Ievads ggplot2 grafiskajā sistēmā

Didzis Elferts

Pamatojums

- ggplot2 pakete atšķiras no citām diagrammu veidošanas paketēm ar to, ka tās pamatā ir noteikta ”gramatika” (Wilkinson, 2005. Grammar of graphics)
- Sistēma sastāv no vairākām atseviškām komponentēm, kuras var dažādā veidā kombinēt
- Ir iespēja veidot jaunus diagrammu veidus, kas parāda Jums aktuālo problēmu

Papildus informācija

Oficiālā dokumentācija:

http://docs.ggplot2.org/current/

R attēlu ”recepšu grāmata”:

http://www.cookbook-r.com/Graphs/

Jautājumi un atbildes:

http://stackoverflow.com/questions/tagged/ggplot2

Sistēmas komponentes

`ggplot2` attēli tiek veidoti no slāņiem:

- **data** – informācija, kuru vēlas attēlot
- **aes** jeb **aesthetics** – datu attēlošanas veids – simbolu veids, krāsa utt
- **geom** – ģeometriskie objekti (līnija, punkti, poligoni), kurus reāli redz
- **stats** – statistiskās transformācijas datiem
- **scale** – nodrošina datu vērtību attēlojumu atbilstoši izvēlētajiem datu parādīšanas veidam (aesthetics), kā arī veido leģendas un asis
- **coord** – koordinātšu sistēma, ko izmanto diagrammā
- **facet** – nosaka kā sadalīt datus pa atsevišķām diagrammām

geom veidi

- `geom_point()` – zīmē punktus, lai veidotu izkliedes diagrammu
- `geom_smooth()` – zīmē līniju, kas izlīdzina datus, kā arī šīs līnijas standartkļūdu
- `geom_boxplot()` – veido vērtībamplitūdas (box-and-whisker) diagrammu
- `geom_path()` un `geom_line()` – zīmē līniju, kas savieno punktus
- `geom_abline()`,`geom_hline()`,`geom_vline()` – zīmē līnijas
- `geom_histogram()` – veido histogrammu
- `geom_density()` – veido blīvuma diagrammu
- `geom_bar()` – veido joslu jeb stabiņu diagrammu

Paketes

```
# Pamatpakete attēlu atveidošanai
if (!require("ggplot2") )  install.packages("ggplot2")
library(ggplot2)
# Pakete nepieciešama, ja jāizmanto funkcija unit() (mērvienības)
if (!require("grid") )  install.packages("grid")
library(grid)
# Pakete, kurā ir papildus skalas un to transformācijas
if (!require("scales") )  install.packages("scales")
library(scales)
```

Dati

```
dati <- read.csv(file="../Dati/augi.csv",header=TRUE,sep=",",dec=".")
head(dati)
```

```
##   gaisma stress lapas garums
## 1   zema    nav   264     91
## 2   zema    nav   200     80
## 3   zema    nav   225     88
## 4   zema    nav   268    109
## 5   zema    nav   215     76
## 6   zema    nav   241     96
```

Attēlu veidošana

`ggplot2` sistēmā attēlus veido ar divām funkcijām - `qplot()` un `ggplot()`. Pirmā funkcija paredzēta ātrākai attēla izveidošanai, bet vienlaicīgi tai ir daudz mazāka iespēja tikt modificētai. `ggplot()` funkcijas gadījumā ir visas iespējas veikt katra elementa modifikāciju.

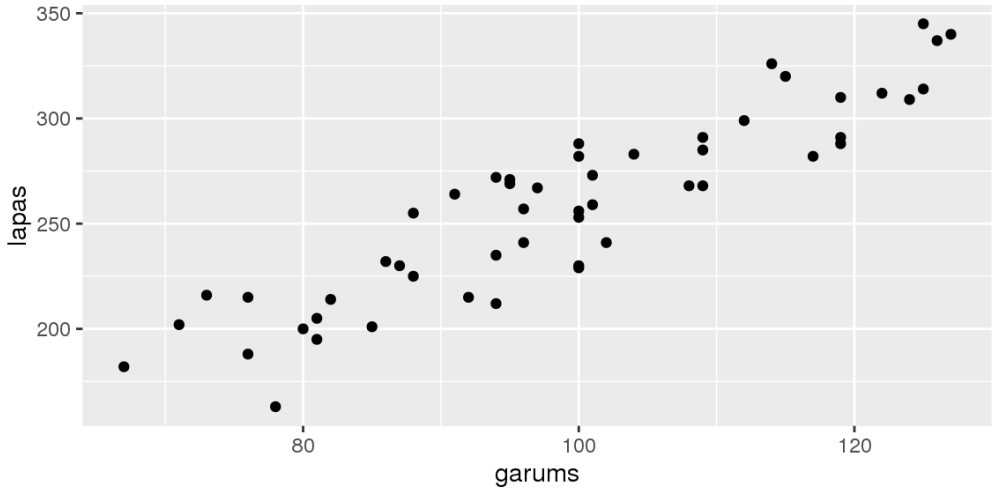
Funkcijā `ggplot()` kā pirmais arguments ir jānorāda datu tabulu (data frame), no kurienes ņemt datus, tad iekšā ir funkcija `aes()`, kur attiecīgi norāda x un y vērtības (atdalītas ar komatu).

`ggplot2` sistēmā **NAV** jāizmanto mainīgo pieraksts ar `$` zīmi.

geom_point()

Lai izveidotu punktu (izkliedes) attēlu, `ggplot()` norāda datu tabulu un x/y vērtības, un tam pieskaita nepieciešamo `geom_...` veidu.

```
ggplot(dati, aes(garums, lapas)) + geom_point()
```



Izskata maiņa

Lai mainītu punktu, līniju, poligonu izskatu (krāsu, lielumu, formu, caurspīdīgumu), ir divas iespējas:

1. Ja izskatam ir jābūt vienādam visiem elementiem, piemēram, visi punkti sarkani, tad atbilstošais arguments ir jāievieto `geom_...` funkcijā ārpus `aes()` iekavām un jānorāda atbilstošā vērtība.

2. Ja izskatam ir jāmainās atbilstoši kādam mainīgajam (piemēram, katrai sugai cita krāsa), tad atbilstošais arguments ir jāievieto funkcijā `geom_...` vai `ggplot()`, bet OBLIGĀTI iekšā `aes()` iekavām un kā vērtība jānorāda mainīgā nosaukums.

Faktori un skaitliskas vērtības

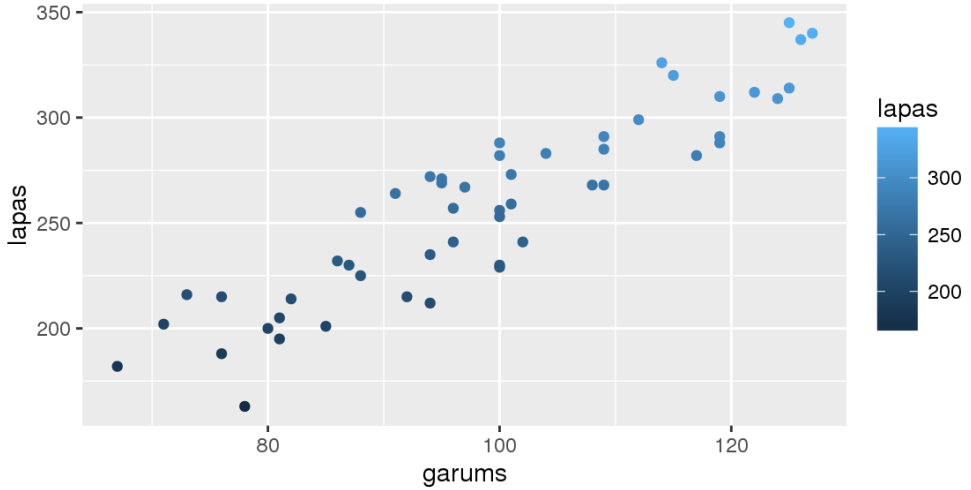
1. Ja pie izskata (krāsas, caurspīdīguma, lieluma) maiņas argumenta norāda skaitliskas vērtības, tad atbilstošie elementi mainās kā gradients (no mazākās uz lielāko vērtību).

2. Ja pie izskata maiņas argumenta norāda faktoru (vai skaitli, kas pārvērsts par faktoru), tad atbilstošie elementi mainās kā diskrētas (atsevišķas) vērtības.

geom_point()

Krāsa kā gradients

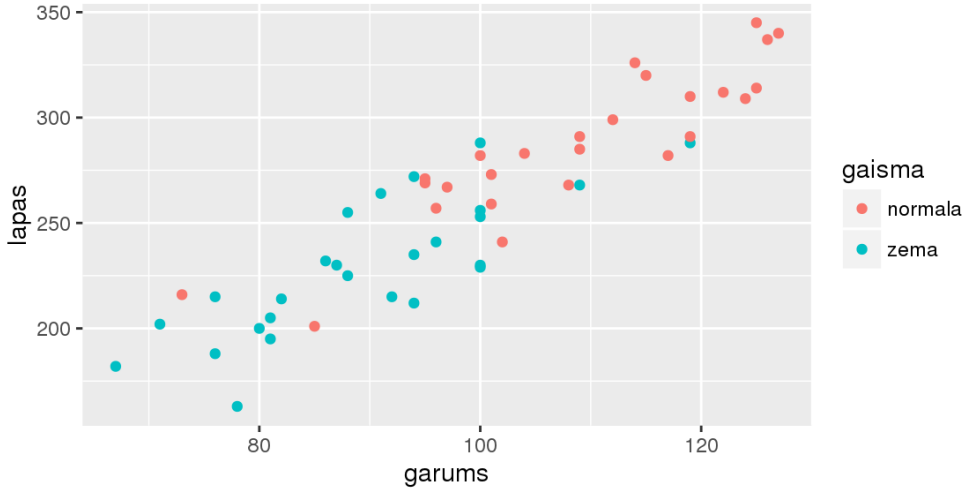
```
ggplot(dati, aes(garums, lapas,color=lapas)) + geom_point()
```



geom_point()

Krāsa kā diskrēta vērtība

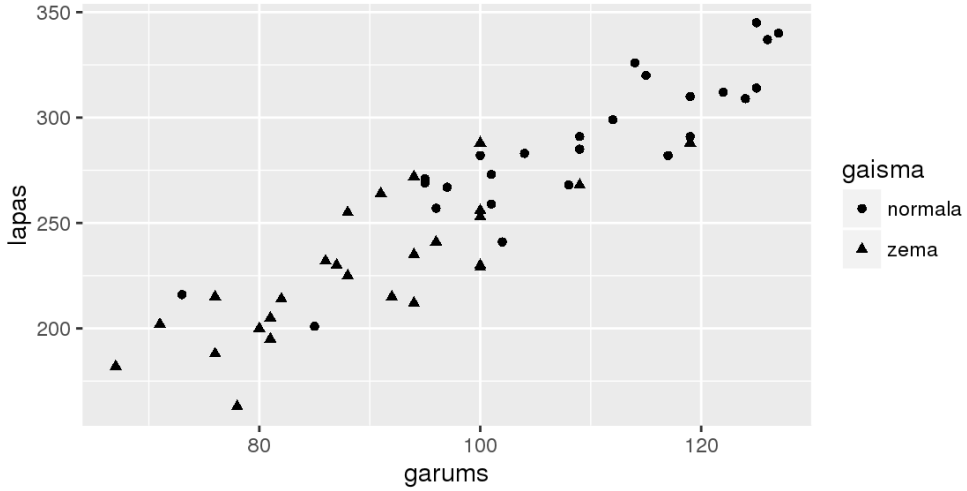
```
ggplot(dati, aes(garums, lapas,color=gaisma)) + geom_point()
```



geom_point()

Forma kā diskrēta vērtība

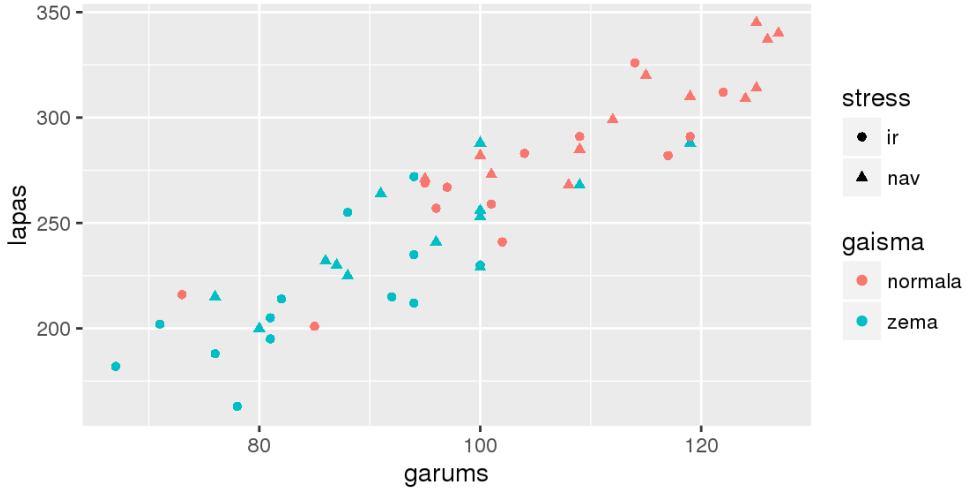
```
ggplot(dati, aes(garums, lapas,shape=gaisma)) + geom_point()
```



geom_point()

Krāsas un izmēra kombinācija

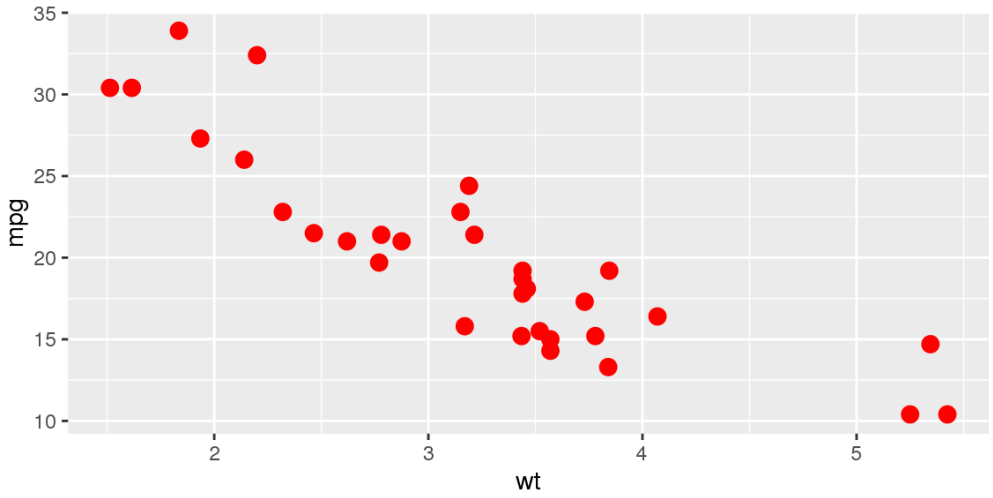
```
ggplot(dati, aes(garums, lapas,color=gaisma,shape=stress)) + geom_point()
```



geom_point()

Krāsa un izmērs, kas noteikts visiem punktiem vienāds

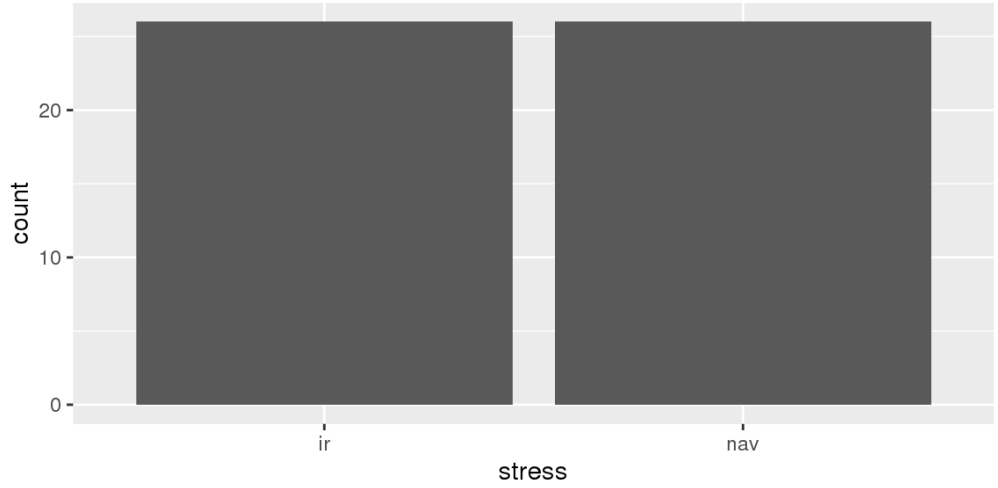
```
ggplot(mtcars, aes(wt, mpg)) + geom_point(colour = "red", size = 3)
```



geom_bar()

Veidojot stabiņu diagrammu (`geom_bar()`) ir nepieciešamas tikai x vērtības (faktors), jo atkārtojumu skaitu nosaka automātiski.

