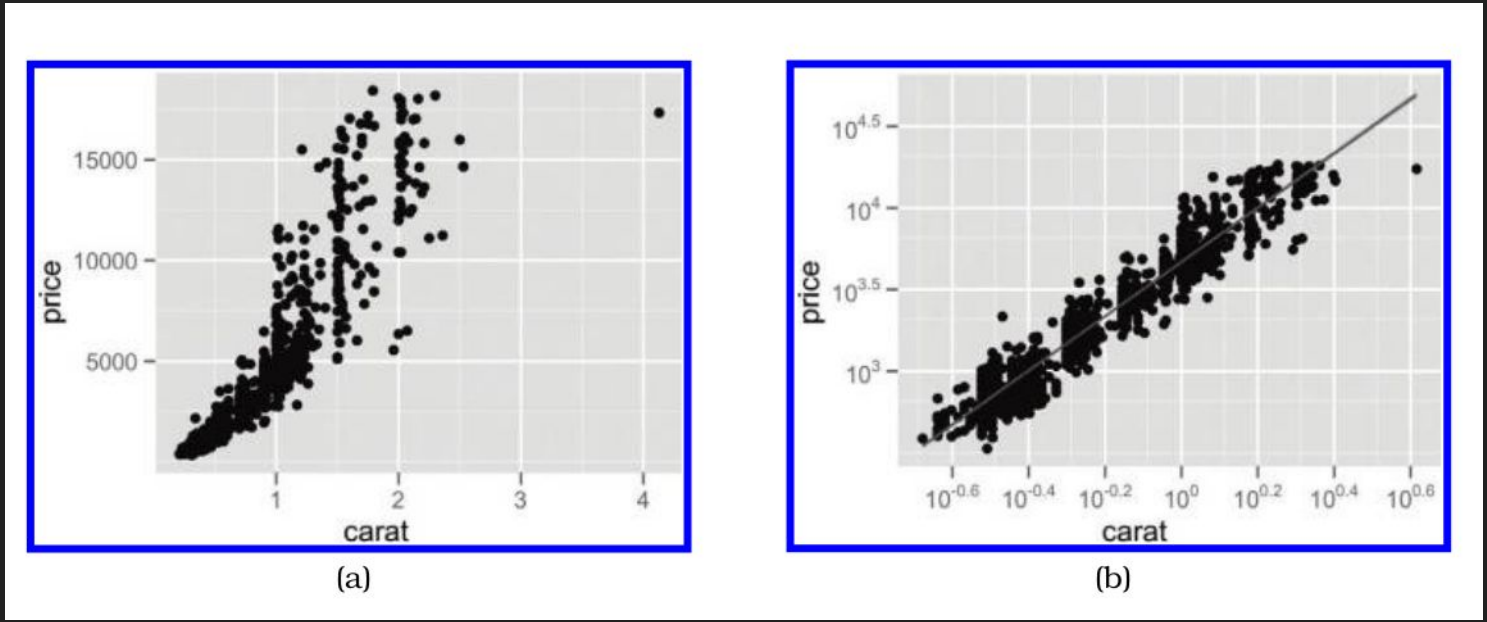
- Como vimos, los *scatterplots* logran expresar, mediante la marca de un punto, dos variables cuantitativas usando las posiciones espaciales, verticales y horizontales.
- Los *scatterplots* son efectivos para los *tasks* de entregar *overviews*, caracterizar distribuciones y específicamente para encontrar *outliers* y valores extremos.

1) Expresar los valores (*scatterplot*) II

- Además, nos permiten, con alta efectividad, juzgar la **correlación** entre dos atributos, ya que este *task* corresponde a percibir si es que los puntos forman una línea en la diagonal.
    - Una correlación **positiva** es cuando la diagonal tiene una pendiente *hacia arriba*.
    - Una correlación **negativa** es cuando la diagonal tiene una pendiente *hacia abajo*.

1) Expresar los valores (*scatterplot*)

- Es posible realizar transformaciones que nos entreguen más información sobre los datos.



1) Expresar los valores (*scatterplot*)

- La figura (a) muestra la relación entre el precio de los diamantes y su peso.
- La figura (b) muestra la relación de estos mismos atributos, luego de haberles aplicado una transformación logarítmica a las escalas.
- Podemos notar fácilmente que los atributos transformados están fuertemente correlacionados, de forma positiva.
- Además, cuando la tarea (*task)* principal es exponer la correlación de las variables, generalmente se superpone una línea que muestra la regresión calculada.

1) Expresar los valores (*scatterplot*)

- Como mencionamos anteriormente, los *scatterplots* pueden ser *aumentados* usando colores para mostrar un atributo adicional. O bien, también se les puede agregar tamaño a las marcas para exponer otro atributo: este tipo de *idiom* se conoce generalmente como ***bubble chart*** o ***bubble plot***.

- La escalabilidad de un *scatterplot* está limitada por la necesidad de distinguir entre los puntos. Por lo tanto, un *scatterplot* debería ser capaz de aceptar hasta una centena de ítems.

1) Expresar los valores (*scatterplot*)

Idiom	**Scatterplots**
What: Data	Table: two quantitative value attributes.
How: Encode	Express values with horizontal and vertical spatial position and point marks.
Why: Task	Find trends, outliers, distribution, correlation; locate clusters.
Scale	Items: hundreds.

