In this documents, we show how we generate the weather forecast example data and how the visualization stimuli is generated.

Data generating Process

We generate the data from a Gaussian distribution with a fixed mean and varying variance drawn from a uniform distribution.

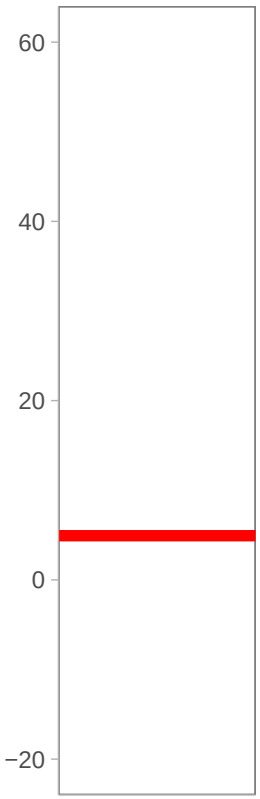
```
# Configuration of data generating model
# Daily low temperature follows a Gaussian distribution with a fixed mu and sigma drawn uniformly from
mu = 5
sigma_choices = c(2, 3, 4, 5)
sigma = sample(sigma_choices, 1)

# Size of sample
n_size = 100

low_temp = data.frame(temp = rnorm(n_size, mu, sigma), x=factor(0))
dense_temp = density(low_temp$temp)
```

Stimuli

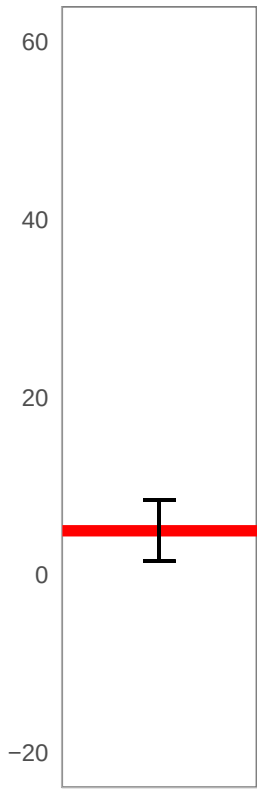
We generate four types of stimuli (mean, mean + CI, mean + gradience, and mean + HOPs).

```
theme_set(theme_ggdist())
options(repr.plot.width = 10, repr.plot.height =2)

ggplot(data = low_temp, mapping = aes(x=0)) +
  geom_hline(aes(yintercept=mean(temp)),
             size=2, color="red") +
  labs(x = "", y = "") +
  ylim(-20, 60) + theme(aspect.ratio=4) +
  theme(panel.background = element_rect(fill = "white", colour = "grey50"))
```

