

ggplot2 grafiskā sistēma

Didzis Elferts

2024-02-27

Saturs

1	Par grāmatu	5
2	Ievads	7
2.1	Dati	7
2.2	Attēlu saglabāšana	8
3	Formas	9
3.1	geom_point()	9
3.2	geom_bar()	11
3.3	geom_col()	14
3.4	geom_line()	16
3.5	geom_path()	16
3.6	geom_boxplot()	18
3.7	geom_count()	21
3.8	geom_histogram()	23
3.9	geom_abline(), geom_hline() un geom_vline()	25
3.10	geom_jitter()	27
3.11	geom_smooth()	29
3.12	geom_violin()	33
4	Skalas	37
4.1	scale_x_continuous() un scale_y_continuous()	37
4.2	scale_x_discrete() un scale_y_discrete()	38
4.3	scale_color_...() un scale_fill...()	40
4.4	scale_alpha_...()	46
4.5	scale_linetype_...()	47
4.6	scale_shape_...()	47
4.7	scale_size_...()	48
4.8	Identitātes skalas	50
5	Koordinātu sistēmas	55
5.1	coord_cartesian()	55
5.2	coord_fixed()	57

5.3	<code>coord_flip()</code>	57
6	Attēlu sadalīšana	61
6.1	<code>facet_grid()</code>	61
6.2	<code>facet_wrap()</code>	63
7	Attēla noformēšana	67
7.1	Definētās attēla tēmas	67
7.2	Atsevišķu attēla elementu mainīšana	70
7.3	Noformējuma saglabāšana kā objektu	90
8	ggplot2 papildinājumi	93
8.1	<code>ggthemes</code>	93

Nodaļa 1

Par grāmatu

Programmā R (R Core Team (2023)) ir iespējams veidot attēlus izmantojot dažādas attēlu veidošanas sistēmas, no kurām viena ir `ggplot2` (Wickham (2016)). Šīs sistēmas pamatā ir attēlu veidošanas gramatika.

Nodaļa 2

Ievads

Veidojot attēlus `ggplot2` sistēmā, var izmantot divas funkcijas: `qplot()` vai `ggplot()`. Pirmā funkcija ir paredzēta ātrai attēla veidošanai, bet tai ir mazākas iespējas tikt modificētai, tāpēc šīs grāmatas ietvaros tā netiks izmantota. Šajā grāmatā visi piemēri balstīsies uz funkciju `ggplot()`. Šai funkcijai kā pirmais arguments ir jānorāda datu tabula/rāmis (var arī nenorādīt, bet tad tā jānorāda kā arguments `geom_...()` vai `stat_...()` funkcijās). Šis objektam būtu jābūt tādām, ko programma R uztver kā data frame. Nākamie argumenti ir `x` un `y` vērtības, kā arī citi mainīgie, ja no tiem ir jābūt atkarīgai krāsai, formai, utt. Visi mainīgie tiek norādīti funkcijā `aes()`. Ir jāatceras, ka `aes()` jānorāda tikai mainīgā (kolonnas) nosaukums, neliekot to pēdiņās, kā arī neizmantojot pieraktu `dati$mainigais`. Pieraksts ar `$` zīmi var radīt dīvainu (nepareizu rezultātu). Funkcijā `aes()` nav obligāti rakstīt `x=...` un `y=...` - trūkstot šiem argumentiem, pirmais mainīgais tiks uztverts kā `x`, bet otrais kā `y`.

2.1 Dati

`ggplot2` sistēmas iespēju apskatīšanai izmantoti R iekļautie datu objekti `CO2` un `mpg` (iekļauts paketē `ggplot2`). `CO2` ir eksperimenta rezultāti par sala tolerenci sugai *Echinochloa crus-galli*. Datu objektā ir piecas kolonnas: (1) `Plant` - auga identifikators; (2) `Type` - auga izcelsmes vieta; (3) `Treatment` - eksperimenta apstākļi (divas kategorijas); (4) `conc` - vides `CO2` koncentrācija; (5) `uptake` - uzņemtā `CO2` apjoms.

```
data(CO2)
head(CO2)
```

```
##   Plant   Type Treatment conc uptake
## 1  Qn1 Quebec nonchilled   95   16.0
## 2  Qn1 Quebec nonchilled  175   30.4
## 3  Qn1 Quebec nonchilled  250   34.8
```

```
## 4   Qn1 Quebec nonchilled  350   37.2
## 5   Qn1 Quebec nonchilled  500   35.3
## 6   Qn1 Quebec nonchilled  675   39.2
```

Objektā mpg ir informācija par degvielas patēriņu, kas pārādīta 11 kolonnās: (1) manufacturer - ražotājs; (2) model - automašīnas modelis; (3) displ - dzinēja tilpums; (4) year - ražošanas gads; (5) cyl - cilindru skaits; (6) trans - transmissijas tips; (7) drv - velkošie riteņi; (8) cty - jūdžu skaits/gallons pilsētā; (9) hwy - ūdžu skaits/gallons uz šosejas; (10) fl - degvielas veids; (11) class - automašīnas veids.

```
library(ggplot2)
data(mpg)
head(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv   cty   hwy fl      class
##   <chr>          <chr> <dbl> <int> <int> <chr>    <chr> <int> <int> <chr> <chr>
## 1 audi          a4      1.8  1999     4 auto(l5) f       18    29 p      compa~
## 2 audi          a4      1.8  1999     4 manual(m5) f       21    29 p      compa~
## 3 audi          a4      2    2008     4 manual(m6) f       20    31 p      compa~
## 4 audi          a4      2    2008     4 auto(av) f       21    30 p      compa~
## 5 audi          a4      2.8  1999     6 auto(l5) f       16    26 p      compa~
## 6 audi          a4      2.8  1999     6 manual(m5) f       18    26 p      compa~
```

2.2 Attēlu saglabāšana

ggplot2 sistēmā izveidoto attēlu saglabāšanu var veikt ar funkciju ggsave(), kuru izpilda pēc attēlā izveidošanas un kurā kā pamatarguments ir jānorāda vēlamais attēlā nosaukums ar nepieciešamo paplašinājumu (png, eps, ps, tex, pdf, jpeg, tiff, bmp, svg, wmf (tikai uz windows)). Papildus var norādīt attēla izmēru (width= un height=). Pēc noklusējuma izmērs ir collās, bet var mainīt uz cm vai mm ar argumentu units=. Arguments dpi= rastra tipa attēliem maina izšķirtspēju.

```
library(ggplot2)
data(CO2)
ggplot(CO2, aes(conc, uptake)) + geom_point()
ggsave("Attels_1.png", width = 10, height = 6, units = "cm")
```


Nodaļa 3

Formas

ggplot2 sistēmā ir iespējams vienus un tos pašus datus attēlot dažādos veidos, izvēloties atbilstošo datu attēlošanas formu jeb `geom_...()`. Vairumā gadījumu ir jānorāda x un y vērtības, bet atsevišķos gadījumos ir nepieciešami arī papildus mainīgie, vai arī nepieciešamas tikai x vērtības (piemēram, histogrammai).

3.1 `geom_point()`

Ar `geom_point()` ir iespējams veidot izkliedes attēlus (scatterplot) (3.1 attēls).

```
library(ggplot2)
ggplot(CO2, aes(conc, uptake)) + geom_point()
```

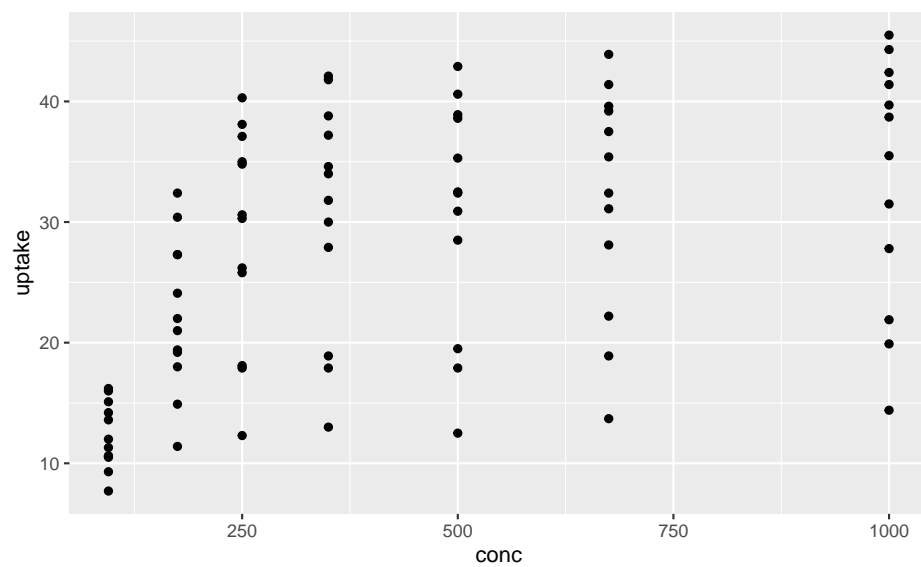
Punktiem ir iespējams mainīt krāsu (`color =`), formu (`shape =`), lielumu (`size =`) un caurspīdīgumu (`alpha =`). Mainot šos parametrus ir jānoņem pēc kādiem principiem tas notiks - parametrs būs vienāds visiem punktiem, vai arī tas mainīsies atkarībā no kāda cita mainīgā datos.

Ja parametram ir jābūt vienādam visiem punktiem, tad tas ir jānorāda ārpus funkcijas `aes()` pašā `geom_...()` vai `ggplot()` funkcijā. Toties, ja parametram ir jāmainās atkarībā no mainīgā, tad tas obligāti jāliek funkcijā `aes()`.

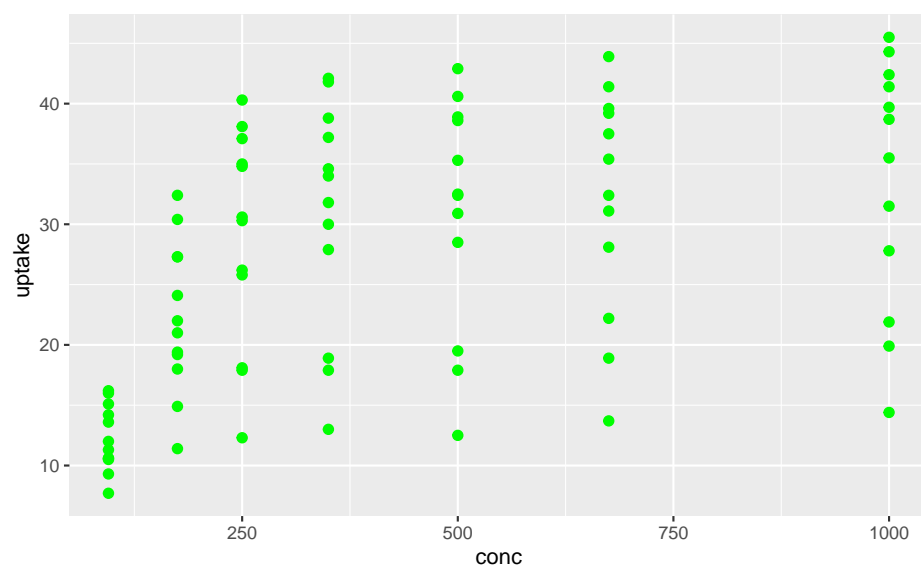
Šajā piemērā punktu krāsa un lielums ir mainīts visiem punktiem uzreiz (3.2 attēls). Krāsu var norādīt kā tās angļu nosaukumu (tos var apskatīt ar funkciju `colors()`) vai arī izmantojot heksadecimālo kodu.

```
ggplot(CO2, aes(conc, uptake)) + geom_point(color = "green", size = 2)
```

Ja arguments `color =` atrodas `aes()` iekavās un tas ir atkarīgs no kāda mainīgā, tad punktu krāsa mainīsies atbilstoši vērtībām, kā arī parādīsies atbilstošā leģenda. Krāsu maiņa un leģendas veids ir atkarīgs no tā, kāda veida mainīgais



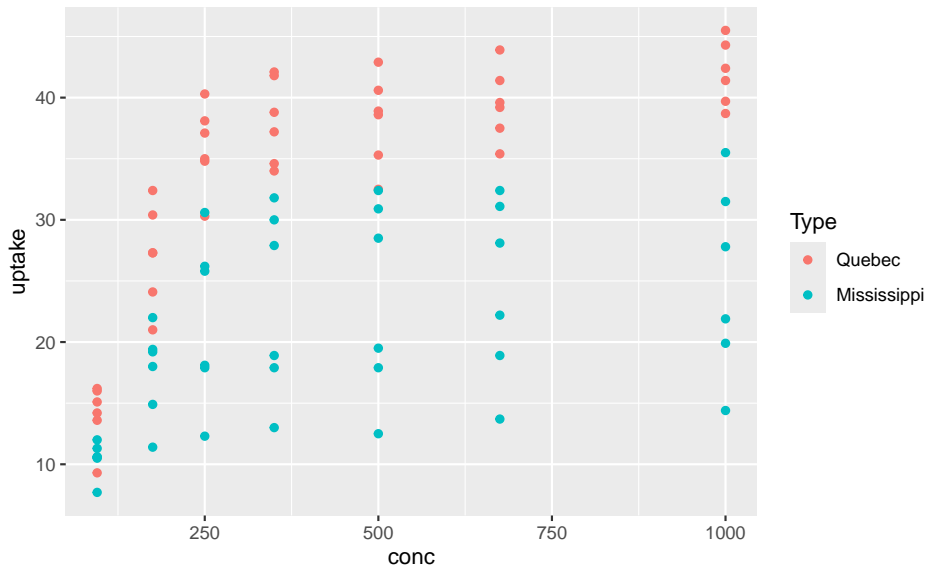
Att. 3.1: Izklīdes attēla piemērs



Att. 3.2: Izklīdes attēls, kurā krāsa un lielums visiem punktiem vienāds

ir izmantots. Ja krāsa ir atkarīga no kategorijas mainīgā, tad krāsas mainīsies diskrēti (3.3 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point()
```



Att. 3.3: Izkliedes attēls, kurā krāsa ir atkarīga no kategorijas mainīgā

Toties norādāt kā mainīgo tādu, kas ir skaitlisks, krāsa mainīsies kā gradients (3.4 attēls).

```
ggplot(CO2, aes(conc, uptake, color = uptake)) + geom_point()
```

Punktu forma var mainīties tikai atkarībā no kategorijas mainīgā (3.5 attēls).

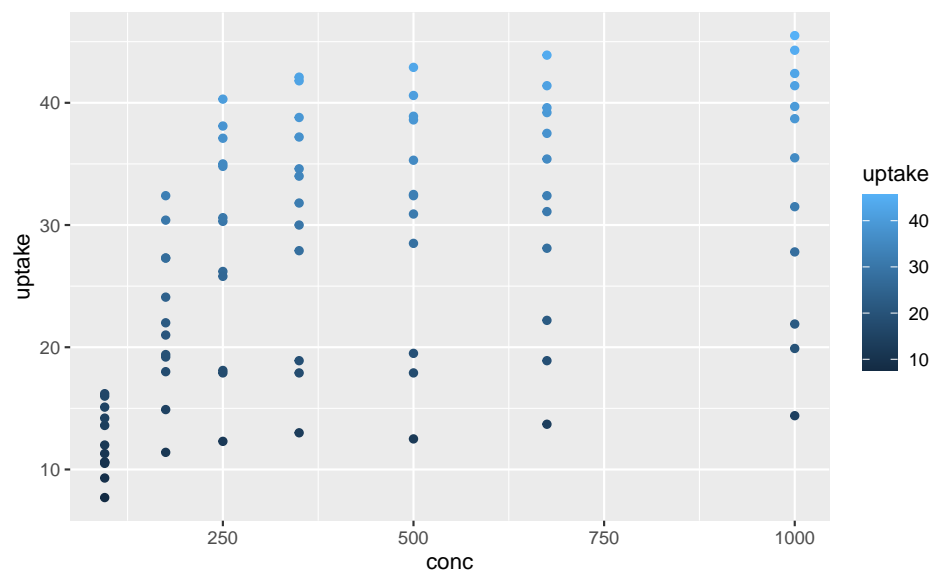
```
ggplot(CO2, aes(conc, uptake, shape = Type)) + geom_point()
```

Ir iespējams panākt arī, ka, piemēram, punktu forma mainās atkarībā no viena mainīgā, bet krāsa atkarībā no cita mainīgā. Šajā gadījumā parādīsies arī divas lēģendas (3.6 attēls).

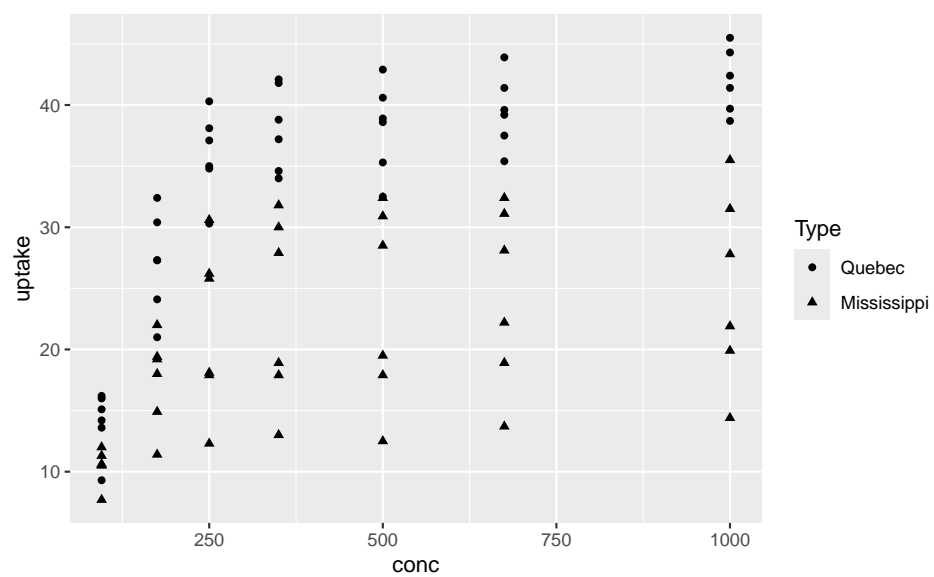
```
ggplot(CO2, aes(conc, uptake, shape = Type, color = Treatment)) +  
  geom_point()
```

3.2 `geom_bar()`

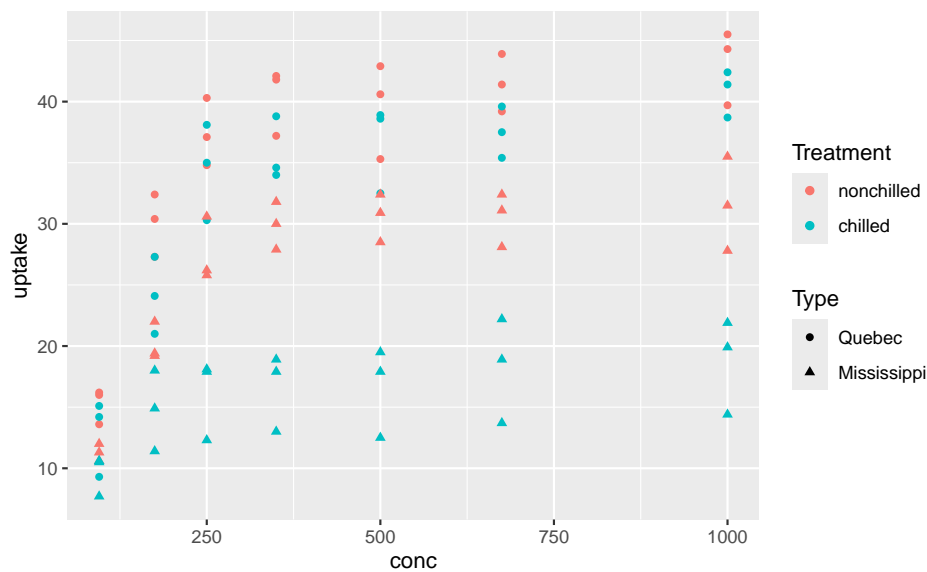
Stabiņu attēlus veido ar funkciju `geom_bar()`. Šai funkcijai `aes()` ir jānorāda tikai x vērtības (diskrētas), jo novērojumu skaits katrā klasē tiek saskaitīts automātiski (`geom_bar()` balstās un `stat_count()`) (?? attēls).



Att. 3.4: Izkliedes attēls, kurā krāsa ir atkarīga no skaitliska mainīgā

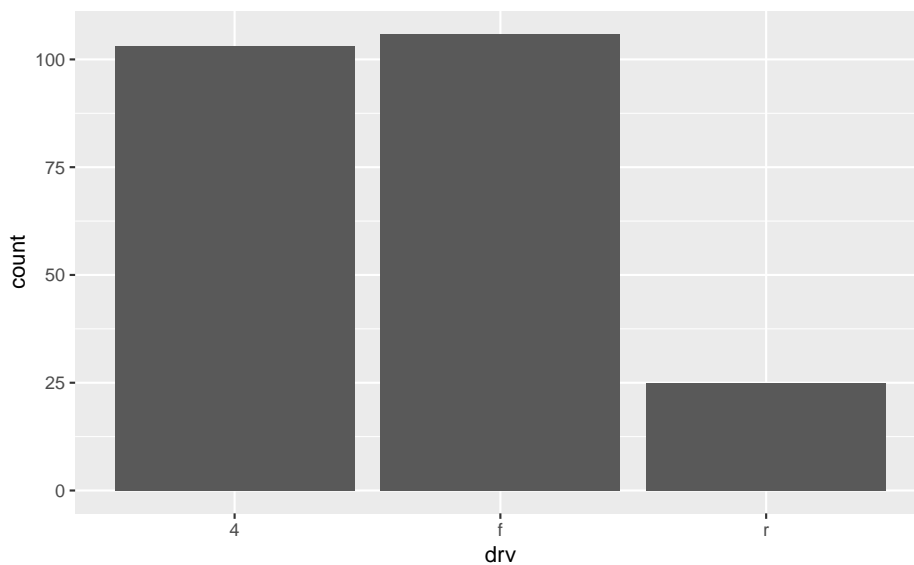


Att. 3.5: Izkliedes attēls, kurā forma ir atkarīga no kategorijas mainīgā



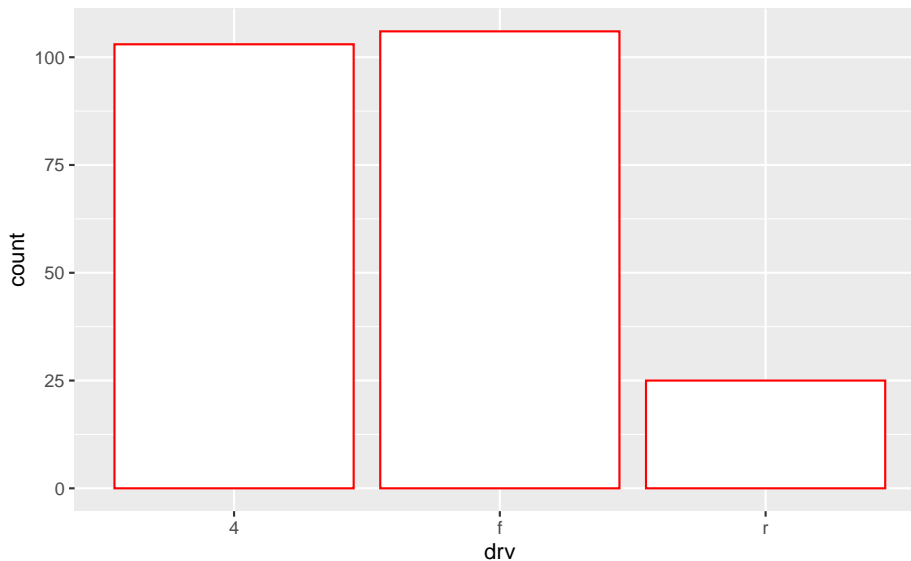
Att. 3.6: Izkliedes attēls, kurā forma un krāsa ir atkarīga no dažādiem kategorijas mainīgajiem

```
ggplot(mpg, aes(drv)) + geom_bar()
```



Stabiņu attēlā katram stabiņam ir iespējams mainīt krāsu (`color =`) un aizpildījumu (`fill =`). Arguments `color =` nosaka līnijas krāsu apkārt katram no stabiņiem, bet `fill =` nosaka paša stabiņa krāsu (aizpildījumu) (3.7 attēls).

```
ggplot(mpg, aes(drv)) + geom_bar(fill = "white", color = "red")
```



Att. 3.7: Stabiņu attēls, kurā stabiņu krāsa un aizpildījums visiem vienāds

Padarot aizpildījumu atkarīgu no kāda kategorijas mainīgā, izveidojas stabiņu attēls, kur pie katras x mainīgā kategorijas, stabiņš ir sadalīts pa daļām balstoties uz jauno mainīgo (3.8 attēls).

```
ggplot(mpg, aes(drv, fill = factor(cyl))) + geom_bar()
```

Pieliekot argumentu `position = "dodge"`, var panākt, ka pie katras x kategorijas stabiņi ir viens otram blakus, nevis viens virs otra (3.9 attēls).

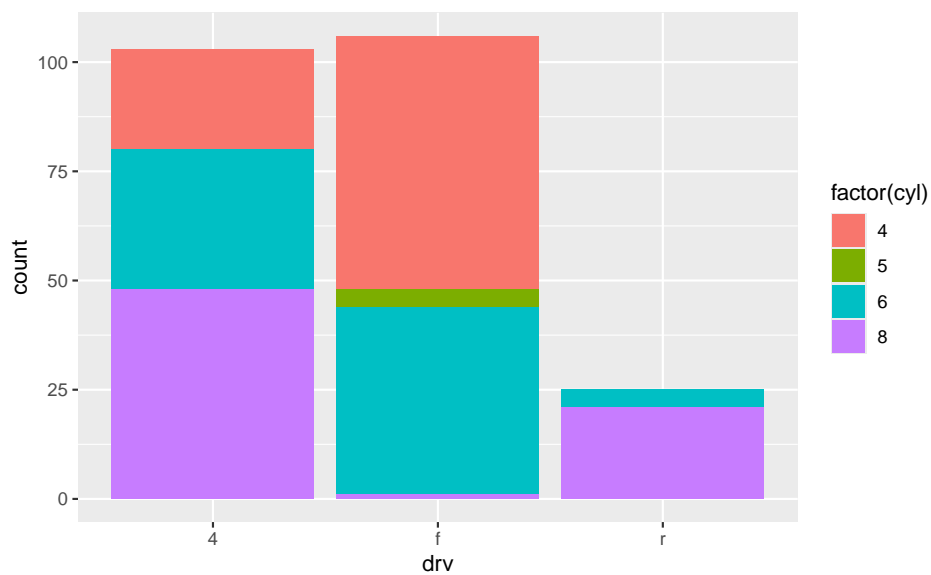
```
ggplot(mpg, aes(drv, fill = factor(cyl))) + geom_bar(position = "dodge")
```

3.3 geom_col()

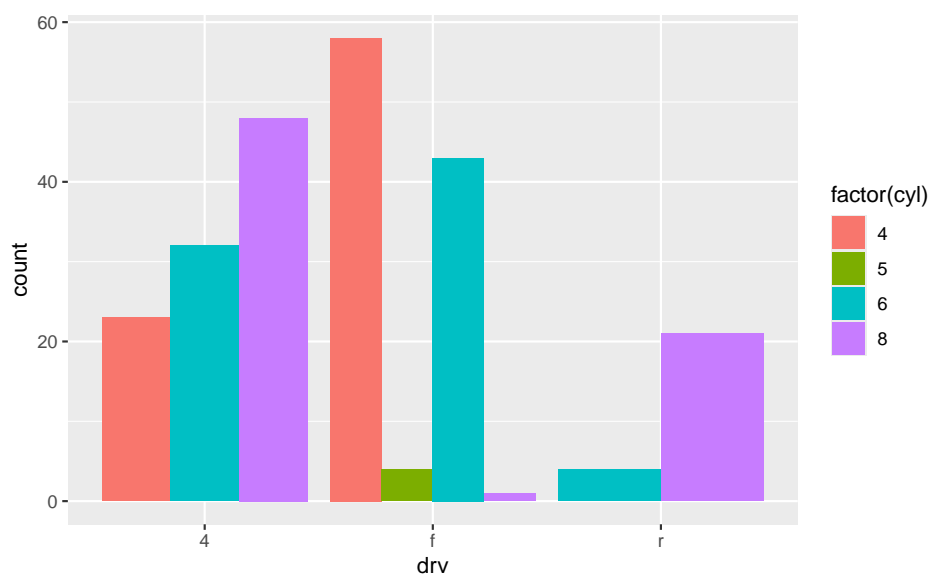
Gadījumos, kad dati ir jau apkopoti un ir nepieciešams izveidot stabiņu attēlu, tad labāk izmantot `geom_col()`, kam jānorāda gan x vērtības, gan arī atbilstošās y vērtības (skaiti) (3.10 attēls).

```
dati <- data.frame(Dzimums = c("S", "V"), Skaits = c(23, 45))
dati
```

```
##   Dzimums Skaits
## 1      S     23
## 2      V     45
```

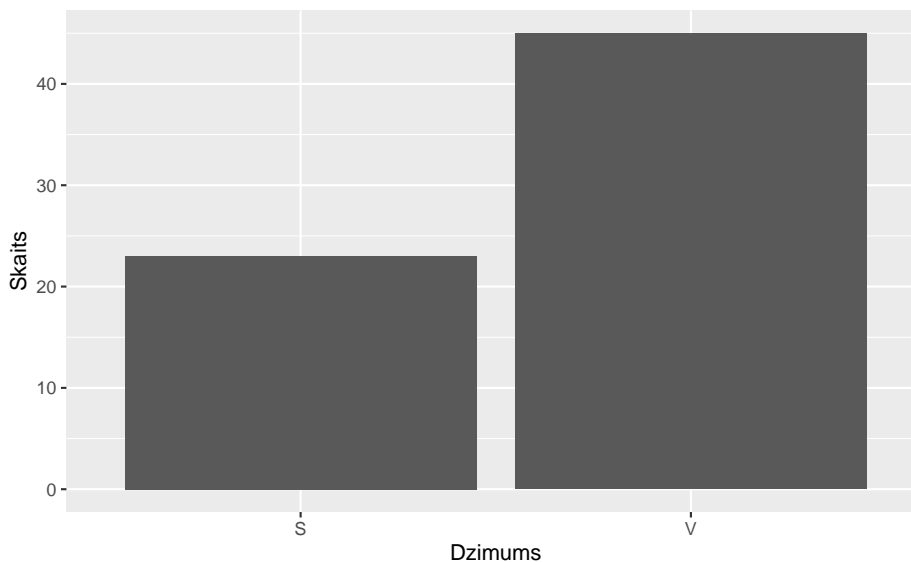


Att. 3.8: Stabiņu attēls, kurā stabiņu aizpildījums atkarīgs no mainīgā



Att. 3.9: Stabiņu attēls, kurā stabiņu aizpildījums atkarīgs no mainīgā

```
ggplot(dati, aes(Dzimums, Skaits)) + geom_col()
```



Att. 3.10: Stabiņu attēls, kurā skaiti jau doti tabulā

3.4 geom_line()

Datu punktu savienošanai ar līniju var izmantot `geom_line()`, kas savieno punktus no mazākās x vērtības līdz lielākajai x vērtībai (3.11 attēls).