2) Separar, ordenar y alinear: Regiones Categóricas

- El uso del espacio para exponer atributos categóricos es más complejo que en el caso de los atributos cuantitativos.
- Dado que la posición es un canal que expresa magnitudes ordenadas, los atributos categóricos no tienen una semántica de orden; por lo tanto, el principio de expresividad podría ser violado si son expuestos con una posición en el espacio.

2) Separar, ordenar y alinear

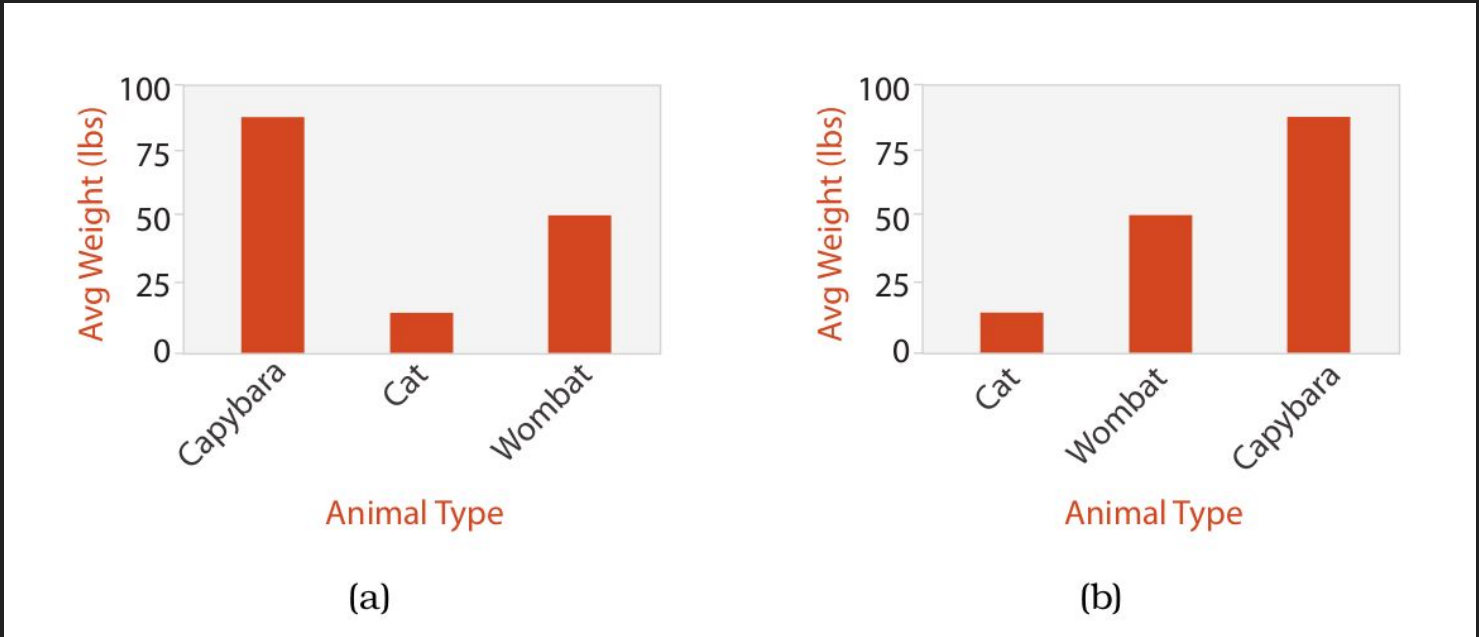
- Sin embargo, la semántica de los atributos categóricos encaja bien con la idea de **regiones** espaciales.
- Estas regiones son áreas contiguas y limitadas, que son distintas entre sí.
- Al dibujar todos los ítems con los mismos valores de un atributo categórico en una misma región, se utiliza la proximidad espacial para exponer información acerca de su similaridad.
- Veamos cómo este problema se vuelve más fácil al descomponerlo en tres operaciones: **separar**, **ordenar** y **alinear** las diferentes regiones.

2) Separar, ordenar y alinear

- La separación y el orden **siempre** deben ocurrir, no así con el alineamiento que es opcional.
- La separación debería ocurrir por un atributo categórico, mientras que los otros dos (i.e. ordenar y alinear) debe hacerse por un atributo que tenga inherentemente algún tipo de orden.

2) Separar, ordenar y alinear (*bar chart*)

Veamos un ejemplo, con una única *key* (i.e. una categoría).



2) Separar, ordenar y alinear (*bar chart*)

- Este conocido *idiom*, llamado diagrama de barras, utilizando una marca de línea, con el atributo cuantitativo expuesto como una posición espacial.
- Además, el otro atributo es categórico (el único *key*) utiliza una región **separada** por cada elemento —en el ejemplo, animales.
- Además, las marcas están **alineadas** en un marco común para resaltar las diferencias del tamaño de las líneas.
- Finalmente, en (a) cada región está **ordenada** alfabéticamente por el nombre de las especies. Por otra parte, en (b) está **ordenado** por el tamaño de las barras.