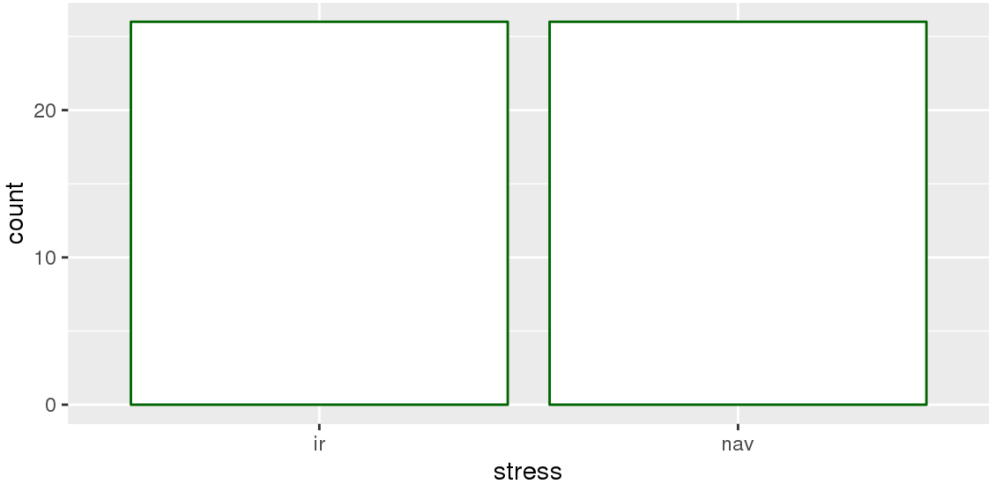
```
ggplot(dati, aes(stress))+geom_bar()
```



geom_bar()

Stabiņam ir iespējams noteik krāsu (līniju, kas ir apkārt) un aizpildījumu (`fill=`).

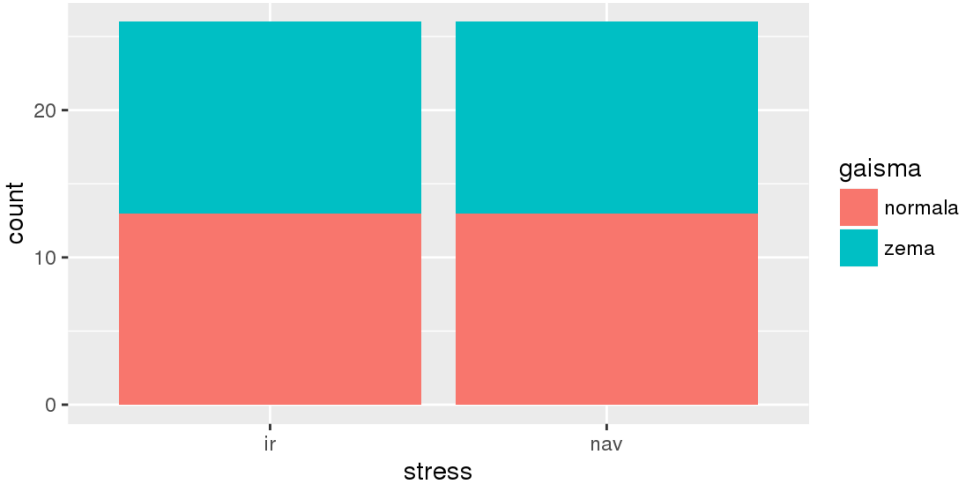
```
ggplot(dati, aes(stress)) + geom_bar(fill="white", colour="darkgreen")
```



geom_bar()

Ja aizpildījumu norāda kā mainīgo `aes()`, pēc noklusējuma stabiņi tiek sadalīti pa daļām atbilstoši mainīgajam.

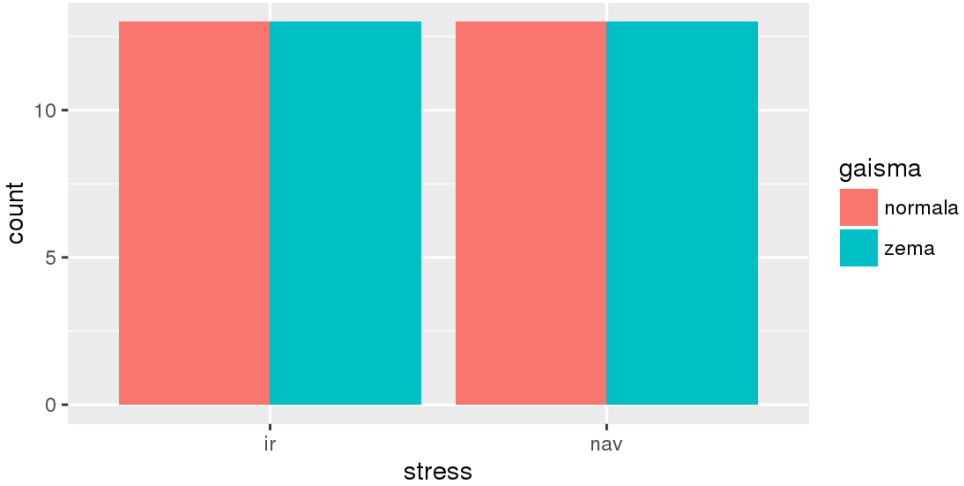
```
ggplot(dati, aes(stress,fill=gaisma)) + geom_bar()
```



geom_bar()

Lai iegūtu stabiņus, kas sadalīti pēc mainīgā un būtu novietoti blakus, papildus jānorāda arguments `position="dodge"`.

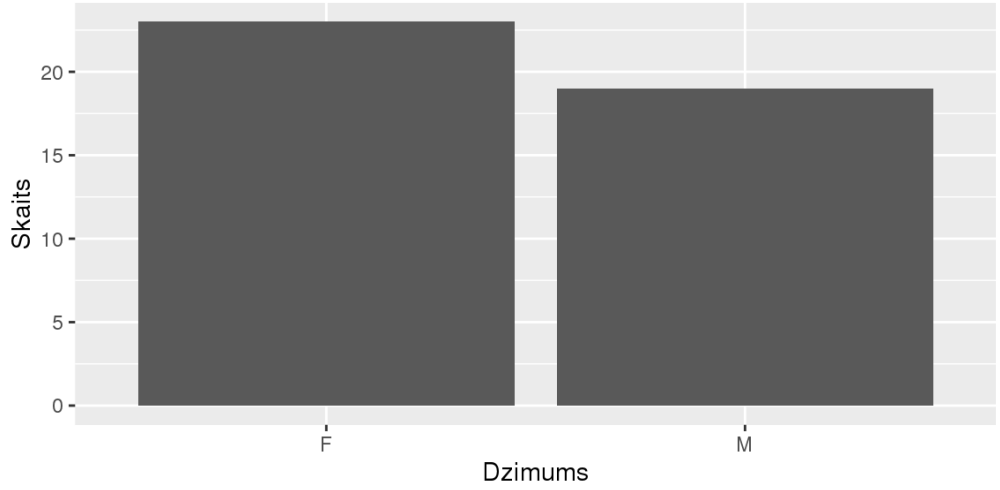
```
ggplot(dati, aes(stress,fill=gaisma)) + geom_bar(position="dodge")
```



geom_bar()

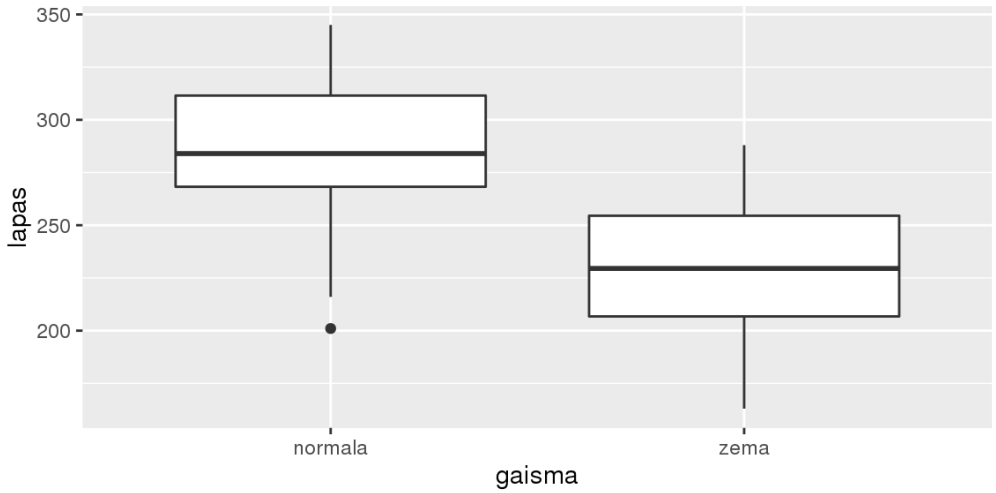
Ja y vērtības stabiņu attēlam jau ir zināmas, tad kā papildus arguments jānorāda `stat="identity"`.

```
df<-data.frame(Dzimums=c("F","M"),Skaits=c(23,19))
ggplot(df,aes(Dzimums,Skaits))+geom_bar(stat="identity")
```



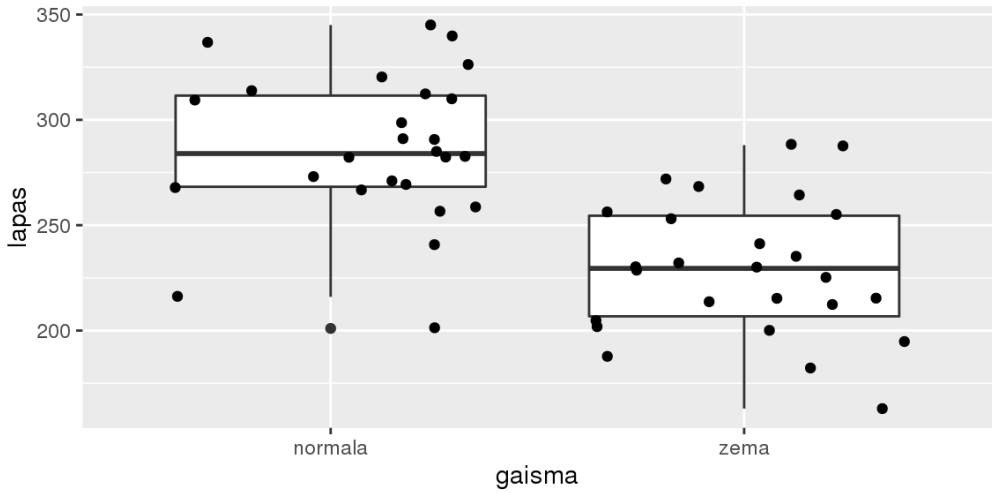
geom_boxplot()

```
ggplot(dati, aes(gaisma, lapas)) + geom_boxplot()
```



geom_boxplot()

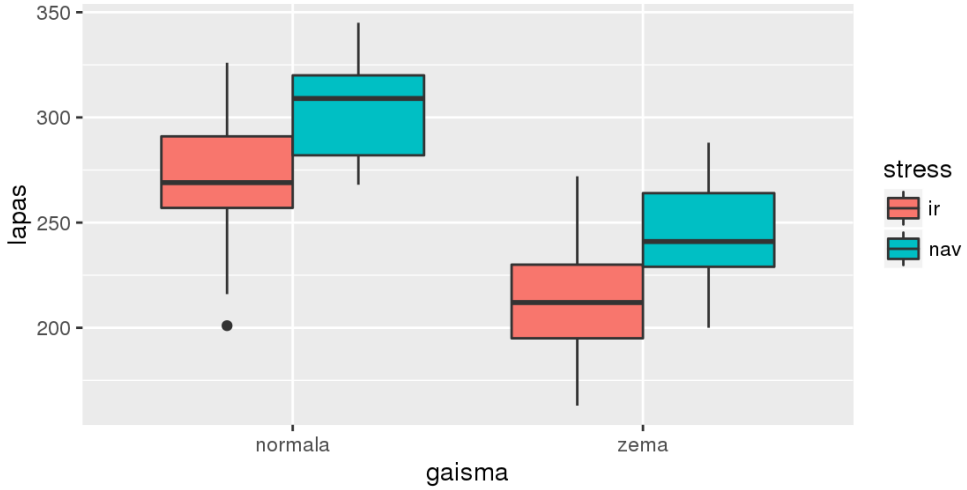
```
ggplot(dati, aes(gaisma, lapas)) + geom_boxplot() + geom_jitter()
```



geom_boxplot

Boxplot attēliem līdzīgi kā stabiņu attēliem krāsa (`colour=`) attiecas uz ārējo malu un aizpildījums (`fill=`) uz iekšējo krāsojumu. Ja kā aizpildījumu norāda mainīgo, kas nav y vērtība, pie katras y vērtības boxplot attēli tiek sadalīti atbilstoši šim mainīgajam.

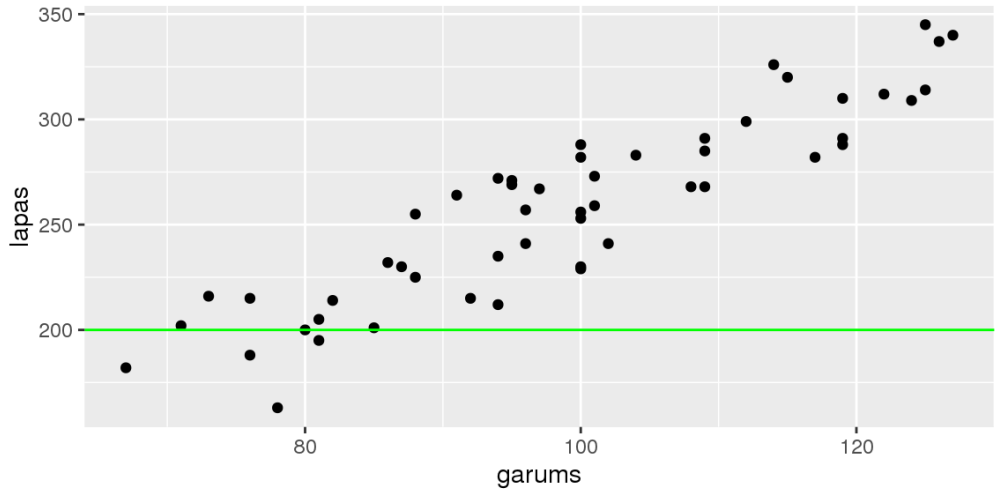
```
ggplot(dati, aes(gaisma, lapas,fill=stress)) + geom_boxplot()
```



geom_hline()

Horizontālu līniju pievienošanai izmanto funkciju `geom_hline()`, kurai kā argumentu norāda `yintercept=`, kas var būt gan iekļauts `aes()`, gan ārpus tā.