```
ggplot(mpg, aes(cty, hwy)) + geom_line()
```

Līnijām ir iespējams mainīt tās platumu (`linewidth =`), krāsu (`color =`) un līnijas veidu (`linetype =`) (3.12 attēls).

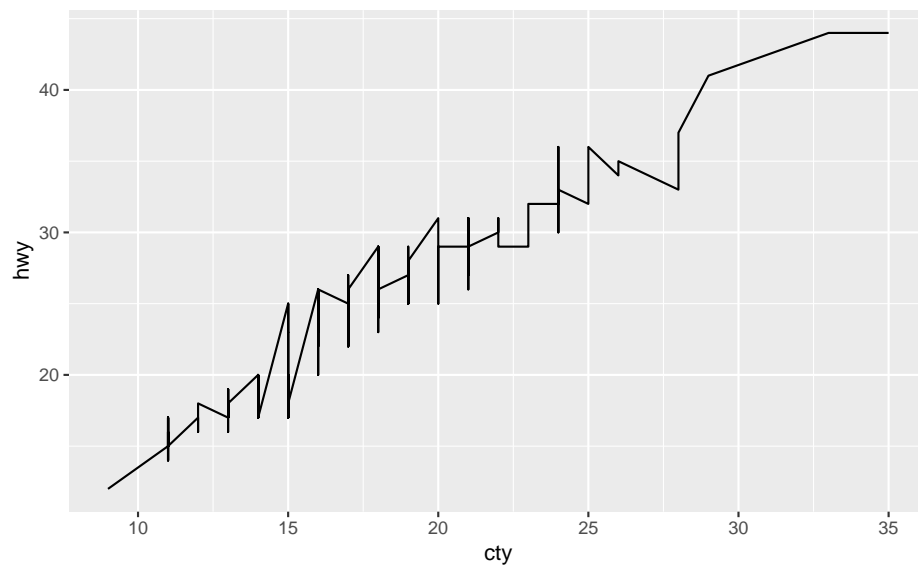
```
ggplot(mpg, aes(cty, hwy)) +  
  geom_line(color = "red", linewidth = 1.5, linetype=2)
```

Ja kāds no līnijas parametriem ir atkarīgs no diskrēta trešā mainīgā, tad parādīsies tik daudz līnijas, cik mainīgajam ir līmeņi (3.13 attēls).

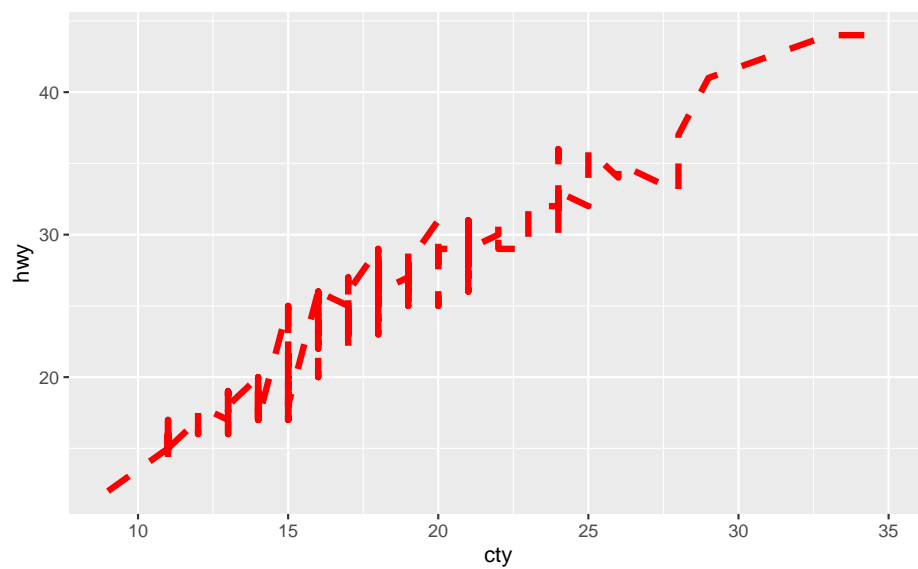
```
ggplot(mpg, aes(cty, hwy, color = drv)) + geom_line()
```

3.5 geom_path()

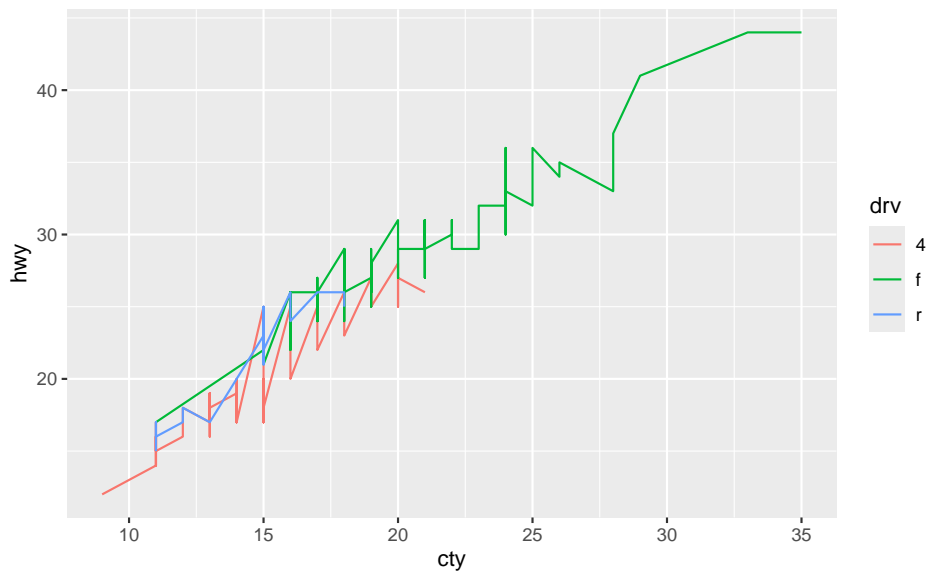
Līniju diagrammai līdzīgs ir arī `geom_path()`, bet šajā gadījumā punkti tiek savienoti tādā secībā, kādā tie parādās datu tabulā (3.14 attēls). `geom_path()`



Att. 3.11: Līniju diagrammas piemērs



Att. 3.12: Līnija ar izmainītiem parametriem



Att. 3.13: Līnija, kuras krāsa atkarīga no mainīgā

ir īpaši noderīgs gadījumos, ja jāsavieno x un y koordinātes, piemēram, attēlojot pārvietošanās ceļam.

```
ggplot(mpg, aes(cty, hwy)) + geom_path()
```

3.6 geom_boxplot()

Vērtībamplitūdas diagrammas veidošanai izmanto `geom_boxplot()`. Šim attēla veida x vērtībām ir jābūt kvalitatīviem datiem, vai arī skaitliskiem datiem, kas pārvērtēti par faktoru. y vērtībām obligāti ir jābūt skaitliskām (3.15 attēls).

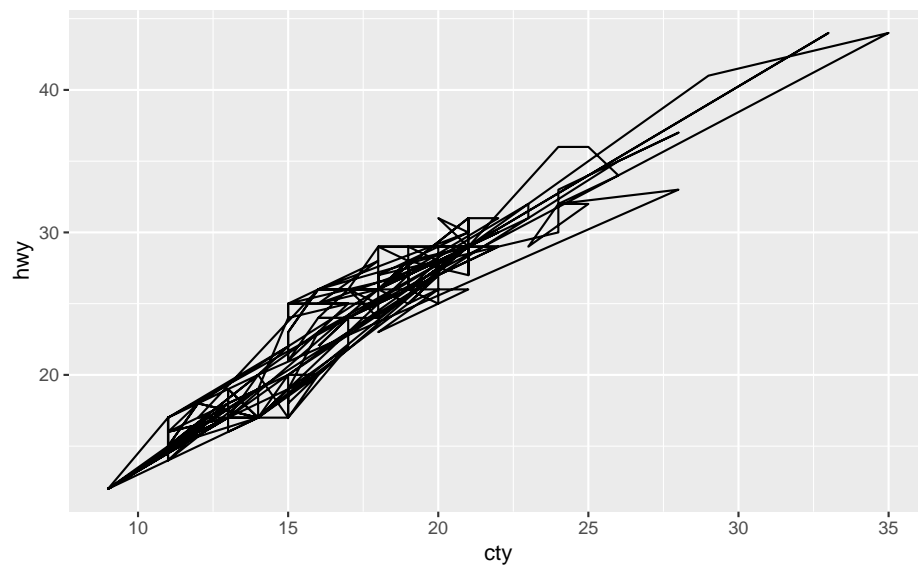
```
ggplot(CO2, aes(Type, uptake)) + geom_boxplot()
```

Līdzīgi kā stabiņu attēlam vērtībamplitūdas diagrammā var mainīt līniju un punktu krāsu (`color =`) vai arī “kastītes” aizpildījumu (`fill =`) (3.16 attēls).

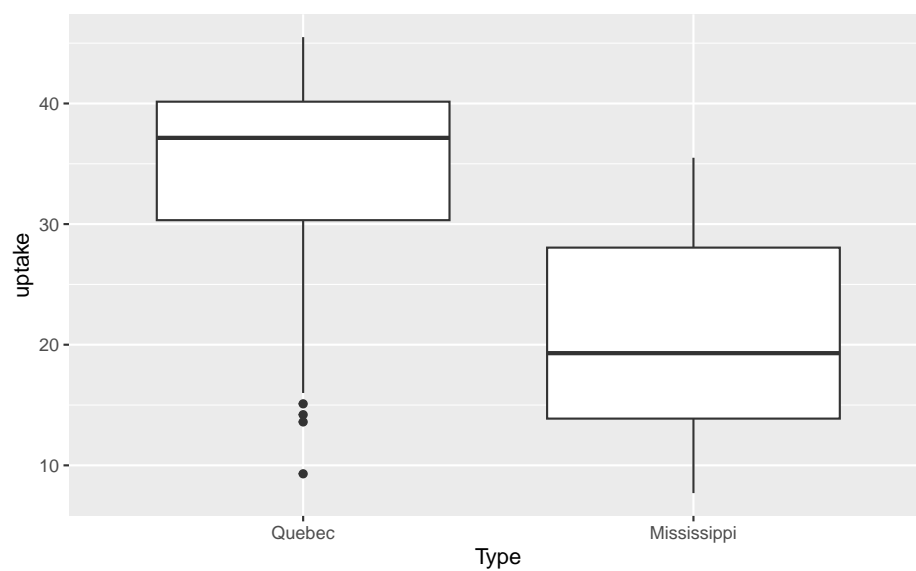
```
ggplot(CO2, aes(Type, uptake)) +  
  geom_boxplot(color = "darkgreen", fill = "salmon")
```

Izlēcēju (neraksturīgo vērtību) punktu krāsu, formu un izmēru var mainīt arī atsevišķi, izmantojot argumentus `outlier.color =`, `outlier.shape =` un `outlier.size =` (3.17 attēls).

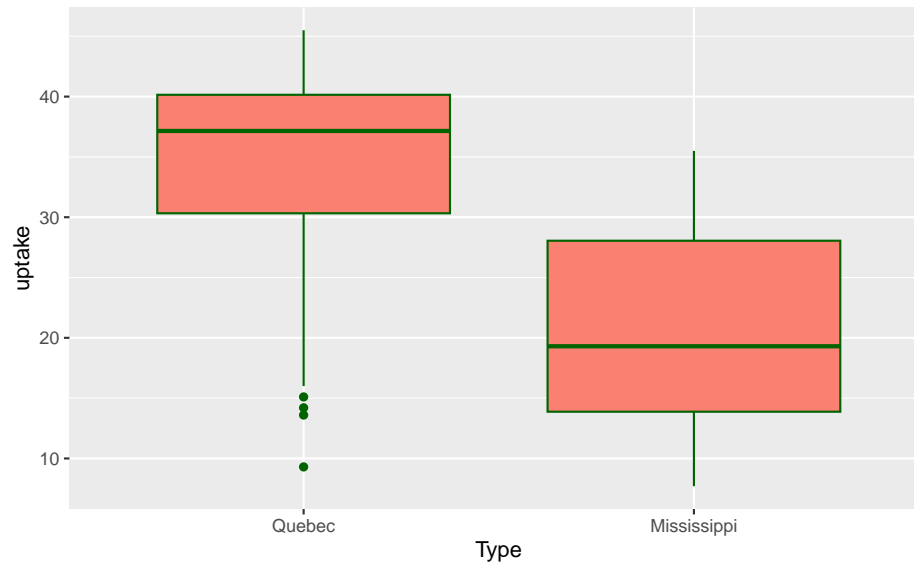
```
ggplot(CO2, aes(Type, uptake)) +  
  geom_boxplot(outlier.color = "red", outlier.shape = 13, outlier.size = 3)
```



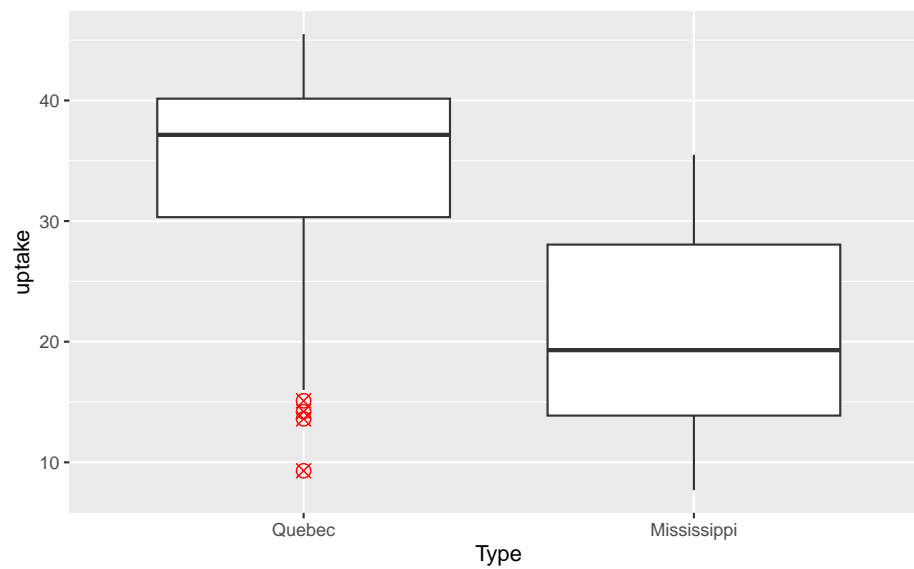
Att. 3.14: Punkti, kas savienoti ar līniju to izvietojuma secībā



Att. 3.15: Vērtībamplitūdas diagrammas piemērs



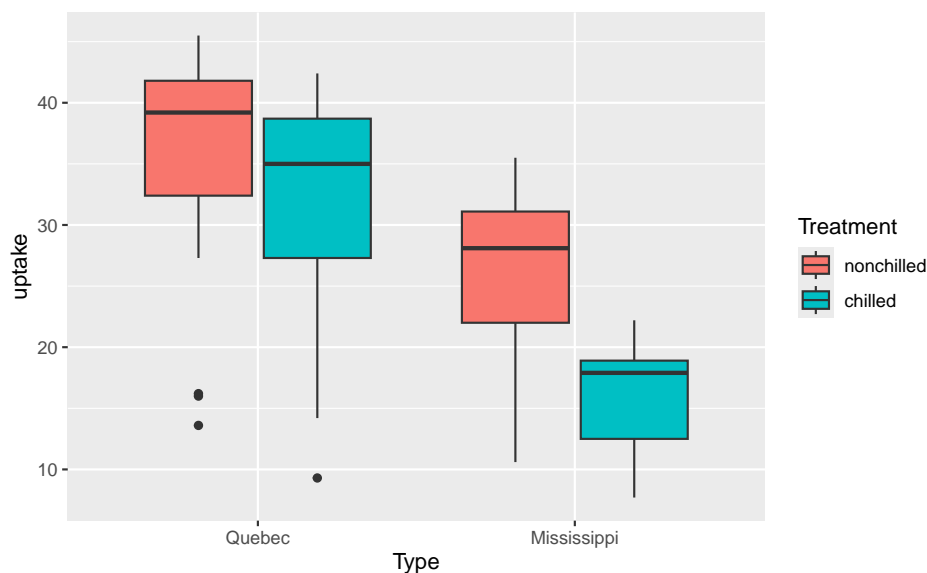
Att. 3.16: Vērtībamplitūdas diagramma ar izmainītu līniju un kastītes krāsu



Att. 3.17: Vērtībamplitūdas diagramma ar izmainītu izlēcēju krāsu, formu un izmēru

Ja arguments `fill` = atrodas funkcijas `aes()` iekavās un tas ir atkarīgs no kāda mainīgā, tad pie katras atbilstošās x vērtības, vērtībamplitūdas diagramma tiek sadalīta tik daļās, cik līmeņi ir papildus mainīgajam, kā arī parādās atbilstošā lēģenda ar izmantotajām aizpildījuma krāsām (3.18 attēls).

```
ggplot(CO2, aes(Type, uptake, fill = Treatment)) +  
  geom_boxplot()
```



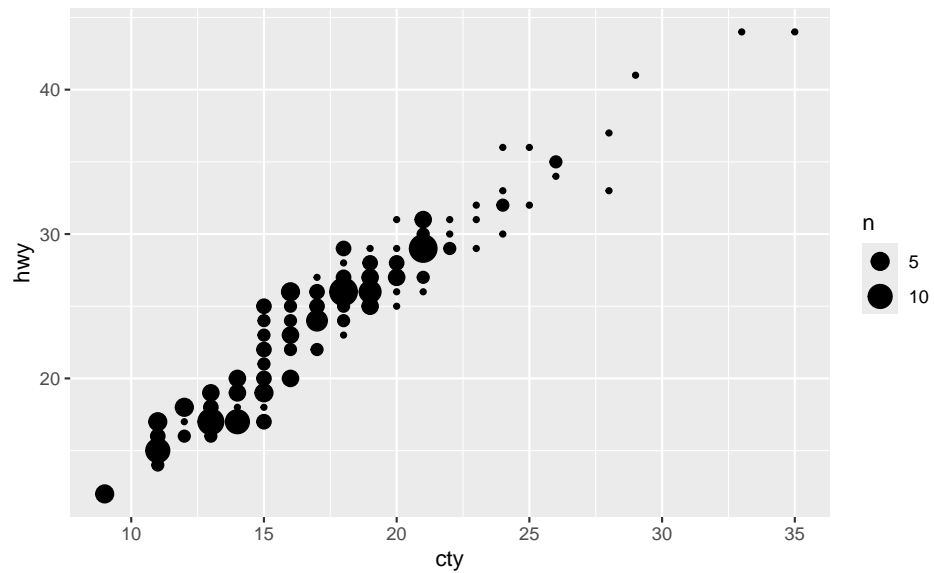
Att. 3.18: Vērtībamplitūdas diagramma, kurā katram faktora līmenim diagramma sadalīta daļās

3.7 `geom_count()`

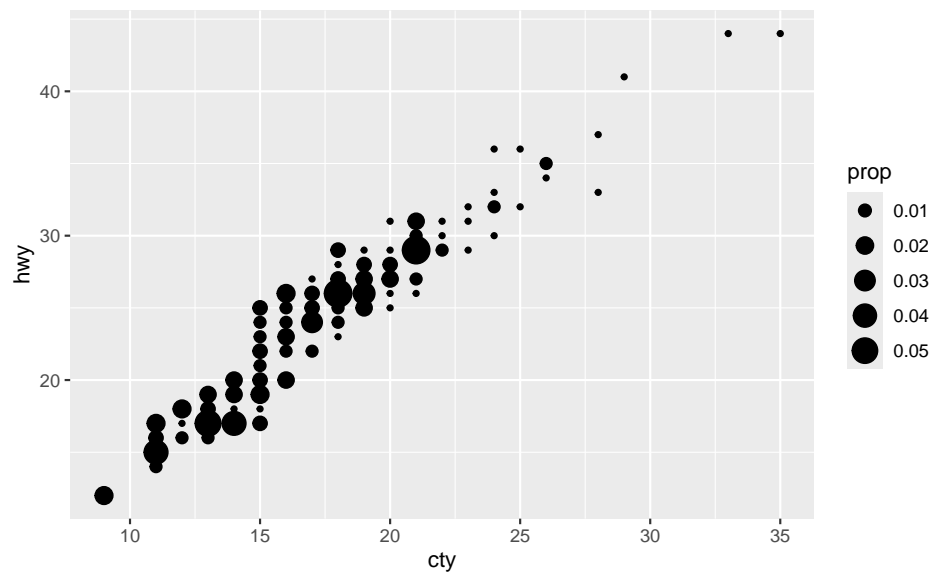
Gadījumos, kad nepieciešams attēlot izkliedes diagrammu, bet ir vērojama punktu pārklāšanās (pie vienādām x un y vērtībām ir vairāki novērojumi), var izmantot `geom_count()`, kas parāda cik daudz novērojumu ir konkrētajām x un y vērtībām (3.19 attēls).

```
ggplot(mpg, aes(cty, hwy)) + geom_count()
```

Ja `aes()` funkcijā norāda argumentu `size = after_stat(prop)`, tad punktu lielums ir parāda proporciju, nevis skaitu (3.20 attēls).



Att. 3.19: Izkliedes diagramma, kur punktu lielums atkarīgs no novērojumu skaita



Att. 3.20: Izkliedes diagramma, kur punktu lielums atbilst novērojumu proporcijai

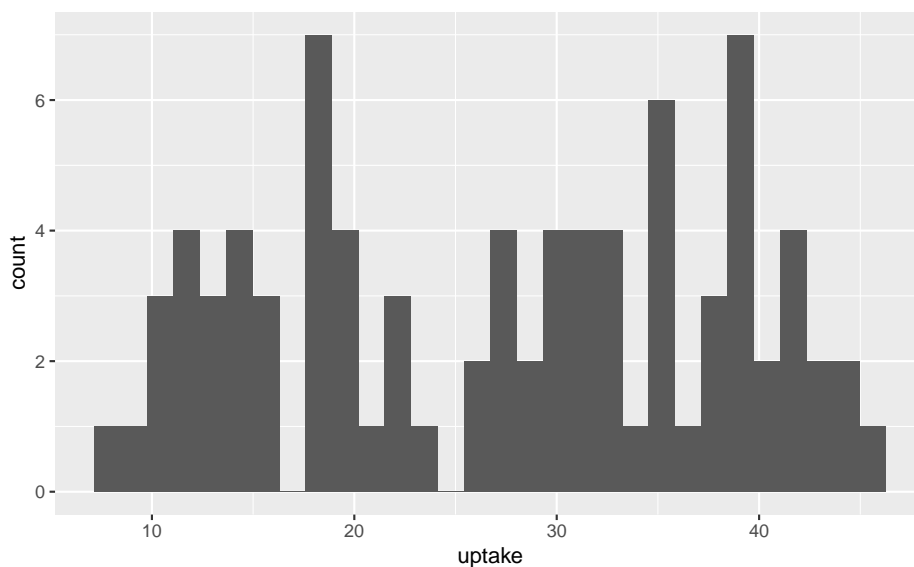
```
ggplot(mpg, aes(cty, hwy)) + geom_count(aes(size=after_stat(prop)))
```

3.8 `geom_histogram()`

Histogrammas veidošanai izmanto funkciju `geom_histogram()`, kam ir nepieciešamas tikai x vērtības. Pēc noklusējuma dati tiek dalīti trīsdesmit klasēs (3.21 attēls).

```
ggplot(CO2, aes(uptake)) + geom_histogram()
```

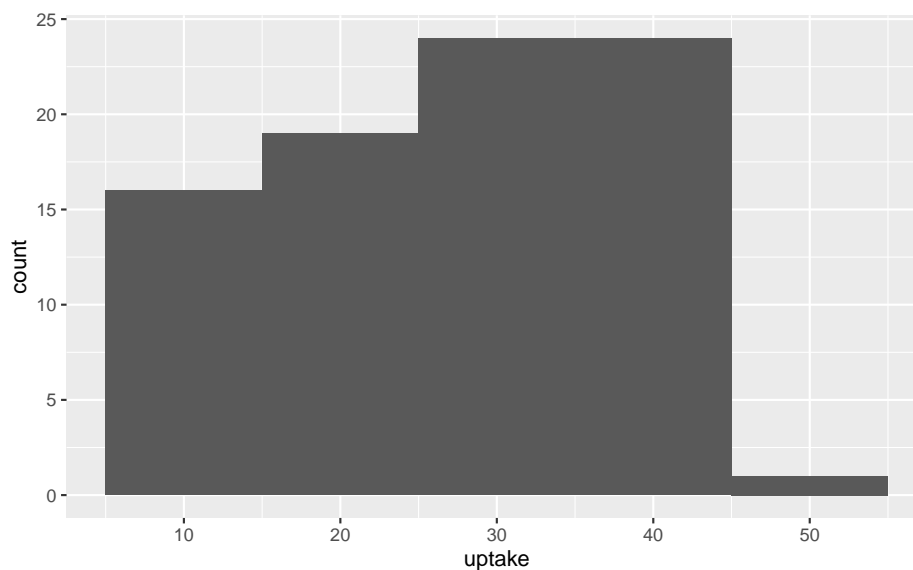
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Att. 3.21: Histogrammas piemērs

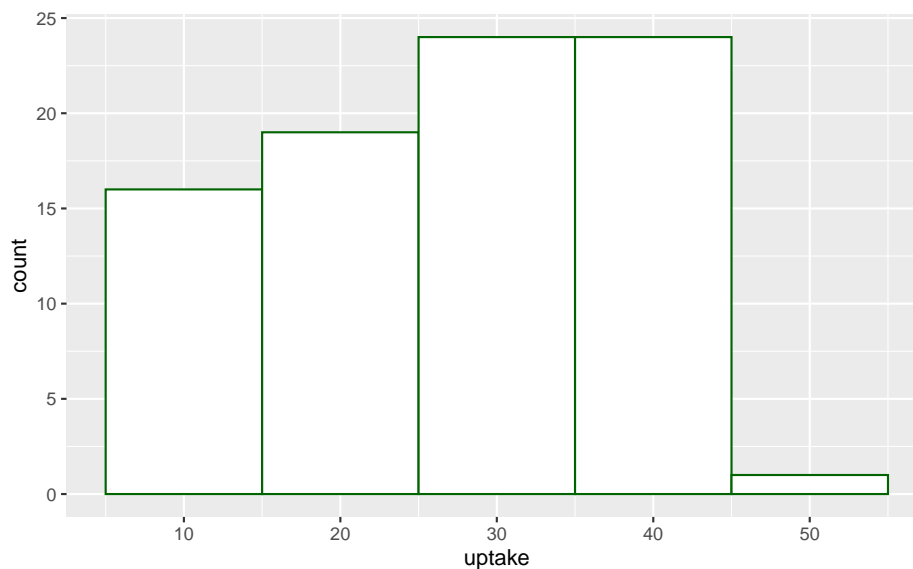
Ar argumentu `binwidth =` ir iespējams mainīt dalījuma klases lielumu, tādēji mainot klašu skaitu un histogrammas izskatu (?? attēls). Var arī norādīt vēlamo klašu skaitu ar argumenti `bins =`.

```
ggplot(CO2, aes(uptake)) + geom_histogram(binwidth = 10)
```



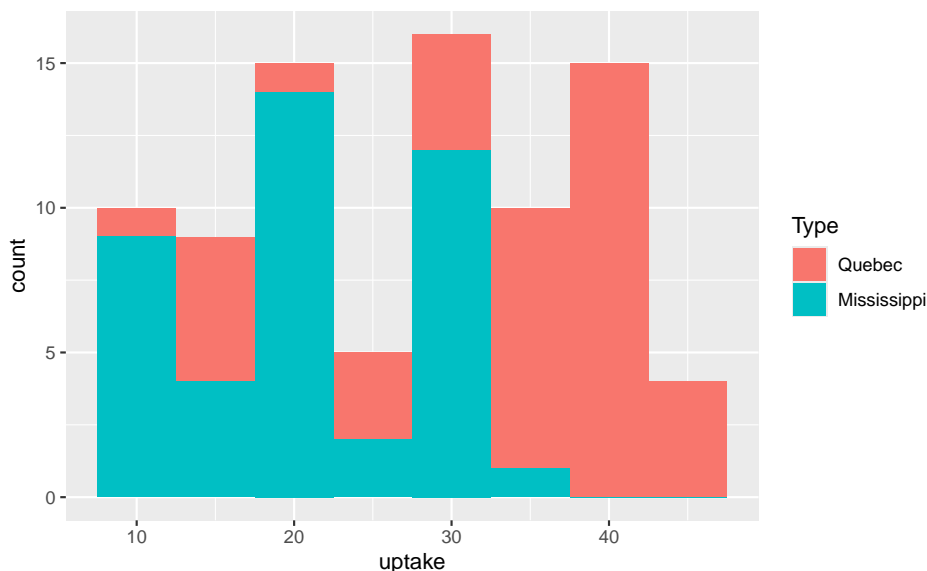
Tā kā histogrammā parādās stabiņi, tad tiem ir iespējams mainīt gan līnijas krāsu (`color =`), gan arī aizpildījumu (`fill =`) (?? attēls).

```
ggplot(CO2, aes(uptake)) +  
  geom_histogram(color = "darkgreen", fill = "white", binwidth = 10)
```



Norādot, ka aizpildījums ir atkarīgs no mainīgā, izveidosies histogramma, kurā katrs stabiņš sadalīts daļās atbilstoši novērojumu skaitam katrā no līmeņiem (3.22 attēls).


```
ggplot(CO2, aes(uptake, fill = Type)) +  
  geom_histogram(binwidth = 5)
```



Att. 3.22: Histogrammas, kur aizpildījums atkarīgs no mainīgā

3.9 `geom_abline()`, `geom_hline()` un `geom_vline()`

Ja attēlam ir nepieciešams pievienot diagonālu, horizontālu vai vertikālu līniju, tad jāizmanto attiecīgi `geom_abline()`, `geom_hline()` vai `geom_vline()`.

Diagonālas līnijas pievienošanai jānorāda divas vērtības: `slope` = (norāda slīpumu) un `intercept` = (norāda, kur krusto y asi, ja $x = 0$) (3.23 attēls).

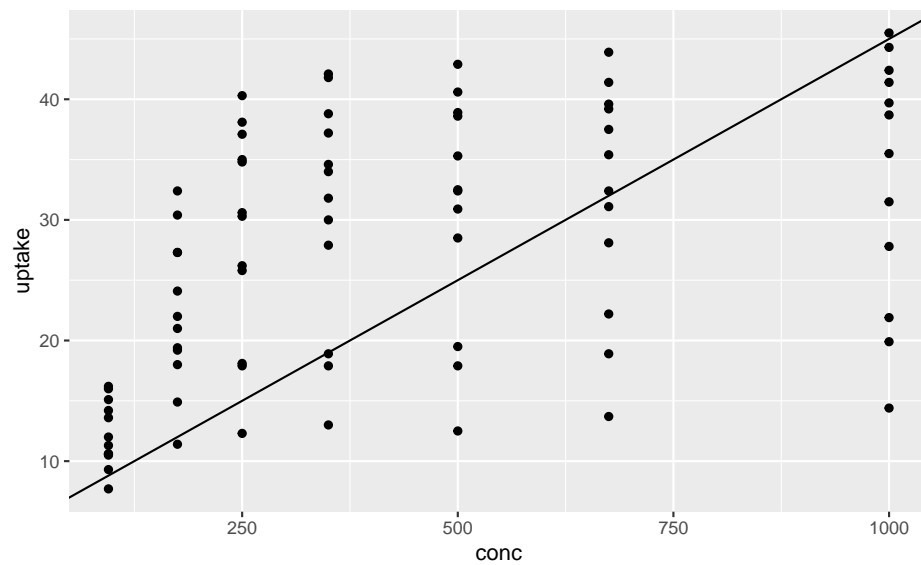
```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_abline(intercept = 5, slope = 0.04)
```

Horizontālas līnijas pievienošanai izmanto `geom_hline()`, kurai kā arguments jānorāda `yintercept` = (kādam y vērtībai atbilst līnija) (3.24 attēls).

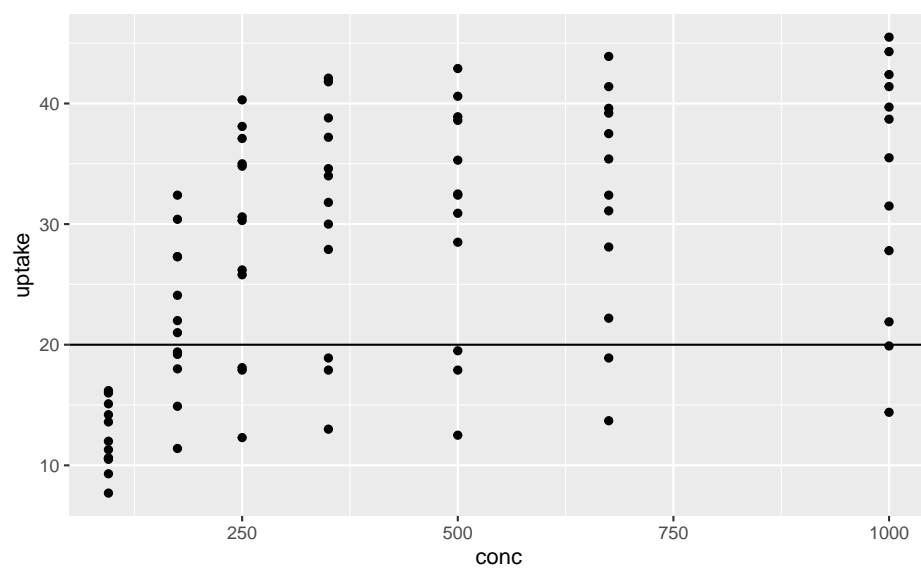
```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_hline(yintercept = 20)
```

Pie argumentu `yintercept` = var norādīt arī uzreiz vairākas vērtības, kā redzultātā parādīsies vairākas līnijas (3.25 attēls).

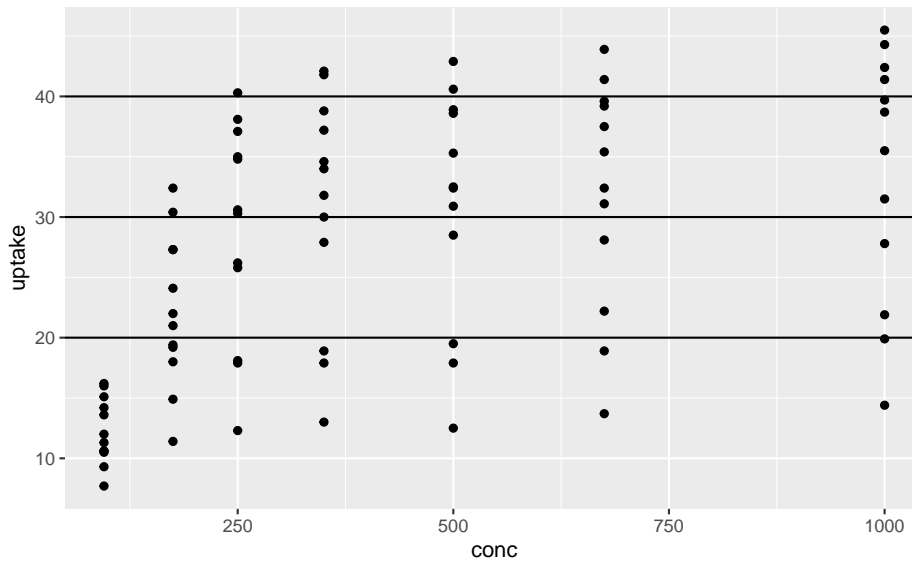
```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_hline(yintercept = c(20, 30, 40))
```



Att. 3.23: Izkļiedes diagramma ar pievienotu diagonālu līniju



Att. 3.24: Izkļiedes diagramma ar pievienotu horizontālu līniju



Att. 3.25: Izkļiedes diagramma ar pievienotām vairākām horizontālām līnijām

Līnijas novietojums var būt atkarīgs no kāda mainīgā datus, tikai šajā gadījumā argumentam `yintercept` = jāatrodas funkcijā `aes()` (3.26 attēls).