2) Separar, ordenar y alinear (*bar chart*)

Idiom	**Bar Charts**
What: Data	Table: one quantitative value attribute, one categorical key attribute.
How: Encode	Line marks, express value attribute with aligned vertical position, separate key attribute with horizontal position.
Why: Task	Lookup and compare values.
Scale	Key attribute: dozens to hundreds of levels.

Stacked Bar Chart

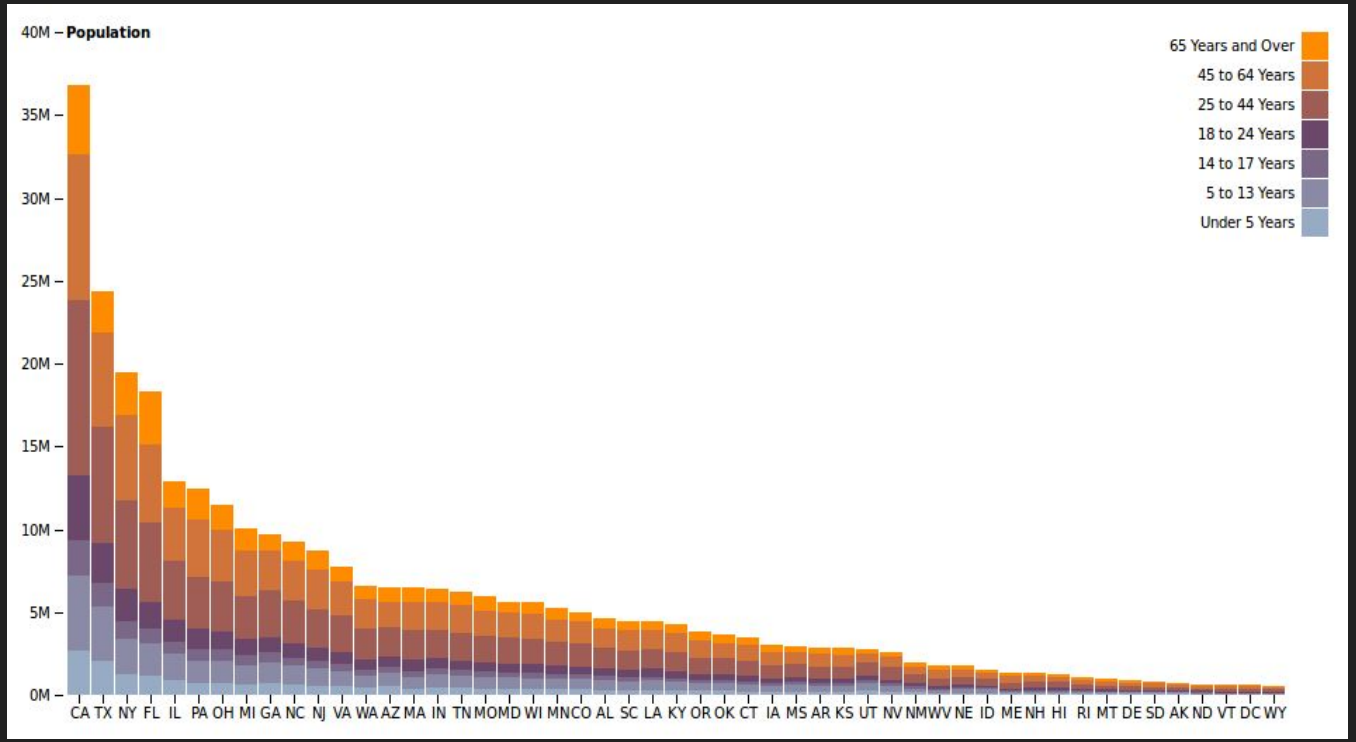
2) Separar, ordenar y alinear (*stacked bar chart*)

- Los *stacked bar charts* son similares a los *bar charts*, pero usan un *glifo* más complejo para cada barra, donde múltiples sub-barras son apiladas verticalmente.
- El largo del glifo compuesto todavía codifica un valor (como un *bar chart*), pero también lo hace cada subcomponente.
- Los *stacked bar charts* ofrecen información sobre tablas multidimensionales, específicamente una tabla bidimensional con **dos** llaves.
- Los glifos compuestos están ordenados por la llave primaria (e.g. ciudad), mientras que la llave secundaria (e.g. segmento etario) sirve para construir la estructura vertical de cada glifo.

2) Separar, ordenar y alinear (*stacked bar chart*)

- Ese *idiom* generalmente también utiliza el color, además del tamaño, para cada glifo. De esta forma, cada subcomponente se colorea según la misma llave que está determinada por el ordenamiento vertical.
- Este color no agrega ninguna información adicional, ya que el orden vertical ya muestra a qué llave secundaria pertenece. Sin embargo, comparar subcomponentes entre diferentes glifos sería mucho más difícil sin colores.
- La escalabilidad de este *idiom* es similar al de los diagramas de barras, en término del número de llaves primarias, pero sí está más limitada con las llaves secundarias, que no deberían ser más de una docena.

