GGPLOT2

kurs: Programowanie w R (2010/2011)
autor: Adam Pytel

Niniejszy „Reference Card” dotyczy pakietu graficznego ggplot2 z programu R. Opisana została w krótki sposób funkcjonalność poszczególnych funkcji, metod oraz parametrów.

Tworzenie objektów ggplot

Przedstawione tu funkcje tworzą w rzeczywistości obiekty z klasy „ggplot”. Jest to szkielet pod dalszą obróbkę graficzną.

`ggfluctuation(table,type,...)` wizualizuje tabele kontyngencji `table`; `type` określa czy częstotliwości będą rozróżniane przy pomocy wielkości kwadratów(`"size"`), lub też nasycenia kolorem(`"colour"`).

`ggmissing(data,avoid,order)`wizualizacja braków danych w zbiorze dla poszczególnych zmiennych; `avoid` określa czy liczba braków danych i liczba pozostałych wartości ma być wyświetlana w jednym słupku bądź w różnych(`stack`, `dodge`); `order` odpowiada za porządek w jakim są wyświetlane słupki.

`ggorder(data,scale)` rysuje wykresy zależności zmiennych od numeru obserwacji; parametr `scale` zmieniamy za pomocą funkcji `rescale`.

`ggpcp(data,vars,scale)` szuka wzorców pomiędzy poszczególnymi zmiennymi(`vars`) ze zbioru; `scale` służy do skalowania zmiennych(dostępne `"range"`,`"var"`,`"I"`).

`ggstructure(data,scale)` wykres pozwalający wykrywać anomalie w danych; `scale` odpowiada za skalowanie.

`ggplot(data, ...)` tworzy nowy ggplot; jest szkieletem dla nakładania kolejnych warstw graficznych.

`ggplot.data.frame(data, mapping,...)` tworzy nowy obiekt ggplot.

`ggsave(filename,plot,scale...)` zapisuje konkretny obraz(`plot`) do pliku w podanej skali(`scale`).

`last_plot()` wyświetla ostatnio utworzony, bądź zmodyfikowany obiekt ggplot.

`plotmatrix(data,...)` wizualizuje macierz danych za pomocą wykresów punktowych.

`qplot(x,y,z,...,data,facets = . ~ .,...)` funkcja służąca do podstawowego wyświetlania wykresów o współrzędnych `x,y,z`(domyślnie `y,z=NULL`); `facets` grupuje wykresy wg zadanej formuły(używamy danych jakościowych).

Tworzenie grafiki

W tej sekcji zajmiemy się tworzeniem różnego rodzaju wykresów: po przez wykresy punktowe, schodkowe, aż po wykresy powierzchniowe,konturowe, histogramy itd. Wykresy są nakładane jako warstwy na szkielet, którym jest obiekt z klasy „ggplot”.

`geom_abline(mapping,...)` dorysowuje proste o zadanych parametrach `slope`, `intercept` (np. `mapping=aes(slope=4,intercept=15)`).

`geom_area(mapping,...)` tworzy wykres wypełniony do osi współrzędnych x (area plot).

`geom_bar(mapping,...)` tworzy wykresy słupkowe, histogramy, itp.

`geom_bin2d(mapping,...)` tworzy np. „heat” mapę.

`geom_blank(mapping,...)` nie rysuje nic poza współrzędnymi i standardowym polem na którym powstaje wykres.

`geom_boxplot(mapping,...)` tworzy wykres skrzynkowy; dodatkowe parametry odpowiadają za kolor obserwacji odstających,kształt i wielkość(`outlier.colour`, `outlier.shape`, `outlier.size`) .

`geom_conour(mapping,...)` tworzy wykres konturowy; dodatkowe parametry to `linemitre`, `lineend`, `linejoin`.

`geom_crossbar(mapping,...,fatten,...)` wykres w postaci skrzynek pomiędzy danymi jakościowymi a ilościowymi; `fatten` odpwiada za grubość środkowej lini na skrzynce.

`geom_density(mapping,...)` dorysowuje estymator gęstości.

`geom_density2d(mapping,...)` dorysowuje estymator gęstości dwuwymiarowej.

`geom_errorbar(mapping,...)` wykres w postaci przedziałów ufności pomiędzy danymi jakościowymi a ilościowymi.

`geom_errorbarh(mapping,...)` taki sam wykres jak `geom_errorbar`, z tym, że przedziały są poziomo.