```
dati.papildus <- data.frame(limeni = c(10,20,30,40))
ggplot(CO2, aes(conc, uptake)) + geom_point() +
  geom_hline(data = dati.papildus, aes(yintercept = limeni))
```

Vertikālas līnijas pievieno ar funkciju `geom_vline()` un argumentu `xintercept` = (kādam x vērtībai atbilst līnija) (3.27 attēls). Pārējie darbības principi ir līdzīgi `geom_hline()`.

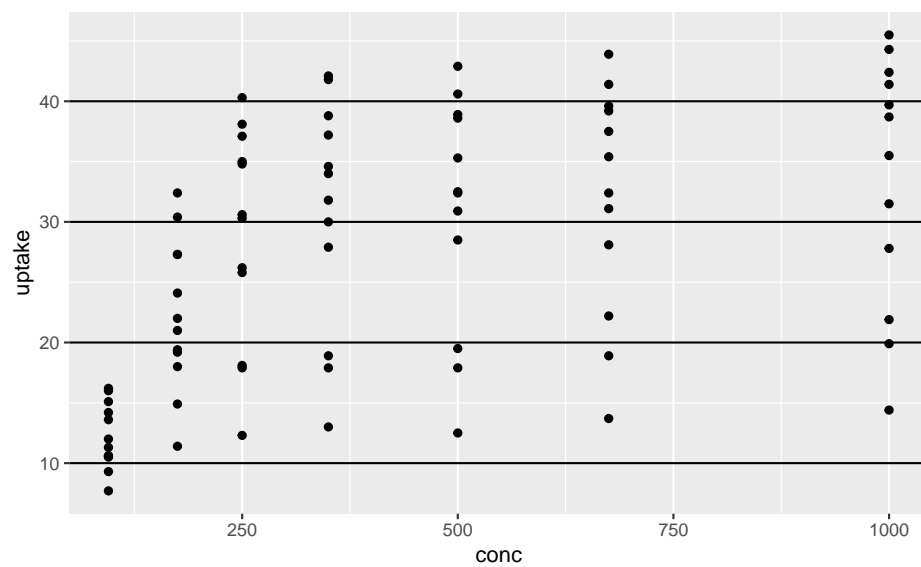
```
ggplot(CO2, aes(conc, uptake)) + geom_point() +
  geom_vline(xintercept = 500)
```

3.10 `geom_jitter()`

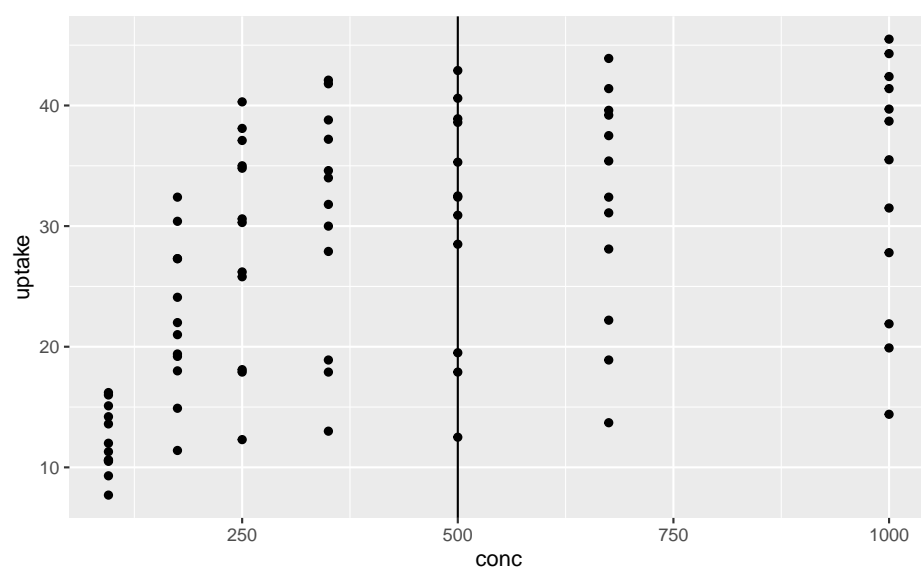
Gadījumos, kad nepieciešams izveidot izkļiedes (punktu) diagrammu, bet vērojama vērtību pārklāšanās (daudz identisku vērtību), var izmantot `geom_jitter()`, kur punktiem tiek veikta neliela nobīde x vai y (vai abu) ass virzienā, lai novērstu pārklāšanos. Šādu attēlošanas veidu sevišķi ērti izmantot, ja x vērtības ir kategorijas mainīgais, jo tad izkļiede notiek tikai x ass virzienā, bet y ass virzienā redzamas reālās vērtības (3.28 attēls).

```
ggplot(CO2, aes(Type, uptake)) + geom_jitter()
```

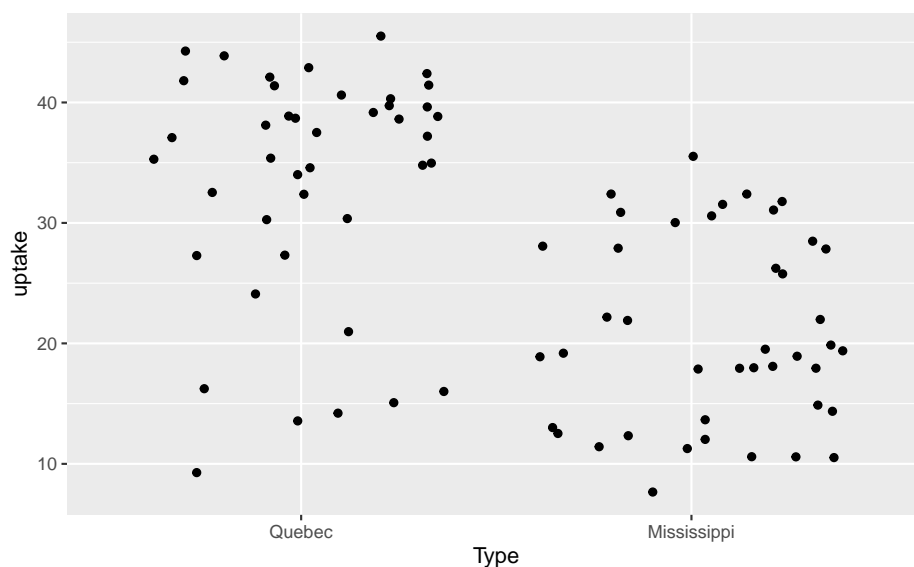
`geom_jitter()` ir labi izmantot kombinācijā ar `geom_boxplot()`, jo tādējādi



Att. 3.26: Izkliedes diagramma ar pievienotu horizontālu līniju



Att. 3.27: Izkliedes diagramma ar pievienotu vertikālu līniju



Att. 3.28: Izkļiedes diagramma, kurā x virzienā nejauši mainīts punktu izvietojums

gan parādās reālās vērtības, gan arī vērtību apkopojums (3.29 attēls).

```
ggplot(CO2, aes(Type, uptake)) + geom_boxplot() +
  geom_jitter()
```

`geom_jitter()` un `geom_boxplot()` var kombinēt arī gadījumos, kad vērtībampplitūdas diagramma ir sadalīta atbilstoši trešā mainīgā līmeņiem, bet šajā gadījumā papildus ir jānorāda arguments `position = position_jitterdodge()`, lai punktu izvietojums atbilstu reālajam vērtību sadalījumam pa līmeņiem (3.30 attēls).

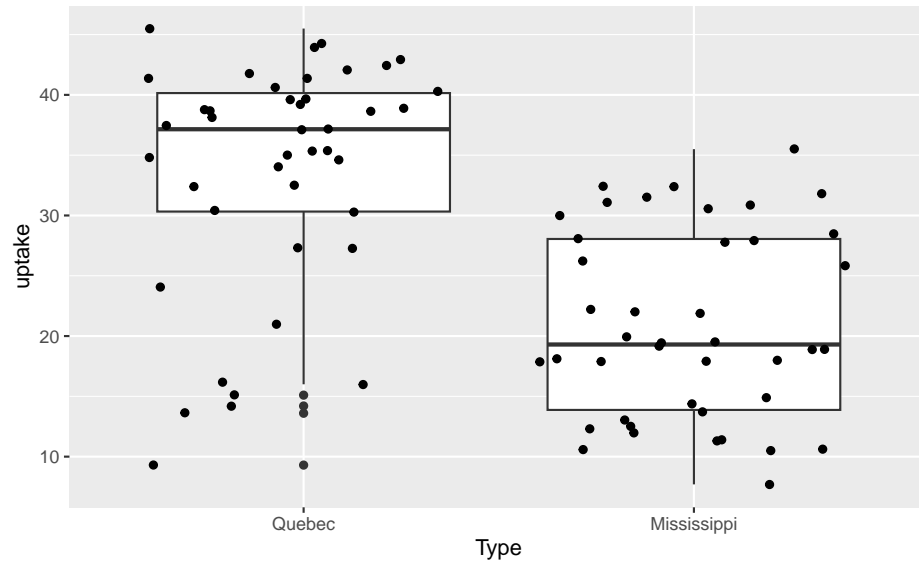
```
ggplot(CO2, aes(Type, uptake, fill = Treatment)) + geom_boxplot() +
  geom_jitter(position = position_jitterdodge())
```

3.11 geom_smooth()

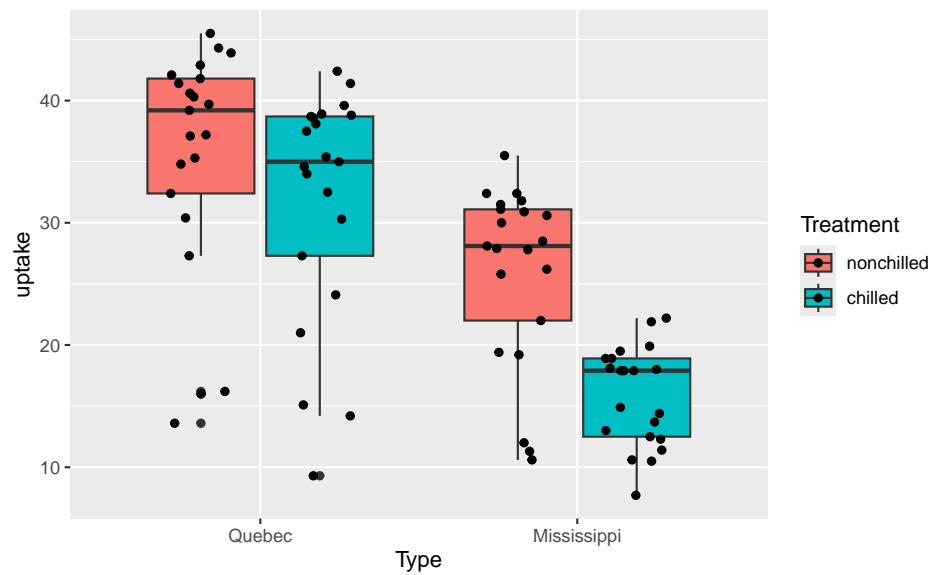
Ja ir vēlme attēlam pievienot trenda līniju, tad jāizmanto `geom_smooth()`. Pēc noklusējuma izveidojas izlīdzinātā trenda līnija un tās ticamības intervāls ar metodi loess vai gam (3.31 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +
  geom_smooth()
```

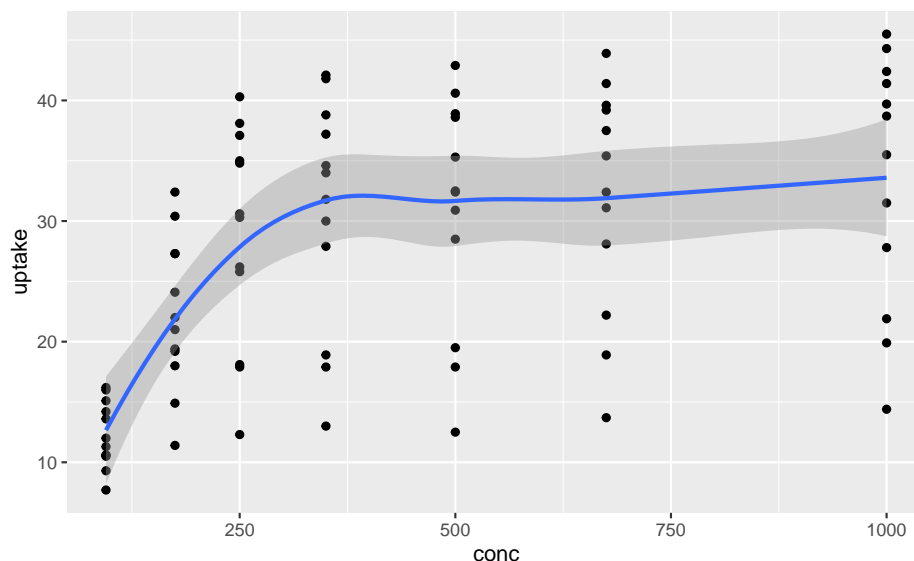
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Att. 3.29: Izklīdes diagrammas un vērtībamplitūdas diagrammas kombinācija



Att. 3.30: Izklīdes diagrammas un vērtībamplitūdas diagrammas kombinācija gadījumā, kad iesaistīts trešais mainīgais dalījuma līmeņiem



Att. 3.31: Izkliedes diagrammas ar pievienotu trenda līniju

Lineārās trenda līnijas pievienošanai, jānorāda arguments `method = "lm"` (3.32 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

Ar argumentu `se = FALSE` var noņemt ticamības intervālu (3.33 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_smooth(method = "lm", se = FALSE)
```

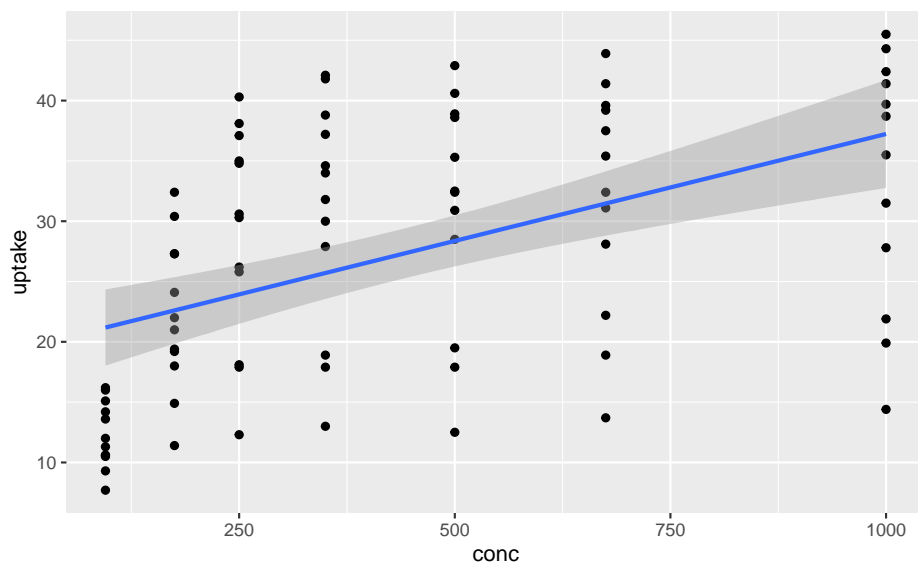
```
## `geom_smooth()` using formula = 'y ~ x'
```

Trenda līnijas krāsu maina ar argumentu `color =`, bet ticamības intervāla aizpildījumu ar argumentu `fill =`. Ja vienu vai abus no šiem argumentiem norāda `aes()` iekavās un tas ir atkarīgs no kāda mainīgā, tad trenda līnijas tiek izveidotas katram līmenim (3.34 attēls).

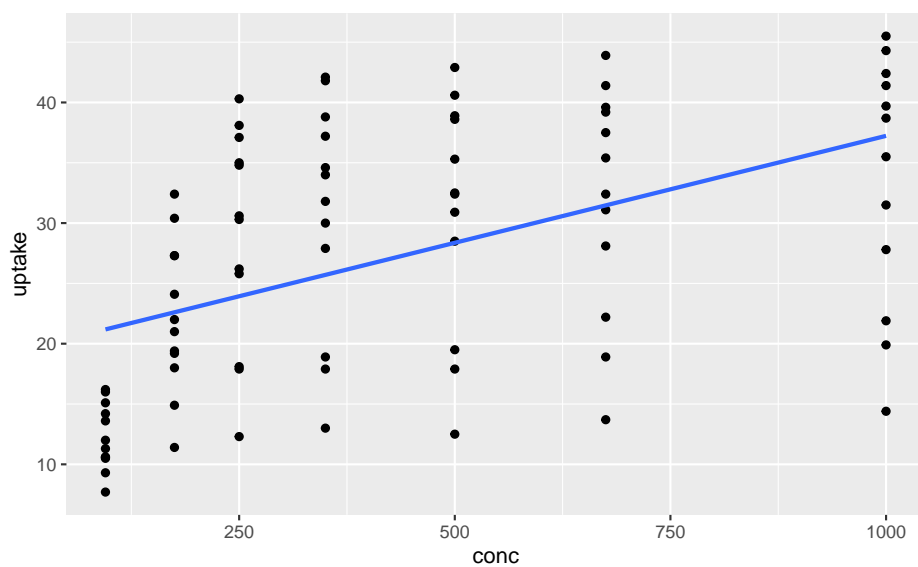
```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_smooth(method = "lm", aes(color = Type, fill = Type))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

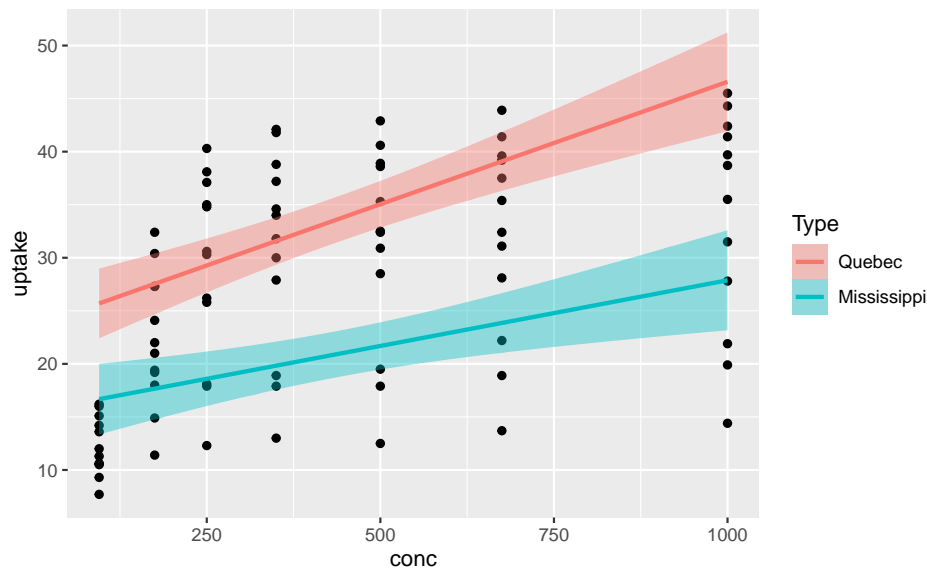
Trenda līniju var veidot ne tikai izmantojot esošo formulu `y ~ x`, bet arī izmantojot kādu citu saistību starp abiem mainīgajiem. Šajā gadījumā jāizmanto arguments `formula =` un jālieto apzīmējumi `x` un `y`, nevis oriģinālie mainīgo



Att. 3.32: Izklīdes diagrammas ar pievienotu lineāro trenda līniju



Att. 3.33: Izklīdes diagrammas ar pievienotu lineāro trenda līniju, toties bez ticamības intervāla



Att. 3.34: Izkliedes diagrammas ar pievienotu lineāro trenda līniju dažādiem līmeņiem

nosaukumi (3.35 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +
  geom_smooth(method = "lm", formula = y ~ x + I(x^2))
```

3.12 geom_violin()

Īpašs datu attēlošanas veids ir `geom_violin()`, kas sevī apvieno gan vērtībamplicitūdas diagrammas īpašības, gan arī blīvuma attēla īpašības. Pēc būtības tas ir blīvuma attēls, kurā vērtību blīvuma funkcijas attēlojums dots spoguļattēlā (3.36 attēls).

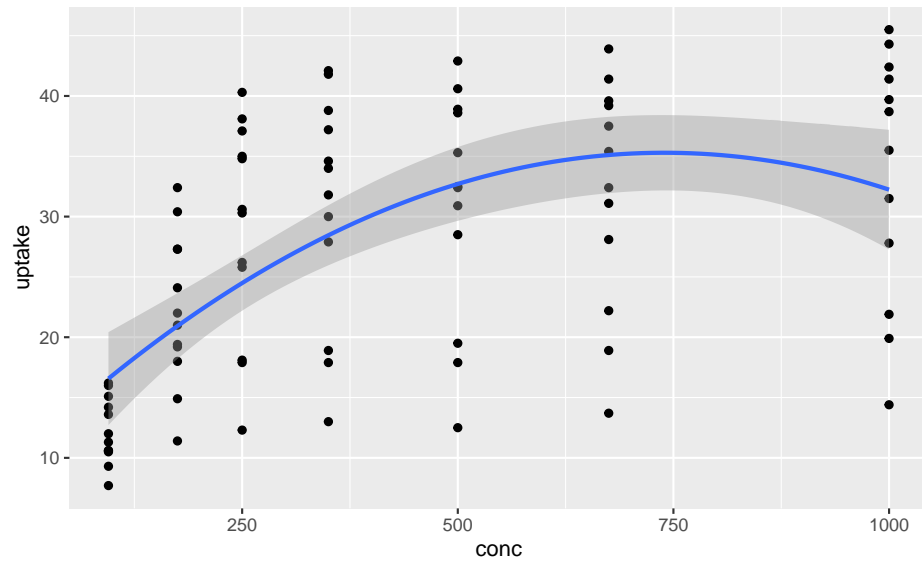
```
ggplot(CO2, aes(Type, uptake)) + geom_violin()
```

Ar argumentu `draw_quantiles` = attēlu var papildināt ar kvantiļu pozīcijām (3.37 attēls).

```
ggplot(CO2, aes(Type, uptake)) +
  geom_violin(draw_quantiles = c(0.25, 0.5, 0.75))
```

Pievienojot argumentu `fill` = funkcijā `aes()`, attēls tiek sadalīts katram no faktora līmeņiem (3.38 attēls).

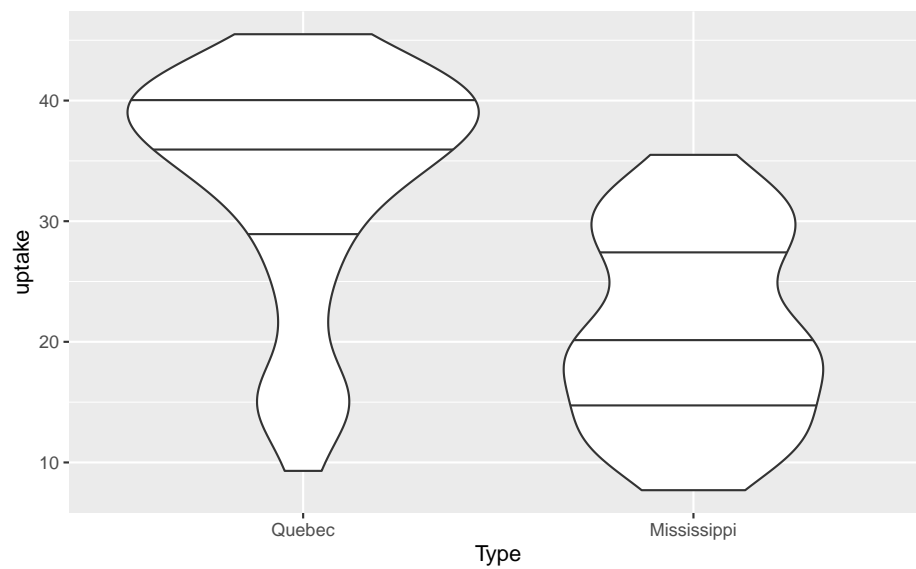
```
ggplot(CO2, aes(Type, uptake)) +
  geom_violin(aes(fill = Treatment))
```



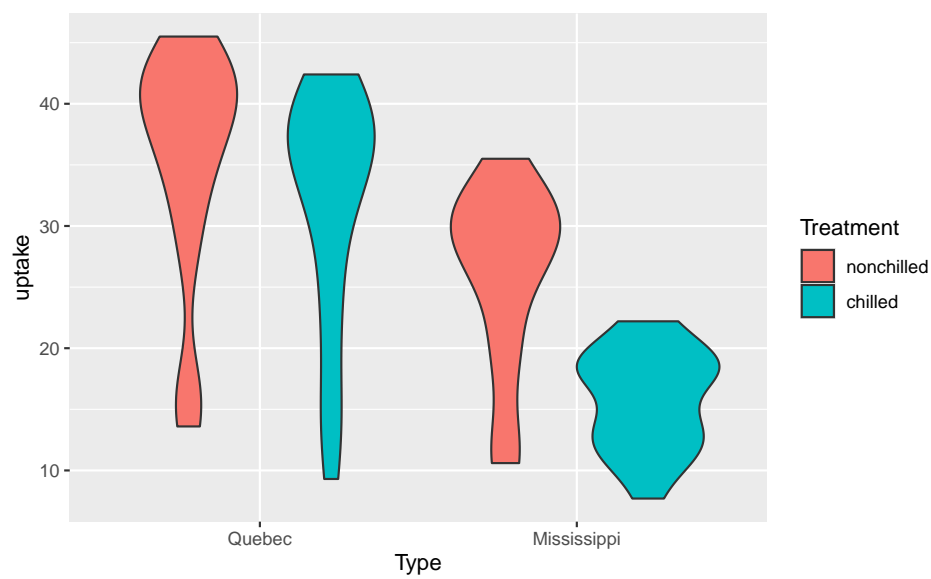
Att. 3.35: Izkļiedes diagrammas ar pievienotu īpašu trenda līniju



Att. 3.36: geom_violin() attēls



Att. 3.37: geom_violin() attēls ar pievienotām kvartilēm



Att. 3.38: geom_violin() attēls sadalīts pa līmeņiem

Nodaļa 4

Skalas

Uz x un y ass esošo vērtību, kā arī punktu, līniju, stabiņu krāsu, formu, izmēru utt. vērtību mainīšanai ir jāizmanto speciālas skalu maiņas funkcijas, kuru nosaukumi sastāv no trīs vārdiem. Visām funkcijām pirmais vārds ir `scale`, otrais vārds parāda, kāda veida skala tā ir - `x`, `y` vai attiecīgi krāsu (`color`), aizpildījuma (`fill`), līniju veida (`linetype`), simbolu veida (`shape`), izmēra (`size`) vai caurspīdīguma (`alpha`). Funkcijas nosaukumā trešais vārds norāda kāda veida vērtības ir izmantotas skalas izveidē - nepārtrauktas (`continuous`) vai diskretas (`discrete`), kā arī ir citi papildus veidi, piemēram, `manual` (vērtības nosaka manuāli), `gradient` (attiecas uz krāsām un aizpildījumiem).

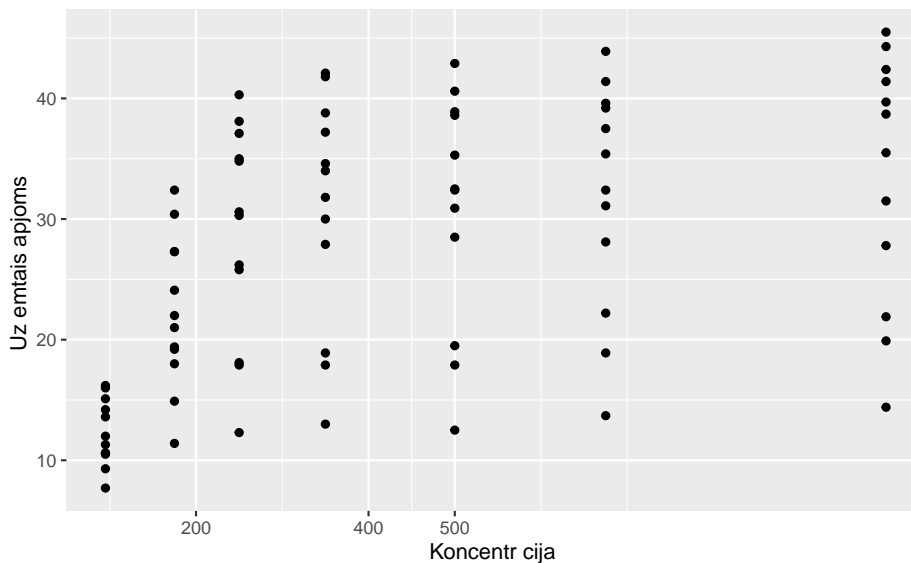
4.1 `scale__x__continuous()` un `scale__y__continuous()`

x un y ass vērtību maiņai, ja tās skaitliskas (nepārtrauktas), izmanto attiecīgi funkcijas `scale_x_continuous()` un `scale_y_continuous()`. Izmantojot šīs funkcijas var mainīt asu parakstus (arguments `name =`), pozīcijas, kurās parādās skaitļi (`breaks =`) (4.1 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  scale_x_continuous("Koncentrācija", breaks = c(200, 400, 500)) +  
  scale_y_continuous("Uzņemtais apjoms")
```

Ar argumentu `limits =` ir iespējams mainīt katras ass garumu, bet jāņem vērā, ka gadījumā, ja jaunais garums būs mazāks nekā vērtību amplitūda, tad vērtības ārpus ass garuma tiks izslēgtas no attēla (to parāda arī brīdinājums par izslēgtām vērtībām), ietekmējot attēlojumu (4.2 attēls). Tas īpaši attiecas uz stabiņu attēliem, vai attēliem ar trenda līniju.