2) Separar, ordenar y alinear (*stacked bar chart*)



2) Separar, ordenar y alinear (*stacked bar chart*)

Idiom	**Stacked Bar Charts**
What: Data	Multidimensional table: one quantitative value attribute, two categorical key attributes.
How: Encode	Bar glyph with length-coded subcomponents of value attribute for each category of secondary key attribute. Separate bars by category of primary key attribute.
Why: Task	Part-to-whole relationship, lookup values, find trends.
Scale	Key attribute (main axis): dozens to hundreds of levels. Key attribute (stacked glyph axis): several to one dozen

Streamgraph

2) Separar, ordenar y alinear (*streamgraph*)

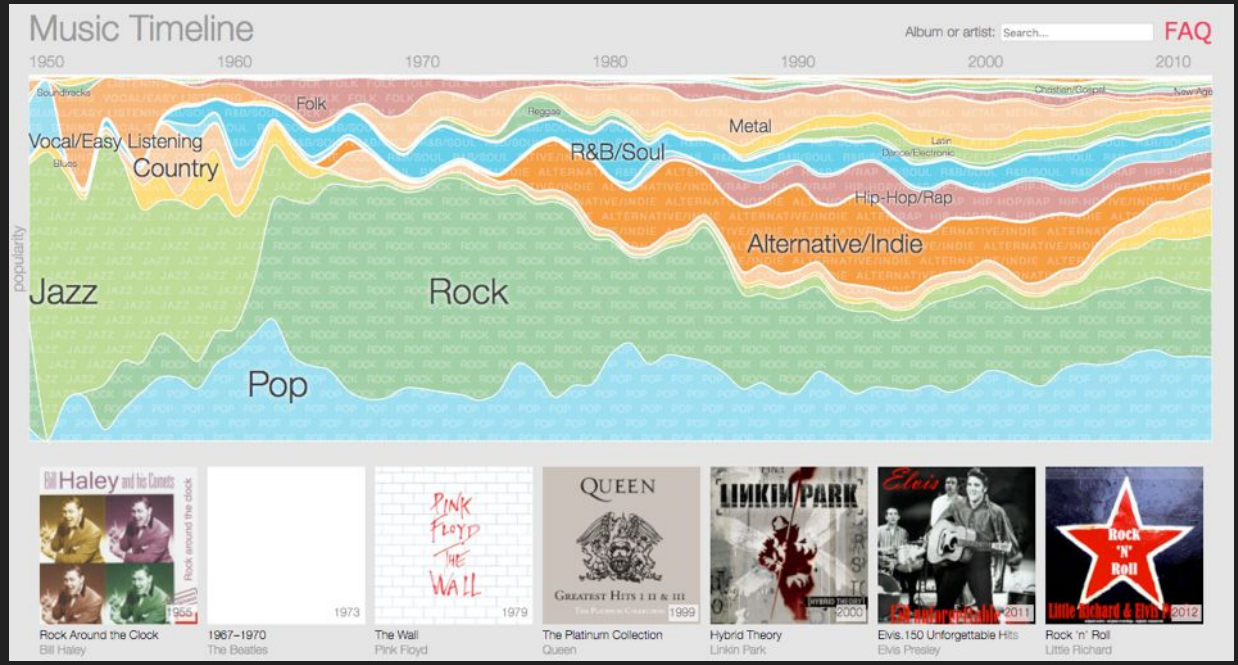
- Los *streamgraphs* (literalmente, gráfico de corriente, torrente o flujo) ofrecen una geometría que hace énfasis en la **continuidad de los datos**.
- Por otra parte, si usáramos un *stacked bar chart*, veríamos glifos verticales que darían énfasis sobre la popularidad en **tiempos específicos**.
- En el ejemplo, cada capa del *streamgraph* es un artista y su altura representa su popularidad en términos de la cantidad de veces que su música fue escuchada semanalmente.
- Es importante notar que tanto los *stacked bar charts* como los *streamgraphs* pueden ser orientados tanto horizontalmente como verticalmente.

2) Separar, ordenar y alinear (*streamgraph*)

- Los *streamgraphs* suelen usarse para representar una serie de tiempo, simplificando el *task* del usuario de rastrear de ítemes individuales a través del tiempo en un flujo continuo de datos.
- Este tipo de idiom es generalmente fácil de entender, ya que es algo familiar y reduce el esfuerzo cognitivo para interpretar la visualización.
- De esta forma, se privilegia la **legibilidad** de cada torrente con un *silueta orgánica*, dejando de lado el *baseline* que ofrece generalmente el eje horizontal.

2) Separar, ordenar y alinear (*streamgraph*)

https://research.google.com/bigpicture/music/



2) Separar, ordenar y alinear (*streamgraph*)