`geom_freqpoly(mapping,...)` wyświetla wykres częstotliwości występowania danej zmiennej.

`geom_hex(mapping,...)` hexagonalna heat mapa.

`geom_histogram(mapping,...)` histogram.

`geom_hline(mapping,...)` dorysowuje proste horyzontalne (odpowiada za to parametr `yintercept`).

`geom_jitter(mapping,...)` uzyskanie efektu rozproszenia skategoryzowanych danych na wykresie, tak by nie zachodziły na siebie.

`geom_line(mapping,...)` tworzy ścieżkę zgodnie ze wzrostem współrzędnej `x`.

`geom_linerange(mapping,...)` rysuje wykres (np. dla danych jakościowych), dla którego obserwacje w poszczególnych kategoriach przedstawione są w postaci pionowych lini.

`geom_path(mapping,...)` tworzy ścieżkę łączącą kolejne obserwacje.

`geom_point(mapping,...)` tworzy wykres punktowy.

`geom_pointrange(mapping,...)` reprezentacja przedziałów przy pomocy pionowych lini wraz z kropką w środku.

`geom_polygon(mapping,...)` wypełnione kolorami wielokąty.

`geom_quantile(mapping,...)` służy do wyświetlania na wykresie prostych kwantylowych pochodzących z regresji kwantylowej.

`geom_rect(mapping,...)` rysuje prostokąty o zadanych wierzchołkach.

`geom_ribbon(mapping,...)` dorysowuje do wykresu wstęgi, bądź pasy ufności.

`geom_rug(mapping,...)` dorysowuje słupki do osi współrzędnych, oznaczajace położenie obserwacji.

`geom_smooth(mapping,...)` dorysowuje krzywe wielomianowe najlepiej dopasowujące się do danych.

`geom_step(mapping,...)` łączy obserwacje schodkami; dodatkowy parametr to `direction`, odpowiadający za ułożenie schodków.

`geom_segment(mapping,...)` tworzy chmurę lini o początkach w (`x`,`y`) i końcach w (`xend`,`yend`).

`geom_text(mapping,position,...)` dodaje tekst do wykresu.

`geom_tile(mapping,...)` rysuje obrazki z płytek takich samych wymiarów.

`geom_vline(mapping,...)` dorysowuje proste pionowe (odpowiada za to parametr `xintercept`).

Wszystkie powyższe warstwy mają wspólne parametry:

- `mapping` - służy do definiowania aspektów estetycznych wykresów, właściwości zmiennych itd.
- `data` - zbiór, na którego bazie, tworzymy grafikę.
- `stat` - statystyka, której używamy do tworzenia tej grafiki.
- `position` - pozycja, w której umieścimy wykres.

`update_geom_defaults(geom,new)` modyfikuje wybrane ustawienia geometryczne(`geom`) na nowe `new`.

Tworzenie grafiki cd.

W poprzednim rozdziale używaliśmy warstw geometrycznych do wizualizacji obiektów. Parametrem dodatkowym był rodzaj statystyk, które dodawaliśmy do obiektu. Tutaj zaprezentowana zostanie technika odwrotna - chcemy uzyskać pewien rodzaj wykresu, mając do dyspozycji zmianę geometrii. W części przypadków efekt jest identyczny jeśli byśmy używali samego `geom`, jednakże wykorzystując różne geometrie można te same statystyki przedstawić w różnych formach:

`stat_abline(mapping,...)` dorysowuje proste o zadanych parametrach `slope`, `intercept`.

`stat_bin(...,bins,binwidth,...)` zlicza obserwacje z poszczególnych przedziałów, oblicza średnie z tych przedziałów, odpowiada za tworzenie estymatora gęstości; `bins` odpowiada za liczbę przedziałów, a `binwidth` za szerokość.

`stat_bin2d(...,bins,binwidth,...)` dwuwymiarowa wersja `stat_bin`; `bins` odpowiada za liczbę prostokątów, a `binwidth` za rozmiary.

`stat_bin2d(...,bins,...)` wersja `stat_bin2d` rysujaca zamiast prostokątów - sześciokąty; `bins` odpowiada za liczbę sześciokątów.

`stat_boxplot(mapping,...)` tworzy wykres skrzynkowy.

`stat_contour(...,bins,binwidth,...)` tworzy wykres konturowy; `bins` odpowiada za liczbe konturów, `binwidth` za odległości pomiędzy kolejnymi konturami.