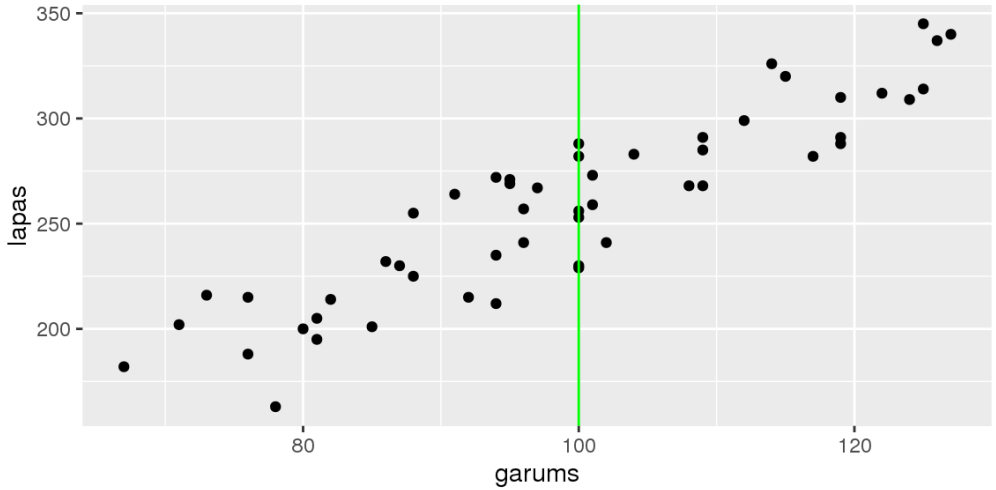
```
ggplot(dati, aes(garums,lapas)) + geom_point()+geom_hline(yintercept=200,color="green")
```



geom_vline

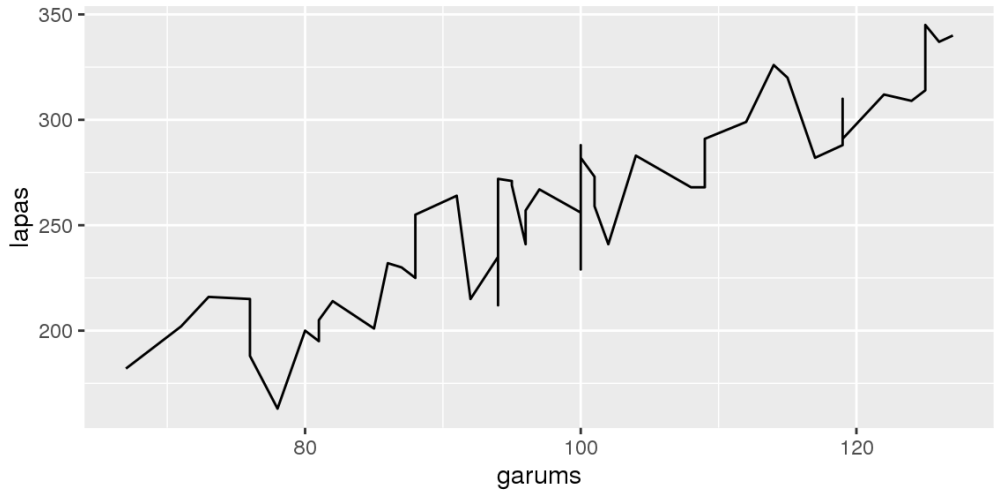
Vertikālu līniju pievienošanai izmanto `geom_vline()` un argumentu `xintercept=`.

```
ggplot(dati, aes(garums,lapas)) + geom_point()+geom_vline(xintercept=100,color="green")
```



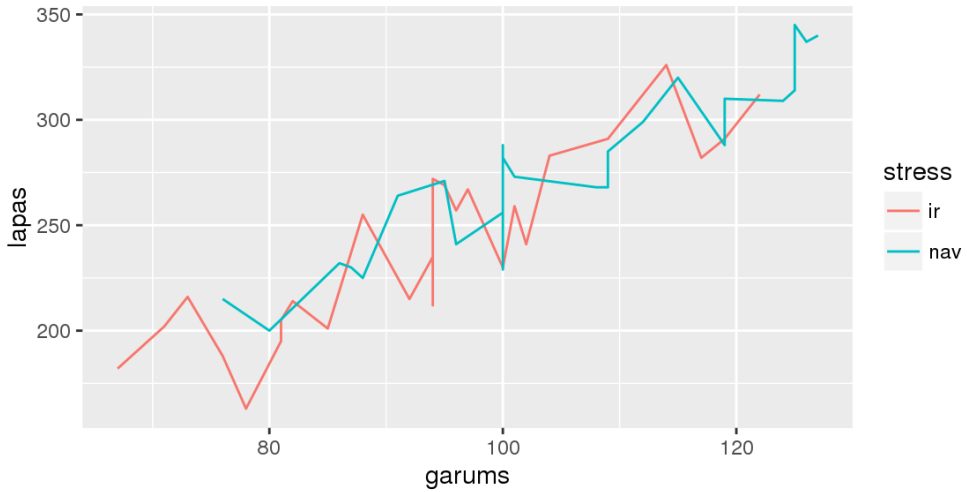
geom_line()

```
ggplot(dati, aes(garums,lapas)) + geom_line()
```



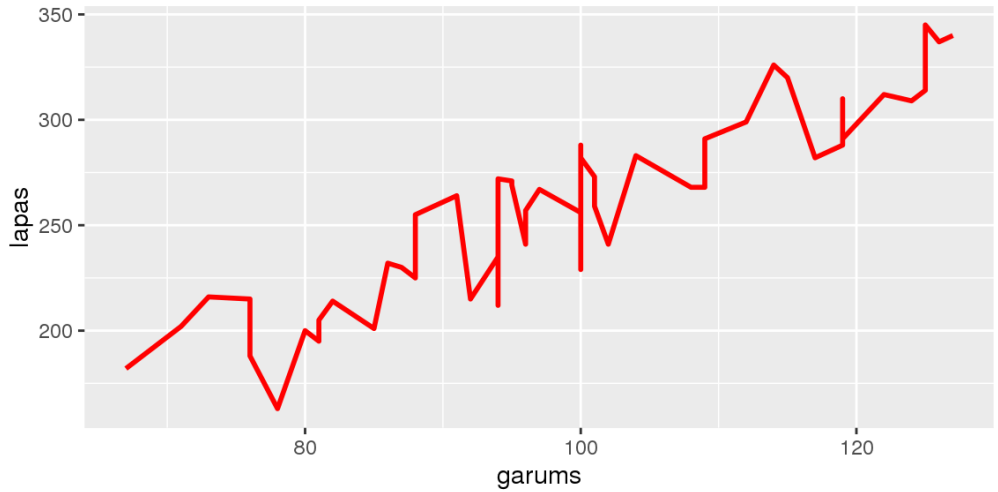
geom_line()

```
ggplot(dati, aes(garums,lapas,color=stress)) + geom_line()
```



geom_line()

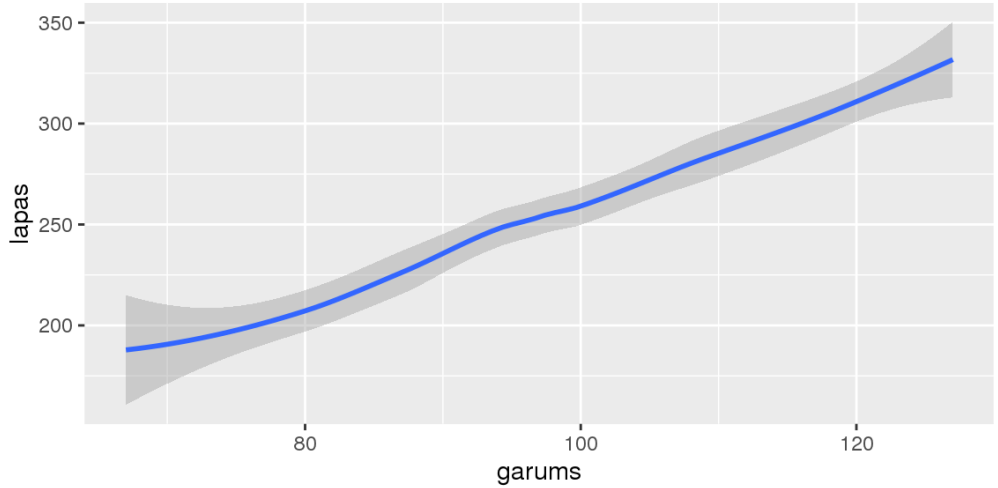
```
ggplot(dati, aes(garums,lapas)) +  geom_line(colour = "red", size = 1)
```



geom_smooth()

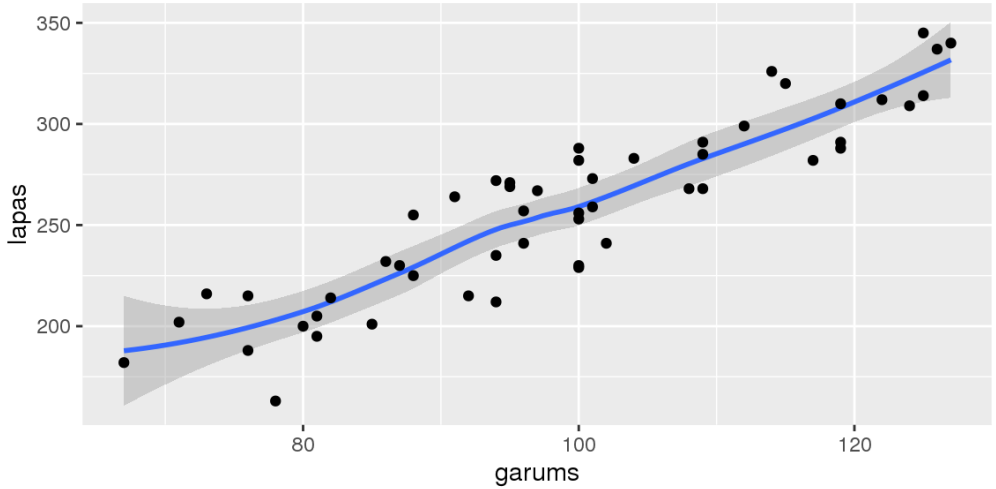
Ar funkciju `geom_smooth()` var pievienot "trenda" līniju kopā ar ticamības intervālu. Izlīdzinātās līnijas veids tiek noteikts automātiski.

```
ggplot(dati,aes(garums,lapas)) + geom_smooth()
```



stat_smooth()

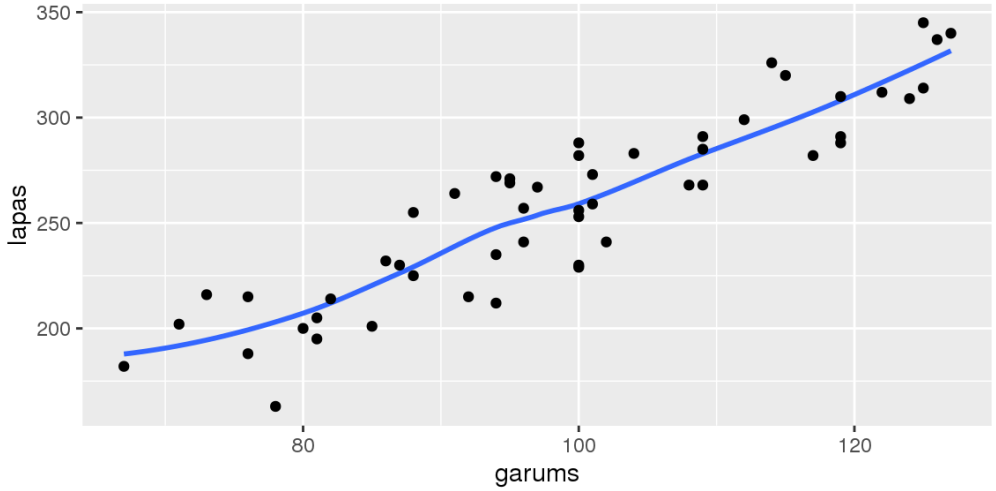
```
ggplot(dati,aes(garums,lapas)) + geom_smooth() + geom_point()
```



geom_smooth()

Ar argumentu se=FALSE var noņemt ticamības intervālu.

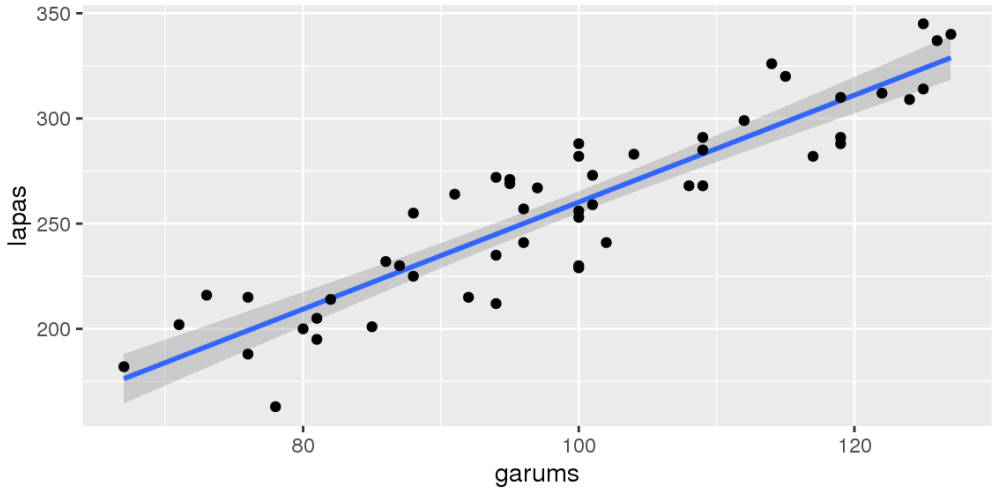
```
ggplot(dati,aes(garums,lapas)) + geom_smooth(se = FALSE) + geom_point()
```



geom_smooth()

Ja nepieciešams pievienot tieši lineāru trendu, tad jālieto arguments `method="lm"`.