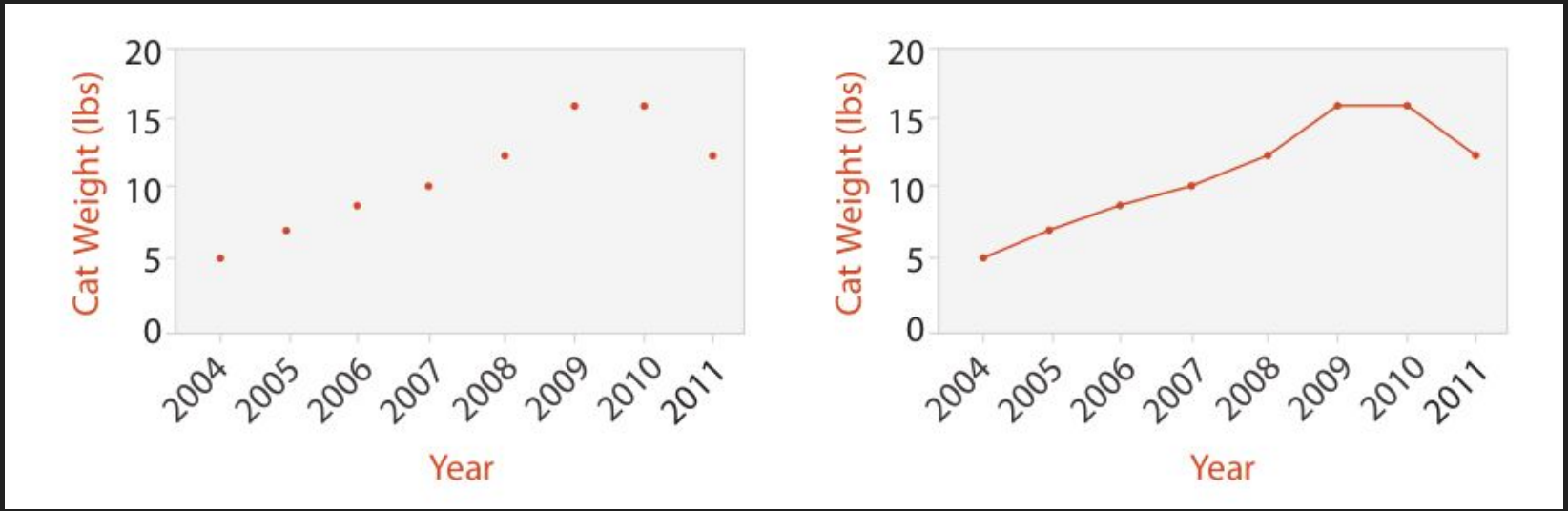
Idiom	**Streamgraphs**
What: Data	Multidimensional table: one quantitative value attribute (counts), one ordered key attribute (time), one categorical key attribute (artist).
What: Derived	One quantitative attribute (for layer ordering).
How: Encode	Use derived geometry showing artist layers across time, layer height encodes counts.
Scale	Key attributes (time, main axis): hundreds of time points. Key attributes (artists, short axis): dozens to hundreds

Dot chart & Line chart

2) Separar, ordenar y alinear (*dot chart & line chart*)

- Este *idiom* es un *visual encoding* de un atributo cuantitativo usando la posición espacial en frente a un atributo categórico.
- Es similar a un *bar chart*, en donde el atributo cuantitativo es codificado por puntos en vez de líneas. O también puede ser visto como un *scatterplot* en donde uno de los ejes muestra un atributo categórico.
- Por otra parte, un *line chart* aumenta un *dot chart*, al conectar los puntos con líneas. Estas líneas sirven para darle énfasis a una cierta **tendencia** que queramos mostrar desde los datos.

2) Separar, ordenar y alinear (ejemplos)