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  scale_x_continuous(limits = c(200,600)) +  
  scale_y_continuous(limits = c(0,50))
```



Att. 4.1: Nepārtraukto asu piemērs

```
## Warning: Removed 48 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

y un x asi ir iespējams arī pārvietot attiecīgi uz labo pusi vai uz augšu, norādot argumentu `position = (4.3 attēls)`.

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +
  scale_y_continuous(position = "right") +
  scale_x_continuous(position = "top")
```

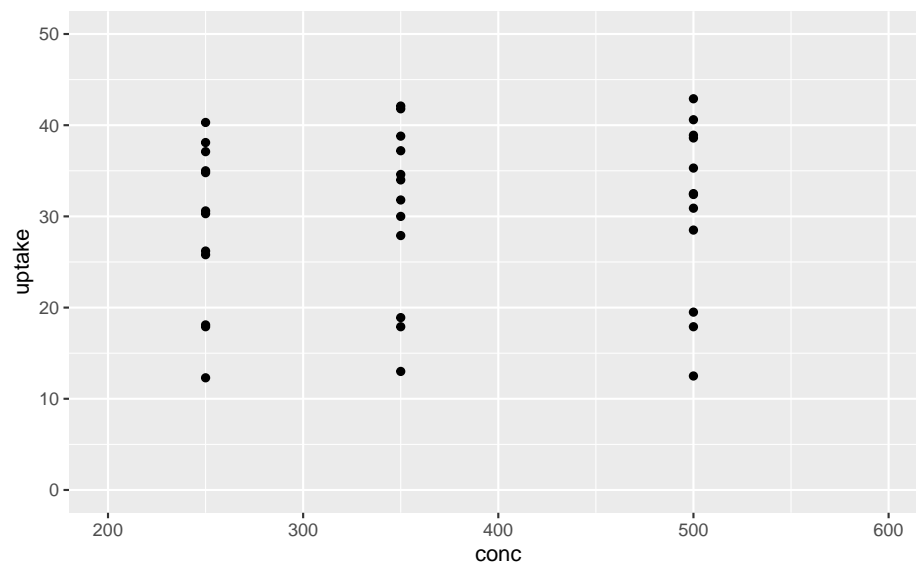
Ar argumentu `sec.axis =` gan x, gan y asij ir iespējams izveidot otro asi, bet tikai ar nosacījumu, ka otrā ass ir tieša pamatass transformāciju (4.4 attēls). Tas nozīmē, ka nevar izveidot otru asi, kas parāda pavisam citas vērtības.

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +
  scale_y_continuous(sec.axis = sec_axis(~./100, name = "Otrā y ass"))
```

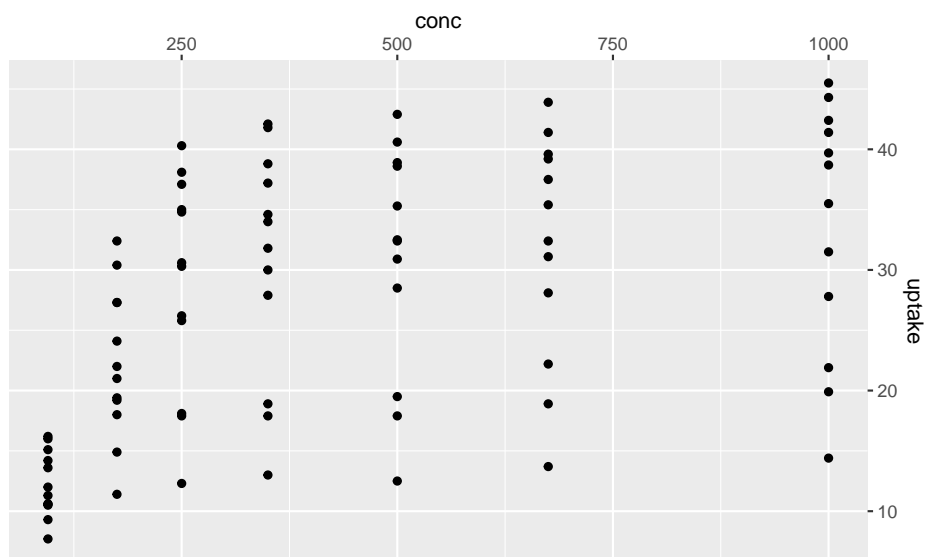
4.2 scale_x_discrete() un scale_y_discrete()

Gadījumos, kad uz x vai y ass attēlotas kategorijas mainīgā vērtības, ir jāizmanto attiecīgi funkcijas `scale_x_discrete()` un `scale_y_discrete()`, lai mainītu šo asu izskatu.

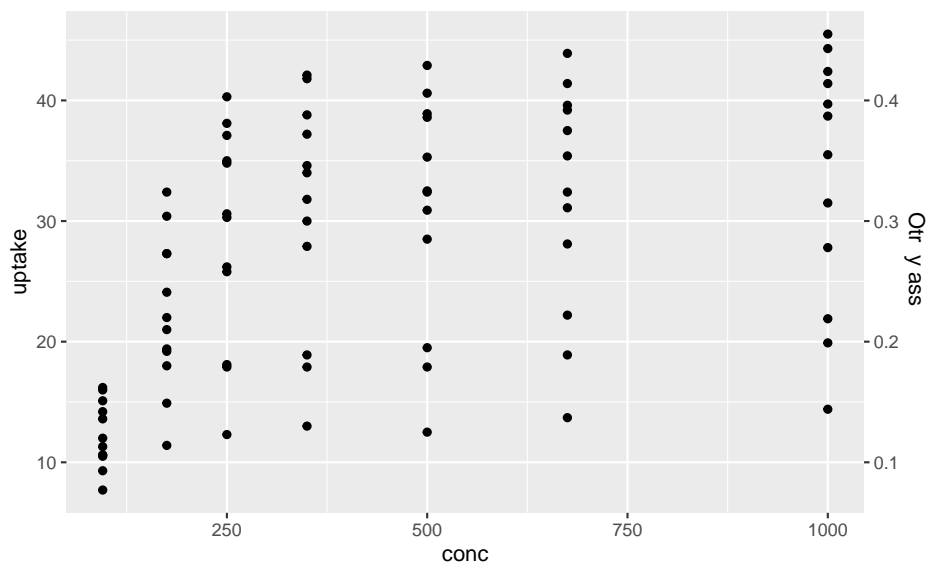
Ar argumentu `limits =` ir iespējams norādīt, kuras tieši vērtības attēlot uz ass (atmest kādu no līmeņiem) (4.5 attēls).



Att. 4.2: Izmainītas nepārtrauktās ass piemērs



Att. 4.3: Pārvietotas x un y asis



Att. 4.4: Attēls ar otru y asi, kas ir pirmās transformācija

```
ggplot(mpg, aes(drv, hwy)) + geom_boxplot() +
  scale_x_discrete(limits = c("f", "r"))
```

```
## Warning: Removed 103 rows containing missing values or values outside the scale range
## (`stat_boxplot()`).
```

Arguments `limits` = ļauj arī mainīt secību, kādā parādās līmeņi pie atbilstošās ass (4.6 attēls).

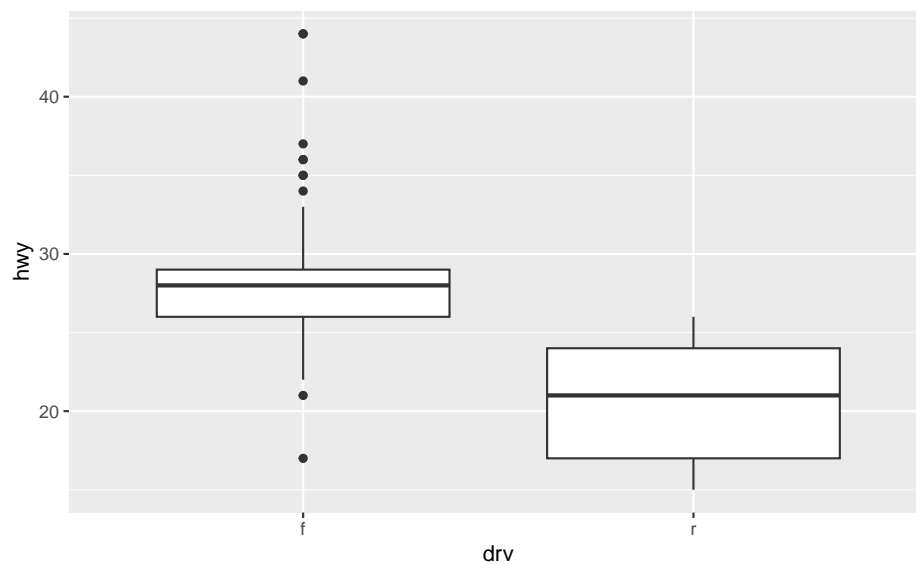
```
ggplot(mpg, aes(drv, hwy)) + geom_boxplot() +
  scale_x_discrete(limits = c("r", "4", "f"))
```

Līmeņu nosaukumu maiņai izmanto argumentu `labels` =, kur jānorāda jaunie līmeņu nosaukumi tādā secībā, kā tie parādās pie atbilstošās ass, vai arī jānorāda vecais un jaunais nosaukums un tad secībai nav nozīmēs (4.7 attēls). Jāņem vērā, ka arguments `labels` = nemaina līmeņu attēlojuma secību, pat ja mainīta kārtība to nosaukumiem.

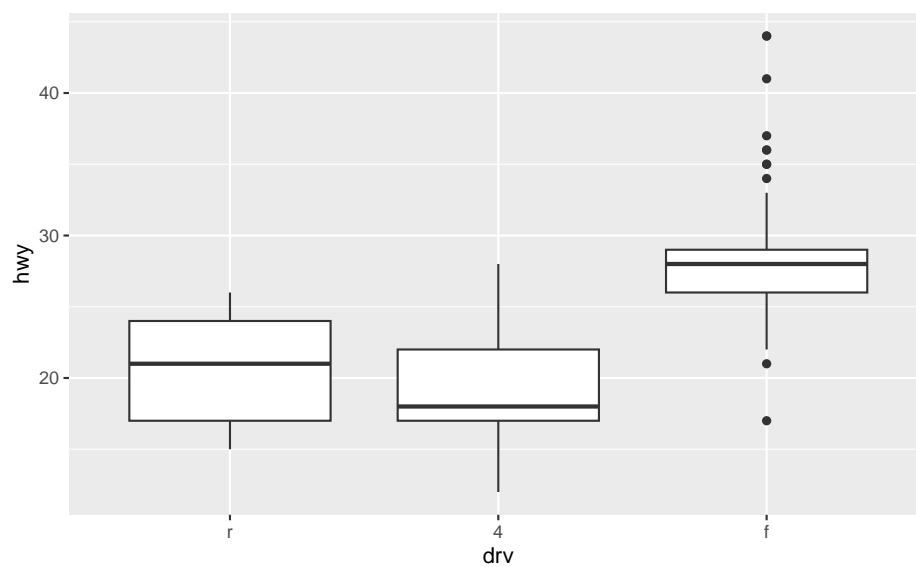
```
ggplot(mpg, aes(drv, hwy)) + geom_boxplot() +
  scale_x_discrete(labels = c("4" = "Četri", "r" = "Pakaļējie", "f" = "Priekšējie"))
```

4.3 `scale_color_...()` un `scale_fill_...()`

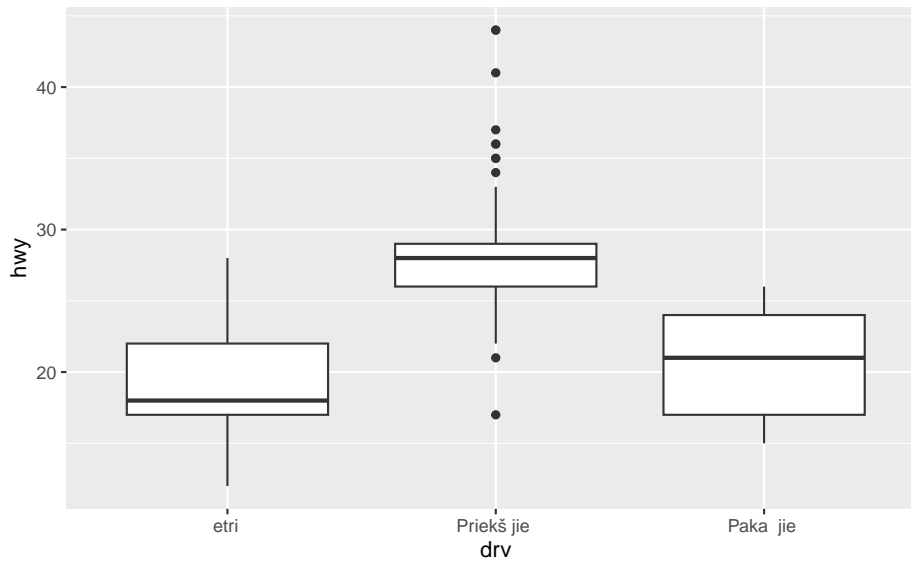
Gadījumos, kad formu (punktu, līniju) krāsa ir atkarīga no kāda mainīgā ar argumentu `color` =, šo krāsu un izveidoto leģendu mainīšanai ir jāizmanto funkcijas



Att. 4.5: Attēls ar kategorijas x asi, kur attēloti tikai daži līmeņi



Att. 4.6: Attēls ar kategorijas x asi, kur mainīta līmeņu secība

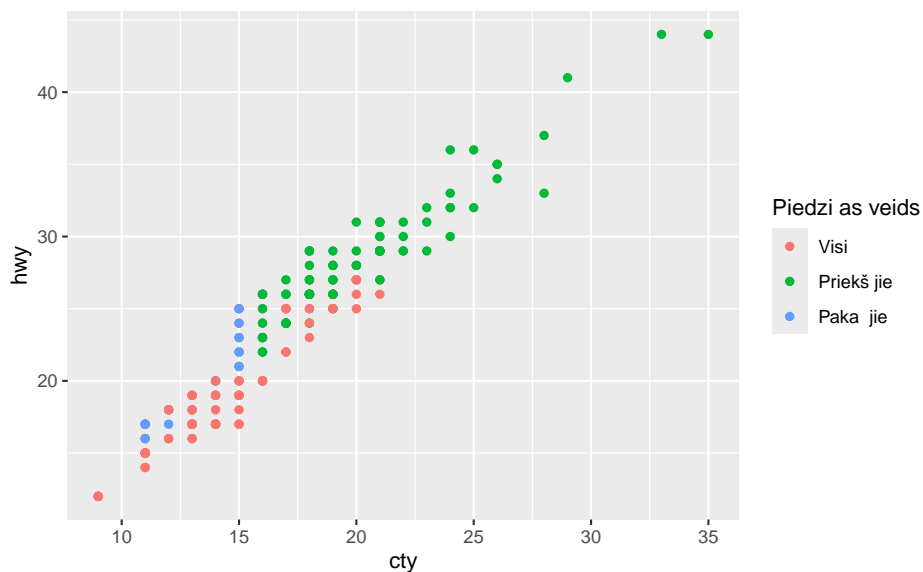


Att. 4.7: Attēls ar kategorijas x asi, kur mainīti līmeņu nosaukumi

no grupas `scale_colur...()`. Analogiski, ja aizpildījuma krāsa ir noteikta ar argumentu `fill =`, tad jāizmanto funkcijas no grupas `scale_fill..()`

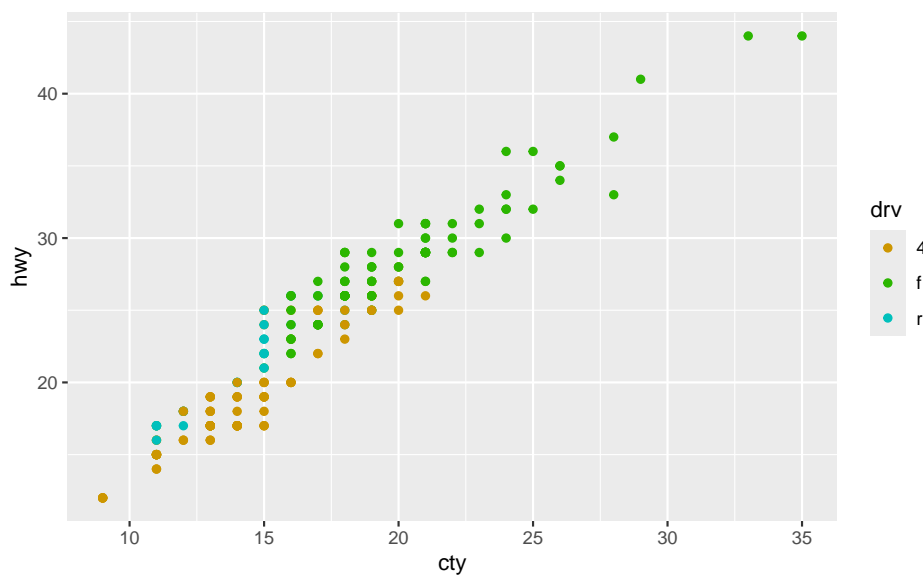
Krāsu (aizpildījuma) skalas veids ir atkarīgs no mainīgā veida, kas izmantots nosakot krāsu. Ja tas ir diskrets (kategorijas), tad jāizmanto funkcija `scale_color_discrete()` vai `scale_fill_discrete()`, kurā kā citās kategorijas skalās var norādīt nosaukumu, kas parādīsies leģendā (`name =`), kādā veidā nosaukt līmeņus leģendā (`labels =`) un kurus līmeņus parādīt attēlā (`limits =`) (?? attēls).

```
ggplot(mpg, aes(cty, hwy, color = drv)) + geom_point() +
  scale_color_discrete(name = "Piedzīņas veids",
    labels = c("Visi", "Priekšējie", "Pakaļējie"))
```



Pašu krāsu (aizpildījumu) mainīšanai var izmantot vairākas pieejas, piemēram, izmantot funkciju `scale_color_hue()` vai `scale_fill_hue()` un mainīt trīs parametru vērtības: `l =`, `h =` un `c =`, kas attiecīgi apzīmē *luminance*, *hues* un *chroma* (4.8 attēls).

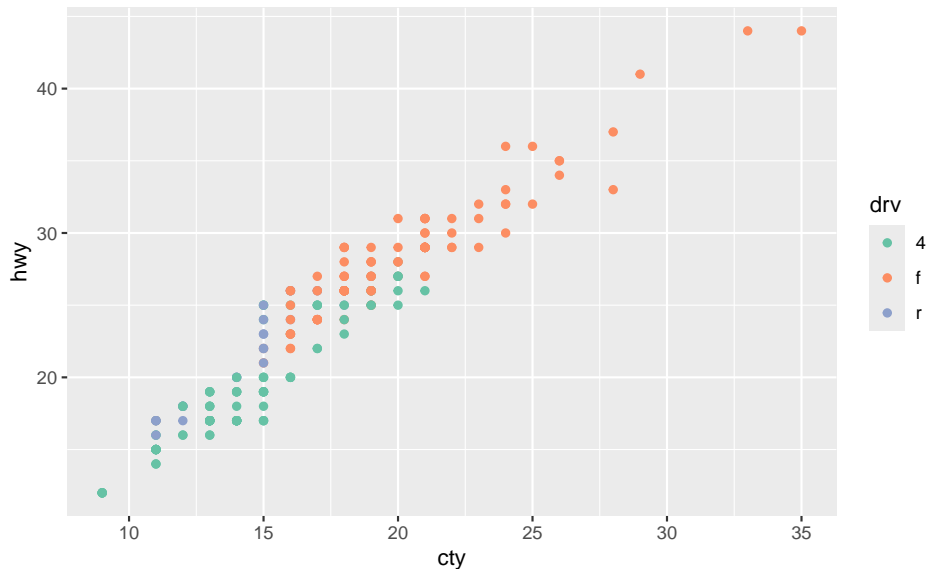
```
ggplot(mpg, aes(cty, hwy, color = drv)) + geom_point() +
  scale_color_hue(h=c(60, 190))
```



Att. 4.8: Izklādes attēls, kurā mainītas krāsu gamma

Gatavu krāsu palešu izmantošanai, jāizmanto funkcija `scale_color_brewer()` vai `scale_fill_brewer()`, norādot vēlamo krāsu paleti (`palette =`) (4.9 attēls).

```
ggplot(mpg, aes(cty, hwy, color = drv)) + geom_point() +
  scale_color_brewer(palette = "Set2")
```



Att. 4.9: Izklides attēls ar noteiktu krāsu paleti

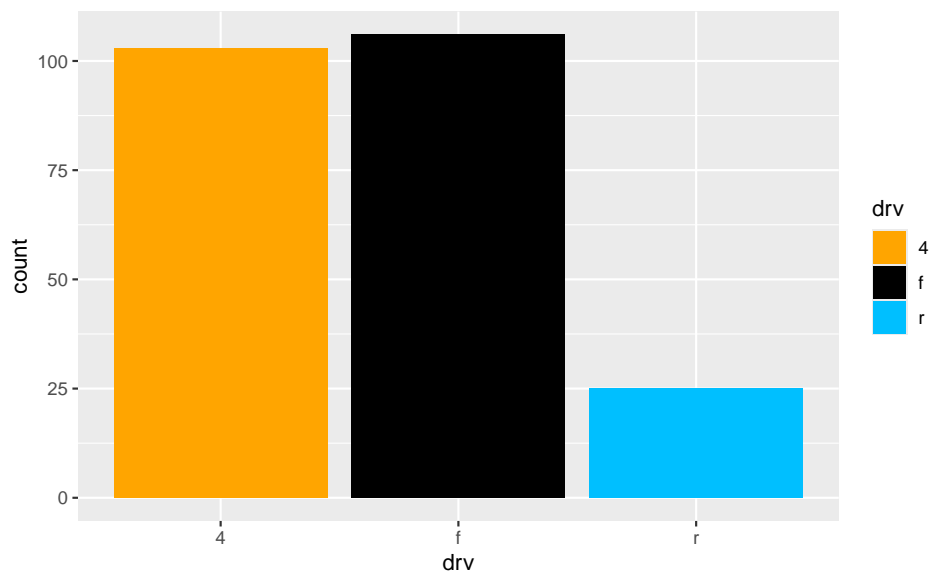
Pilnīgai kontrolei pār attēla krāsām jāizmanto `scale_color_manual()` vai `scale_fill_manual()`, kas ļauj izmantot paša izvēlētās krāsas (arguments `values =`), rakstot to angļiskos nosaukumus vai izmantojot Hex kodus. Krāsu skaitam ir jābūt identiskam, līmeņu skaitam datos (4.10 attēls).

```
ggplot(mpg, aes(drv, fill = drv)) + geom_bar() +
  scale_fill_manual(values = c("orange", "black", "deepskyblue"))
```

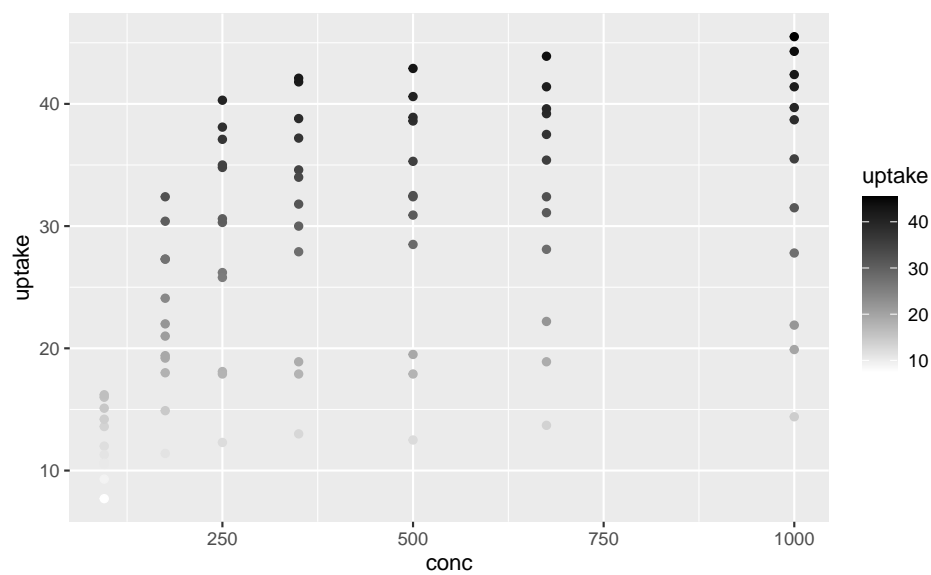
Ja krāsa vai aizpildījums ir atkarīgs no skaitliska mainīgā, tad to mainīšanai var izmantot, piemēram, funkcijas `scale_color_gradient()`, `scale_color_gradient2()` un `scale_color_gradientn()`, vai to analogus `scale_fill_gradient()`, `scale_fill_gradient2()` un `scale_fill_gradientn()`. Šīs funkcijas attiecīgi noteiks krāsu kā pāreju no zemākā uz augstāko toni, kā pāreju starp divām atšķirīgām krāsām, vai arī kā gradientu no vairāk kā divām krāsām (4.11 attēls).

```
ggplot(CO2, aes(conc, uptake, color = uptake)) + geom_point() +
  scale_color_gradient(low="white", high = "black")
```

Norādot divas ļoti atšķirīgas krāsas, veidosies pāreja no vienas krāsas uz otru

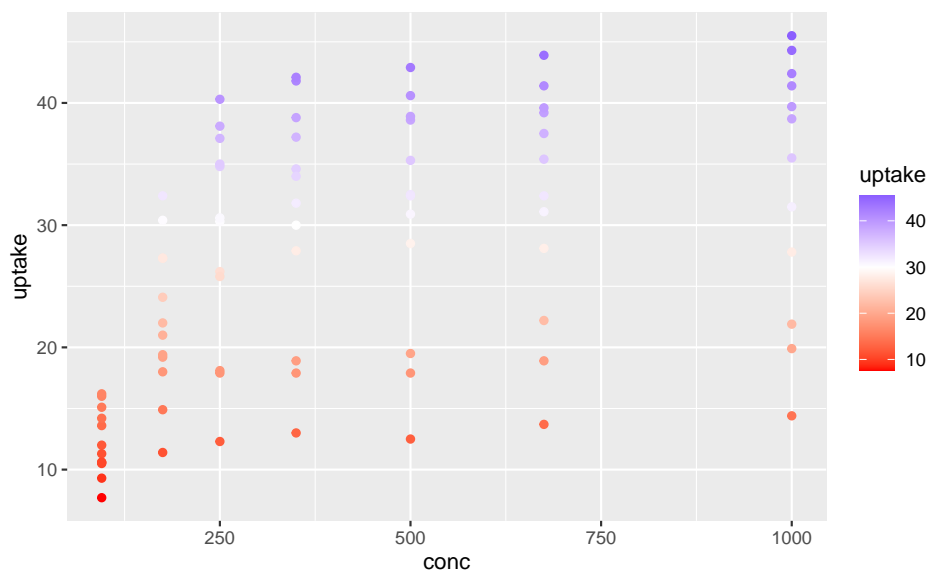


Att. 4.10: Stabiņu attēls ar manuāli noteiktām aizpildījuma krāsām



krāsu ar balto pa vidu. Pēc noklusējuma baltā krāsa atbildīs vērtībām 0, bet, ja nepieciešams cits viduspunkts, tas jānorāda ar argumentu `midpoint =` (4.12 attēls).

```
ggplot(CO2, aes(conc, uptake, color = uptake)) + geom_point() +
  scale_color_gradient2(low="red", high = "blue", midpoint = 30)
```

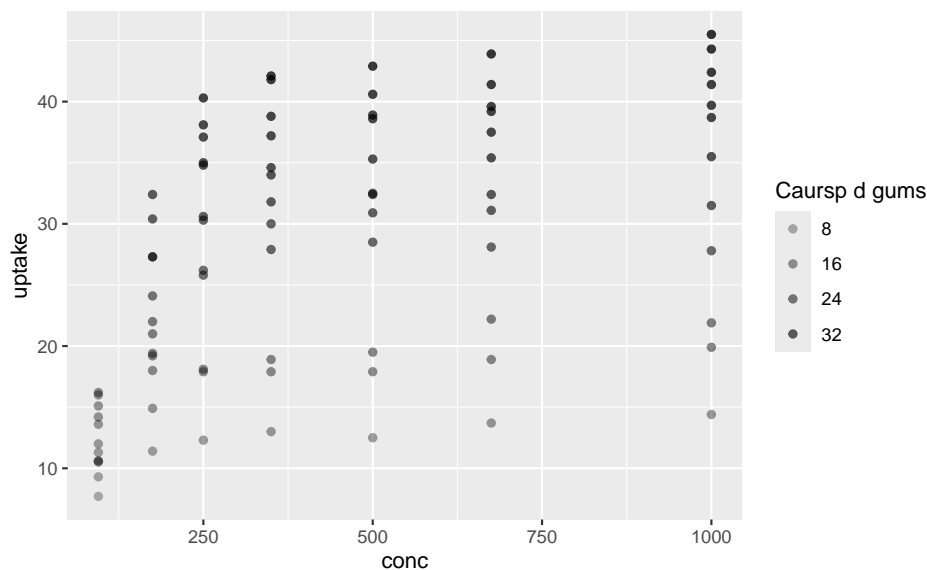


Att. 4.12: Izklīdes attēls, kurā krāsa mainās kā gradients no sarkanās uz zilo

4.4 `scale_alpha_...()`

Ja veidojot attēlu funkcijā `aes()` ir izmantots arguments `alpha =`, tad caurspīdīguma vērtību un atbilstošo leģendu mainīšanai izmanto funkcijas `scale_alpha_continuous()`, `scale_alpha_discrete()` un `scale_alpha_manual()`. Visās funkcijās var norādīt leģendas nosaukumu (`name =`) un līmeņu nosaukumus (`labels =`). Papildus var norādīt arī argumentu `range =`, kas nosaka mazāko un lielāko vērtību caurspīdīguma skalai (4.13 attēls).

```
ggplot(CO2, aes(conc, uptake, alpha = uptake)) + geom_point() +
  scale_alpha_continuous("Caurspīdīgums", breaks = c(8, 16, 24, 32),
    range = c(0.3, 0.8))
```



Att. 4.13: Izkliedes attēls, kurā krāsa mainās kā gradients no sarkanās uz zilo

4.5 `scale_linetype_...()`

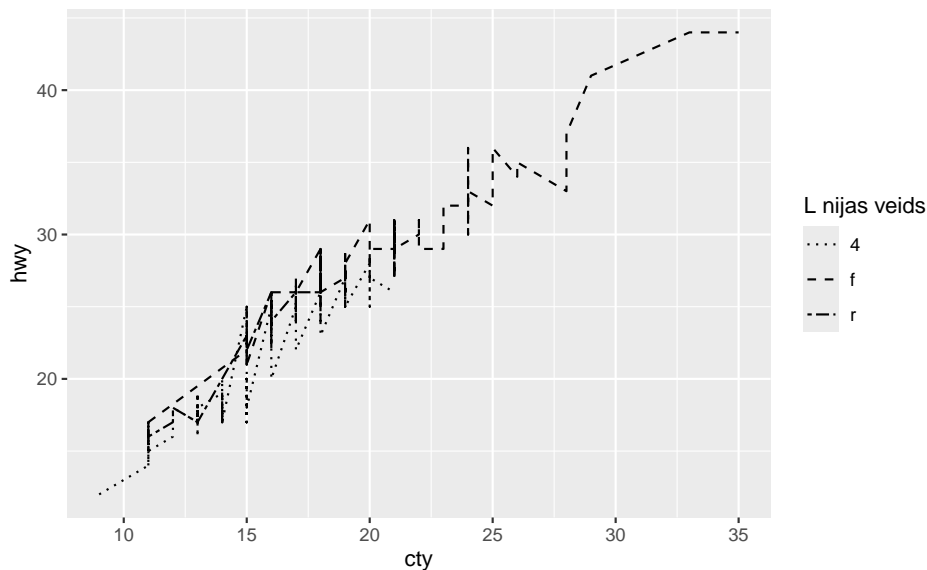
Attēlā izmantoto līniju veida ietekmēšanai izmanto funkcijas `scale_linetype_discrete()` un `scale_linetype_manual()`. Ar pirmo funkciju var ietekmēt tikai leģendu un līmeņu nosaukumus, bet manuālā skala ļauj jau definēt konkrētos līniju veidus katram no līmeņiem, tos norādot kā skaitļus no 1 līdz 6, vai to angliskos nosaukumus (4.14 attēls).

```
ggplot(mpg, aes(cty, hwy, linetype = drv)) +
  geom_line() +
  scale_linetype_manual("Līnijas veids",
    values= c(3, 2, 6))
```

4.6 `scale_shape_...()`

Izkliedes attēlos (punktu attēlos) simbolu maiņai izmanto funkcijas `scale_shape_discrete()` un `scale_shape_manual()`. Automātiski var tikt izveidots attēls ar sešiem dažādiem simbolu veidiem. Ja līmeņu daudzums ir lielāks par seši, tad tikai sešiem līmeņiem parādīsies simboli un brīdinājums, kā arī pārējie līmeņi attēlā neparādīsies. Šajā gadījumā obligāti jāizmanto funkcija `scale_shape_manual()` un jānedefinē savi simbola veidi.