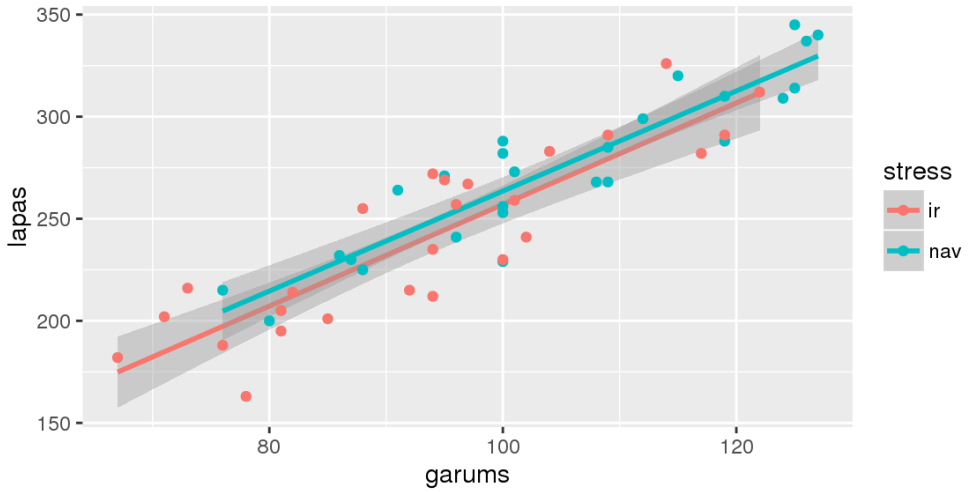
```
ggplot(dati,aes(garums,lapas))+ geom_smooth(method = "lm") + geom_point()
```



geom_smooth()

Nosakot krāsu (līnijai) vai aizpildījumu (ticamības intervāliem) atkarībā no mainīgā, trenda līnijas parādīsies katram līmenim atsevišķi.

```
ggplot(dati,aes(garums,lapas,color=stress))+ geom_smooth(method = "lm") + geom_point()
```



Skalas

Izmantojot funkcijas `scale_..._...()` var mainīt ne tikai x un y asu noformēju un veidu, bet arī ietekmēt visus citus atribūtus, kas tiek noteikti izmantojot `aes()`: krāsas, lielumus, punktu formas, līniju veidus, apzīmējumu caurspīdīgumu.

Skalu piemēri:

- `scale_x_discrete()`,`scale_y_discrete()`
- `scale_x_continuous()`,`scale_y_continuous()`
- `scale_colour_discrete()`,`scale_colour_continuous()`,`scale_colour_grey` `scale_colour_gradient()`
- `scale_fill_discrete()`,`scale_fill_continuous()`,`scale_fill_grey()`, `scale_fill_gradient()`

Skalas

Skalu piemēri:

- `scale_linetype_discrete()`,`scale_linetype_continuous()`, `scale_linetype_manual()`,`scale_linetype_identity()`
- `scale_x_log10()`,`scale_x_reverse()`,`scale_x_sqrt()`

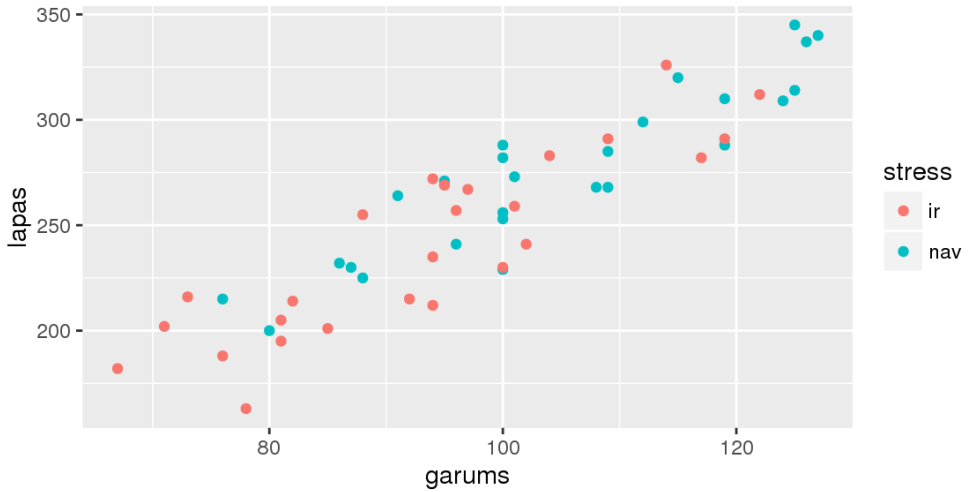
Skalas

Skalām var mainīt sekojošos parametrus:

- name - nosaukums
- breaks - dalījuma vietas
- labels - apzīmējumi dalījuma vietās
- limits - vērtību diapozons
- values - lietotāja definētās krāsas/līniju veidi/simbolu veidi - izmanto tikai ar `scale_..._manual()`

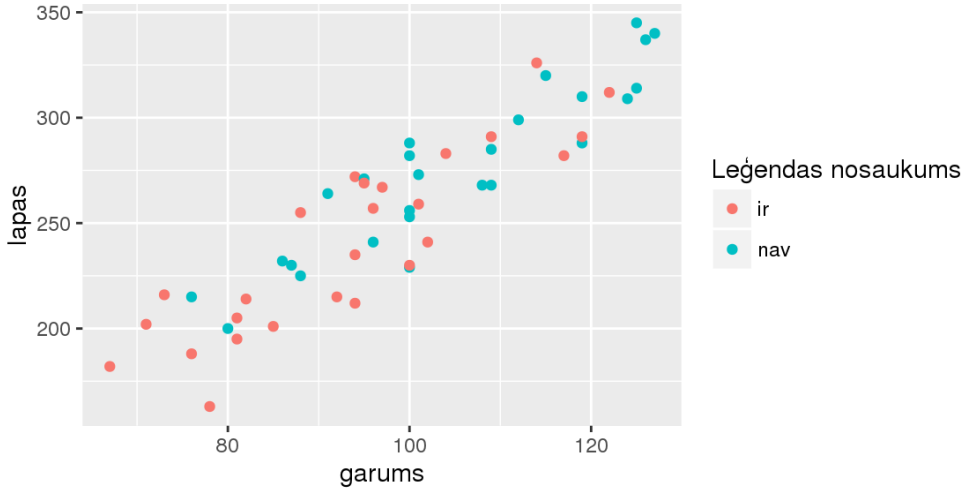
scale_colour_discrete()

```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point()
```



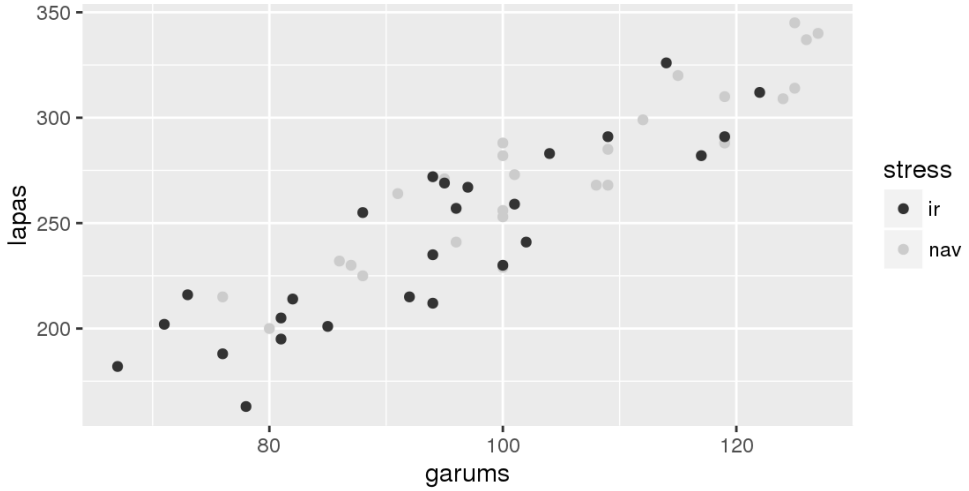
scale_colour_discrete()

```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point() +
      scale_colour_discrete("Leģendas nosaukums")
```



scale_colour_grey()

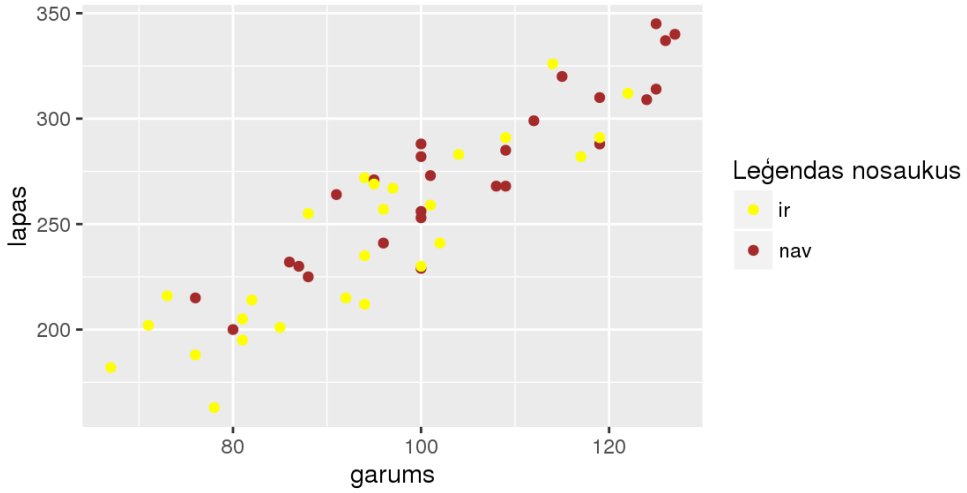
```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point() +
      scale_colour_grey()
```



scale_colour_manual()

Izmantojot `scale_..._manual()` ir iespējams izraudzīties paša definētās vērtības (to skaitam jāatbilst līmeņu skaitam).

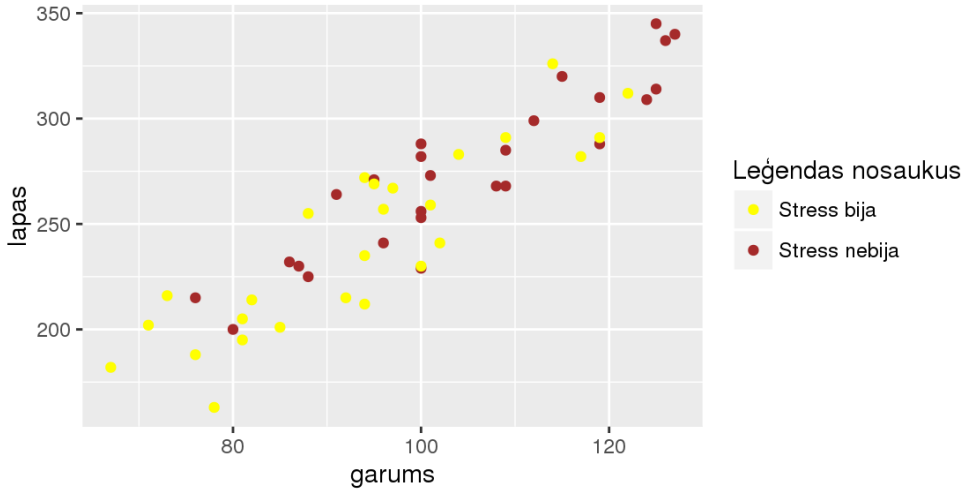
```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point()  +
      scale_colour_manual("Leģendas nosaukus",values = c("yellow","brown"))
```



scale_colour_manual()

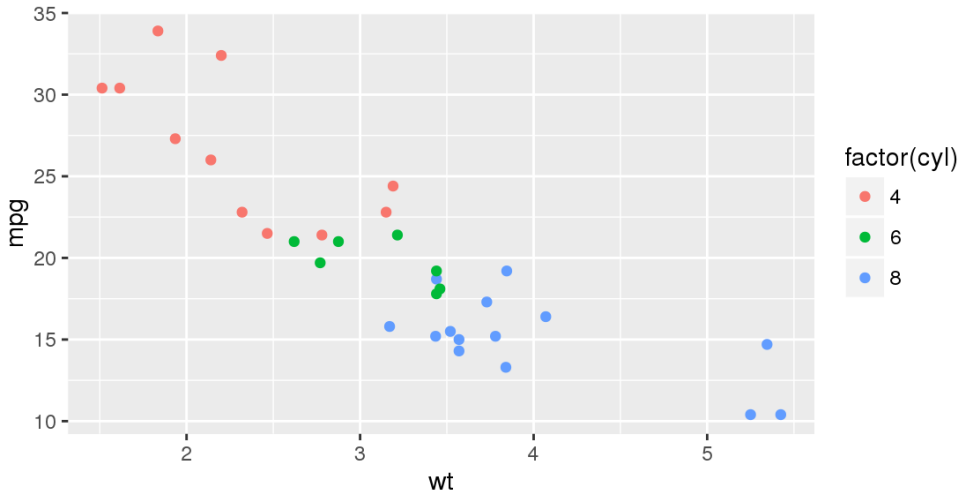
Lai mainītu līmeņu nosaukumus leģendā, jāizmanto argumentu `labels=`.

```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point()  +
      scale_colour_manual("Leģendas nosaukus",values = c("yellow","brown"),
                          labels=c("Stress bija","Stress nebija"))
```



scale_continuous()

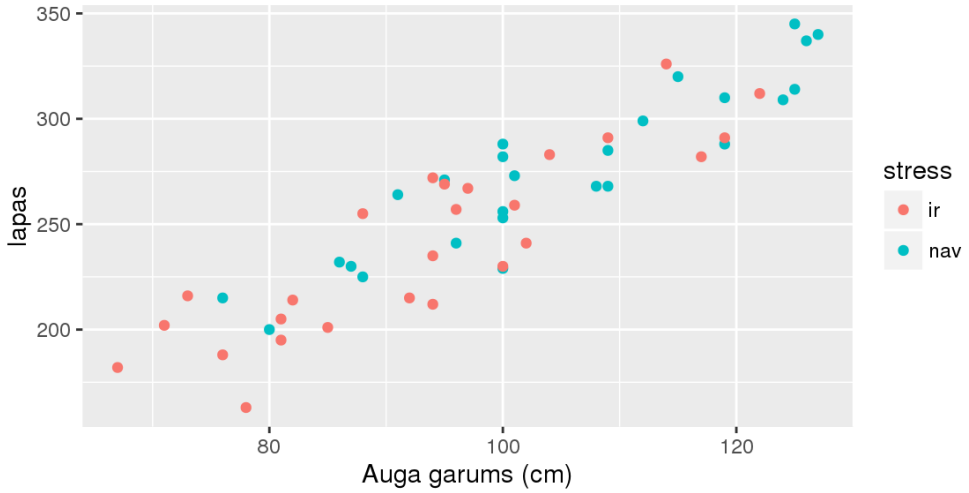
```
ggplot(mtcars, aes(x = wt, y = mpg,color=factor(cyl))) + geom_point()
```



scale_continuous()

Skaitliskām vērtībām izmanto `scale_.._continuous()`.

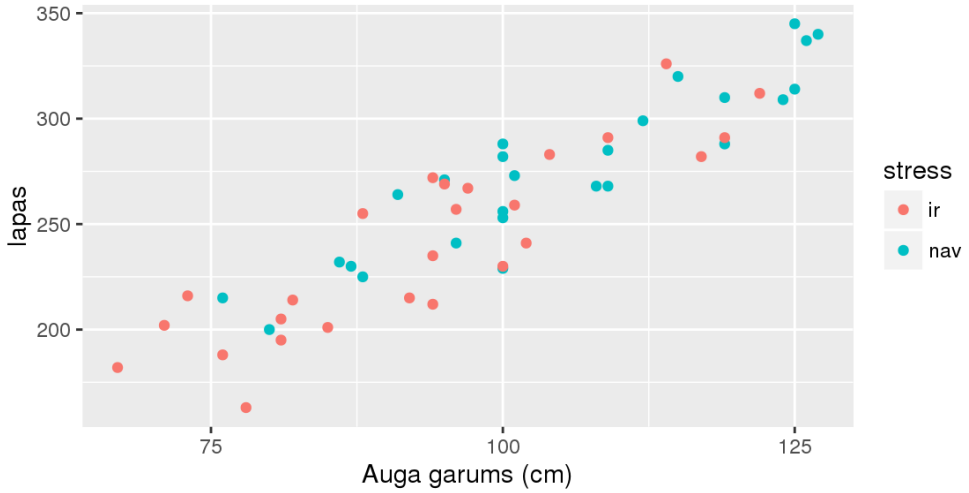
```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point() +
      scale_x_continuous("Auga garums (cm)")
```



scale_continuous()

Ar argumentu `breaks=` var noteikt vērtības, pie kādām veikt skalas dalījumu.