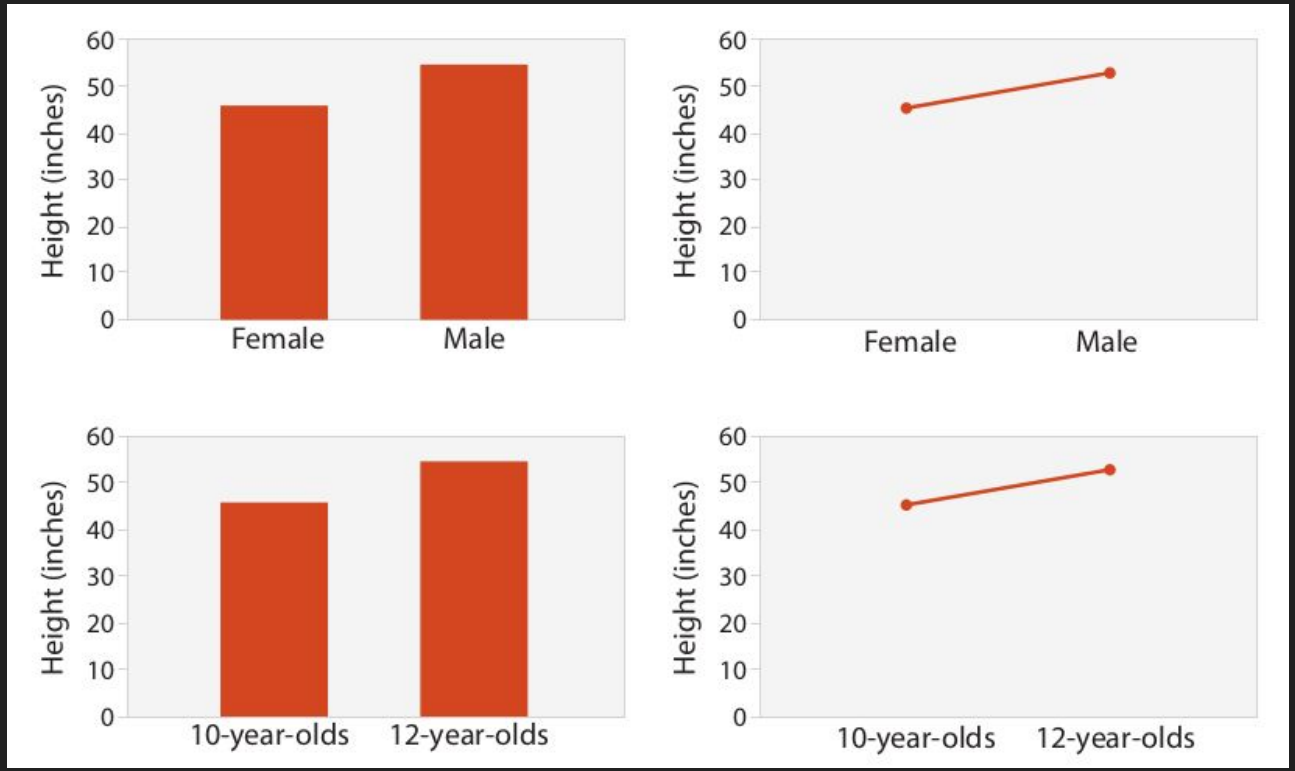
2) Separar, ordenar y alinear

Idiom	**Dot Charts**
What: Data	Table: one quantitative value attribute, one ordered key attribute.
How: Encode	Express value attribute with aligned vertical position and point marks. Separate/order into horizontal regions by key attribute.

2) Separar, ordenar y alinear

Idiom	**Line Charts**
What: Data	Table: one quantitative value attribute, one ordered key attribute.
How: Encode	Dot chart with connection marks between dots.
Why	Show trend.
Scale	Key attribute: hundreds of levels.

2) Separar, ordenar y alinear



Matrix alignment

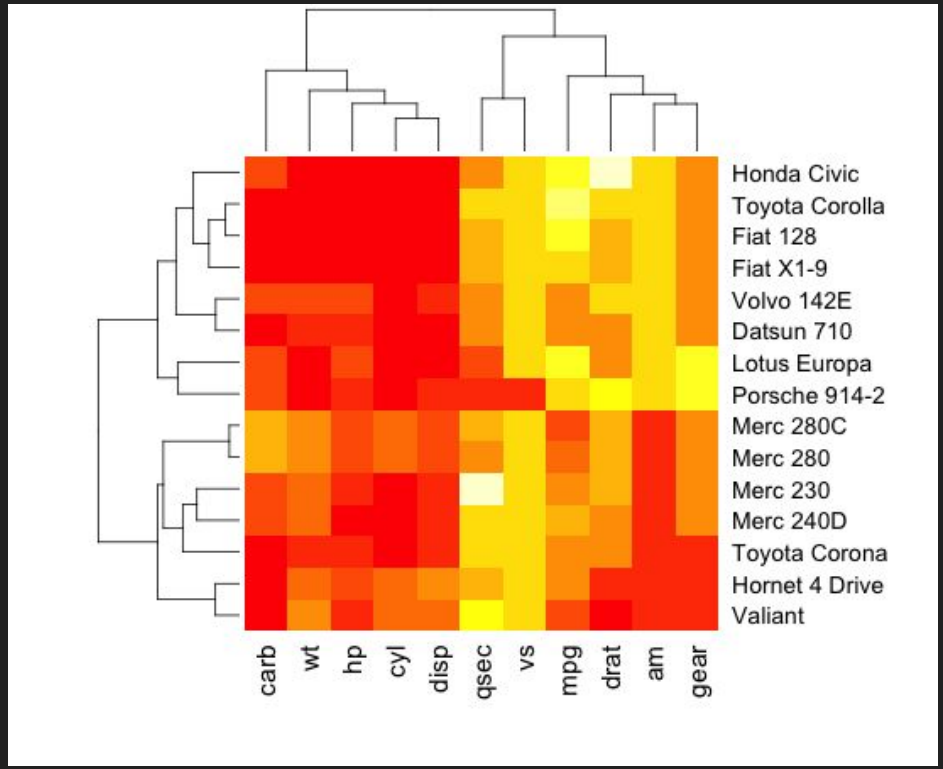
Matrix alignment, alinear con dos llaves

- Los *datasets* tabulares que están compuestos por dos llaves son, generalmente, mostrados en una matriz bidimensional, con una llave distribuida en las filas y otra en las columnas.
- Así, la celda en esta matriz es la región para codificar el valor del ítem.

Ejemplo (*heatmaps*)

- Este *idiom* es uno de los más simples de este tipo: cada celda está ocupada por una marca que codifica un único valor cuantitativo.
- La ventaja de los *heatmaps* es que usar este tipo de marcas cromáticas para codificar valores es compacta, por lo que permite ofrecer *overviews* con alta densidad de información.
- Los principales *tasks* que es posible desarrollar a partir de un *heatmap* son:
    - Encontrar *outliers*
    - Encontrar *clusters*
    - Tener un *overview*

Ejemplo (*heatmaps*)



Ejemplo (*heatmaps*)