Ar `scale_shape_discrete()` var definēt leģendas nosaukumu un līmeņu nosaukumus (4.15 attēls).



Att. 4.14: Līniju attēls ar mainītiem līniju veidiem

```
ggplot(CO2, aes(conc, uptake, shape = Type)) + geom_point() +
  scale_shape_discrete("Tips", labels = c("Pirmais", "Otrais"))
```

Funkcijā `scale_shape_manual()` papildus nāk arguments `values =`, kas ļauj definēt izmantojamo simbola veidu (4.16 attēls).

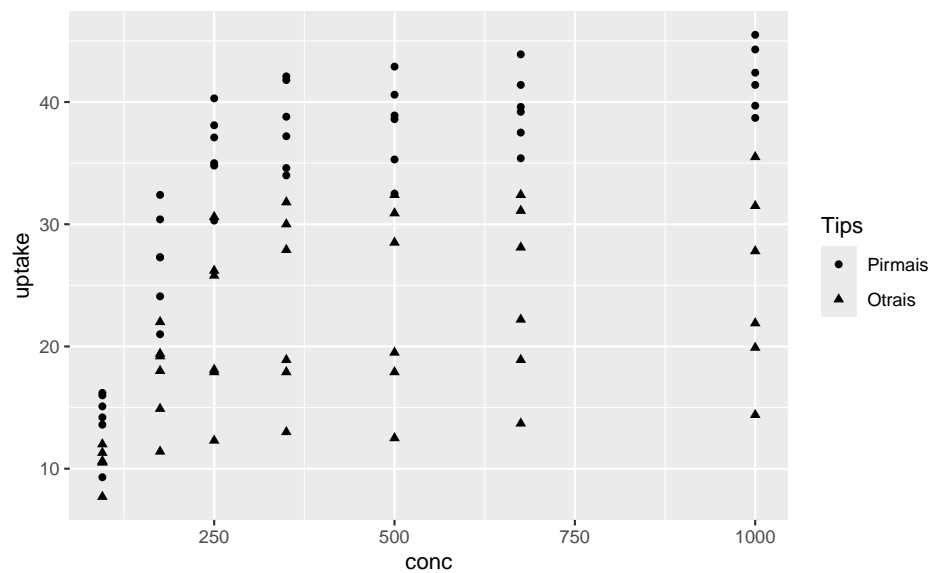
```
ggplot(CO2, aes(conc, uptake, shape = Type)) + geom_point() +
  scale_shape_manual("Tips", labels = c("Pirmais", "Otrais"),
    values = c(11, 23))
```

4.7 scale_size_...()

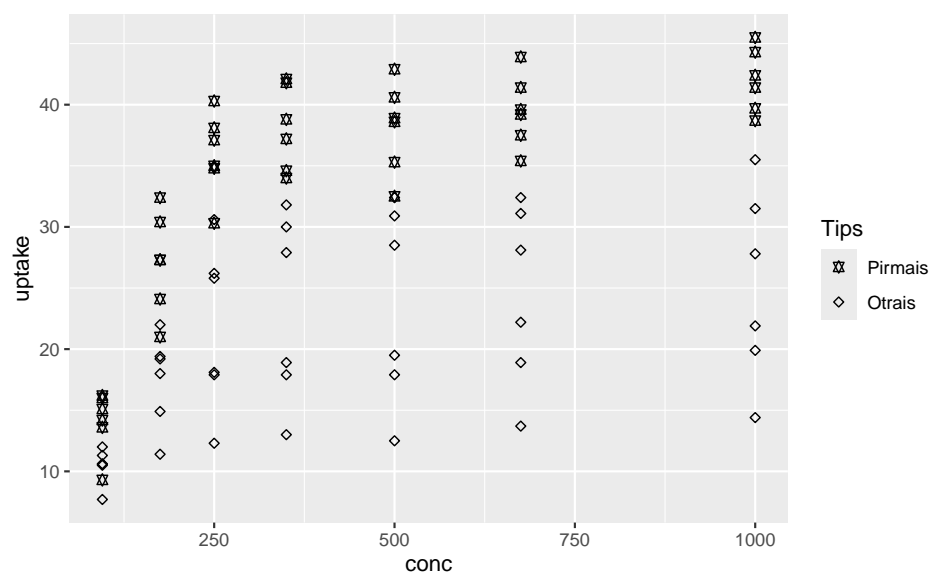
Gadījumos, kad kādai formai attēlā mainās lielums atbilstoši argumentam `size =`, ir jāizmanto kāda no `scale_size_...()` funkcijām. Ar funkciju `scale_size()` var definēt ne tikai leģendas nosaukumu un dalījuma līmeņus, bet arī ar argumentu `range =` izmēru robežas kādās pārvērst vērtības (4.17 attēls).

```
ggplot(CO2, aes(conc, uptake, size = uptake)) + geom_point() +
  scale_size("Izmēri", range = c(2, 8))
```

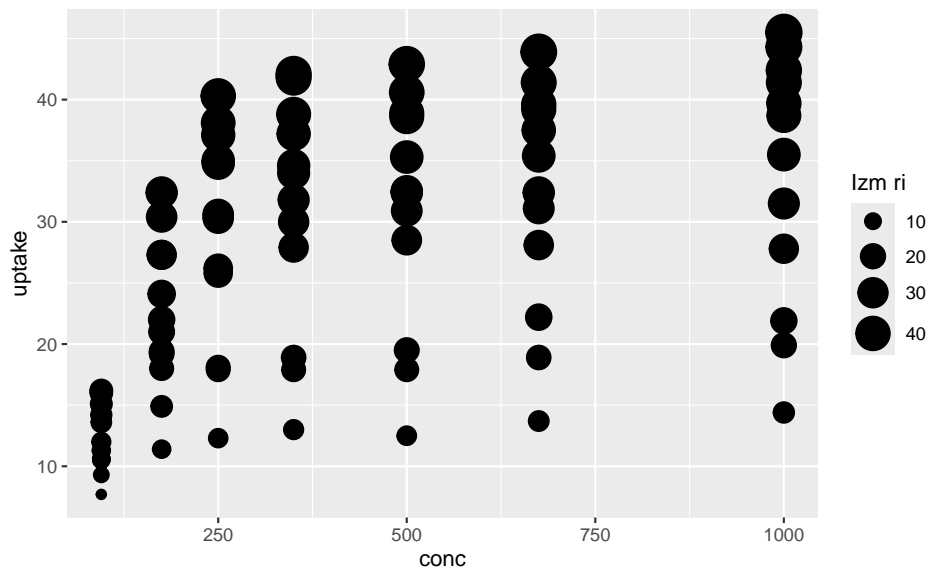
Ja grib precīzi definēt izmērus tā, ka 0 vērtība atbilst punkta platībai (izmēram) 0, tad ir jāizmanto funkcija `scale_size_area()` (4.18 attēls).



Att. 4.15: Izkliedes attēls, kurā forma ir atkarīga no kategorijas mainīgā



Att. 4.16: Izkliedes attēls ar mainītu simbolu veidu



Att. 4.17: Izkliedes attēls, kurā mainās punktu izmērs

```
ggplot(CO2, aes(conc, uptake, size = uptake)) + geom_point() +
  scale_size_area()
```

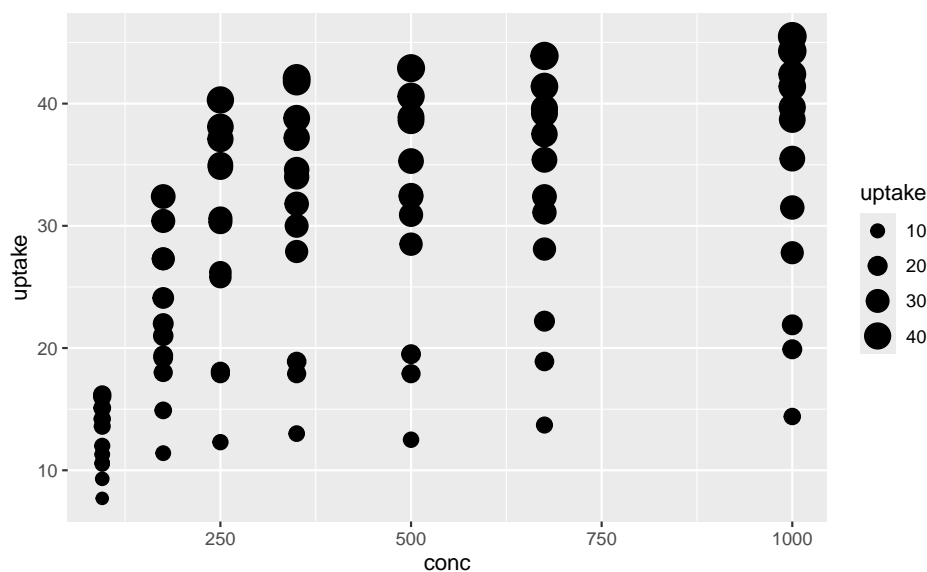
4.8 Identitātes skalas

Ir vesela grupa skalu funkciju, kuru nosaukumā pēdējais vārds ir `identity`. Šīs skalas izmanto gadījumos, kad datos ir kolonna, kuras vērtības ir jāizmanto kā reālās attēlojamā veida iespējamās vērtības, nevis sistēmai pašai jāizdomā attēlojums šim mainīgajam. Piemērs būtu, ka datos ir kolonna ar krāsu nosaukumiem, kurus tiešā veidā ir arī jāattēlo.

Piemēram izveidota datu tabula, kurā ir kolonnas `izmers` un `krasa` (ar reāliem krāsu nosaukumiem).

```
df <- data.frame(x = 1:3,
                 y = 1:3,
                 izmers = 4:6,
                 krasa = c("green", "yellow", "brown"))
df
```

```
##   x y izmers  krasa
## 1 1 1      4  green
## 2 2 2      5 yellow
## 3 3 3      6  brown
```



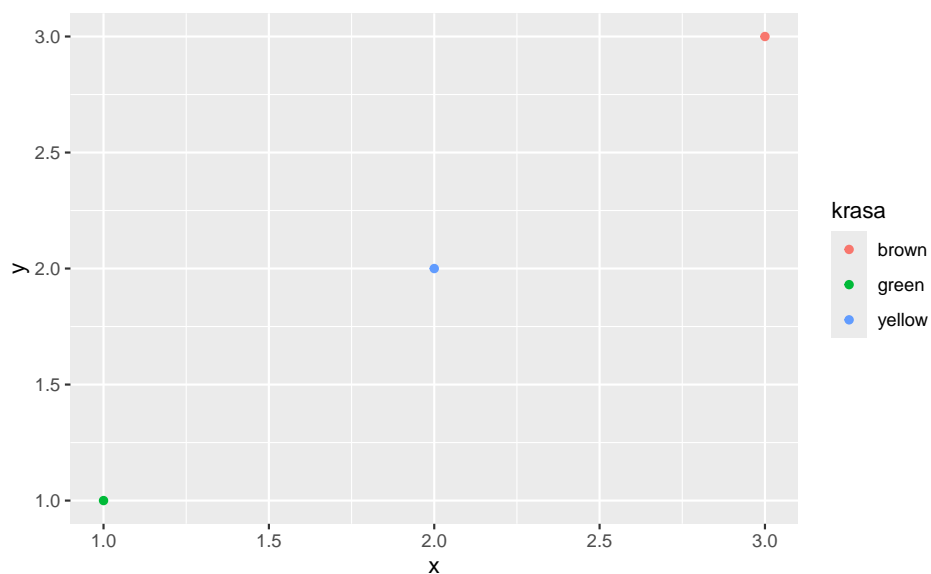
Att. 4.18: Izkliedes attēls, kurā mainās punktu izmērs sākot ar 0 vērtību

Izmantojot piemēra datus izkliedes attēlā un nosakot, ka punktu krāsa ir atkarīga no mainīgā `krasa`, redzams, ka izveidotajā attēlā un leģendā nosaukumi un krāsas nesakrīt (4.19 attēls), jo `ggplot2` sistēma tos uztver kā vienkāršus nosaukumus, nevis reālus krāsu nosaukumus.

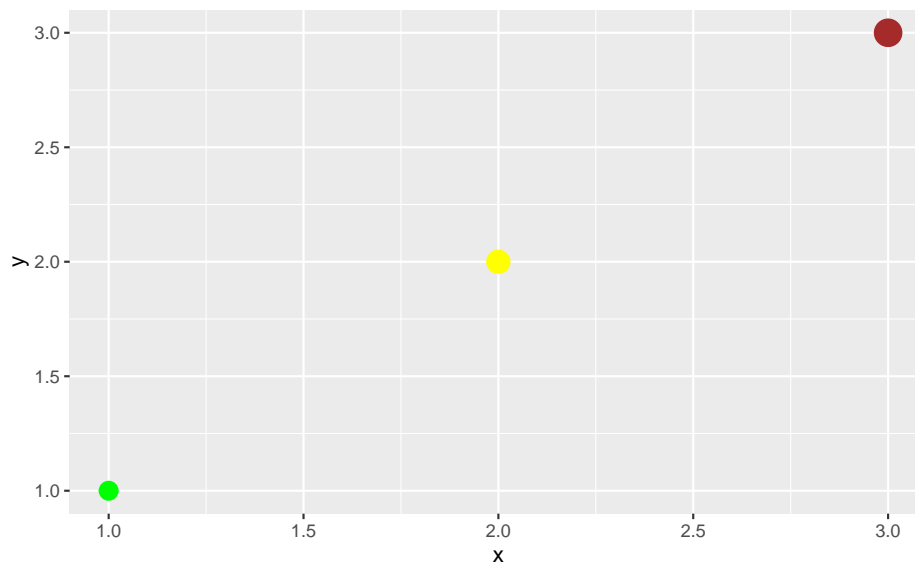
```
ggplot(df, aes(x, y, color = krasa)) +  
  geom_point()
```

Pievienojot funkciju `scale_color_identity()` un `scale_size_identity()`, attēlā punktu izmērs un krāsa ir tāda, kāda tā norādīta datu tabulā, kā arī pēc noklusējuma nerādās leģenda (?? attēls).

```
ggplot(df, aes(x, y, color = krasa, size = izmers)) +  
  geom_point() +  
  scale_color_identity() +  
  scale_size_identity()
```



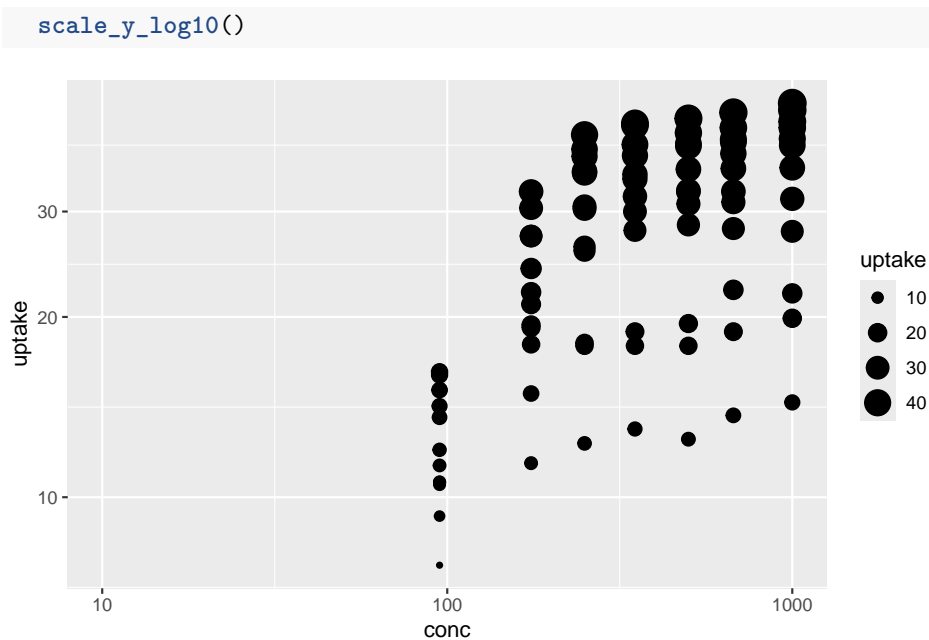
Att. 4.19: Izkliedes attēls ar mainītu krāsu



Transformāciju skalas

Ir vairākas skalu funkcijas, kas ļauj veikt x un y ass vērtību transformācijas. Tādas ir skalas `scale_x_log10()` un `scale_y_log10()`, kas attiecīgi pārvērs x un y asi par logaritmisko asi (4.20 attēls).

```
ggplot(CO2, aes(conc, uptake, size = uptake)) + geom_point() +
  scale_x_log10(breaks = c(10, 100, 1000), limits = c(10, 1000)) +
```



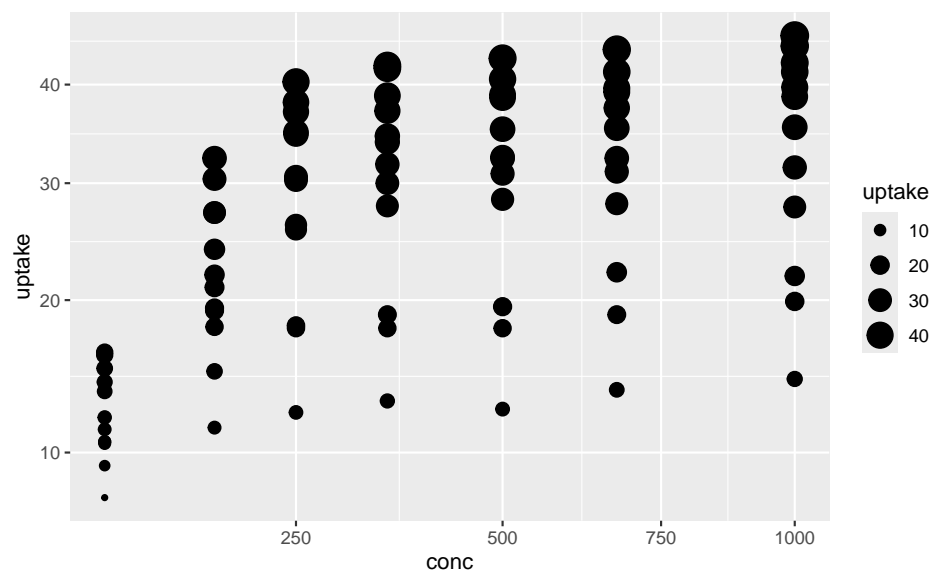
Att. 4.20: Izkliedes attēls ar logaritmisko x un y asi

Skalas `scale_x_sqrt()` un `scale_y_sqrt()` attiecīgi x un y asij veic kvadrātsaknes transformāciju (4.21 attēls).

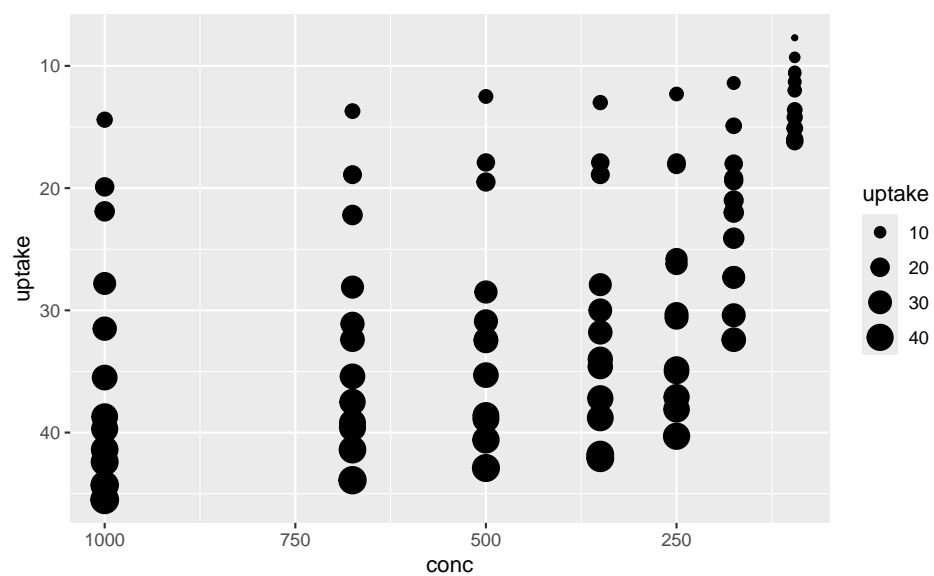
```
ggplot(CO2, aes(conc, uptake, size = uptake)) + geom_point() +  
  scale_x_sqrt() + scale_y_sqrt()
```

Skalu funkcijas `scale_x_reverse()` un `scale_y_reverse()` izmanto, ja attiecīgi x un y asij ir nepieciešams skaitliskās vērtības attēlot pretējā secībā, tas ir, sākot ar lielāko vērtību un beidzot ar mazāko vērtību (4.22 attēls).

```
ggplot(CO2, aes(conc, uptake, size = uptake)) + geom_point() +  
  scale_x_reverse() + scale_y_reverse()
```



Att. 4.21: Izklīdes attēls ar x un y asi, kurām piemērota kvadrātsaknes transformācija



Att. 4.22: Izklīdes attēls ar reversajām x un y asi

Nodaļa 5

Koordinātu sistēmas

Funkcijas koordinātu sistēmu noteikšanai izmanto, lai mainītu uz ass attēlojamo vērtību diapozonu, kā arī, lai mainītu vērtību attiecības starp x un y asīm.

5.1 coord_cartesian()

Pamatkoordinātu sistēma ir `cartesian`, kas tiek izmantota pēc noklusējuma. Ar funkciju `coord_cartesian()` var mainīt x un y ass diapozonu (to palielinot vai samazinot). Svarīgākais šajā procesā ir tas, ka mainās tikai attēlā redzamā datu daļa, bet netiek mainīts attēla veidošanai izmantotais datu apjoms (strādā līdzīgi kā “zoom”) (5.1 attēls).

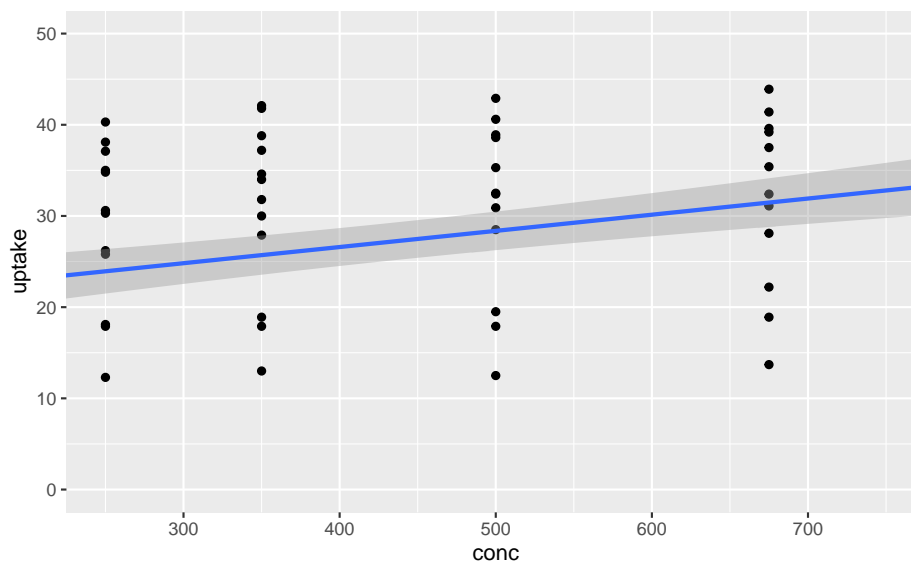
```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_smooth(method = "lm") +  
  coord_cartesian(xlim = c(250, 750), ylim = c(0,50))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

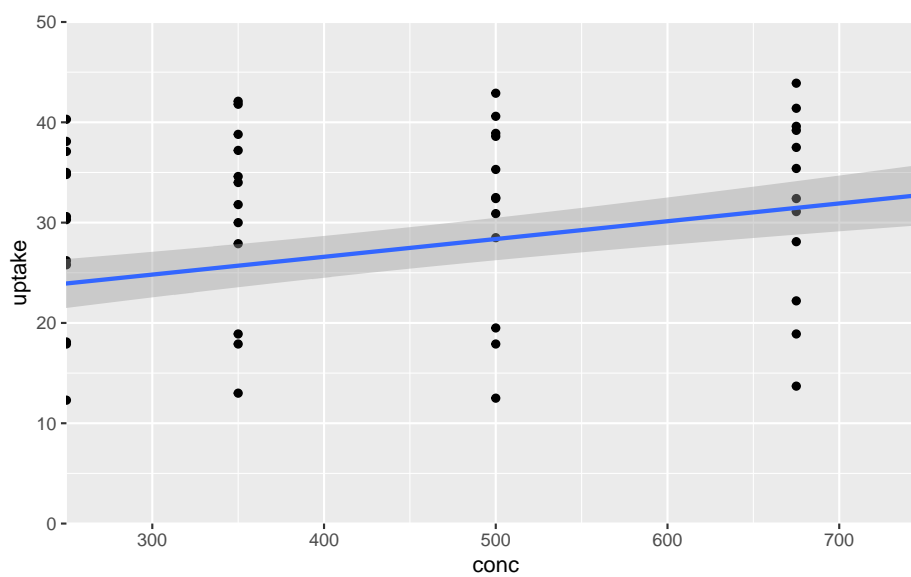
Norādītajiem asu limitiem automātiski tiek pievienota neliela papildus vieta. Ja ir nepieciešams, lai attēls būtu precīzi noteiktajā diapozonā, tad jāpievieno arguments `expand = FALSE` (5.2 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  geom_smooth(method = "lm") +  
  coord_cartesian(xlim = c(250, 750), ylim = c(0,50), expand = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Att. 5.1: Izkliedes attēls ar mainītu asu diapozonu

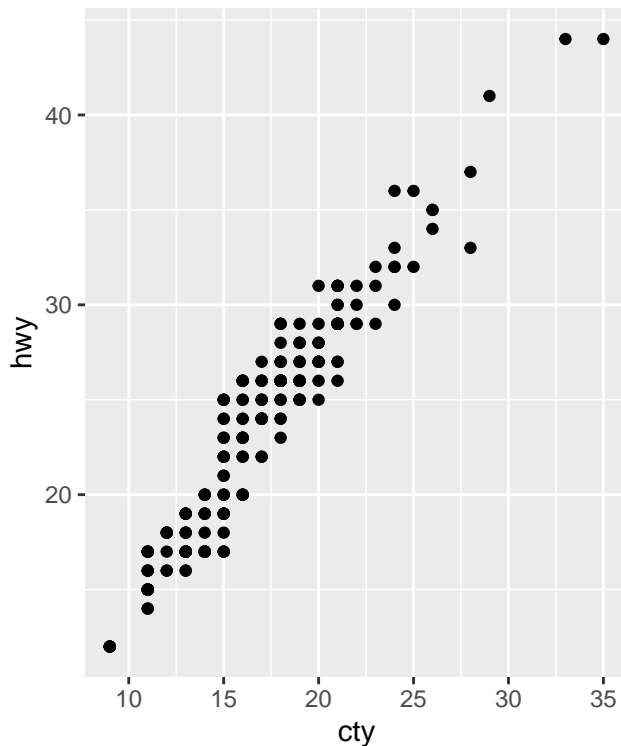


Att. 5.2: Izkliedes attēls ar precīzu mainītu asu diapozonu

5.2 coord_fixed()

Fiksēto koordinātu sistēmu izmanto tad, ja nepieciešams noteikta attiecība starp vienas vienības izmēru uz x ass un y ass. Pēc nokļūšanas vērtība `ratio = 1`, kas nozīmē, ka viena vienība uz x ass ir tikpat gara kā uz y ass (5.3 attēls).

```
ggplot(mpg, aes(cty, hwy)) + geom_point() +  
  coord_fixed()
```



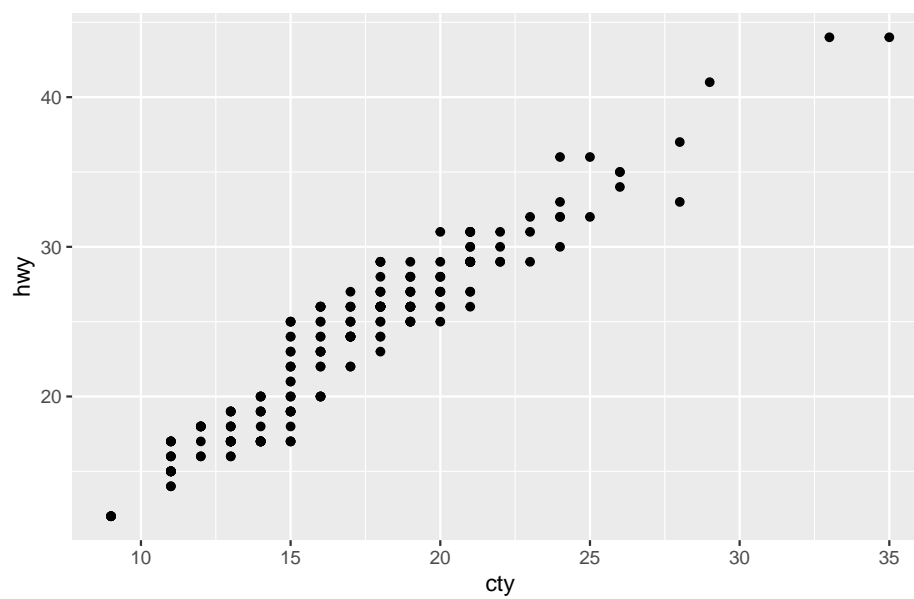
Att. 5.3: Izklādes attēls ar fiksētām asīm

Norādot pie argumenta `ratio = skaitli`, kas lielāks par 1, uz y ass vienībā būs tik reizes lielāka, nekā uz x ass; attiecīgi norādot skaitli, kas mazāks par 1, y ass viena vienība būs mazāka nekā uz x ass (5.4 attēls).