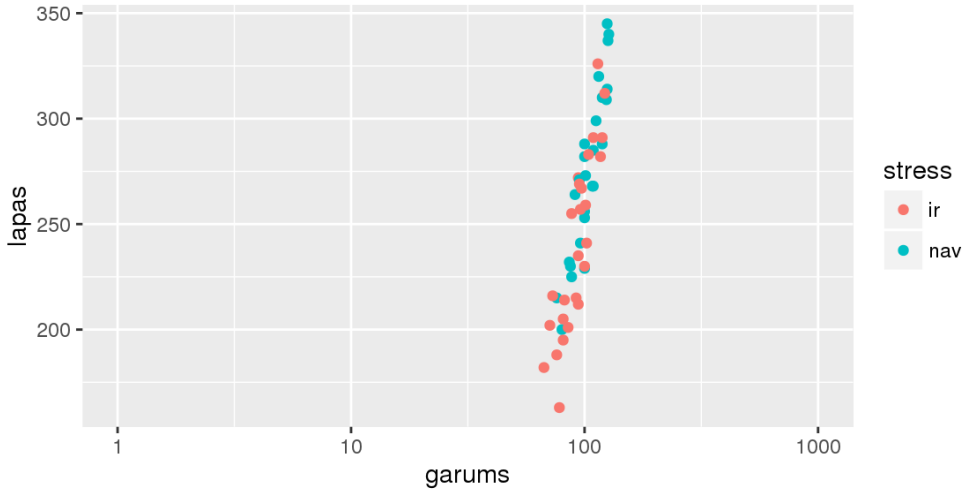
```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point() +
      scale_x_continuous("Auga garums (cm)",breaks=c(75,100,125))
```



scale_x_log10()

Ar funkcijām `scale_..._log10()`, `scale_..._sqrt()` var veikt automātisku skalu (asu) logaritmisko vai kvadrātsaknes transformāciju.

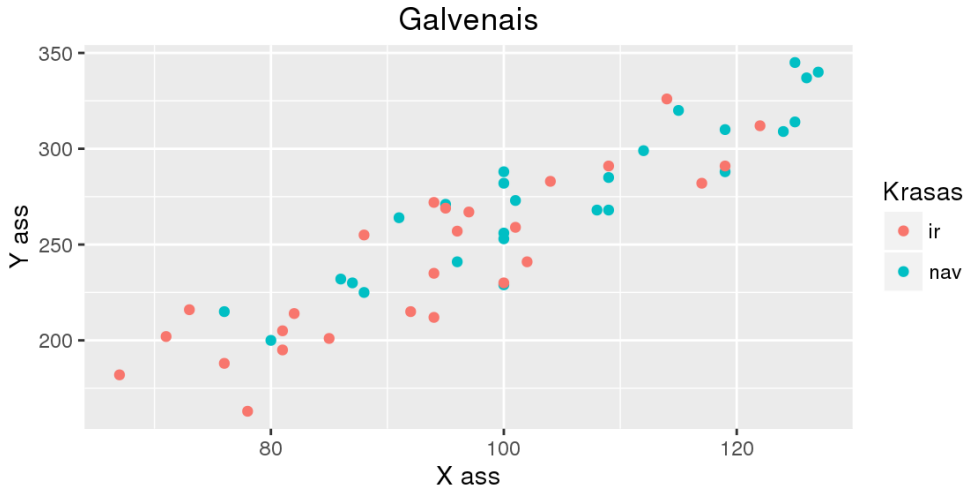
```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point()  +
      scale_x_log10(limits=c(1,1000),breaks=c(1,10,100,1000))
```



Nosaukumi

Nosaukumus asīm, attēlam leģendai var mainīt ne tikai ar skalu funkcijām, bet arī ar funkciju `labs()` un argumentiem, `x=`, `y=`, `title=`, `color=`, `fill=`, `size=`, ...

```
ggplot(dati,aes(garums, lapas, color=stress)) + geom_point()  +
      labs(x = "X ass",y="Y ass",title="Galvenais",color="Krasas")
```



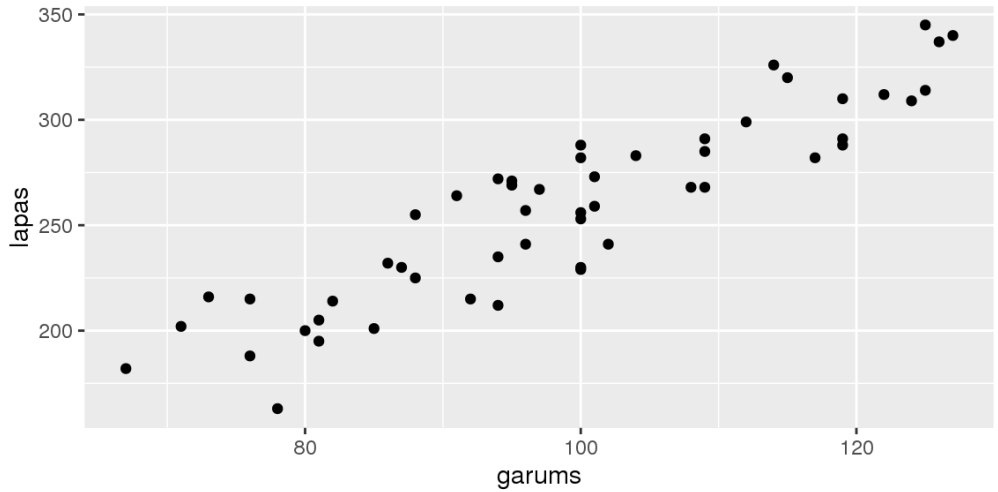
Attēlu sadalīšana daļās

`ggplot2` sistēmā ir iespējams automātiski sadalīt attēlu vairākās daļās balstoties uz vienu vai vairākiem mainīgajiem. To panāk ar funkcijām `facet_grid()` un `facet_wrap()`.

Pirmajā gadījumā tiek izveidots rāmis, kur jānorāda mainīgais, kas dala x ass virzienā un y ass virzienā, bet `facet_wrap()` gadījumā dalījums notiek pēc viena mainīgā, norādot nepieciešamo rindu vai kolonnu skaitu.

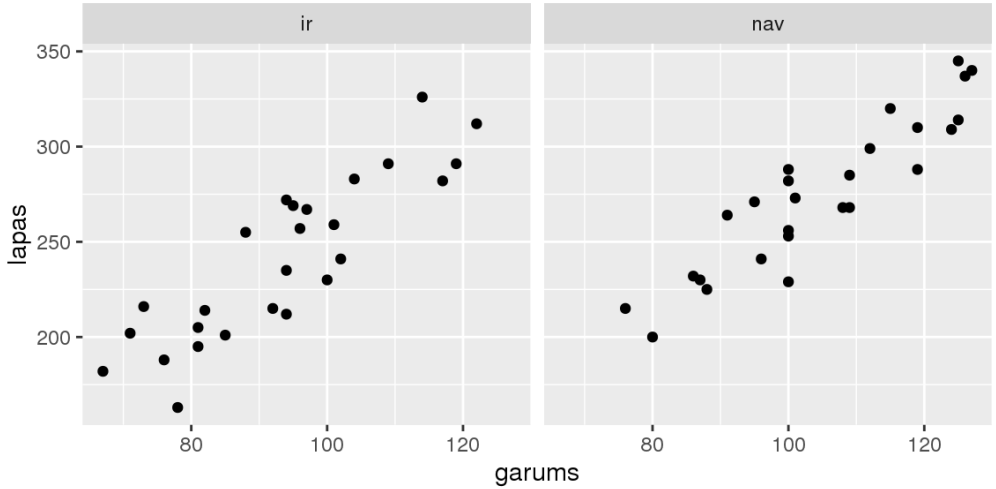
facet_grid()

```
ggplot(dati,aes(garums, lapas)) + geom_point()
```



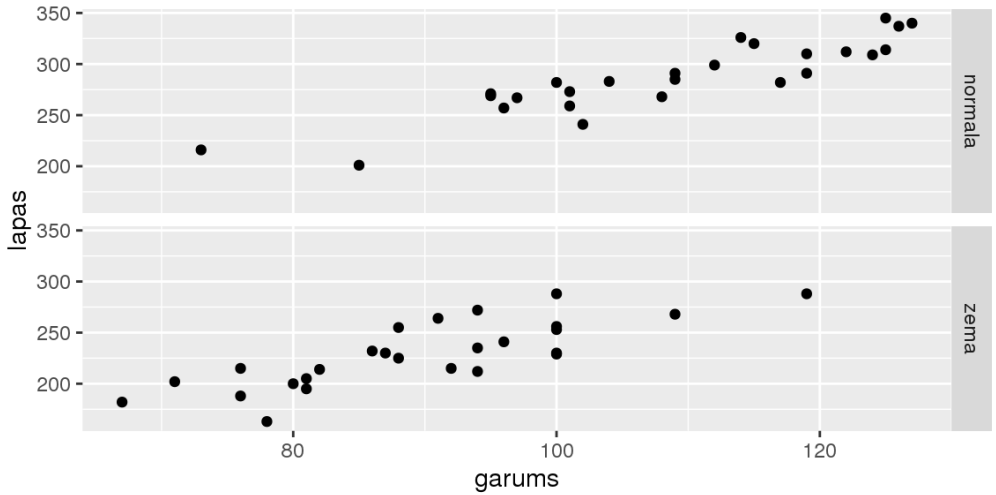
facet_grid()

```
ggplot(dati,aes(garums, lapas)) + geom_point()  + facet_grid(. ~ stress)
```



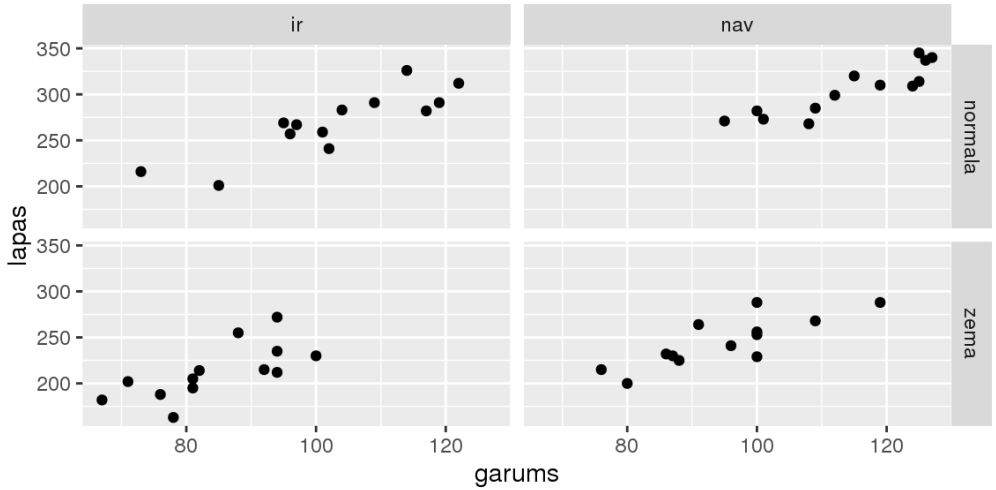
facet_grid()

```
ggplot(dati,aes(garums, lapas)) + geom_point() + facet_grid(gaisma ~ .)
```



facet_grid()

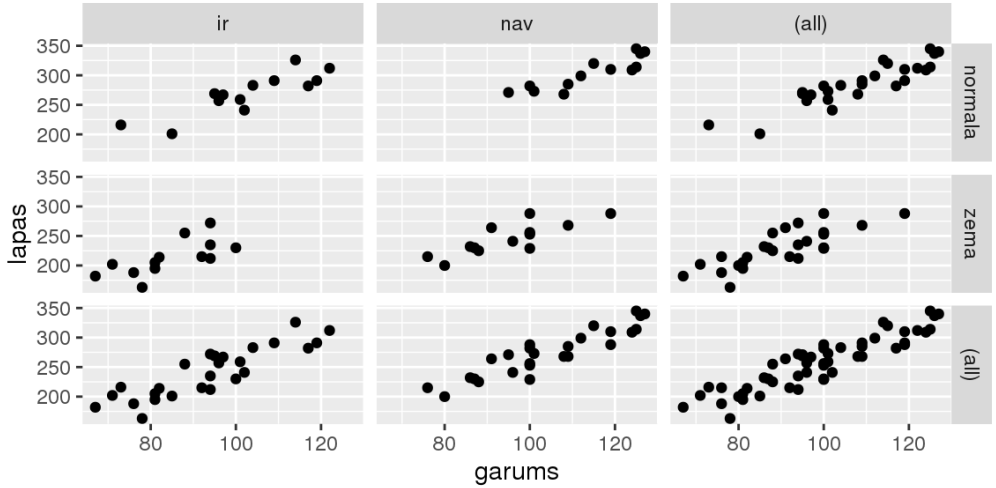
```
ggplot(dati,aes(garums, lapas)) + geom_point() + facet_grid(gaisma ~ stress)
```



facet_grid()

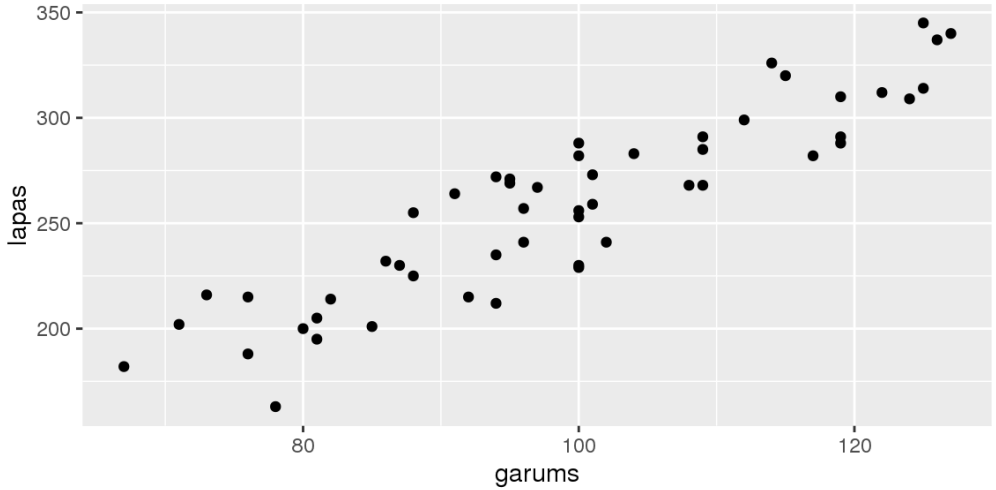
Ja funkcijai `facet_grid()` pieliek argumentu `margins=TRUE`, tad tiek izveidoti arī faktoru kombināciju attēli.

```
ggplot(dati,aes(garums, lapas)) + geom_point() + facet_grid(gaisma ~ stress,margins = TRUE)
```