`stat_density(...,adjust,kernel,...)` dorysowuje estymator gęstości; parametry `adjust`, `kernel` są standardowymi parametrami z funkcji `density`.

`stat_density2d(...,contour,...)` dorysowuje estymator gęstości dwuwymiarowej; `contour` odpowiada za dorysowywanie konturów.

`stat_function(...,fun,args,...)` dorysowuje do danych wykres funkcji `fun` o dodatkowych argumentach `args`.

`stat_hline(mapping,...)` dorysowuje proste horyzontalne.

`geom_identity(mapping,...)` nie transformuje danych.

`stat_qq(...,distribution,dparams,...)` tworzy wykres kwantylowy dla danego rozkładu `distribution` o zadanych parametrach `dparams`.

`stat_quantile(...,quantiles,method,formula,...)` służy do wizualizacji regresji kwantylowej; `quantiles` - warunkowe kwantyle, `method` - użyta metoda, `formula` - formuła nadająca relacje pomiędzy `y` a `x`.

`stat_smooth(...,method,se,n,level,...)` dorysowuje krzywe wielomianowe najlepiej dopasowujące się do danych wg wybranej metody; `se` i `level` odpowiadają za pas ufności; `n` za liczbę obserwacji na których podstawie zostanie stworzony estymator.

`stat_spoke(mapping,...)` transformuje zmienne o współrzędnych polarnych na współrzędne kartezjańskie.

`stat_sum(mapping,...)` sumuje unikalne wartości - przydatne w porównywaniu w grupach.

`stat_summary(mapping,...)` tworzy graficzne podsumowanie dla `y` dla każdej unikalnej wartości `x`.

`stat_unique(mapping,...)` usuwa powtarzające się obserwacje.

`stat_vline(mapping,...)` dorysowuje proste pionowe.

Wszystkie powyższe warstwy mają wspólne parametry:

- `mapping` - służy do definiowania aspektów estetycznych wykresów, właściwości zmiennych itd.
- `data` - zbiór, na którego bazie, tworzymy grafikę.
- `geom` - geometria, którą chcemu wykorzystać.
- `position` - pozycja, w której umieścimy wykres.

`update_stat_defaults(geom,new)` modyfikuje wybrane ustawienia geometryczne(`geom`) na nowe `new`.

Koordynaty i mapy

Najpowszechniejszym układem współrzędnych jest układ kartezjański. Ten dział przedstawia warstwy, które odpowiadają za przekształcenia w tym układzie, a także proponuje inne rodzaje układów. Niektóre przykłady z „manuala" wymagają zainstalowania pakietów „maps", "maptools", "sp".

`borders(database,regions,fill,colour, ...)` dodaje warstwę granic i tło do mapy; `database` jest zbiorem map, `regions` - regionem który chcemy wyświetlić, `fill` odpowiada za kolor wypełnienia mapy, `colour` - kolor granic.

`coord_cartesian(xlim,ylim,wise,...)` współrzędne kartezjańskie na wykresie(ggplot wyświetla domyślnie); `xlim` i `ylim` są zakresami współrzędnych.

`coord_fixed(ratio,...)` określa proporcję pomiędzy skalami współrzędnych `x` a `y`.

`coord_flim(xlim,ylim,wise,...)` zamienia współrzędne miejscami

`coord_map(projection,orientation, ...)` rysuje mapy wg konkretnej projekcji i uwzględniając dany punkt orientacyjny(`orientation`).

`coord_polar(theta,start,direction,expand,...)`rysuje we współrzędnych biegunowych; kąt `theta` jest jedną ze współrzędnych 'x' lub 'y', `start` odpowiada za przesunięcie; `direction` za kierunek (zgodnie ze wskazówkami zegara będź przeciwnie)

`coord_trans(xtrans,ytrans,...)` transformacja współrzędnych, dostępne są miedzy innymi transformacja logarytmiczna, potęgowa, wykładnicza itd.

`fortify.map(model,...)` wzbogaca metody dla map, przekształca mapy na ramki danych.

`fortify.SpatialPolygonsDataFrame(model,region,...)` wzbogaca metody z pakiety sp.

`map_data(map,region)` konwertuje mapy do ramki danych.