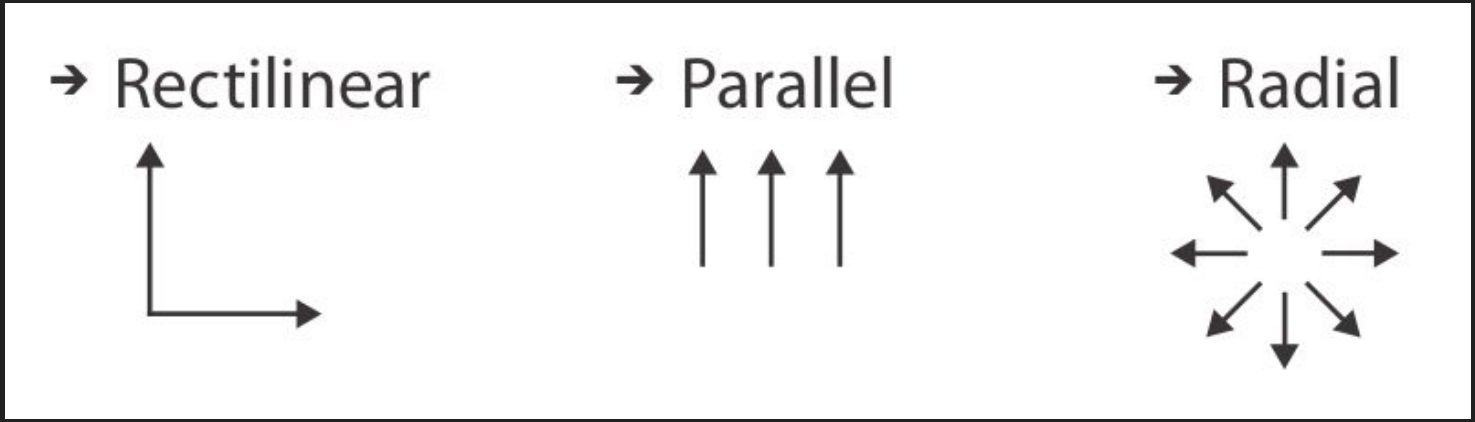
Idiom	**Heatmaps**
What: Data	Table: two categorical key attributes (genes, conditions), one quantitative value attribute (activity level for gene in condition).
How: Encode	2D matrix alignment of area marks, diverging colormap.
Why: Task	Find clusters, outliers; summarize.
Scale	Items: one million. Categorical attribute levels: hundreds. Quantitative attribute levels: 3–11.

¿Qué ocurre con más de dos llaves?

- Así como los datos pueden ser alineados en una lista (1D) o en una matriz (2D), es posible hacer lo mismo entonces con una grilla volumétrica (3D).
- Sin embargo, esta alternativa de diseño **no es recomendada**, ya que provoca problemas de percepción como oclusión y distorsión perspectiva.
- Una buena opción es utilizar **subdivisión recursiva**, que analizaremos en un capítulo futuro.

3) Orientación de ejes

- Para esta decisión, tenemos tres opciones: rectilíneos, paralelos, radiales.



Rectilinear layout

- En esta disposición, las regiones o ítems se distribuyen en **dos ejes perpendiculares**, con posiciones espaciales horizontales y verticales, que van desde un valor mínimo hasta uno máximo.
- Este tipo de disposición es utilizado generalmente en visualizaciones. De hecho, todos los ejemplos vistos hasta ahora en este capítulo utilizan un *rectilinear layout*.