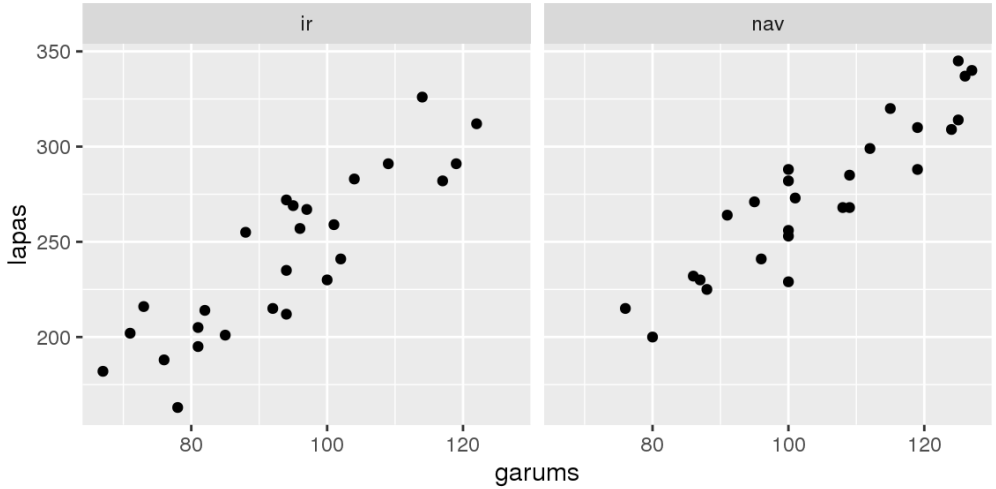
facet_wrap()

```
ggplot(dati,aes(garums, lapas)) + geom_point()
```



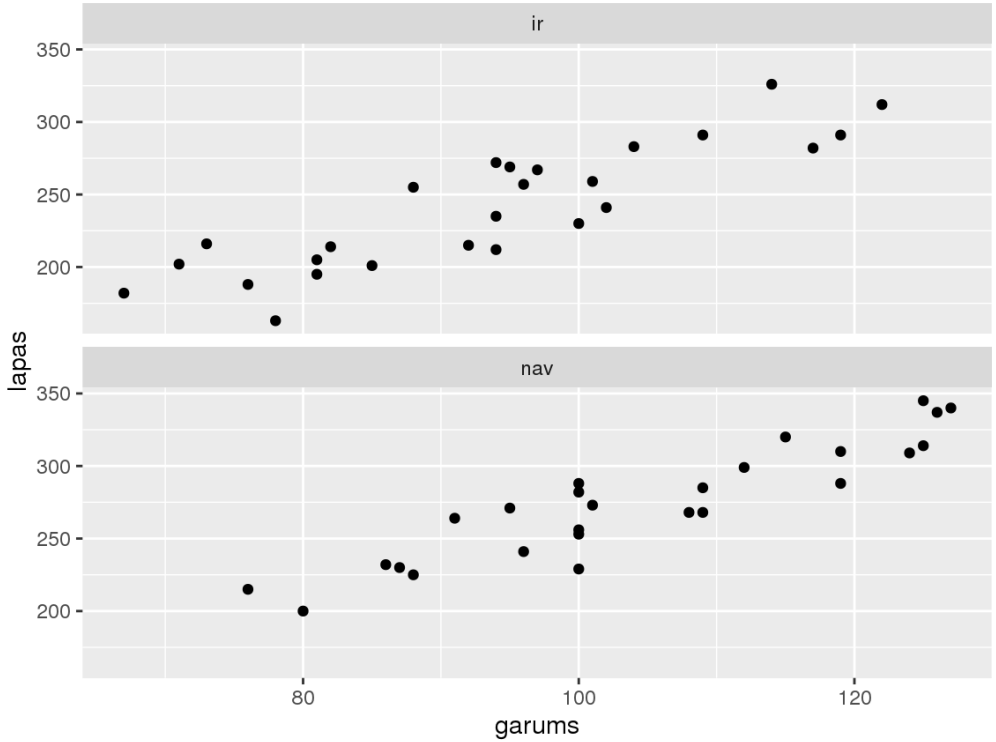
facet_wrap()

```
ggplot(dati,aes(garums, lapas)) + geom_point() + facet_wrap(~ stress)
```



facet_wrap()

```
ggplot(dati,aes(garums, lapas)) + geom_point() + facet_wrap(~ stress,ncol=1)
```



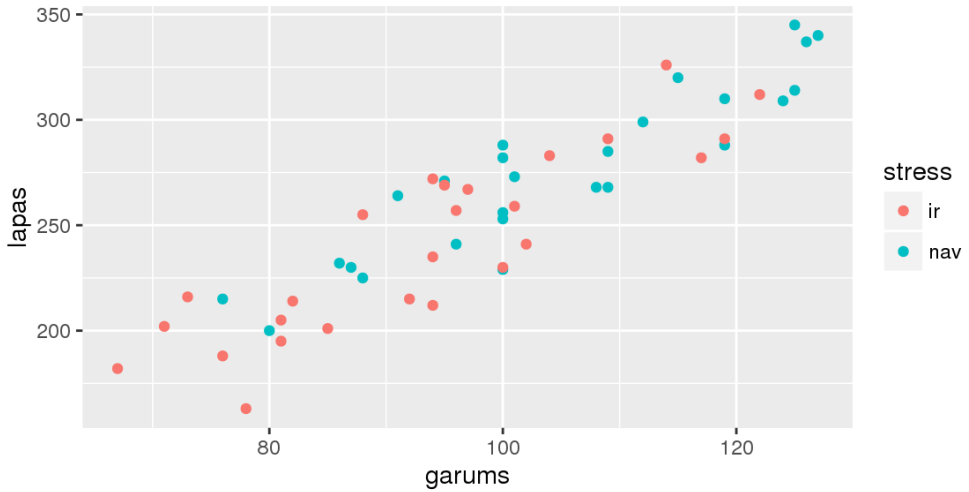
Attēla izskata maiņa

Lai mainītu attēla izskatu, var izmantot gatavas tēmas vai arī mainīt katru elementu atsevišķi izmantojot funkciju `theme()`.

Gatavās tēmas, piemēram, ir `theme_bw()` vai `theme_minimal()`.

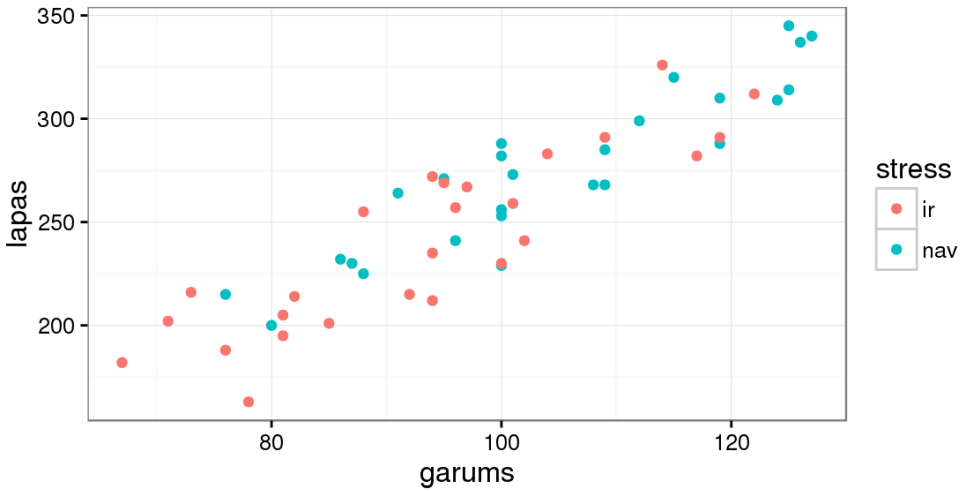
theme_bw()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()
```



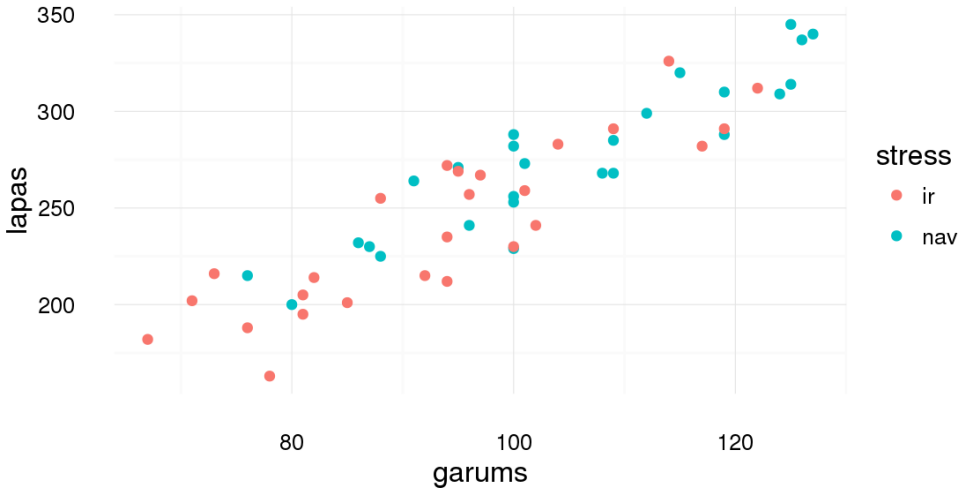
theme_bw()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+ theme_bw()
```



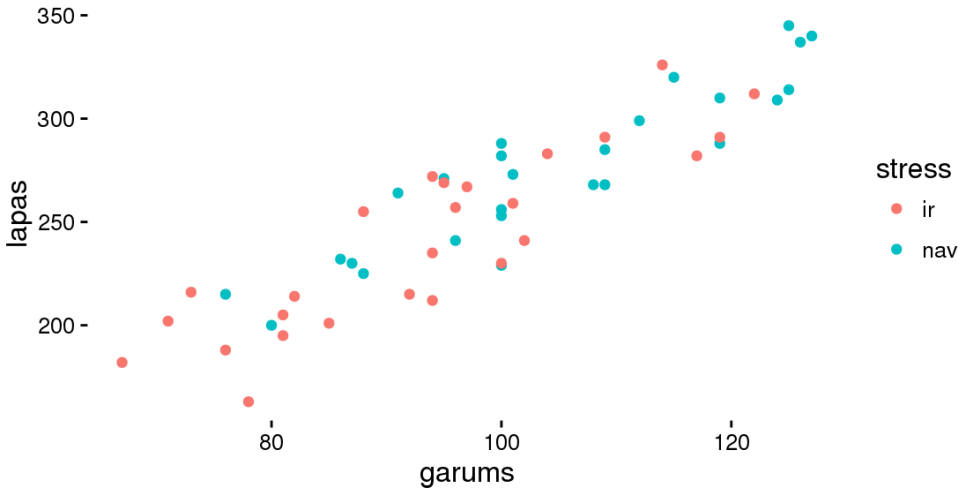
theme_minimal()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme_minimal()
```



theme_classic()

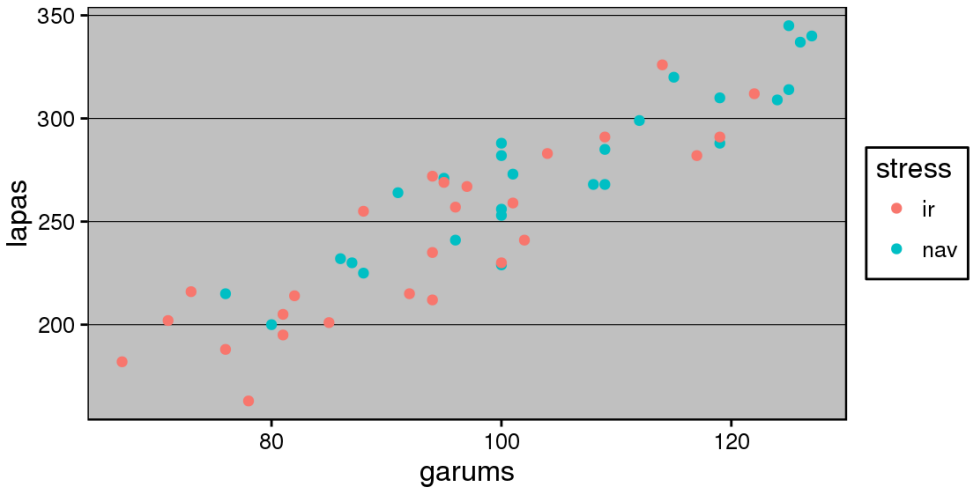
```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+
      theme_classic()
```



theme_excel()

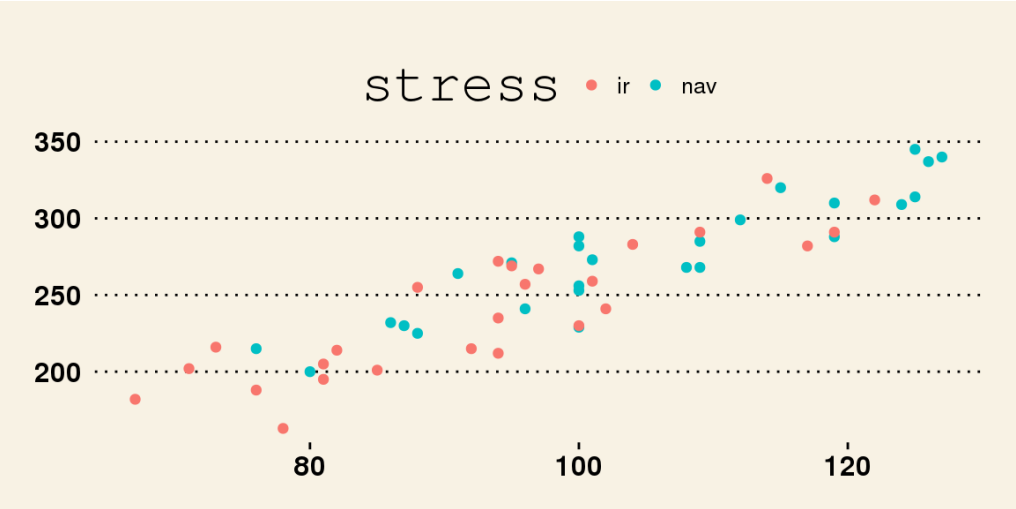
Paketē ggthemes ir vairākas papildus noformējuma tēmas.