Opcje graficzne

Detale, takie jak wygląd osi współrzędnych, tytuł, zakresy zmiennych, szata graficzna odpowiadają za estetyczną jakość wykresu oraz za jego przejrzystość. Poniższe funkcje służą do obsługi owych detali:

`aes(x,y,...)` funkcja odpowiadająca za estetykę wyświetlania obserwacji dla poszczególnych zmiennych (`x,y`).
`expand_limits(x,y)` modyfikuje zakresy współrzędnych.
`facet_grid(facets,scales,space,labeller,...)` prezentuje podgrupy danych w tabelce; grupy stworzone są wg formuły(`facets`), `scales` odpowiada za skale na współrzędnych dla poszczególnych grup, `spaces` odpowiada za ilość przestrzeni zajmowaną przez grupy, `labeller` odpowiada za etykiety.
`facet_wrap(facets,nrow,ncol,scales,...)` prezentuje podgrupy danych w tabelce o sugerowanej liczbie kolumn `ncol` oraz liczbie wierszy `nrow`; dostępne możliwości dla skalowania współrzędnych: `free`, `fixed`, `free_x`, `free_y`.
`label_both(variable,value)` odpowiada za łączenie w etykiecie nazwy zmiennej jakościowej z wartościami dla poszczególnych kategorii.
`label_bquote(variable,value)` wstawia całe wyrażenia matematyczne do etykiety.
`label_parsed(variable,value)` analizuje składnię etykiety i zamienia odpowiednio znaki.
`label_value(variable,value)` etykietowanie jedynie przy pomocy wartości zmiennej.
`labs(...)` funkcja służąca do modyfikacji nazw osi współrzędnych oraz tytułu.
`opts(...)` daje możliwość zmiany opcji wizualnych dla konkretnego wykresy.
`xlim(...)` funkcja określająca dolny i górny próg wartości dla współrzędnej `x`.
`ylim(...)` funkcja określająca dolny i górny próg wartości dla współrzędnej `y`.

Skalowanie

Skalowanie danych, kolorów, ostrości ma służyć poprawie przejrzystości obrazów. Daje możliwość spojrzenia na problem z różnych perspektyw.

`scale_alpha_continuous(name,limits,...)` moduł służący do modyfikacji parametru cieniującego `alpha`, jego zakresu(`limits`) oraz legendy związanej z parametrem `alpha`.
`scale_colour_brewer(name,palette,type,...)` moduł służący skalowaniu barw (`palette`) wyświetlanych punktów, modyfikacji legendy, kolorów itd.
`scale_x_continuous(name,limits,...)` moduł zmieniający właściwości współrzędnej `x` będącej zmienną ilościową; skalowanie i formatowanie .
`scale_y_continuous(name, limits,...)` moduł zmieniający właściwości współrzędnej `y`; skalowanie i formatowanie .
`scale_x_date(name,limits,major,minor,...)` moduł skalujący daty.
`scale_x_datetime(name,limits,major,minor,...)` moduł skalujący zmienne czasowe (dotyczy osi współrzędnych `x`).
`scale_x_discrete(name, limits,...)` to samo co `scale_x_continuous` tyle, że dla zmiennej skategoryzowanej.
`scale_colour_gradient(name,low,high,space,...)` skalowanie powmiędzy dwoma kolorami (`low` i `high`), używając odpowiedniego systemu kolorów (`space`).
`scale_colour_gradient2(...,low,mid,high,space,...)` skalowanie powmiędzy trzema kolorami (`low`, `mid` oraz `high`), używając odpowiedniego systemu kolorów (`space`).
`scale_colour_gradientn(name,colours,values,...)` skalowanie powmiędzy n kolorami (`values`), używając odpowiedniego systemu kolorów (`space`, np. „rgb”) i palety (`colours`).
`scale_colour_grey(...,start,end,...)` skalowanie przy pomocy odcieniów szarości.
`scale_colour_hue(...,h,l,c,...)` zaawansowane skalowanie przy pomocy barw(`h`), luminescencji(`l`), intensywności barwy(`c`).
`scale_linetype_discrete(...)` skalowanie wg różnych wzorców lini.
`scale_..._manual(...)` tworzenie własnej skali(dostępne np. `shape`, `colour`, `linetype`).
`scale_shape_discrete(...,solid,...)` odpowiada za kategoryzowanie obserwacji przy pomocy kształtu punktów wyświetlanych na obrazie; parametr `solid` odpowiada za wypełnienie bądź brak wypełnienia znaków.
`scale_size_continuous(...,trans,...)` skalowanie wielkości ciągłych zmiennych; `trans` odpowiada za transformację zmiennych.