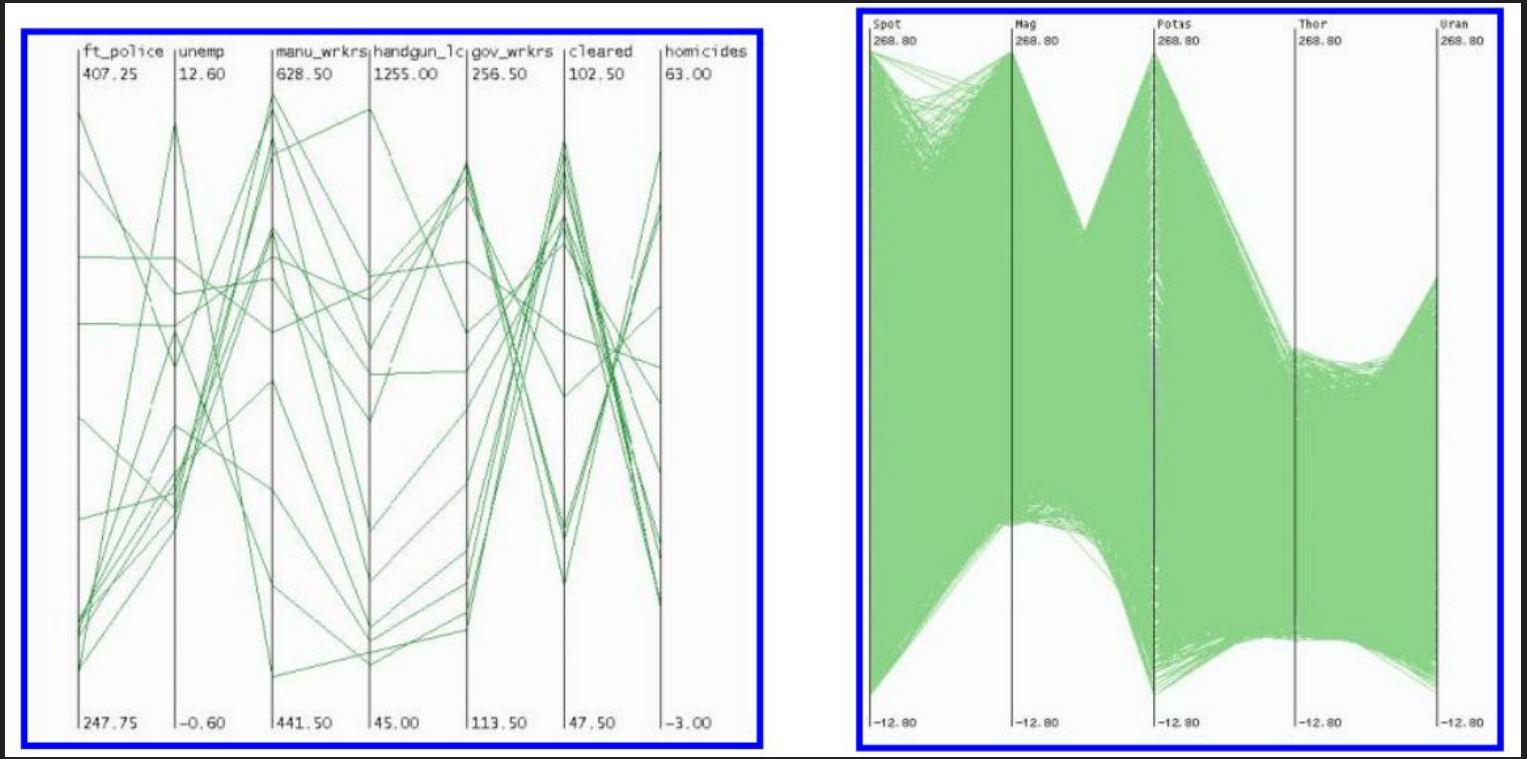
Parallel layout

- El *approach* rectilíneo de un *scatterplot* sólo nos permite analizar *datasets* con dos atributos. Si bien podemos codificar más atributos con canales no-espaciales, seguiremos limitados para atributos cuantitativos.
- Para solucionar este problema, es posible usar un *parallel layout* con el propósito de visualizar atributos multidimensionales.
- Como el nombre lo sugiere, los ejes están dispuestos de forma paralela entre ellos, generalmente de forma equidistante.

Ejemplo: *parallel coordinates*

- Un *idiom* específico de esta disposición son los *parallel coordinates*.
- Mientras un dato en un *scatterplot* es representado con un punto, acá cada ítem se simboliza con una línea que hace *zigzags* a través de los distintos ejes (cruzando exactamente una vez) justo en el lugar que representa el valor del ítem.
- Al igual que los *scatterplots*, este *idiom* también permite ver rápidamente si existe alguna correlación entre los atributos.
- Veamos este ejemplo hecho con D3.

Ejemplo: *parallel coordinates*

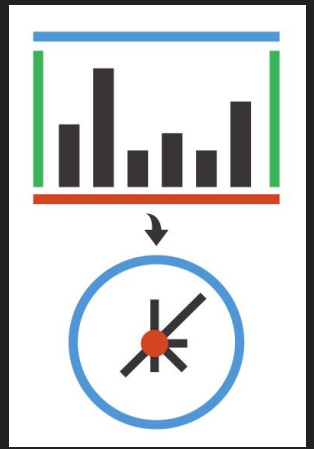


Ejemplo: *parallel coordinates*

Idiom	**Parallel Coordinates**
What: Data	Table: many value attributes.
How: Encode	Parallel layout: horizontal spatial position used to separate axes, vertical spatial position used to express value along each aligned axis with connection line marks as segments between them.
Why: Tasks	Find trends, outliers, extremes, correlation.
Scale	Attributes: dozens along secondary axis. Items: hundreds.

Radial layout

- En una disposición espacial radial, los ítems están distribuidos alrededor de un círculo, utilizando como canal el **ángulo** entre ellos.
- El sistema natural de coordenadas en este tipo de *layout* es el de coordenadas polares, en donde una dimensión es medida como el ángulo desde la línea de partida y la otra como una distancia desde el centro.
- Desde un punto de vista matemático, los *layouts* rectilíneos y radiales son equivalentes después de cierta transformación.



Radial layout

- Sin embargo, desde un punto de vista perceptual, estas disposiciones son bastante diferentes. El cambio de canal visual tiene dos principales consecuencias según los principios ya estudiados.
- Primero, el ángulo es percibido con menor precisión que un canal perteneciente al *layout* rectilíneo.
- Segundo, el canal angular es inherentemente cíclico ya que el punto de inicio y de término es el mismo. Esto se opone a la naturaleza lineal del canal de posición. Por esta misma razón, un *layout* radial es más efectivo para mostrar **periodicidad** de ciertos patrones. Por otra parte, codificar datos no-periódicos con este tipo de canal podría ser **engañoso**.

Referencias

- http://leebyron.com/streamgraph/stackedgraphs_byron_wattenberg.pdf
- http://www.visualisingdata.com/2010/08/making-sense-of-streamgraphs/
- Google music timeline https://research.google.com/bigpicture/music/