```
if (!require("ggthemes") )  install.packages("ggthemes")
library(ggthemes)
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme_excel()
```



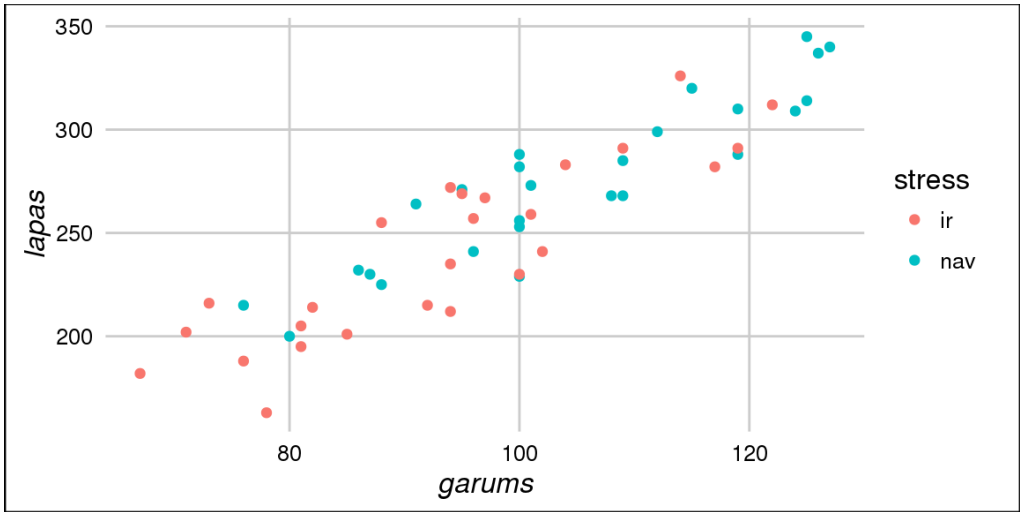
theme_wsj()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme_wsj()
```



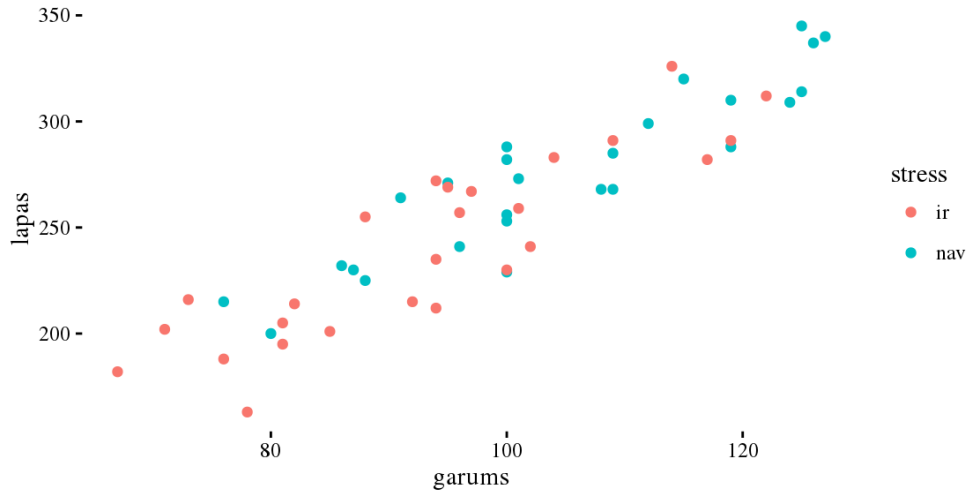
theme_gdocs()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme_gdocs()
```



theme_tufte()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme_tufte()
```



theme()

Argumenti attēla elementu mainīšanai

- axis.title - asu paraksti (element_text)
- axis.title.x - x ass paraksts (element_text)
- axis.title.y - y ass paraksts (element_text)
- axis.text - apzīmējumi pie asīm (element_text)
- axis.text.x - apzīmējumi pie x ass (element_text)
- axis.text.y - apzīmējumi pie y ass (element_text)
- axis.ticks - nogriežņi pie asīm (element_line)
- axis.ticks.x - nogriežņi pie x ass (element_line)
- axis.ticks.y - nogriežņi pie y ass (element_line)
- axis.ticks.length - nogriežņu garums pie asīm (unit)
- axis.ticks.margin - atstarpe starp ass apzīmējumu un nogriežņiem (unit)
- axis.line - līnijas gar asīm (element_line)
- axis.line.x - līnijas pie x ass (element_line)
- axis.line.y - līnijas pie y ass (element_line)

theme()

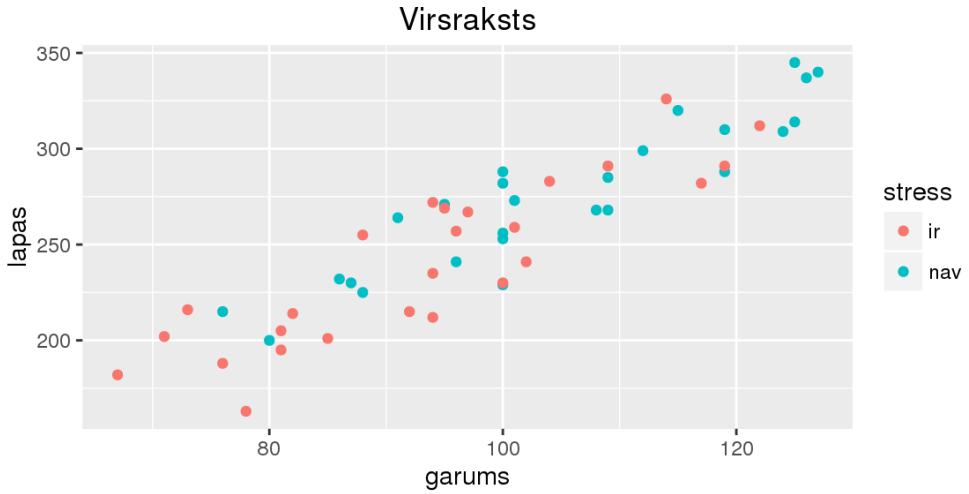
- legend.background - leģendas pamatne (element_rect)
- legend.margin - papildus atstarpe apkārt leģendai (unit)
- legend.key - pamatne zem leģendas ierakstiem (element_rect)
- legend.key.size - leģendas ierakstu izmērs (unit)
- legend.key.height - leģendas ieraksta pamatnes augstums (unit)
- legend.key.width - leģendas ieraksta pamatnes platums (unit)
- legend.text - leģendas ieraksti (element_text)
- legend.text.align - leģendas teksta novietojums (skaitlis no 0 līdz 1)
- legend.title - leģendas virsraksts (element_text)
- legend.title.align - leģendas virsraksta novietojums (skaitlis no 0 līdz 1)
- legend.position - leģendas novietojums ("left", "right", "bottom", "top", vai divu skaitļu vektors)
- legend.direction - leģendas ierakstu izvietojums ("horizontal" or "vertical")
- legend.justification - leģendas novietojums diagrammas iekšienē ("center" vai divu skaitļu vektors)
- legend.box - vairāku leģendu novietojums ("horizontal" or "vertical")
- panel.background - diagrammas iekšienes pametne (element_rect)
- panel.border - robeža apkārt diagrammas iekšienei (element_rect)

theme()

- panel.margin - mala ap atsevišķām diagrammām (facet) (unit)
- panel.grid - grid lines (element_line)
- panel.grid.major - galvenās grid lines (element_line)
- panel.grid.minor - mazās grid lines (element_line)
- panel.grid.major.x - vertikālās galvenās grid lines (element_line)
- panel.grid.major.y - horizontālās galvenās grid lines (element_line)
- panel.grid.minor.x - vertikālās mazās grid lines (element_line)
- panel.grid.minor.y - horizontālās mazās grid lines (element_line)
- plot.background - visas diagrammas pamatne (element_rect)
- plot.title - diagrammas virsraksts (element_text)
- plot.margin - mala apkārt visai diagrammai (unit)
- strip.background - atsevišķu diagrammu (facet) uzrakstu pamatne (element_rect)
- strip.text - atsevišķu diagrammu (facet) uzraksts (element_text)
- strip.text.x - atsevišķu diagrammu (facet) uzraksts horizontālā virzienā (element_text)
- strip.text.y - atsevišķu diagrammu (facet) uzraksts vertikālā virzienā (element_text)

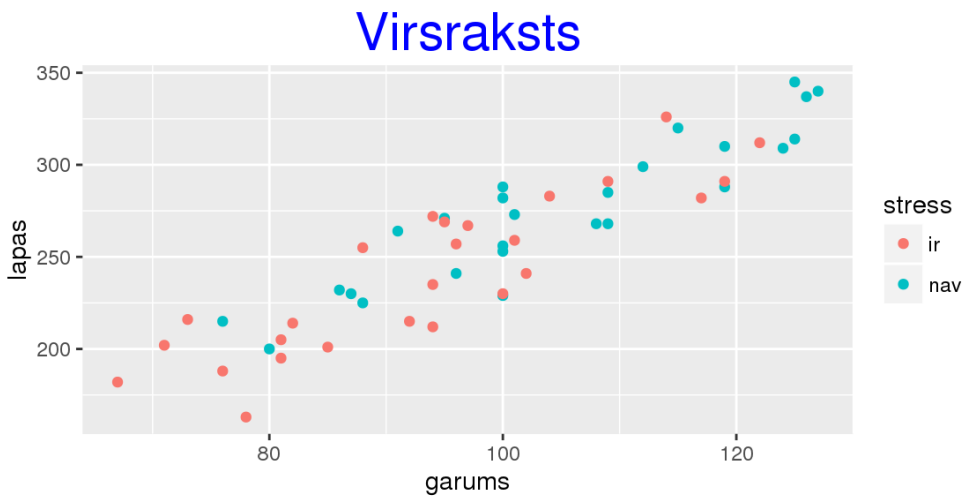
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+
      labs(title="Virsraksts")
```



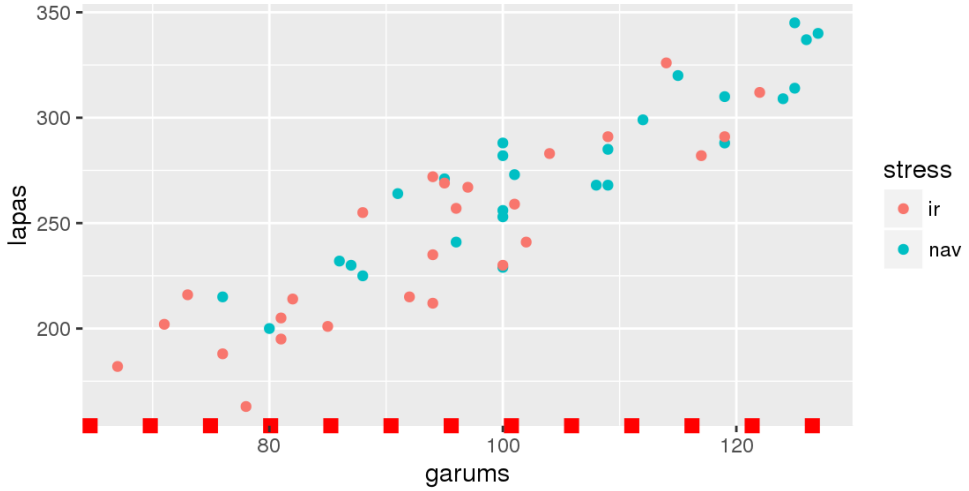
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+
      labs(title="Virsraksts") +
      theme(plot.title = element_text(size = rel(2),colour="blue"))
```



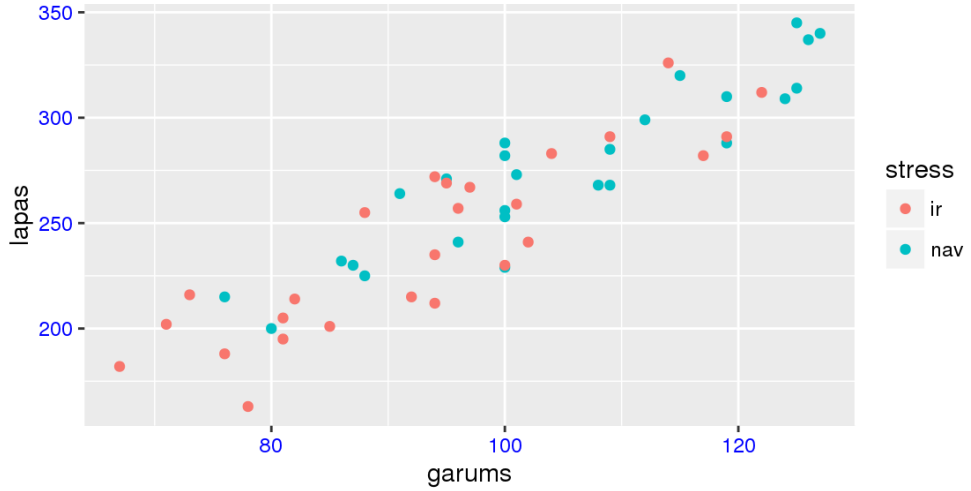
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(axis.line.x = element_line(size = 3, colour = "red", linetype = "dotted"))
```



theme()

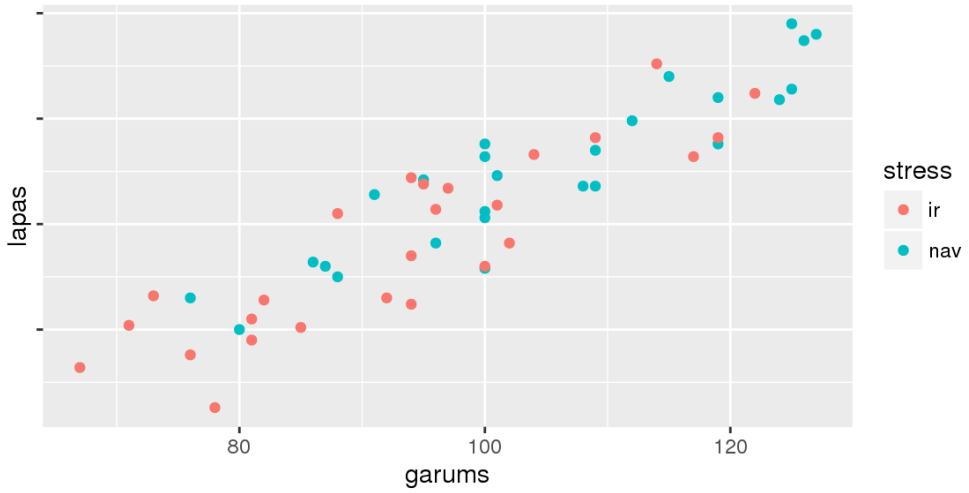
```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(axis.text = element_text(colour = "blue"))
```



theme()

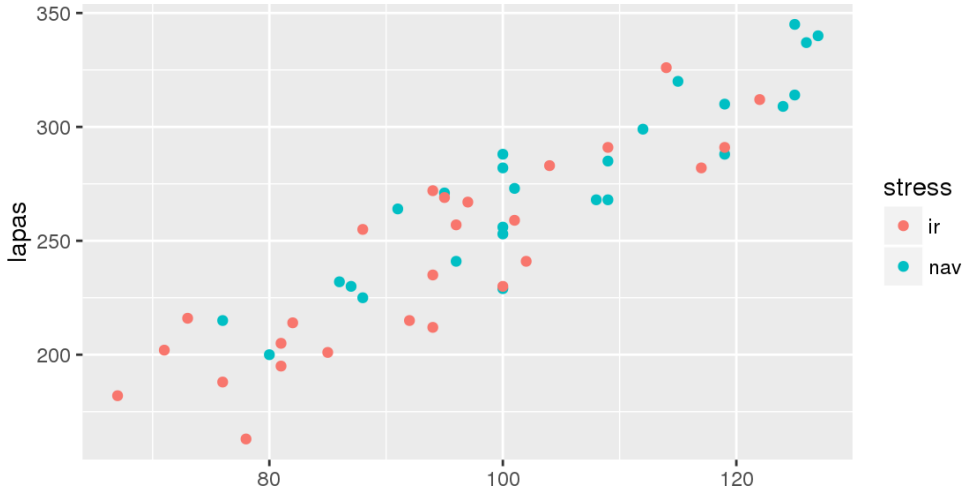
Jā kādu no elementiem nepieciešams pilnībā izslēgt/paslēpt, izmanto argumentu `element_blank()`.

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(axis.text.y = element_blank())
```