Większość z powyższych „warstw” zawiera parametry:

- `name` nazwa legendy.
- `limits` zakres skalowania.
- `breaks` wektor numeryczny odpowiadający za stopień rozbicia.
- `legend` atrybut odpowiadający za powstanie legendy.
- `labels` etykiety opisujące rozbicie.

Tło, siatka, motywy

Siatka na wykresie bywa pomocna przy zczytywaniu dokładniejszych wartości danej zmiennej. Nie trzeba również przekonywać, że manipulacja tłem jest czasem niezbędna do uzyskanie odpowiedniego efektu wizualnego.

`theme_blank()` nie wyświetla tła ani żadnej siatki (pusty motyw).
`theme_bw(base_size,base_family)` tło jest białe, linie siatki są czarne; `base_size` odpowiada za wielkość czcionki; `base_family` za rodzaj czcionki.
`theme_get()` wyświetla opcje dla poszczególnych detali.
`theme_grey(base_size,base_family)` tło szare, linie siatki białe.
`theme_line(colour,size,lintype)` określa linie siatki.
`theme_segment(colour,size,linetype)` siatka rysowana jest przy pomocy prostokątów; parametry z tej funkcji odpowiadają za wygląd tychże prostokątów.
`theme_rect(fill,colour,size,linetype)` odpowiada za obramowanie, kolor, wypełnienie powierzchni okalającej wykres.
`theme_set(...)` ustawia motywy globalnie.

`theme_text(family,face,colour,...)` odpowiada za wygląd tekstu.
`theme_update(...)` modyfikuje poszczególne motywy.

Pozycjonowanie

Z punktu widzenia estetyki, czytelność obrazu jest bardzo ważnym aspektem, np. aby elementy grafiki nie nachodziły na siebie nawzajem. Do pozycjonowanie obiektów na warstwie służą następujące metody:

`position_dodge(width,height,...)` sprawia, że elementy wykresu(np. słupki) nie nachodzą na siebie.
`position_fill(width,height,...)` ustawia objekty na siebie, a następnie wyrównuje poziom(np. pomiędzy grupami).
`position_identity(width,height,...)` nic nie zmienia.
`position_jitter(width,height,...)` zaburza punkty na wykresie tak, aby nie nachodziły na siebie.
`position_dodge(width,height,...)` ustawia w stos.

Podział i wypisywanie zmiennych

Funkcje pomocne do przekształcania i formatowania zmiennych:

`comma(x)` format comma, `x` jest liczbą rzeczywistą.
`cut_interval(x,n,length,...)` zamienia liczby z wektora `x` na przedziały o tej samej długości, `n` jest liczbą przedziałów, a `length` długością.
`cut_number(x,n,...)` dyskretyzuje `x` na przedziały zawierające taką samą ilość elementów wektora `x` .
`dist_central_angle(lon, lat)` liczy kąt środkowy dla danej długości (`lon`) i szerokości geograficznej (`lat`).
`dollar(x)` zaokrągla centy i dodaje znak dolara.
`mean_se(x,mult)` oblicza średnią `x` oraz przedział ufności o szerokości `2*mult*se`.
`percent(x)` wektor liczbowy x zamienia na procenty.
`rescale(x, to=, from=, clip=T)` przeskalowuje wektor numeryczny `x` na zadany przedział.
`scientific(x)` wektor liczbowy przekształca na format naukowy.

Zbiory danych

Poniżej przedstawiono zbiory danych użyte w pakiecie „ggplot2" wraz z funkcjami wzbogacającymi te zbiory o pewne statystyki.

`diamonds` ceny 54000 oszlifowanych diamentów i ich charakterystyki.
`economics` szereg czasowy zawierające wskaźniki ekonomiczne w USA.
`midwest` dane demograficzne ze środkowo-zachodnich rejonów USA.
`movies` zbiór zawierający informacje o filmach i oceny użytkowników z portalu IMDB.com.
`mpg` informacje na temat zużycia paliwa poszczególnych rodzajów samochodów.
`msleep` dane dotyczące snu ssaków.
`presidential` dane dotyczące 10 prezydentów USA.
`seals` informacje na temat ruchu zwierzyny płownej w Oregonie.
`fortify(model, data, ...)` dodaje do zbioru danych `data` statystyki z modelu.
`fortify.lm(model, data, ...)` dodaje do zbioru danych `data` statystyki z modelu liniowego.