```
ggplot(mpg, aes(cty, hwy)) + geom_point() +  
  coord_fixed(ratio = 0.5)
```

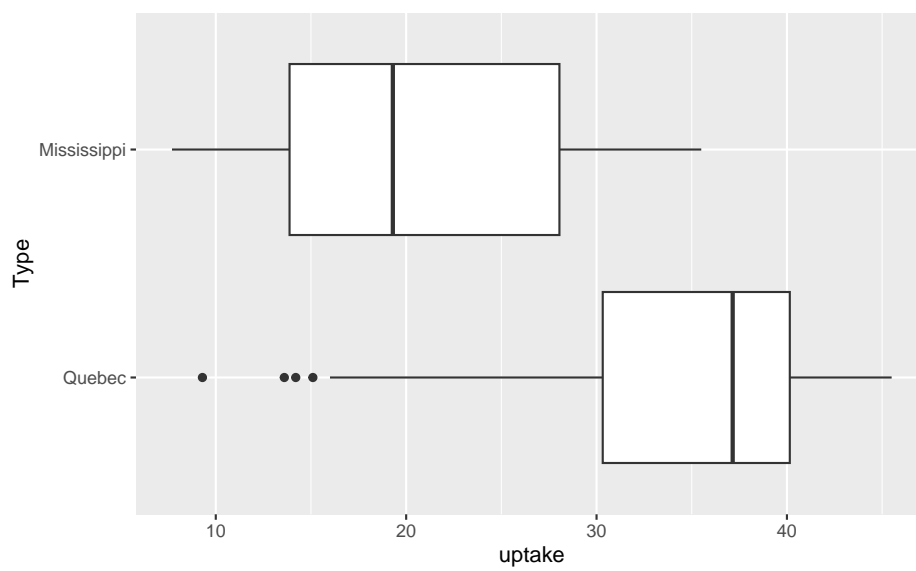
5.3 coord_flip()

Lai samainītu vietām x un y asi, jāizmanto funkcija `coord_flip()` (5.5 attēls).



Att. 5.4: Izkliedes attēls ar fiksētām asīm

```
ggplot(CO2, aes(Type, uptake)) + geom_boxplot() +  
  coord_flip()
```



Att. 5.5: Vērtībamplitūdas diagramma ar mainītu asu novietojumu

Nodaļa 6

Attēlu sadalīšana

Viena no ggplot2 sistēmas lielajām priekšrocībām ir tā, ka izmantojot tam speciāli paredzētas funkcijas (`facet_wrap()` un `facet_grid()`), ir iespējams sadalīt attēlu vairākās daļās balstoties uz vienu vai vairākiem mainīgiem, kur katrs mazais attēls ir daļa no kopējā datu attēlojuma.

6.1 `facet_grid()`

Izmantojot funkciju `facet_grid()`, var norādīt divus mainīgos pēc kuriem dalīt datus. Pirmais mainīgais (pirms tildes zīmes) norāda dalījumu rindās, bet otrais mainīgais aiz tildes zīmes norāda dalījumu kolonnās. Ja ir vēlme dalīt tikai vienā dimensijā, tad neizmantotās dimensijas (mainīgā) vietā jānorāda “.”.

Pirmajā piemēra attēls sadalīts mazākos attēlos balstoties tikai uz mainīgo `Type` kolonnās (6.1 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  facet_grid(. ~ Type)
```

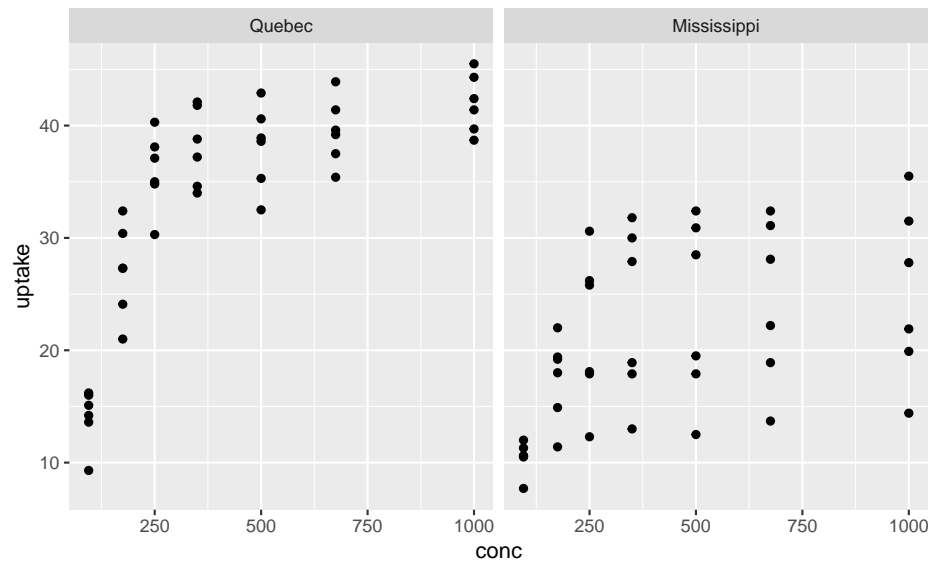
Norādot mainīgo `Treatment` pirms tildes zīmes, izveidojas attēls, kas sadalīts rindās atbilstoši šī mainīgā līmeņiem (6.2 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  facet_grid(Treatment ~ .)
```

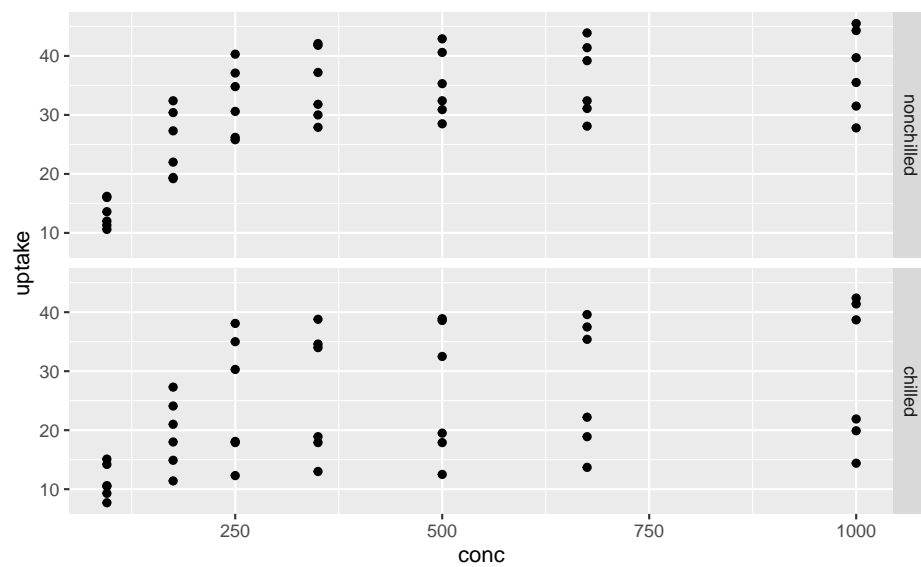
Norādot abus divus mainīgos, izveidojas attēls, kurā mazie attēliņi ir atbilstošo mainīgo līmeņu kombinācijas (6.3 attēls).

```
ggplot(CO2, aes(conc, uptake)) + geom_point() +  
  facet_grid(Treatment ~ Type)
```

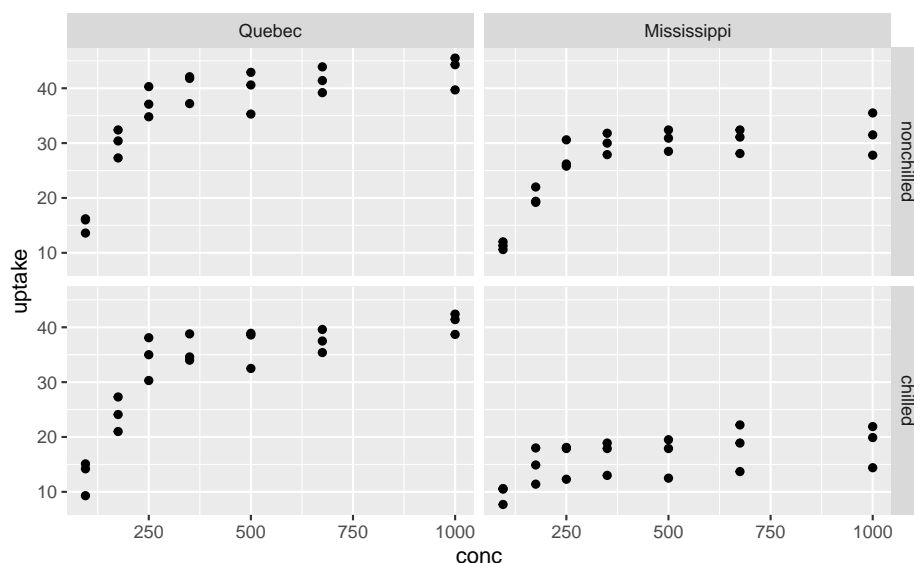
Pievienojot argumentu `margin = TRUE`, var panākt, ka veidojas ne tikai atse-



Att. 6.1: Attēla sadalīšana kolonnās balstoties uz vienu mainīgo



Att. 6.2: Attēla sadalīšana rindās balstoties uz vienu mainīgo



Att. 6.3: Attēla sadalīšana balstoties uz diviem mainīgumiem

višķi mazie attēli, bet arī attēli, kuros mainīgo līmeņi skatīti kopā (6.4 attēls).

```
ggplot(CO2,aes(conc, uptake)) + geom_point() +
  facet_grid(Treatment ~ Type,margins = TRUE)
```

6.2 facet_wrap()

'facet_wrap()' gadījumā mazie attēliņi tiek novietoti viens aiz otra, ar iespēju norādīt cik rindās/kolonnās tos nepieciešams izvietot. Attēlu sadalīšanu var veikt, piemēram, ar vienu mainīgo (nav jāizmanto "." pirms tildes) (6.5 attēls).

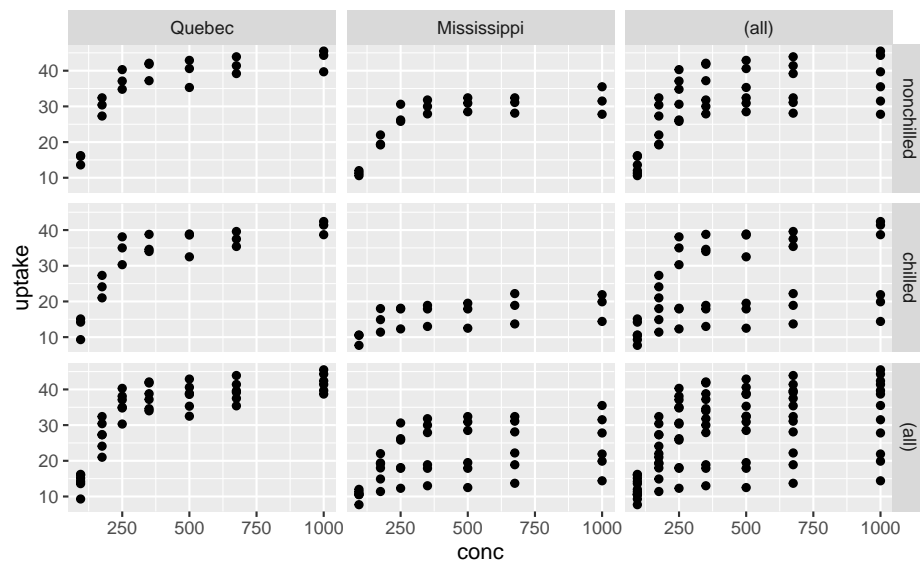
```
ggplot(mpg, aes(cty, hwy)) + geom_point() +
  facet_wrap(~ class, ncol = 4)
```

Dalīšanu daļās var veikt arī ar vairākiem mainīgajiem, norādot tos aiz tildes zīmes (6.6 attēls).

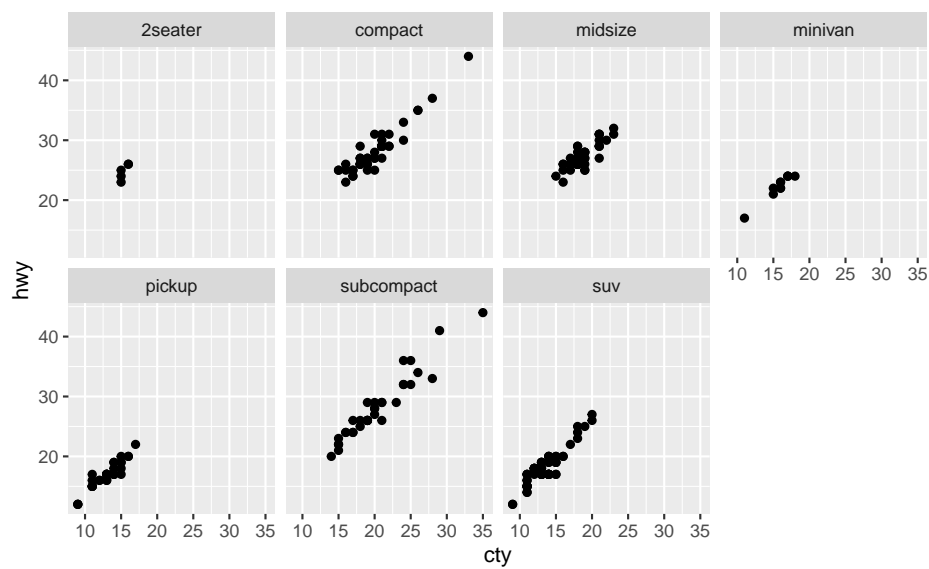
```
ggplot(mpg, aes(cty, hwy)) + geom_point() +
  facet_wrap(~cyl + drv, ncol = 4)
```

Attēla sadalīšanai daļās var izmantot arī papildus funkcijas, piemēram, sadalot skaitlisku mainīgo daļās (6.7 attēls).

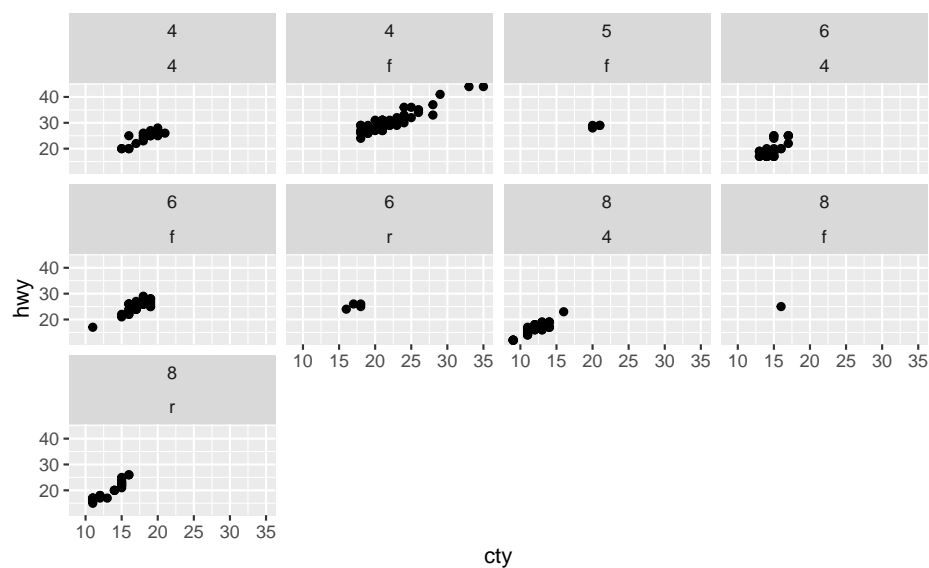
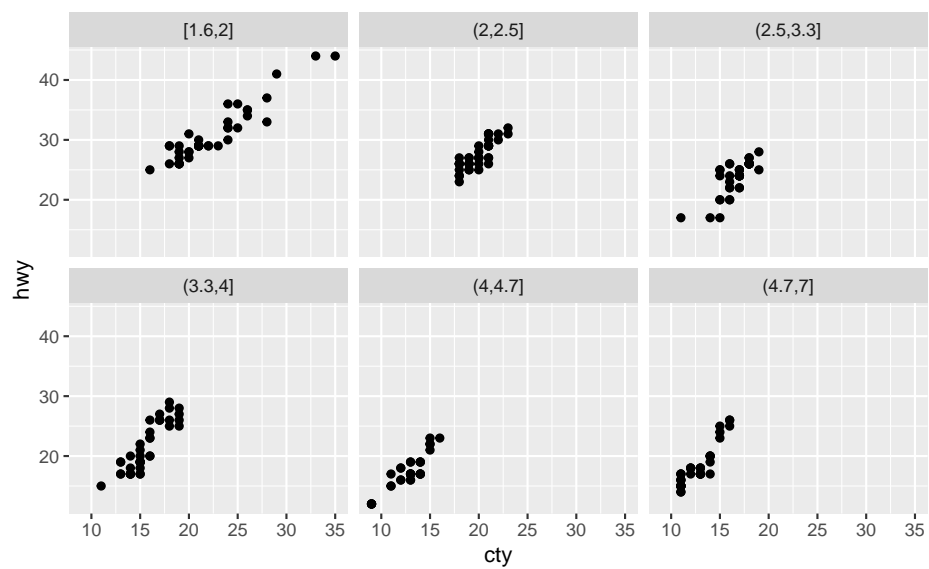
```
ggplot(mpg,aes(cty,hwy)) + geom_point() +
  facet_wrap(~cut_number(displ, 6))
```



Att. 6.4: Attēla sadalīšana balstoties uz diviem mainīgiem, parādot arī kopējos attēlus



Att. 6.5: Attēla sadalīšana daļās ar facet_wrap()

Att. 6.6: Attēla sadalīšana daļās ar `facet_wrap()` un diviem mainīgiemAtt. 6.7: Attēla sadalīšana daļās ar `facet_wrap()` un dalījums balstās uz skaitlisku mainīgo, kas sadalīts intervālos

Nodaļa 7

Attēla noformēšana

ggplot2 sistēmā izveidoto attēlu izskata mainīšanai var izmantot iepriekš sagatavotas attēla noformēšanas tēmas, vai arī var mainīt katru elementu atsevišķi. Mainot noformējumu, mainās tikai tās attēla daļas, kas nav saistītas attēlojamiem datiem.

7.1 Definētās attēla tēmas

Paketē ggplot2 ir definētas gatavas tēmas attēla izskata maiņai. Attēlā noformējums mainās, pieskaitot klāt atbilstošo tēmas funkciju. Katrā tēmā papildus ir iespējams mainīt pamatteksta izmēru (`base_size =`) un pamatfontu (`base_family =`). Gatavos tēmu noformējumus protams var papildināt arī ar citām izmaiņām konkrētiem elementiem. Attēlos 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8 un 7.9 parādīts kā izskatās sākotnējais attēls un kā tas mainās, izmantojot kādu no gatavajām tēmām.

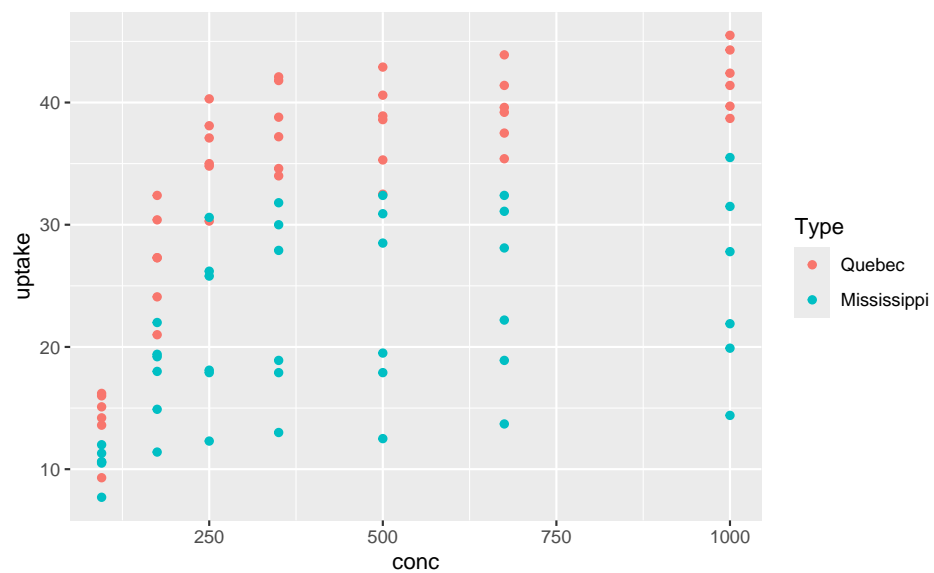
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point()
```

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_bw()
```

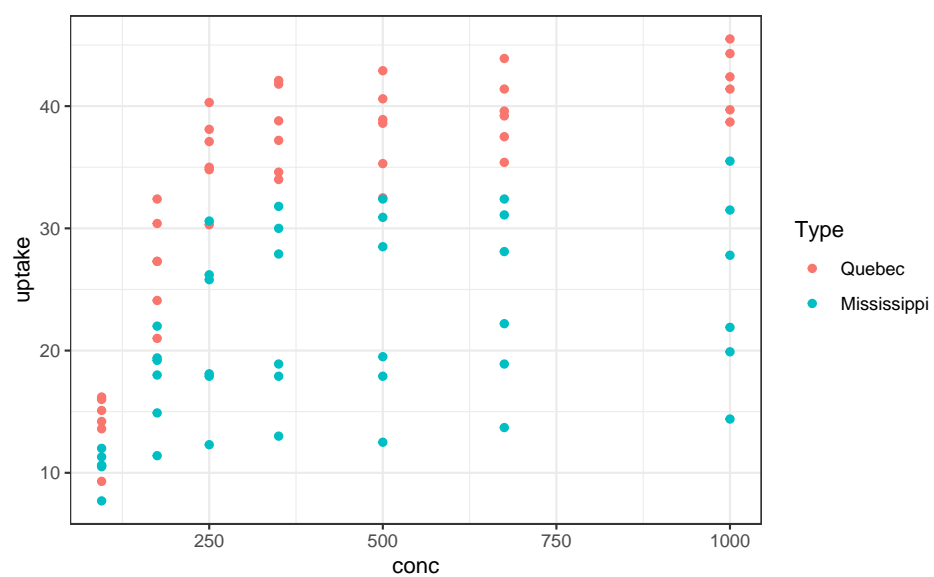
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_classic()
```

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_dark()
```

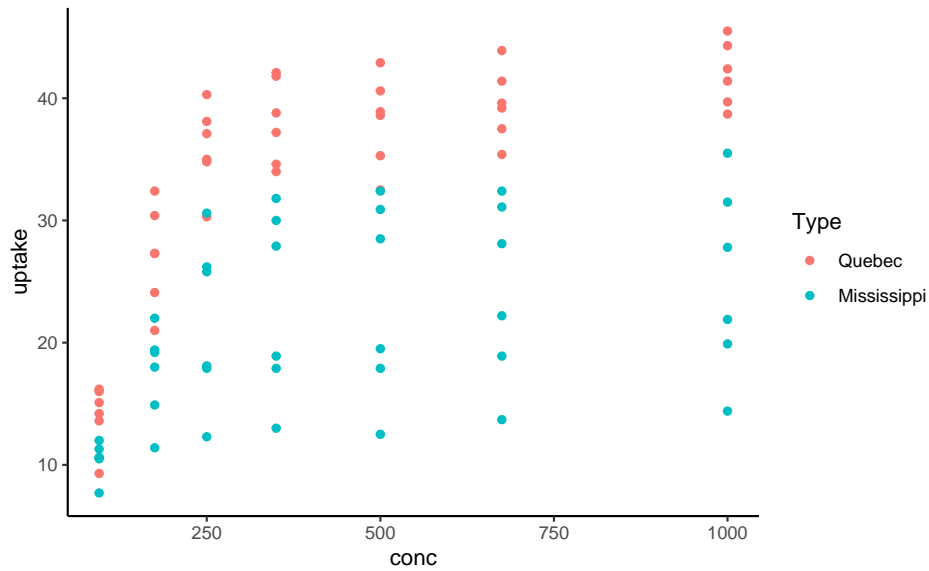
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_grey()
```



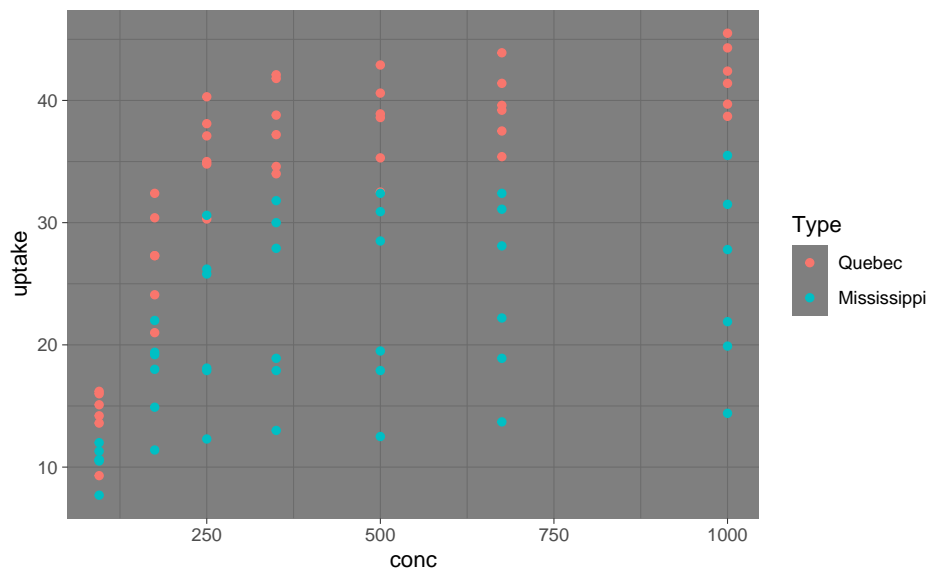
Att. 7.1: Attēls bez papildus noformējuma



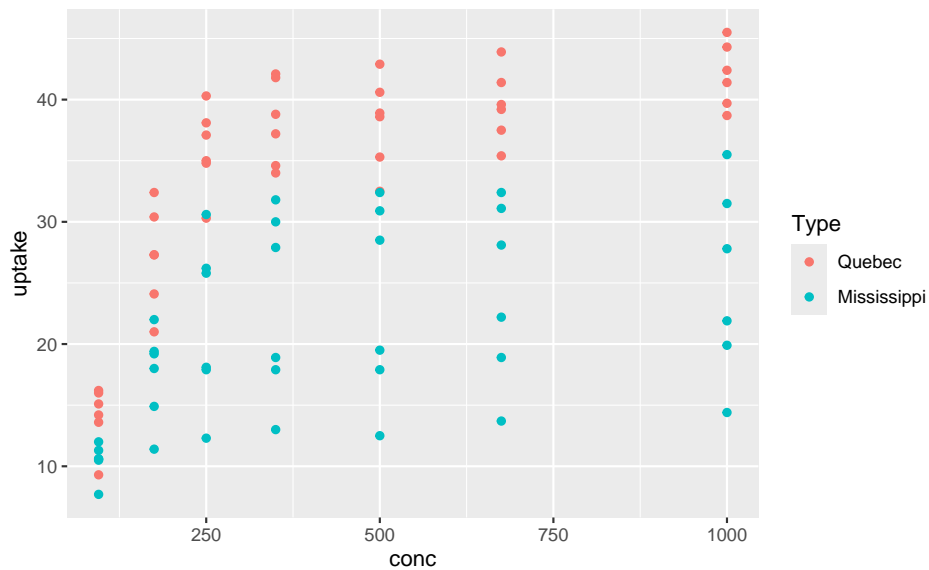
Att. 7.2: Attēls ar theme_bw()



Att. 7.3: Attēls ar theme_classic()



Att. 7.4: Attēls ar theme_dark()

Att. 7.5: Attēls ar `theme_grey()`

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_light()
```

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_linedraw()
```

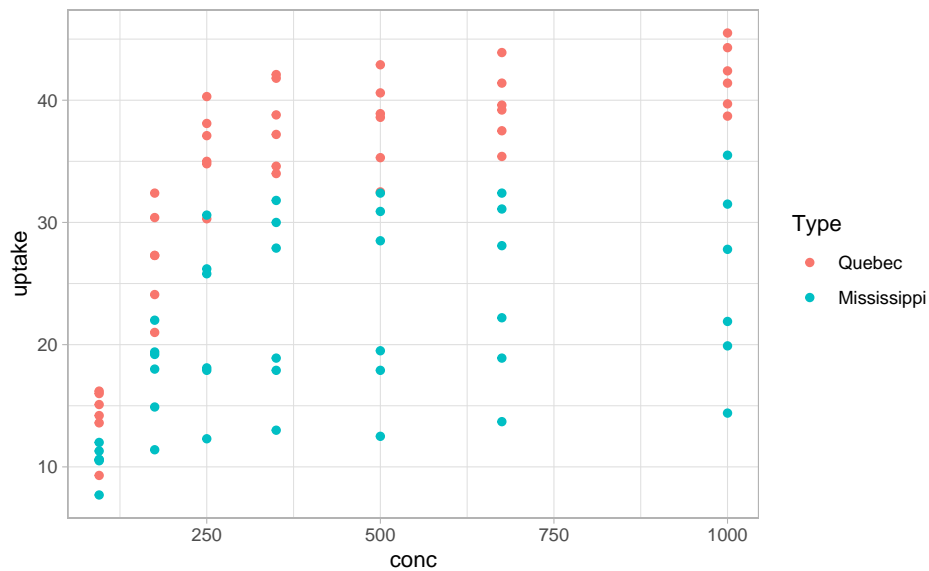
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_minimal()
```

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme_void()
```

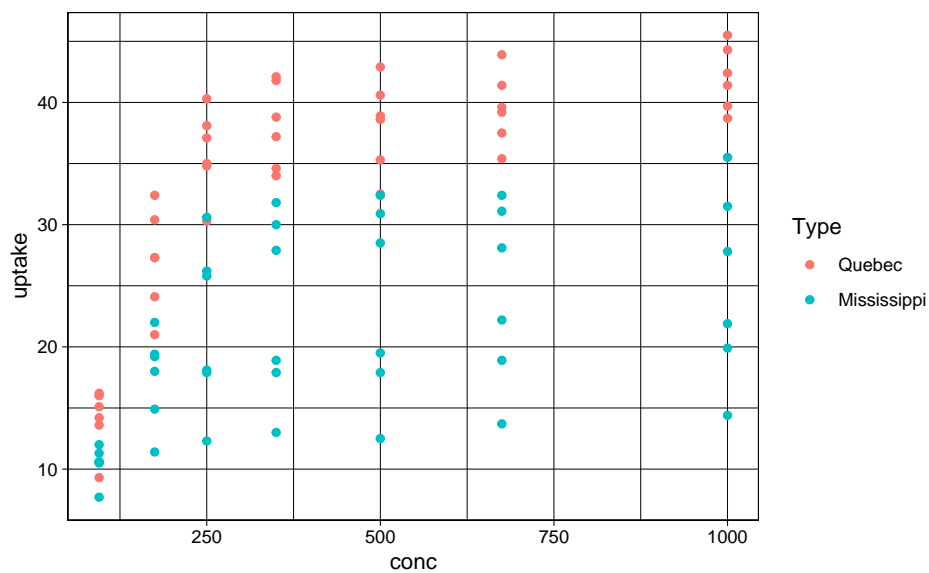
7.2 Atsevišķu attēla elementu mainīšana

Papildus jau definētajām attēla noformējuma tēmām, ir iespējams mainīt gan drīz katru no attēla elementiem atsevišķi, izmantojot funkciju `theme()` un atbilstošo elementa nosaukumu. Pilnu sarakstu ar elementu nosaukumiem var iegūt apskatot funkcijas `theme()` palīdzības lapu.