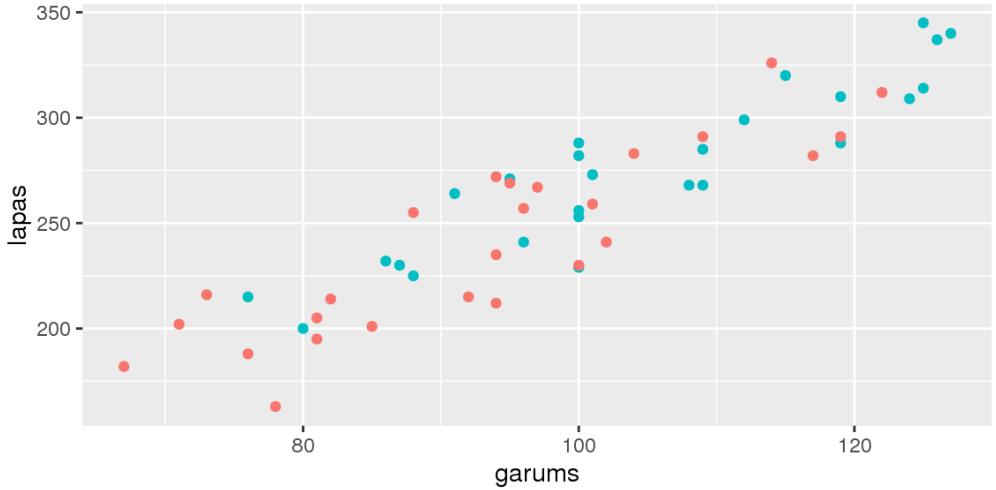
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(axis.title.x = element_blank())
```



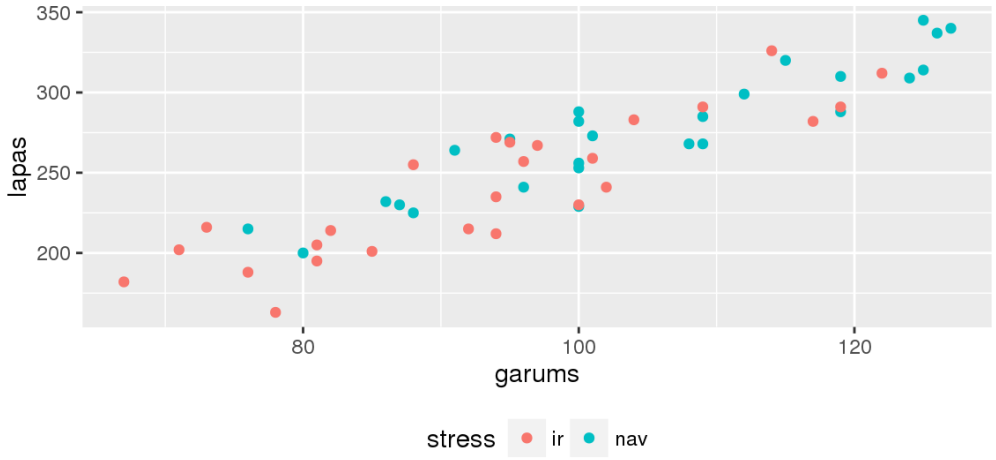
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+
      theme(legend.position = "none")
```



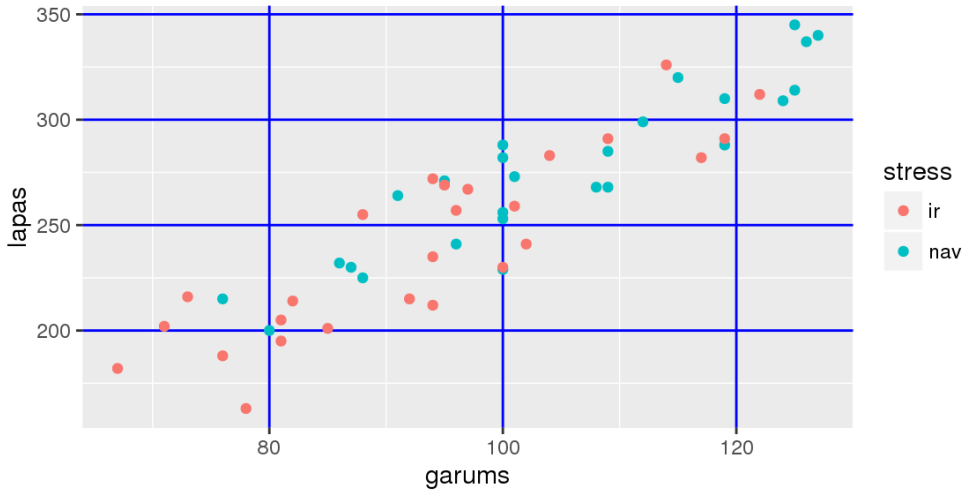
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+
      theme(legend.position = "bottom")
```



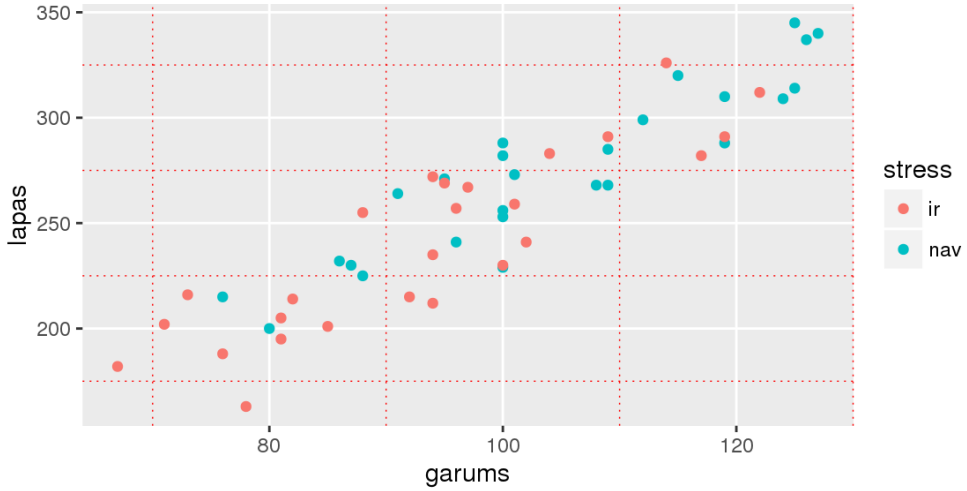
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(panel.grid.major = element_line(colour = "blue"))
```



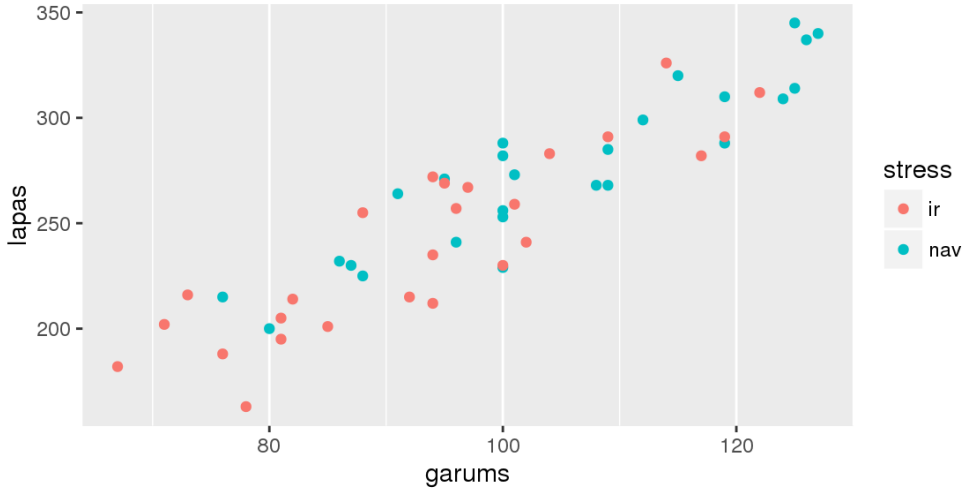
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(panel.grid.minor = element_line(colour = "red", linetype = "dotted"))
```



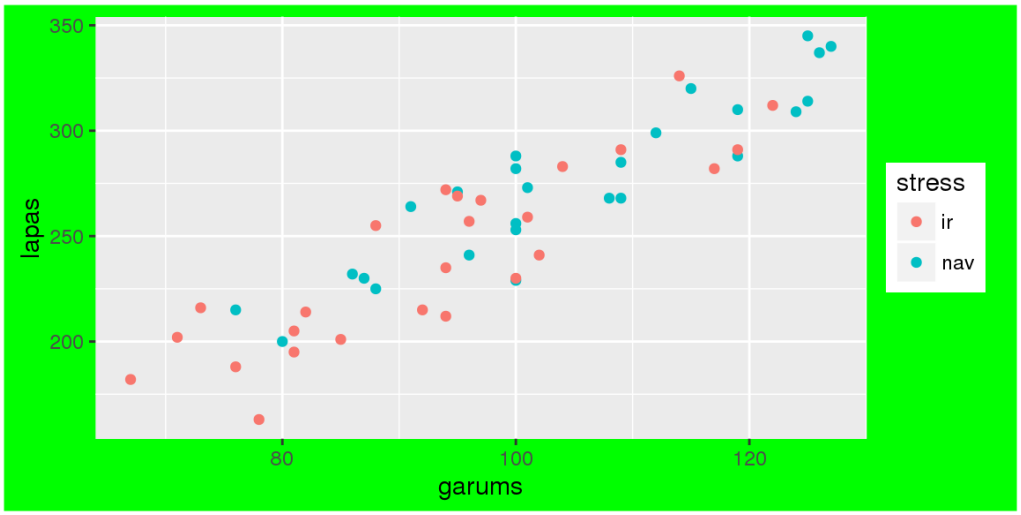
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(panel.grid.major.y = element_blank(),
            panel.grid.minor.y = element_blank())
```



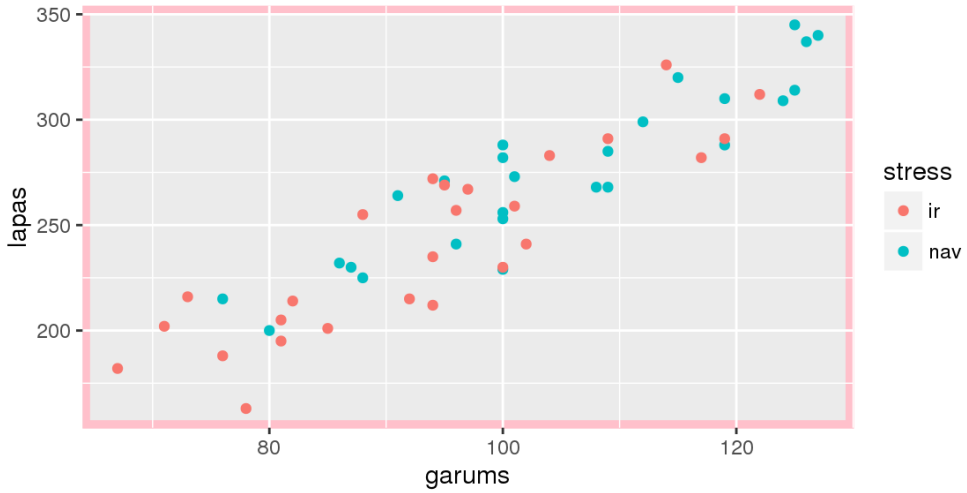
theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+
      theme(plot.background = element_rect(fill = "green"))
```



theme()

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point() +
      theme(panel.background = element_rect(colour = "pink",size=3))
```

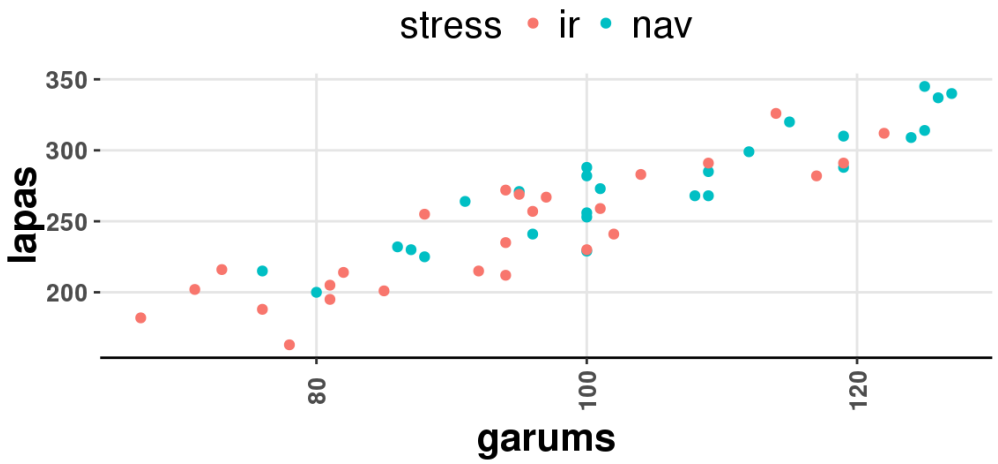


theme()

Katrs elements nav jānorāda savā `theme()` funkcijā, bet tos visus var iekļaut vienā funkcijā.

```
ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()+
  theme(axis.text.y=element_text(size=rel(1.2),face="bold"),
        axis.text.x=element_text(size=rel(1.2),face="bold",angle=90,vjust=0.5),
        axis.title=element_text(size=rel(1.5),face="bold"),
        axis.line.x=element_line(color="black"),
        panel.background=element_blank(),
        panel.grid.minor=element_blank(),
        panel.grid.major=element_line(color="grey90"),
        legend.position="top",
        legend.key=element_rect(fill="white"),
        legend.title=element_text(size=rel(1.5)),
        legend.text=element_text(size=rel(1.5)))
```

theme()



Attēlu saglabāšana

ggplot2 attēlus var saglabāt arī pēc to izveidošanas.

Lai saglabātu attēlu, izmanto funkciju `ggsave()`, kurā jānorāda vēlamais faila nosaukums ar paplašinājumu, izmērs (pēc noklusējuma collās). Šo komandu izpilda kā pēdējo.

```
ggsave("1_attels.png",width=8,height=5)
```

Attēla veidošana no objektiem

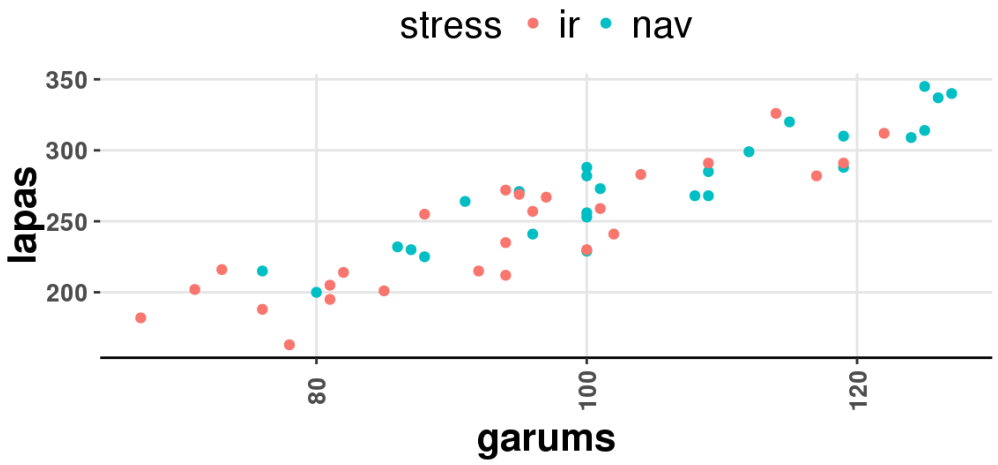
Gan pašu attēlu, gan arī noformējuma nosacījumus var saglabāt kā atsevišķus argumentus un izmantot atkārtoti.

```
p<-ggplot(dati,aes(garums, lapas,color=stress)) + geom_point()

noformejums<-  theme(axis.text.y=element_text(size=rel(1.2),face="bold"),
        axis.text.x=element_text(size=rel(1.2),face="bold",angle=90,vjust=0.5),
        axis.title=element_text(size=rel(1.5),face="bold"),
        axis.line.x=element_line(color="black"),
        panel.background=element_blank(),
        panel.grid.minor=element_blank(),
        panel.grid.major=element_line(color="grey90"),
        legend.position="top",
        legend.key=element_rect(fill="white"),
        legend.title=element_text(size=rel(1.5)),
        legend.text=element_text(size=rel(1.5)))
```

Attēla veidošana no objektiem

```
p+noformejums
```



Attēla veidošana no objektiem

```
ggplot(dati,aes(garums))+geom_histogram()+noformejums
```

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