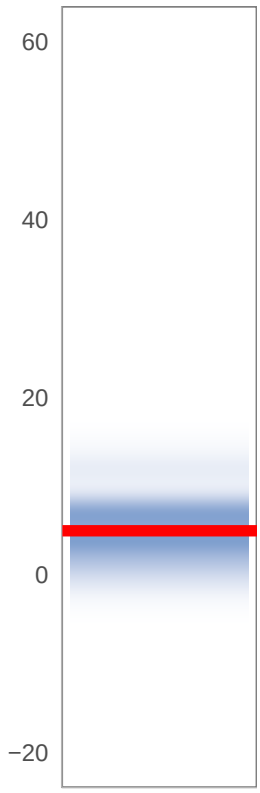
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
ggplot(data = low_temp, mapping = aes(x=factor(0)), colour=x) +
  geom_hline(aes(yintercept=mean(temp)),
             size=2, color="red")+
  geom_errorbar(aes(ymin = mean(temp) - sd(temp), ymax = mean(temp) + sd(temp)),
                width = 0.2) +
  labs(x = "", y = "") +
  theme(axis.text.x = element_blank(),
    axis.ticks = element_blank()) +
  ylim(-20, 60) + theme(aspect.ratio=4) +
  theme(panel.background = element_rect(fill = "white", colour = "grey50"))
```

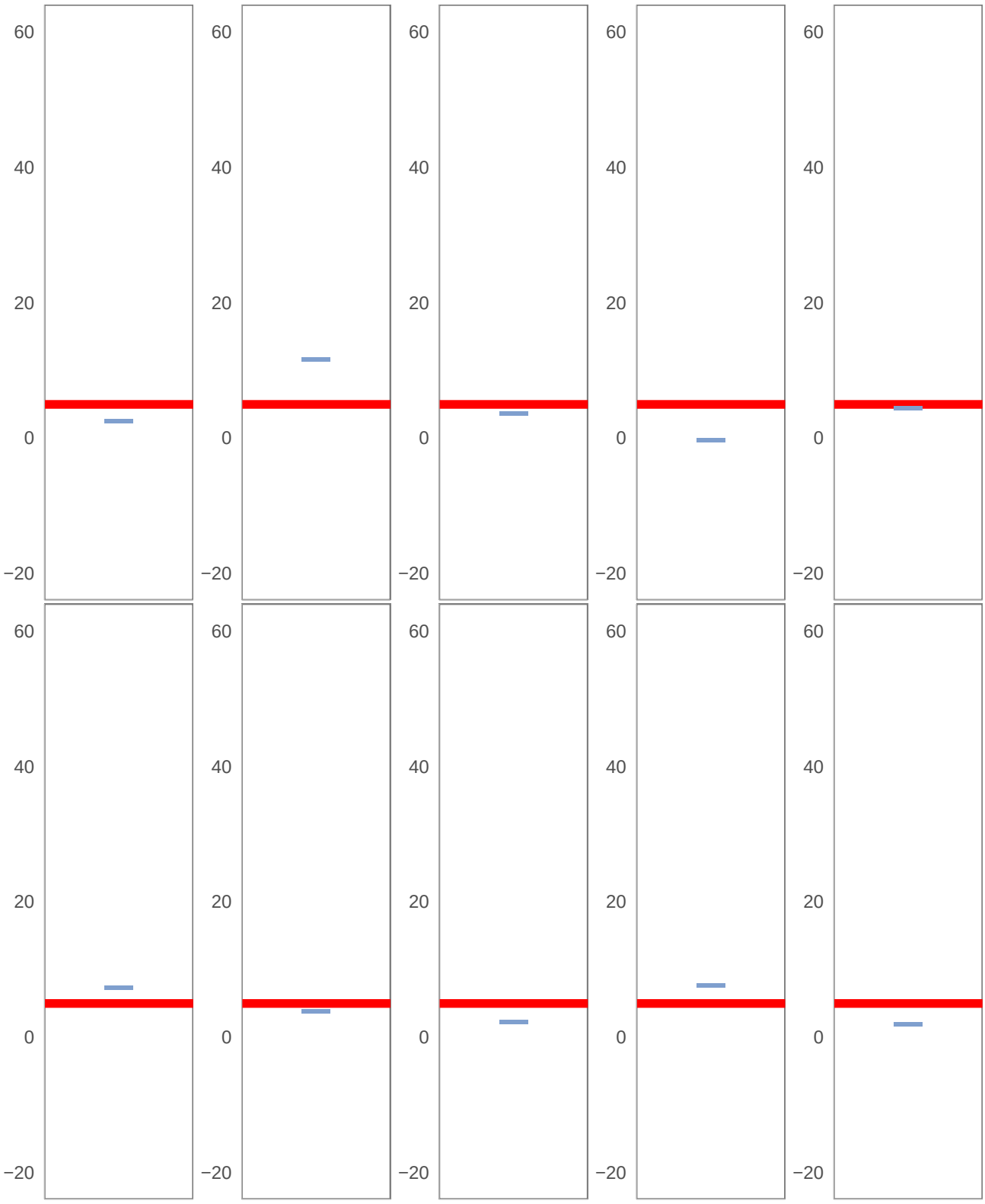


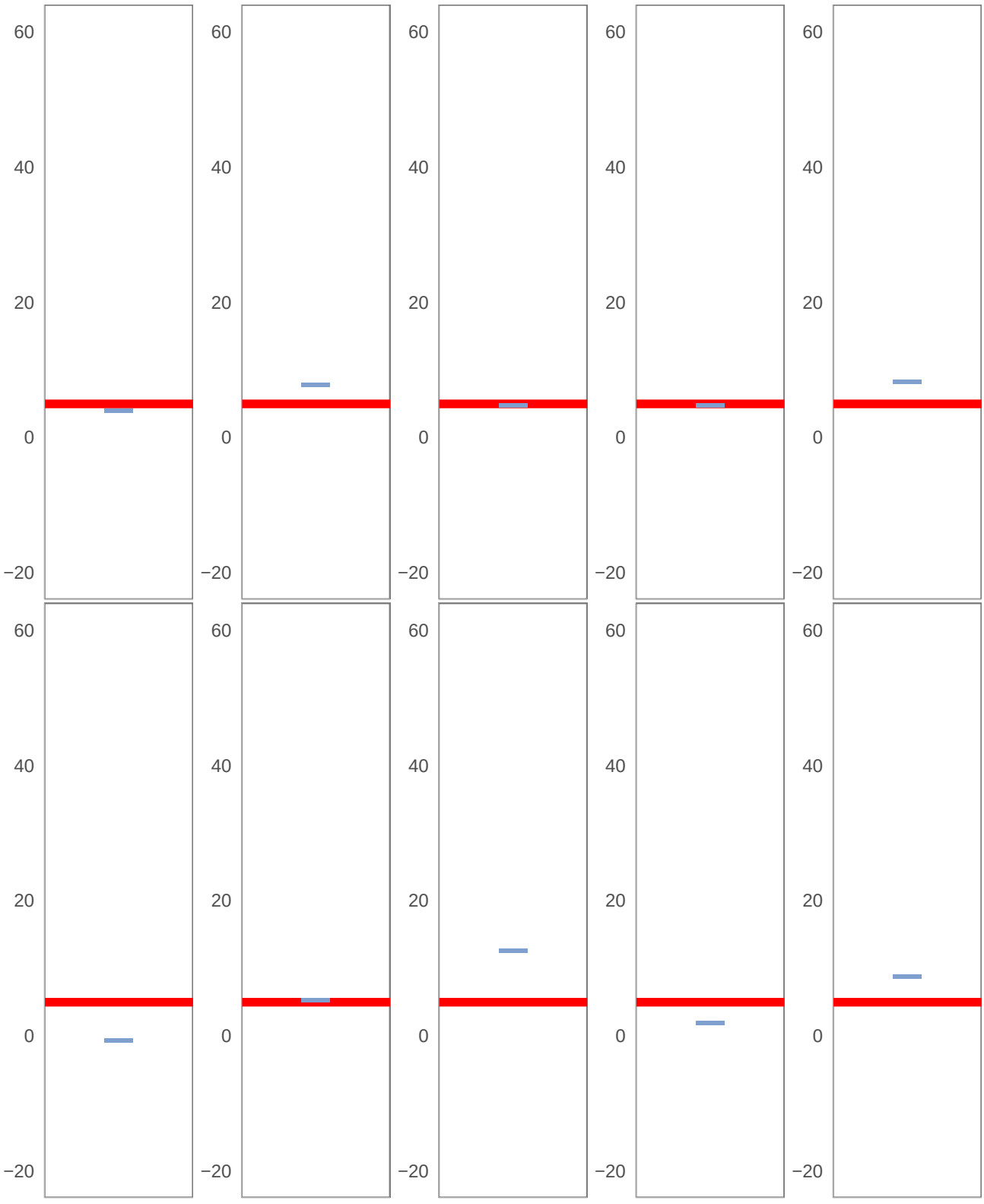