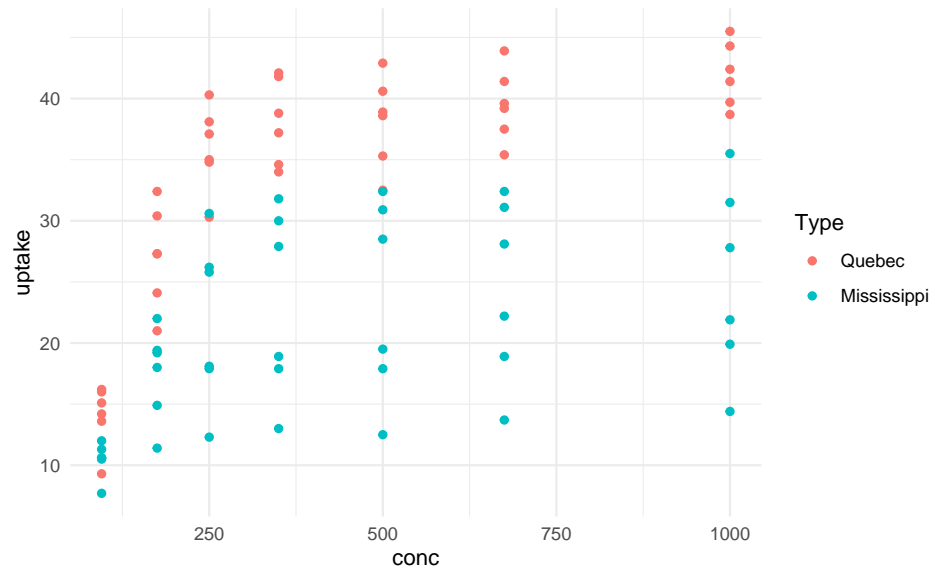
Elementu izskatu maiņa notiek pēc vienota principa - sākotnēji ir funkcija `theme()`, kurā kā arguments jānorāda maināmais elements, piemēram, `axis.text.x` = un tad nāk funkcija, kas norāda kāda izmaiņas veikt. Ir pieejamas trīs funkcijas `element_text()`, `element_line()` un `element_rect()`, kuras ir jāizraugas atbilstoši tam, kāda veida maināmais elements tas ir - teksts,



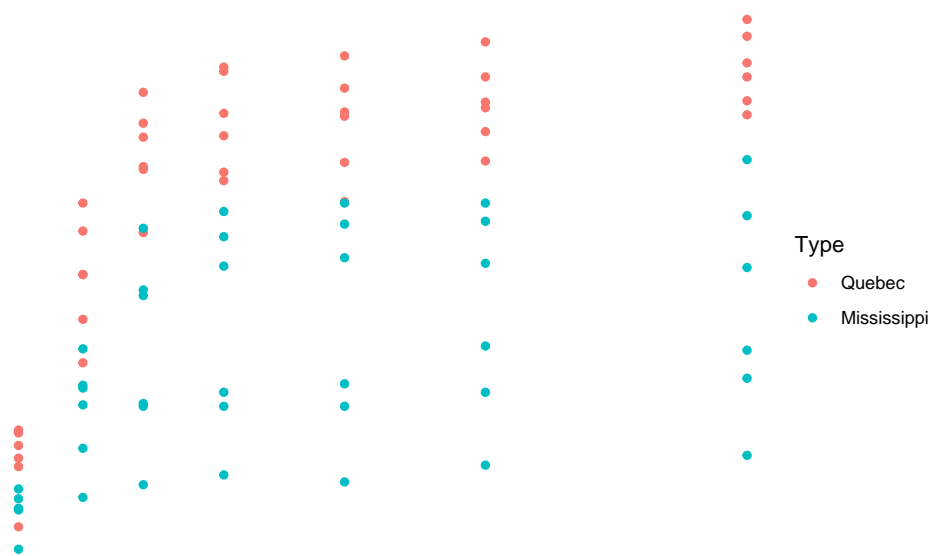
Att. 7.6: Attēls ar theme_light()



Att. 7.7: Attēls ar theme_linedraw()



Att. 7.8: Attēls ar theme_minimal()



Att. 7.9: Attēls ar theme_void()

līnija vai reģions (poligons). Katra elementa maiņa nav jāraksta savā `theme()` funkcijā, bet tie var būt vairāki argumenti vienā funkcijā.

Teksta elementiem (`element_text()`) var mainīt teksta fontu grupu (`family =`), fonta veidu (`face =`), krāsu (`color =`), izmēru (`size =`), novietojumu horizontāli (`hjust =`) un vertikāli (`vjust =`), teksta leņķi (`angle =`), līnijas augstumu (`lineheight =`), atstarpes (`margin =`).

Līniju elementiem (`element_line()`) var mainīt krāsu (`color =`), līnijas biezumu (`linewidth=`), līnijas veidu (`linetype =`), līnijas nobeigumu (`lineend =`) un pievienot bultu (`arrow =`).

Reģiona jeb poligona elementiem (`element_rect()`) var mainīt aizpildījumu (`fill =`), krāsu līnijai apkārt reģionam (`color =`), līnijas biezumu (`linewidth =`) un līnijas veidu (`linetype =`).

Ir pieejama arī speciāla funkcija `element_blank()`, kas no attēla noņems atbilstošo elementu, turklāt “pazudīs” arī šim elementam atvēlēta vieta, ja tas būs, piemēram, asu paraksts, vai asu apzīmējumi.

7.2.1 Attēla virsraksts

Attēla virsraksta izskatu (bet ne tā saturu) maina ar argumentu `plot.title` =.

Pirmajā piemēra teksta virsraksts pārveidots zaļā krāsā, 20 fonta izmērā un slīprakstā (7.10 attēls).

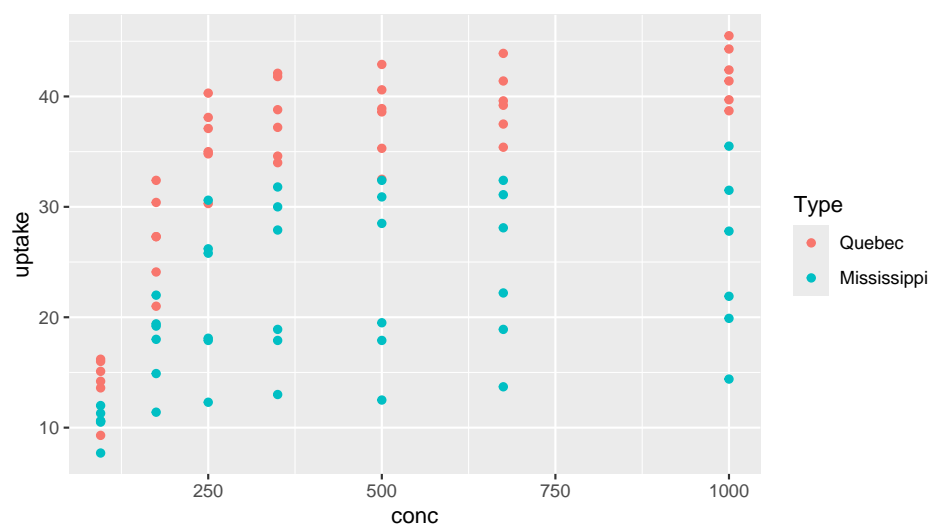
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  labs(title = "CO2 attēls") +
  theme(plot.title = element_text(colour = "darkgreen",
                                   size = 20,
                                   face = "italic"))
```

Virsraksts automātiski ir novietots attēla kreisajā pusē. To var izmainīt ar argumentu `hjust` = un vērtību 0.5 (0 - kreisā mala, 1 - labā mala, 0.5 - pa vidu) (7.11 attēls).

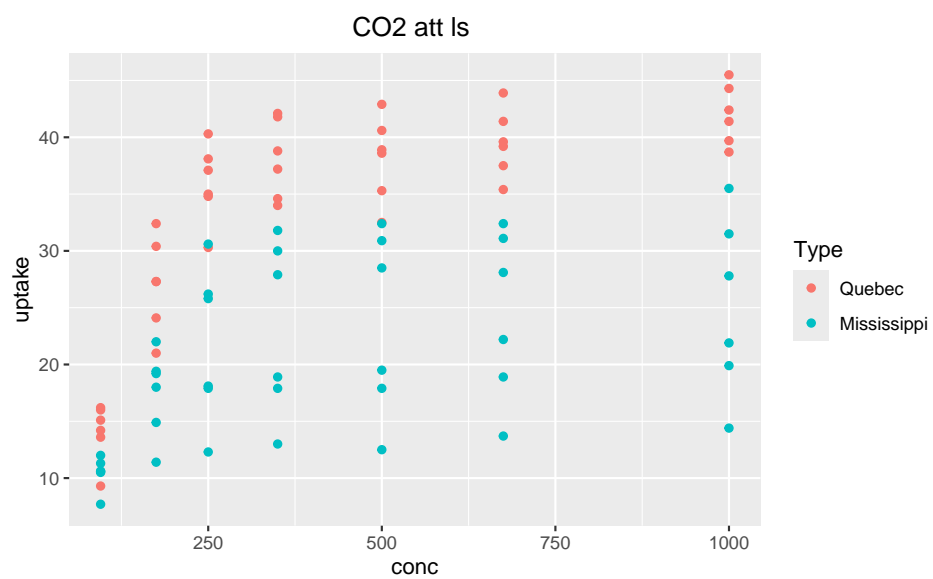
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  labs(title = "CO2 attēls") +  
  theme(plot.title = element_text(hjust = 0.5))
```

Ja attēlam ir arī apakšvirsraksts, tad tā izskatu maina ar argumentu `plot.subtitle = (7.12 attēls)`.

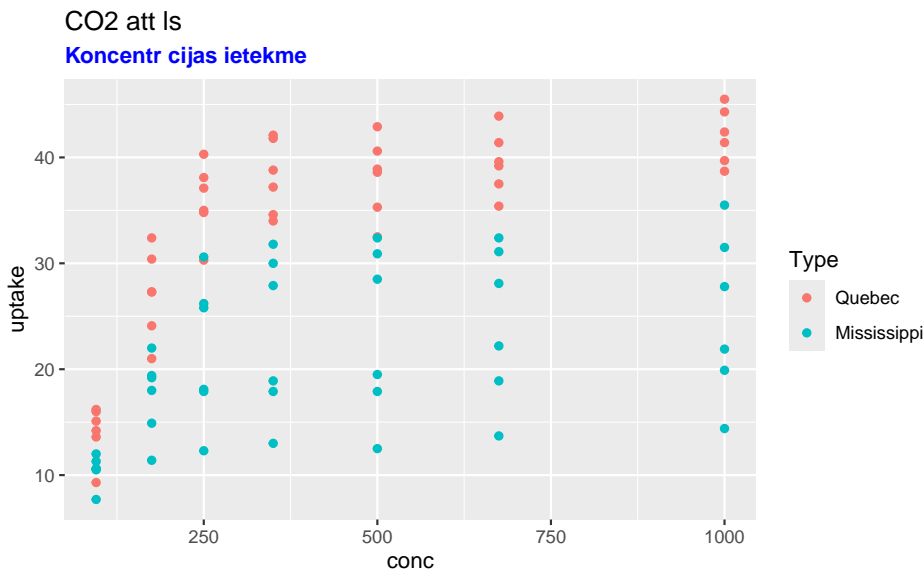
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  labs(title = "CO2 attēls", subtitle = "Koncentrācijas ietekme")+
  theme(plot.subtitle = element_text(face = "bold",
                                     colour = "blue"))
```

CO₂ att ls

Att. 7.10: Attēls ar mainītu virsraksta izskatu



Att. 7.11: Attēls ar mainītu virsraksta novietojumu



Att. 7.12: Attēls ar mainītu apakšvirsrakstu

7.2.2 Asu paraksti

Asu parakstu maiņa notiek ar elementu `axis.title =` (abas ass vienlaicīgi), vai `axis.title.x =` un `axis.title.y =` mainot katras ass izskatu atsevišķi.

Asu paraksti arī ir teksts, tāpēc izmantojama funkcija `element_text()` (7.13 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(axis.title = element_text(size = 15,
                                   face = "bold"))
```

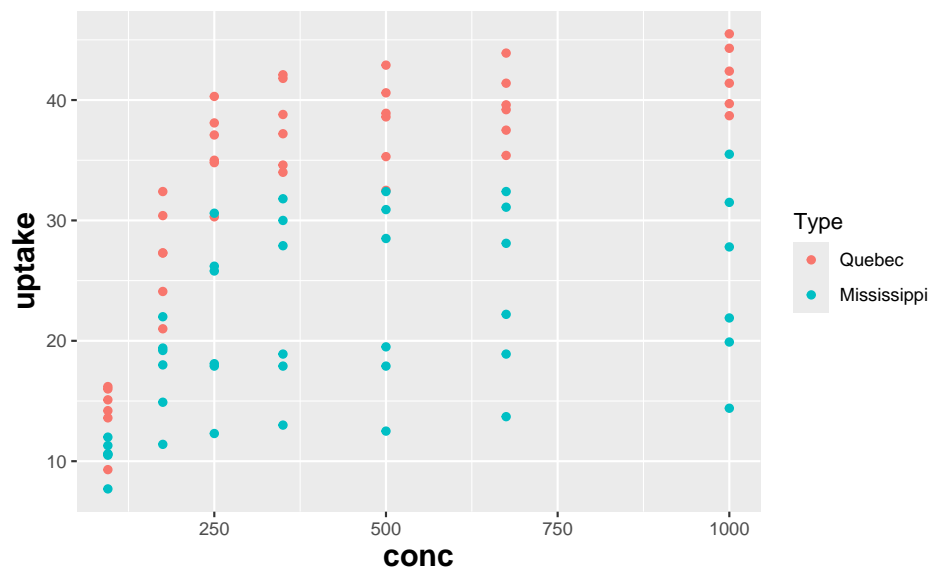
Izmantojot argumentus `axis.title.x =` un `axis.title.y =` var mainīt katru asi atsevišķi, piemēram, viens asij noņemot parakstu, bet otrai to izmainot (7.14 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_text(colour = "red"))
```

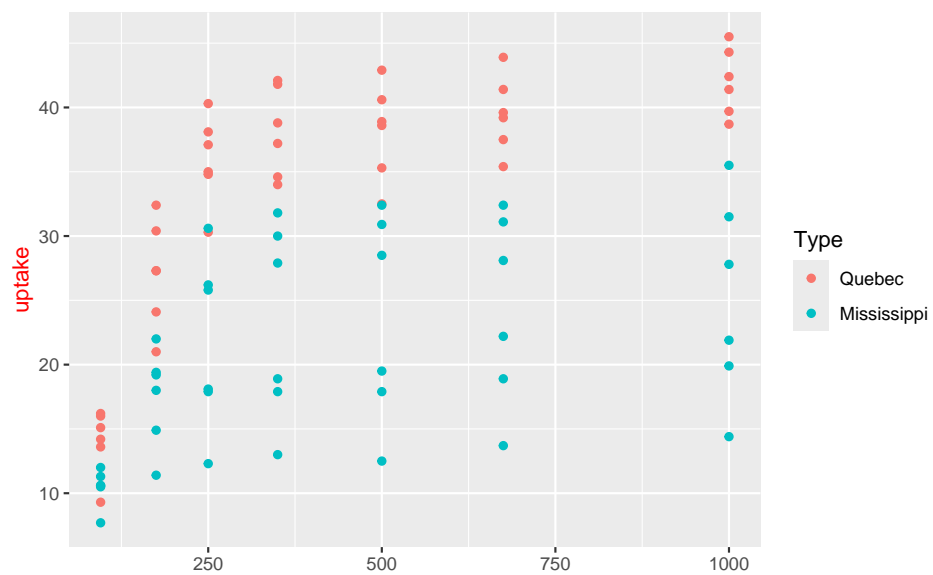
7.2.3 Asu apzīmējumi

Apzīmējumus pie asīm (skaitļus vai līmeņu nosaukumus) maina ar argumentiem `axis.text =`, `axis.text.x =`, vai `axis.text.y =` (7.15 attēls).

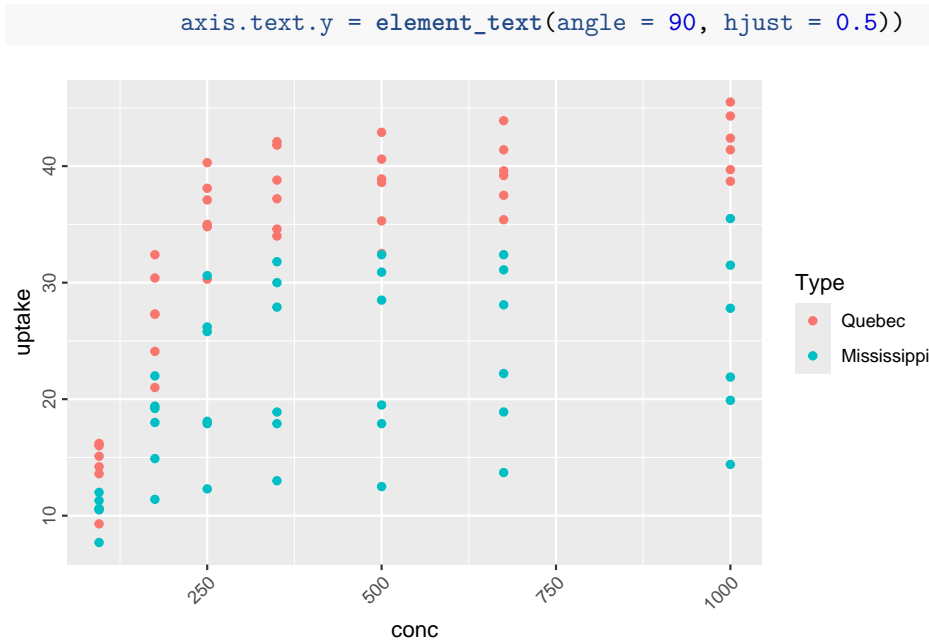
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
```



Att. 7.13: Attēls ar vienādu asu parakstu noformējumu



Att. 7.14: Attēls ar x asi bez paraksta



Att. 7.15: Attēls ar mainītiem asu apzīmējumiem

7.2.4 Asu līnijas

Pēc noklusējuma vairākām tēmām nerādās asu līnijas. Tās var pievienot ar argumentu `axis.line =`, vai `axis.line.x =` un `axis.line.y =`. Tā kā tās ir līnijas, tad to izskata maiņai izmanto funkciju `element_line()` (7.16 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme(axis.line = element_line(linetype = "longdash"))
```

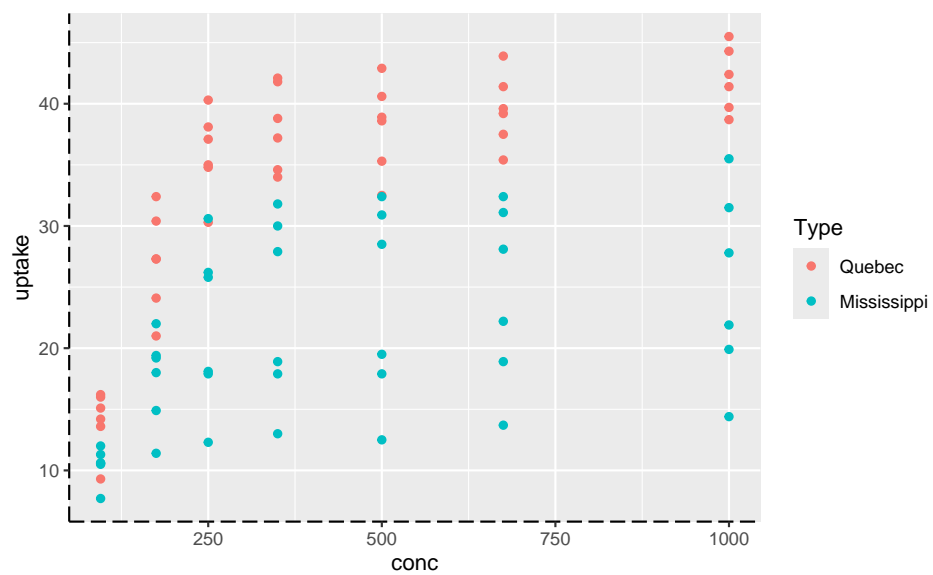
Asu līniju galos ir iespējams arī pievienot bultiņu ar argumentu `arrow =` un funkciju `arrow()`, kurā norāda argumentu `length =` un bultiņas garumu un mērvienību (funkcijā `unit()`) (7.17 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme(axis.line = element_line(arrow = arrow(length = unit(2, 'mm'))))
```

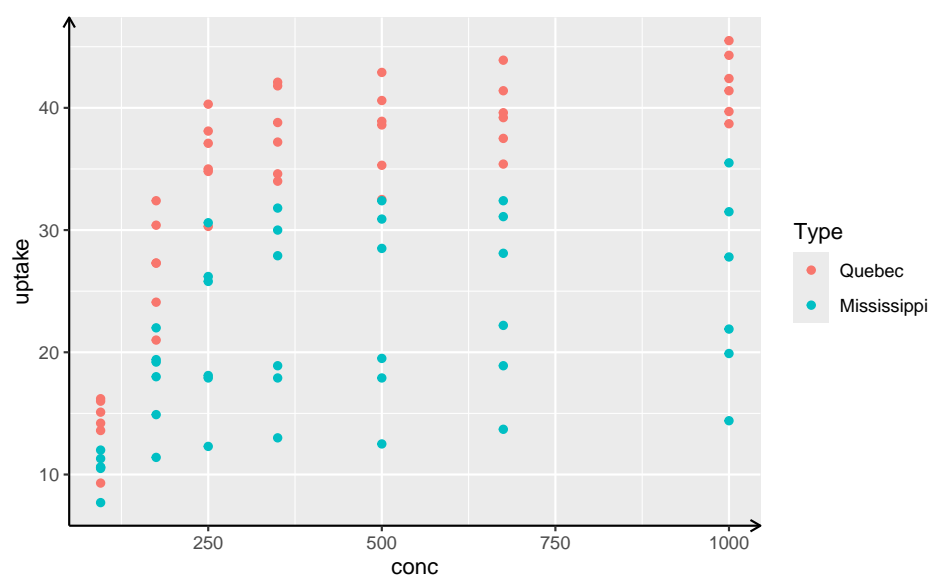
7.2.5 Lēģendas novietojums

Lēģendas novietojuma nosaka ar argumentu `legend.position =` un iespējamām vērtībām "none", "left", "right", "bottom" un "top" (7.18 attēls).

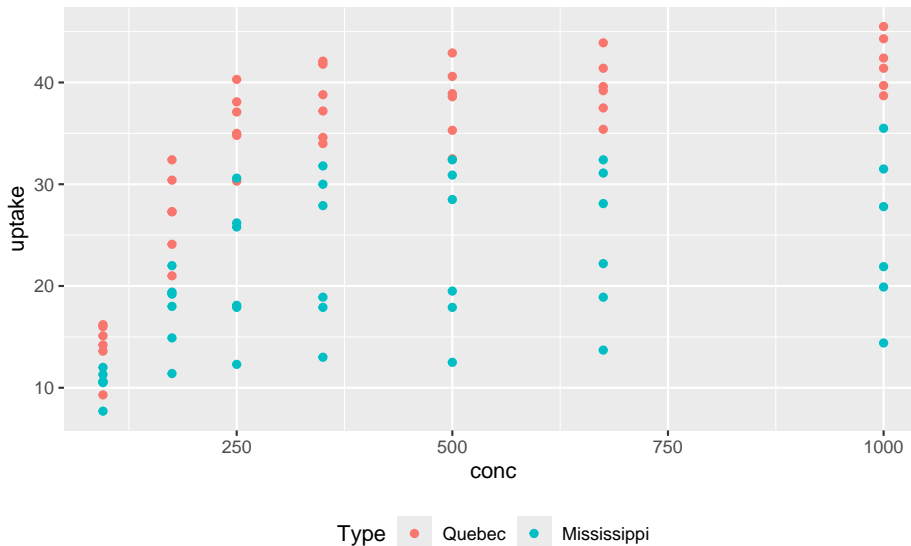
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme(legend.position = "bottom")
```



Att. 7.16: Attēls ar asu līnijām



Att. 7.17: Attēls ar bultiņām pie asīm



Att. 7.18: Attēls ar leģendu apakšā

Ja nepieciešams novietot leģendu attēla iekšienē, tad jānorāda relatīvās x un y koordinātes, kur $c(0,0)$ atbilst attēla apakšējam kreisajam stūrim un $c(1,1)$ attēla augšējam labajam stūrim (7.19 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.position = c(0.1, 0.8))
```

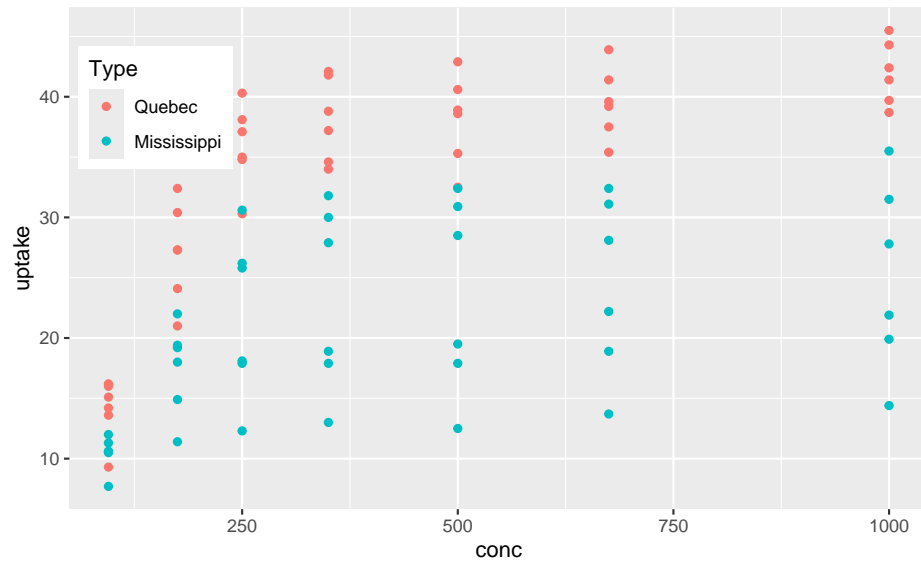
```
## Warning: A numeric `legend.position` argument in `theme()` was deprecated in ggplot2
## 3.5.0.
## i Please use the `legend.position.inside` argument of `theme()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

Leģendas novietojumā var mainīt arī leģendas ierakstu izvietojumu horizontāli vai vertikāli ar argumentu `legend.direction` = (7.20 attēls)

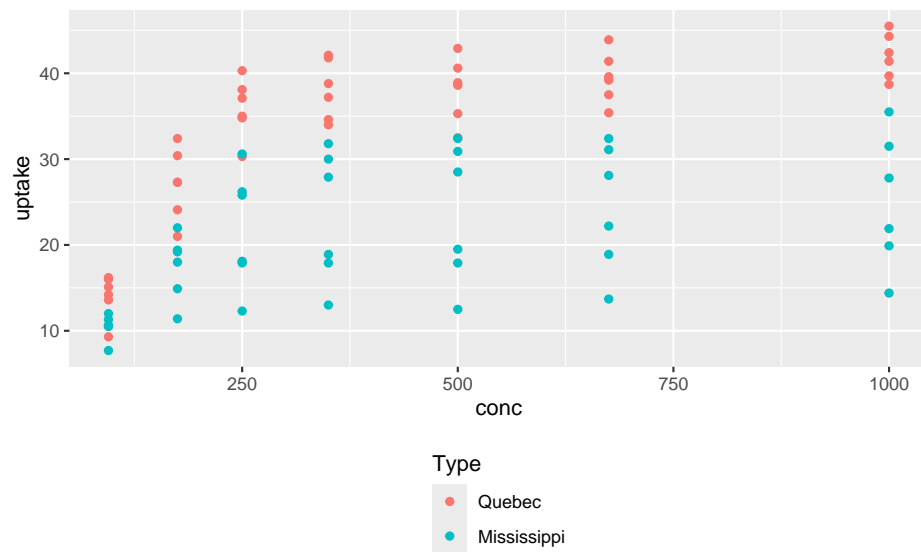
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.position = "bottom",
        legend.direction = "vertical")
```

Leģendas novietojumu var ietekmēt arī mainot atstarpi starp leģendu un pamattattēlu ar argumentu `legend.box.spacing` = un funkciju `unit()`, kurā norāda vēlamā atstarpes izmēru un mērvienību (7.21 attēls).

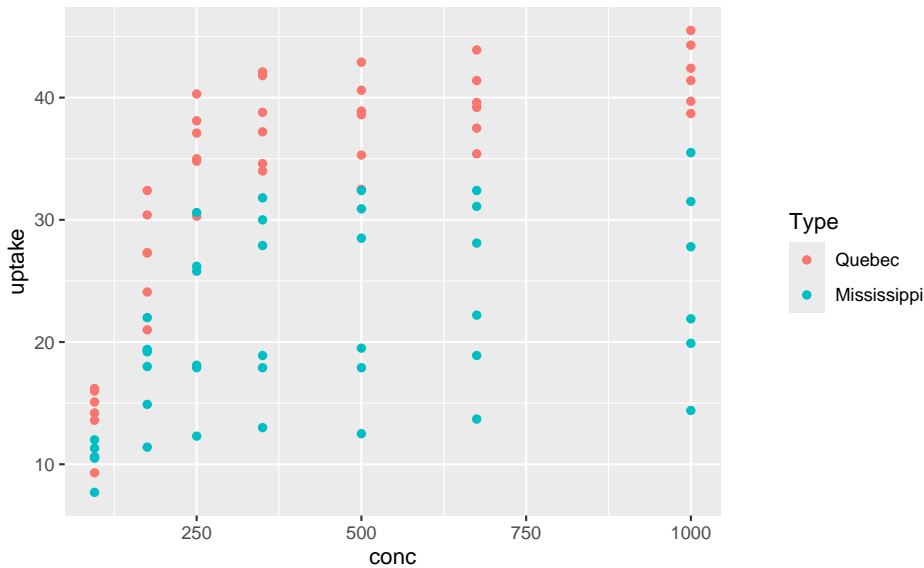
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.box.spacing = unit(1, "cm"))
```



Att. 7.19: Attēls ar leģendu attēla iekšienē



Att. 7.20: Attēls ar leģendu apakšā un leģendas ierakstiem izkārtotiem vertikāli



Att. 7.21: Attēls ar tālāk novietotu leģendu

7.2.6 Leģendas virsraksts

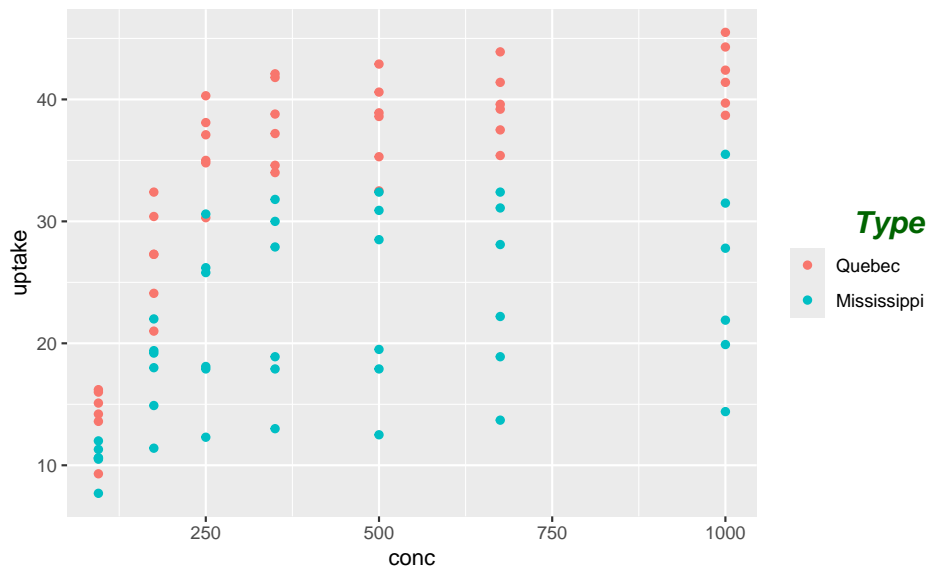
Leģendas virsrakstam var mainīt izskatu (`legend.title =`) līdzīgi kā citiem teksta elementiem, kā arī mainīt leģendas virsraksta novietojumu (`legend.title.align =`) ar vērtībām no 0 (pa kreisi) līdz 1 (pa labi) (7.22 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.title = element_text(size = 15,
                                     color = "darkgreen",
                                     face = "bold.italic"),
        legend.title.align = 1)
```

```
## Warning: The `legend.title.align` argument of `theme()` is deprecated as of ggplot2
## 3.5.0.
## i Please use theme(legend.title = element_text(hjust)) instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

7.2.7 Leģendas pamatne

Kopējo leģendas pamatnes izskatu var mainīt ar argumentu `legend.background` = un funkciju `element_rect()` (7.23 attēls).



Att. 7.22: Attēls ar mainītu leģendas virsrakstu

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.background = element_rect(fill = "grey73",
                                          colour = "black"))
```

Ja attēlam ir divas vai vairākas leģendas, tad `legend.background` = maina katras atsevišķas leģendas pamatni, bet kopējo pamatni maina ar argumentu `legend.box.background` = (7.24 attēls).

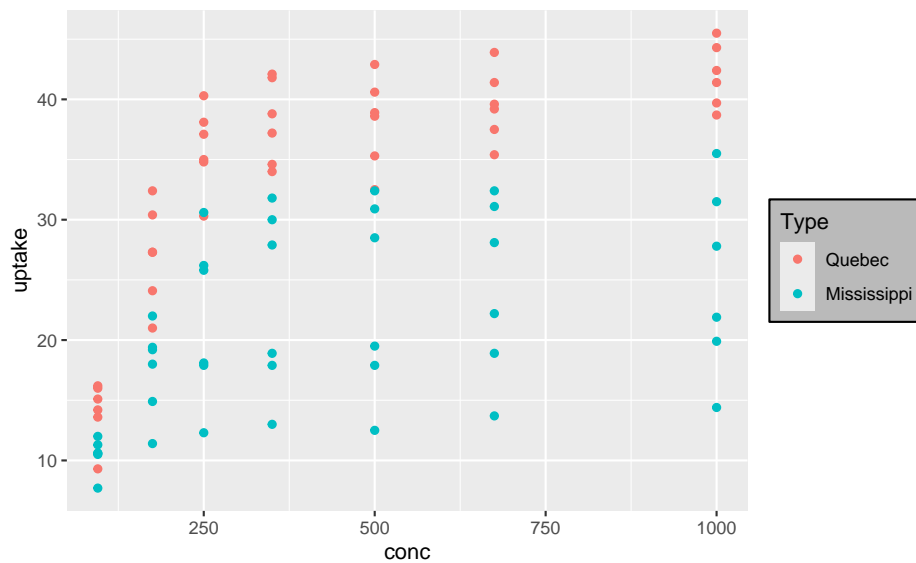
```
ggplot(CO2, aes(conc, uptake, color = Type, shape = Treatment)) + geom_point() +
  theme(legend.box.background = element_rect(fill = "grey73",
                                              colour = "black"),
        legend.background = element_rect(fill = "grey90"))
```

7.2.8 Leģendas teksti

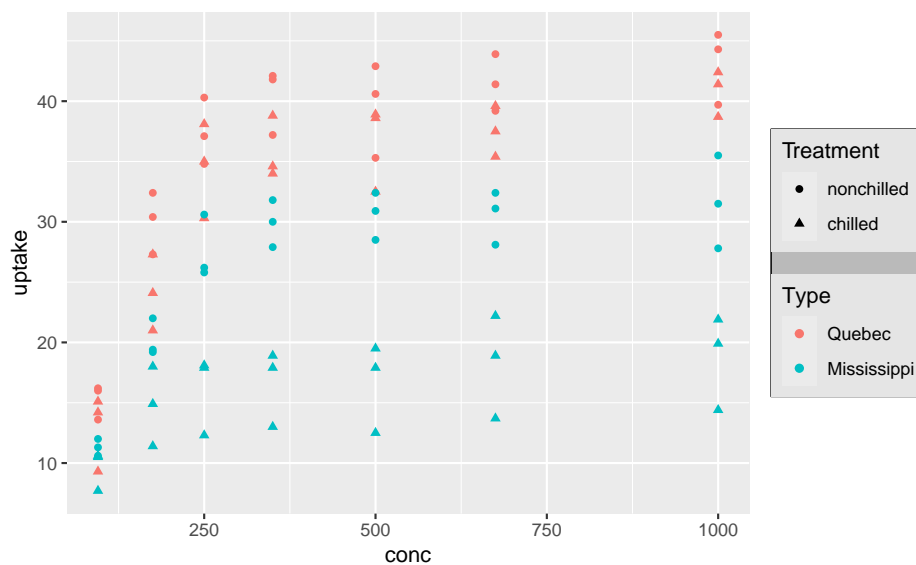
Leģendas ierakstu tekstus maina ar argumentiem `legend.text` = (to izskats) un `legend.text.align` = (to novietojums pa kreisi (0) vai labi (1)) (7.25 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.text = element_text(size = 7,
                                    angle = 5),
        legend.text.align = 0.8)
```

```
## Warning: The `legend.text.align` argument of `theme()` is deprecated as of ggplot2
## 3.5.0.
```

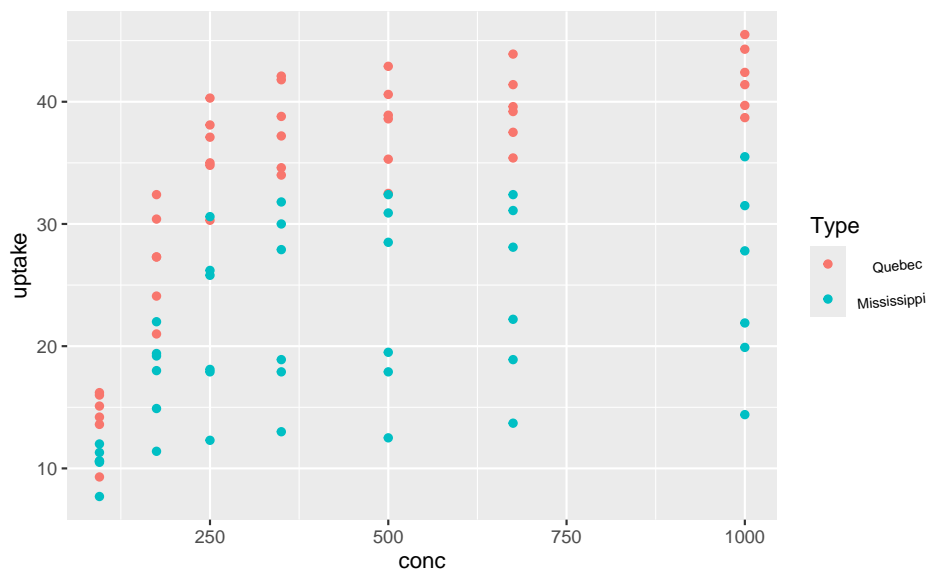


Att. 7.23: Attēls ar mainītu lēģendas pamatni



Att. 7.24: Attēls ar mainītu abu lēģendu pamatni

```
## i Please use theme(legend.text = element_text(hjust)) instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Att. 7.25: Attēls ar mainītu leģendu tekstu

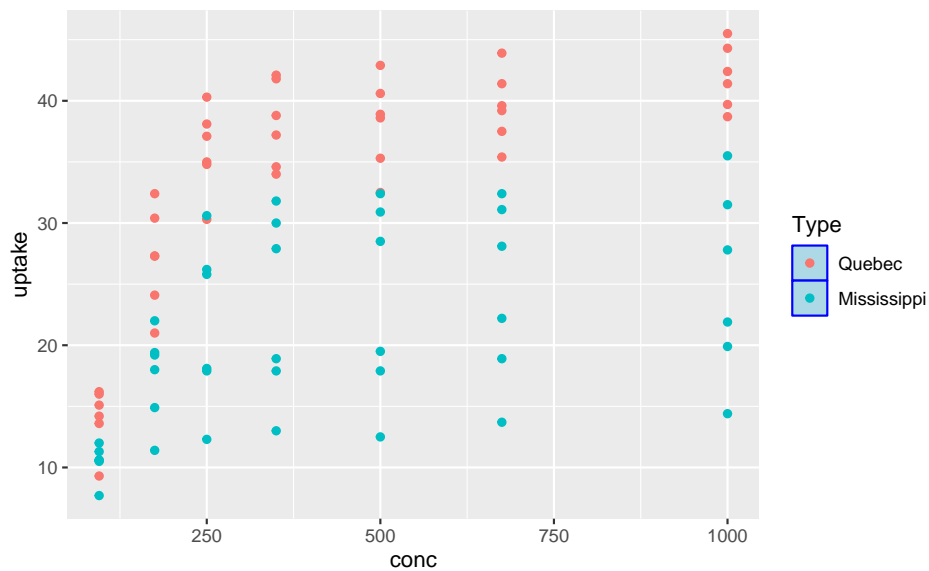
7.2.9 Leģendas ieraksti

Pašu leģendas ierakstu, kuros parādās, piemēram, krāsu līmeņi vai simbolu veidi, mainīšanai ir pieejams parametrs `legend.key =`, kas maina ieraksta pamatnes izskatu (7.26 attēls).

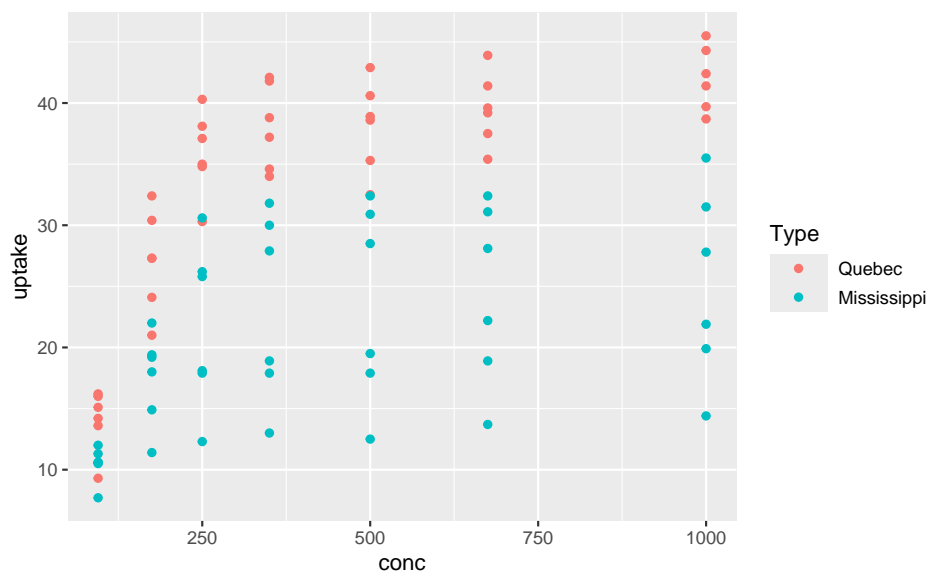
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.key = element_rect(fill = "lightblue",
                                   color = "blue"))
```

Leģendas ierakstu pamatnes izmēru maiņai izmanto argumentu `legend.key.size =`, kas uzreiz maina vienādā apjomā gan platumu, gan augstumu, vai arī parametrus `legend.key.height =` un `legend.key.width =`, kas attiecīgi maina tikai augstumu, vai tikai platumu. Visiem šiem argumentiem ir jāizmanto funkcija `unit()`, lai noteiktu šo izmēru (7.27 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(legend.key.height = unit(0.5, "cm"),
        legend.key.width = unit(1, "cm"))
```



Att. 7.26: Attēls ar mainītu lēģendu ieraksta pamatni

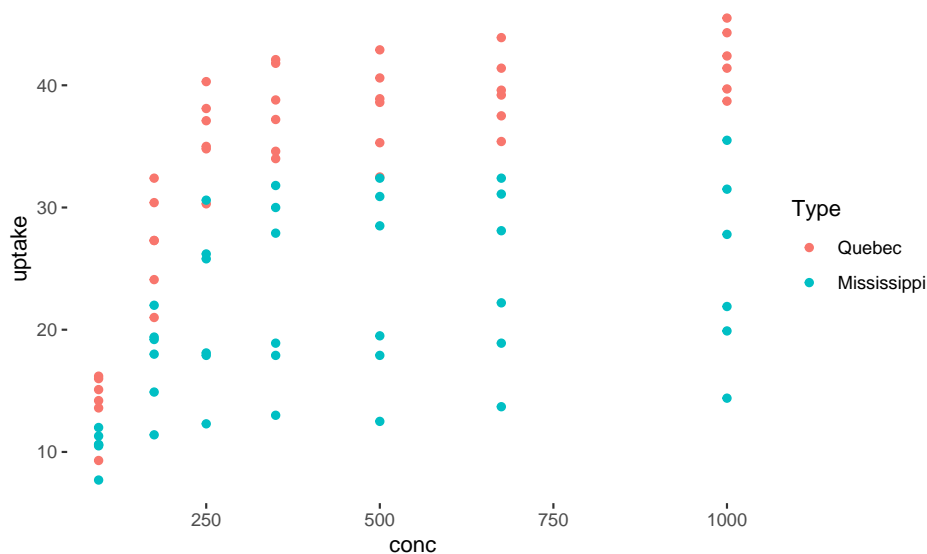


Att. 7.27: Attēls ar mainītu lēģendu ierakstu izmēru

7.2.10 Attēla iekšējās daļas pamatne

Attēla iekšējās daļas pamatnes izskata maina ar argumentu `panel.background =`. Ja šo argumentu lieto kopā ar funkciju `element_blank()`, tad tiek noņemta pamatne, tai skaitā, arī krāsa (aizpildījums) (7.28 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme(panel.background = element_blank())
```



Att. 7.28: Attēls ar noņemta iekšējās daļas pamatni

Lai noņemtu tikai krāsu, arguments `fill =` jānorāda kā `NA`. Ja papildus norāda argumentu `color =`, tad apkārt attēla iekšējai daļai parādās līnija (7.29 attēls).

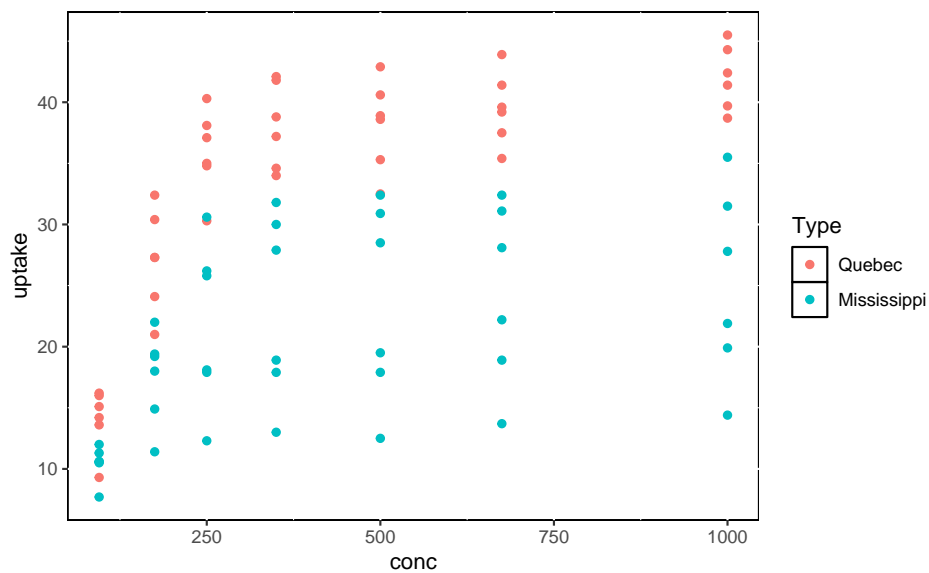
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme(panel.background = element_rect(fill = NA, color = "black"))
```

7.2.11 Palīglinijas (gridlines)

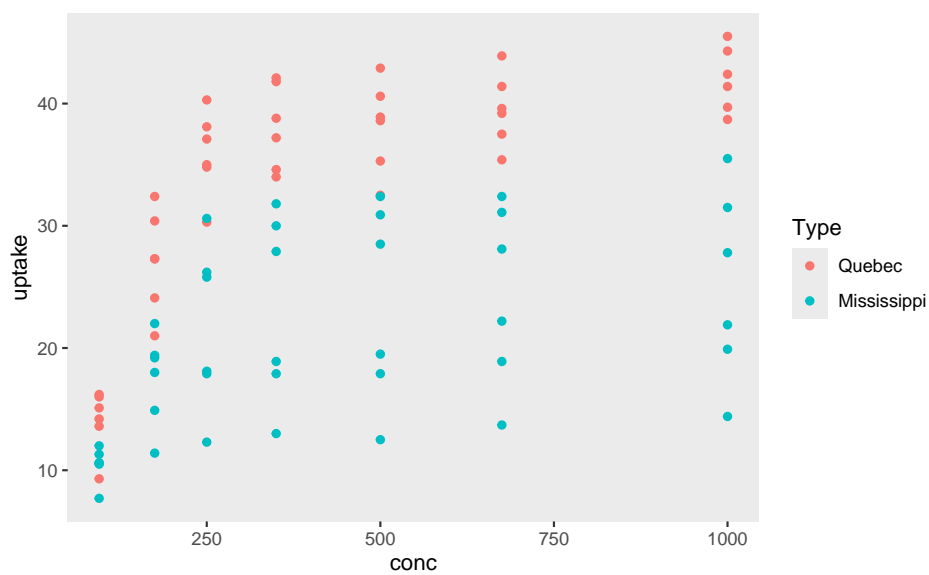
Visu palīgliniju ietekmēšanai jāizmanto arguments `panel.grid =`, piemēram, tā vērtību norādot kā `element_blank()` tiks noņemtas visas palīglinijas (7.30 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme(panel.grid = element_blank())
```

Ir iespējams arī ietekmēt atsevišķi visas galvenās palīglinijas un mazās palīglinijas, attiecīgi ar argumentiem `panel.grid.major =` un `panel.grid.minor =`. Var ietekmēt arī atsevišķi šīs palīglinijas attiecībā pret x un y ass ar argumentiem kā `panel.grid.major.x =` (7.31 attēls).

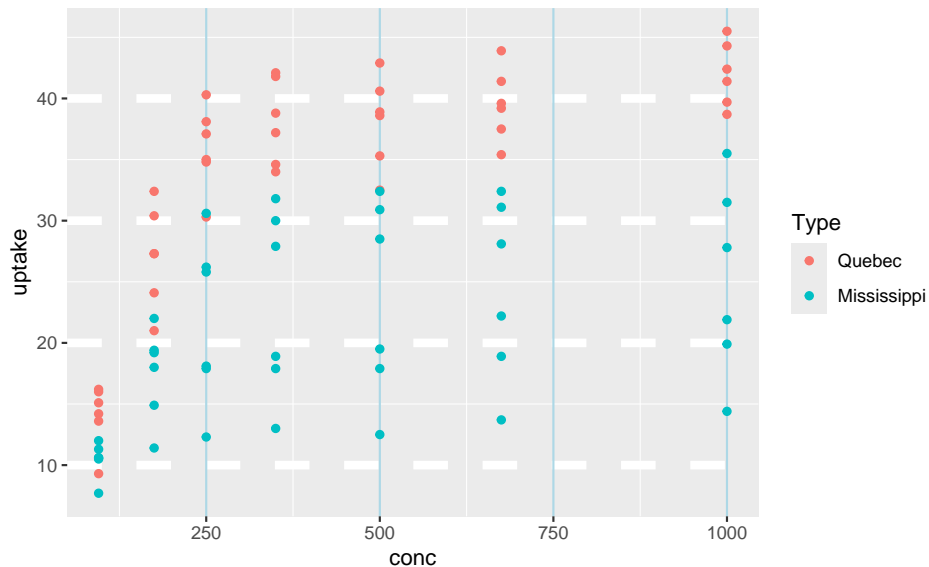


Att. 7.29: Attēls ar līniju apkārt iekšējai daļai



Att. 7.30: Attēls bez palīglinijām

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(panel.grid.major.x = element_line(colour = "lightblue"),
        panel.grid.major.y = element_line(linetype = "dashed",
                                          linewidth = 2))
```

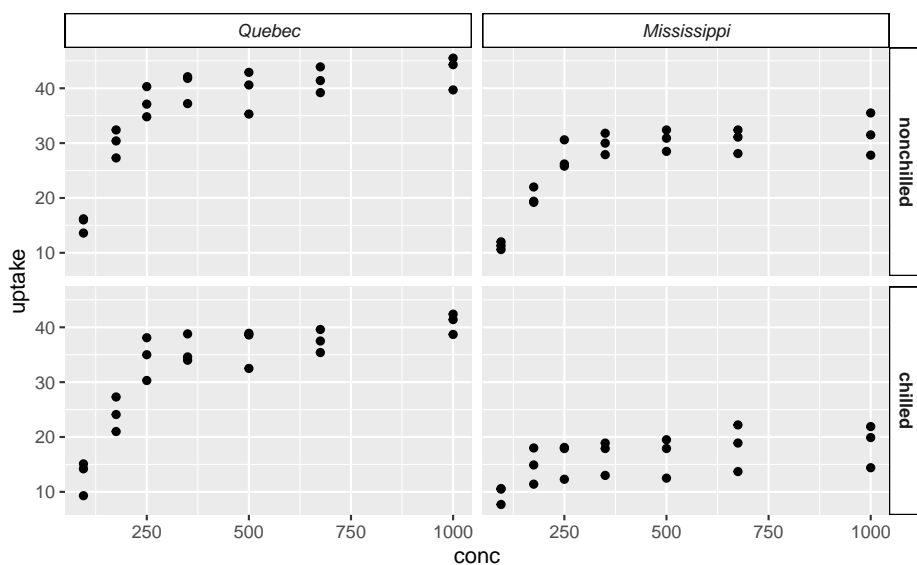


Att. 7.31: Attēls ar izmainītām palīglinijām

7.2.12 Attēla daļu virsraksti

Tiem attēliem, kas ir sadalīti daļās izmantojot `facet_...()` funkcijas, atsevišķu attēlu nosaukumu tekstu pamatnes maina ar elementu `strip.background =`, bet pašu tekstu izskatu ar `strip.text =` (visus kopā), vai arī ar argumentiem `strip.text.x =` (horizontālie nosaukumi) un `strip.text.y =` (vertikālie nosaukumi) (?? attēls).

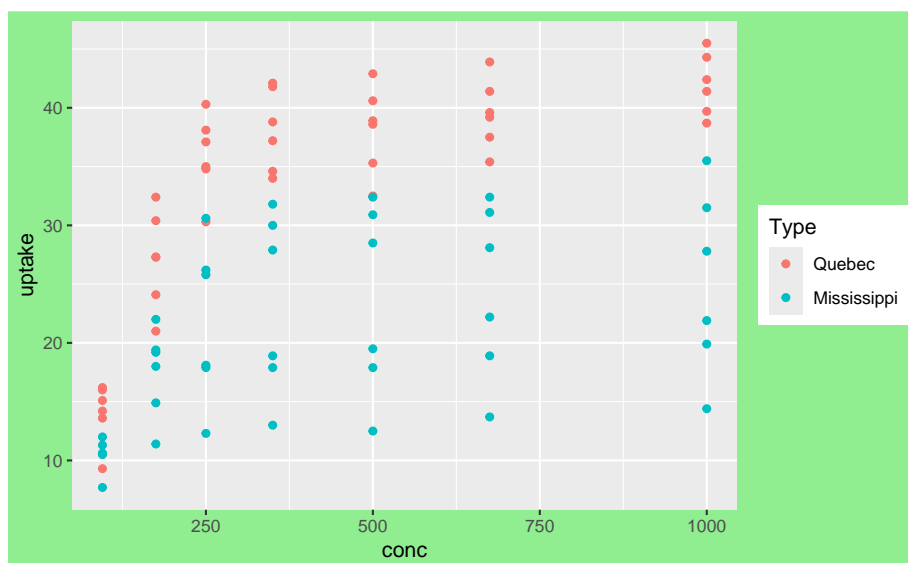
```
ggplot(CO2, aes(conc, uptake)) + geom_point() +
  facet_grid(Treatment ~ Type) +
  theme(strip.background = element_rect(colour = "black", fill = NA),
        strip.text.x = element_text(face = "italic"),
        strip.text.y = element_text(face = "bold"))
```

Visa attēla pamatne

Visa kopējā attēla pamatnes (fona) noteikšanai izmanto argumentu `plot.background =` (7.32 attēls).

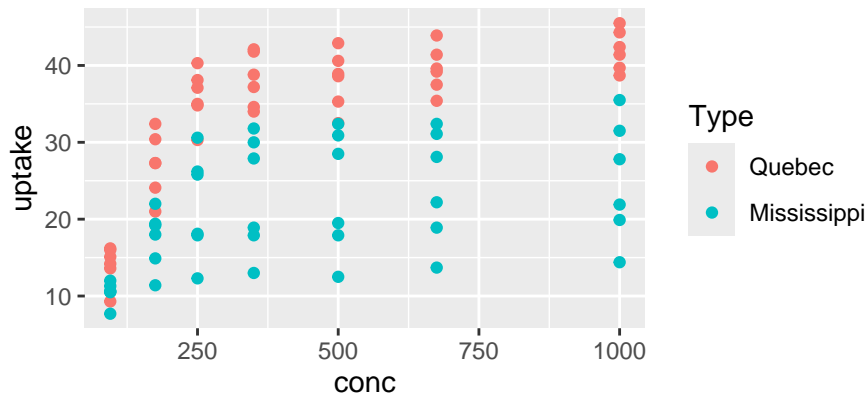
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +  
  theme(plot.background = element_rect(fill = "lightgreen"))
```



Att. 7.32: Attēls ar mainītu pamatnes krāsu

Visam attēlam var mainīt arī malas izmērus, kas atrodas apkārt. To nosaka ar argumentu `plot.margin` un funkcijā `unit()` norādot četrus skaitļus, kas atbilst attiecīgi augšējai, labajai, apakšējai un kreisajai malai (7.33 attēls).

```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(plot.margin = unit(c(2, 2, 3, 3), "cm"))
```



Att. 7.33: Attēls ar mainītiem malas izmēriem

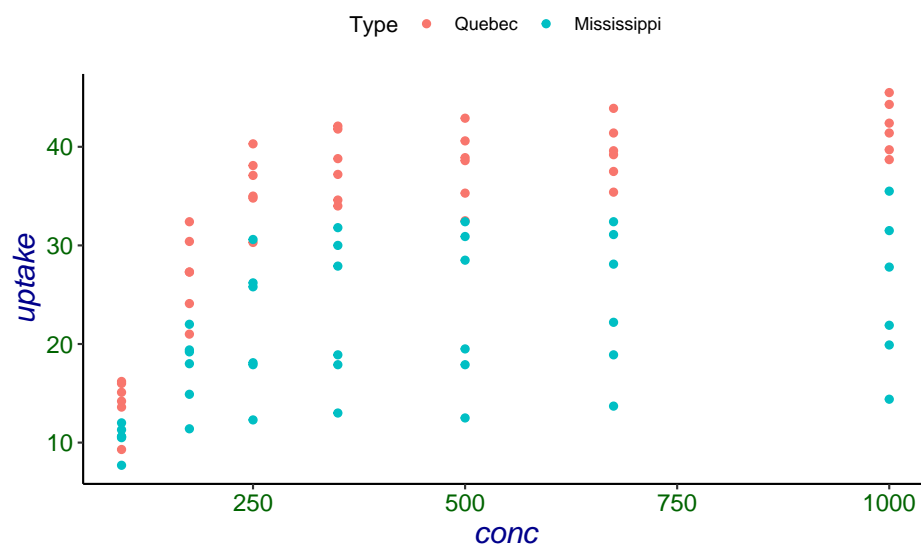
7.3 Noformējuma saglabāšana kā objektu

Attēla noformējuma nosacījumus nav obligāti jāpievieno attēlam kā atsevišķas funkcijas `theme()`, bet visus nepieciešamos elementus var norādīt vienās `theme()` iekavās (7.34 attēls).

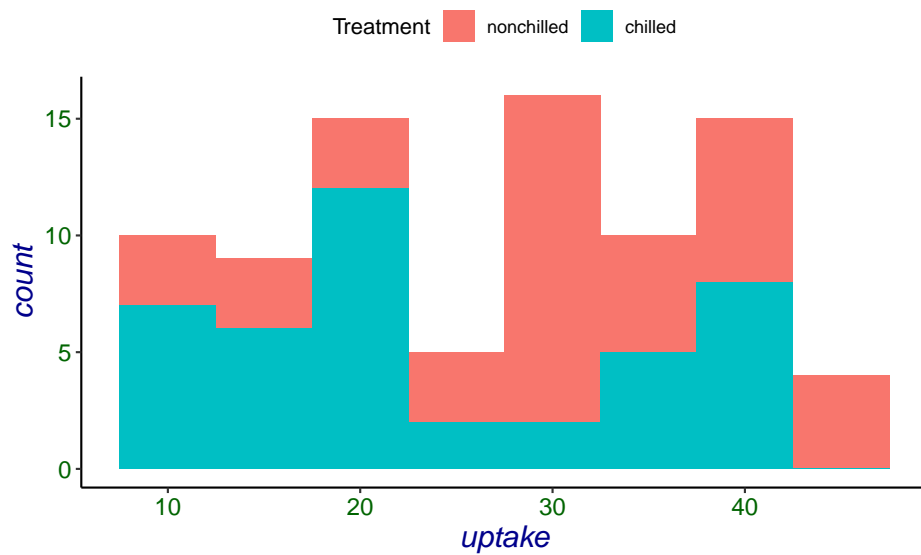
```
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme(axis.text = element_text(size = 12, color = "darkgreen"),
        axis.title = element_text(size = 15, color = "darkblue", face = "italic"),
        panel.background = element_rect(fill = NA),
        axis.line = element_line(color = "black"),
        legend.position = "top")
```

Noformējumu ir iespējams arī definēt kā atsevišķu objektu, kuru tad var izmantot atkāroti, lai vairākiem attēliem iegūtu identisku noformējumu (7.35 attēls).

```
tema <- theme(axis.text = element_text(size = 12, color = "darkgreen"),
              axis.title = element_text(size = 15, color = "darkblue", face = "italic"),
              panel.background = element_rect(fill = NA),
              axis.line = element_line(color = "black"),
              legend.position = "top")
ggplot(CO2, aes(uptake, fill = Treatment)) + geom_histogram(binwidth = 5) +
  tema
```



Att. 7.34: Attēls, kuram mainīti vairāki attēla elementi



Att. 7.35: Attēls, kuram mainīti vairāki attēla elementi

Nodaļa 8

ggplot2 papildinājumi

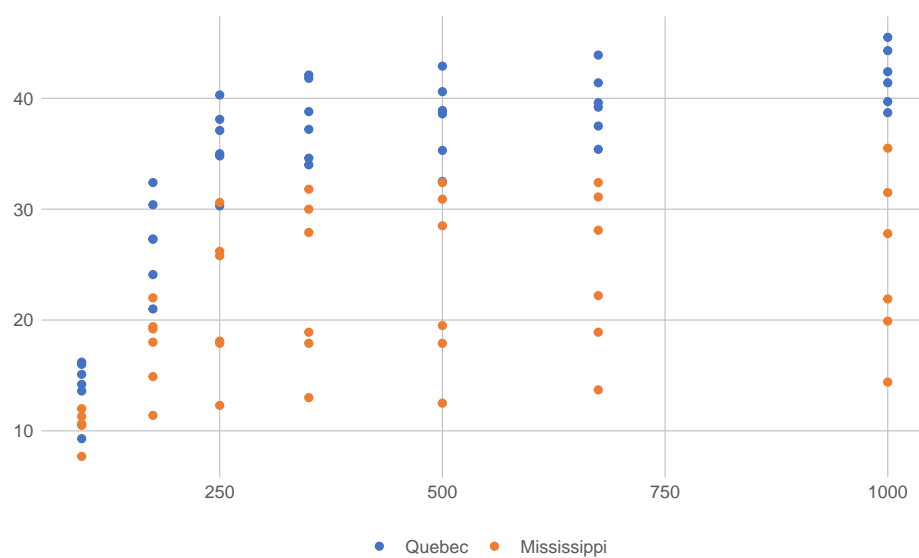
ggplot2 grafiskajai sistēmai ir ļoti daudzi papildinājumi (R paketes), kas ievieš jaunas formas, skalas, noformējuma tēmas un citas papildus iespējas. Mājaslapā <https://exts.ggplot2.tidyverse.org> var iepazīties ar daļu no šīm papildus paketēm.

8.1 ggthemes

Pakete `ggthemes` (Arnold (2024)) satur funkcijas, kas piedāvā dažas jaunas formas, vairākas gatavās noformējuma tēmas, skalas un krāsu paletes.

Piemērā (8.1 attēls) attēlam piemērota jaunā Excel tēma (funkcija `theme_excel_new()`) un atbilstošā krāsu skala (funkcija `scale_color_excel_new()`).

```
library(ggthemes)
ggplot(CO2, aes(conc, uptake, color = Type)) + geom_point() +
  theme_excel_new() +
  scale_color_excel_new()
```



Att. 8.1: Attēls ar Excel tēmu

Literatūra

Arnold, J. B. (2024). *ggthemes: Extra Themes, Scales and Geoms for 'ggplot2'*.
R package version 5.1.0.

R Core Team (2023). *R: A Language and Environment for Statistical Computing*.
R Foundation for Statistical Computing, Vienna, Austria.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-
Verlag New York.