```
ggplot() +
  geom_raster(aes(x = 0, y = dense_temp$x, fill = dense_temp$y)) +
  guides(fill=guide_legend(title="Density")) +
  scale_fill_gradient(low = "white", high = "#7F9FCE") +
  geom_hline(yintercept=mean(low_temp$temp),
             size=2, color="red") +
  labs(x = "", y = "") +
  theme(axis.text.x = element_blank(),
        axis.ticks = element_blank()) +
  ylim(-20, 60) + theme(aspect.ratio=4) + theme(legend.position = "none") +
  theme(panel.background = element_rect(fill = "white", colour = "grey50"))
```

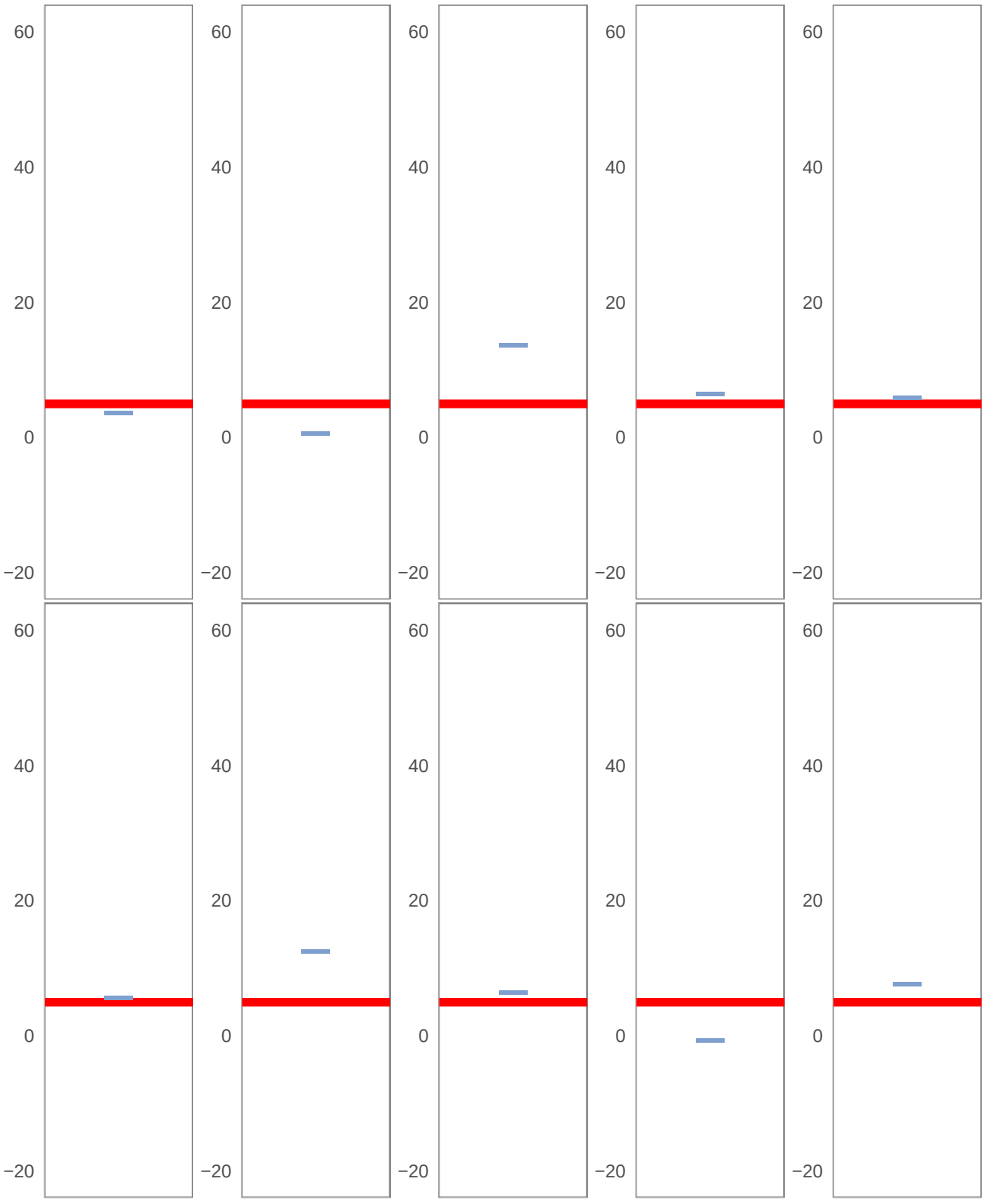


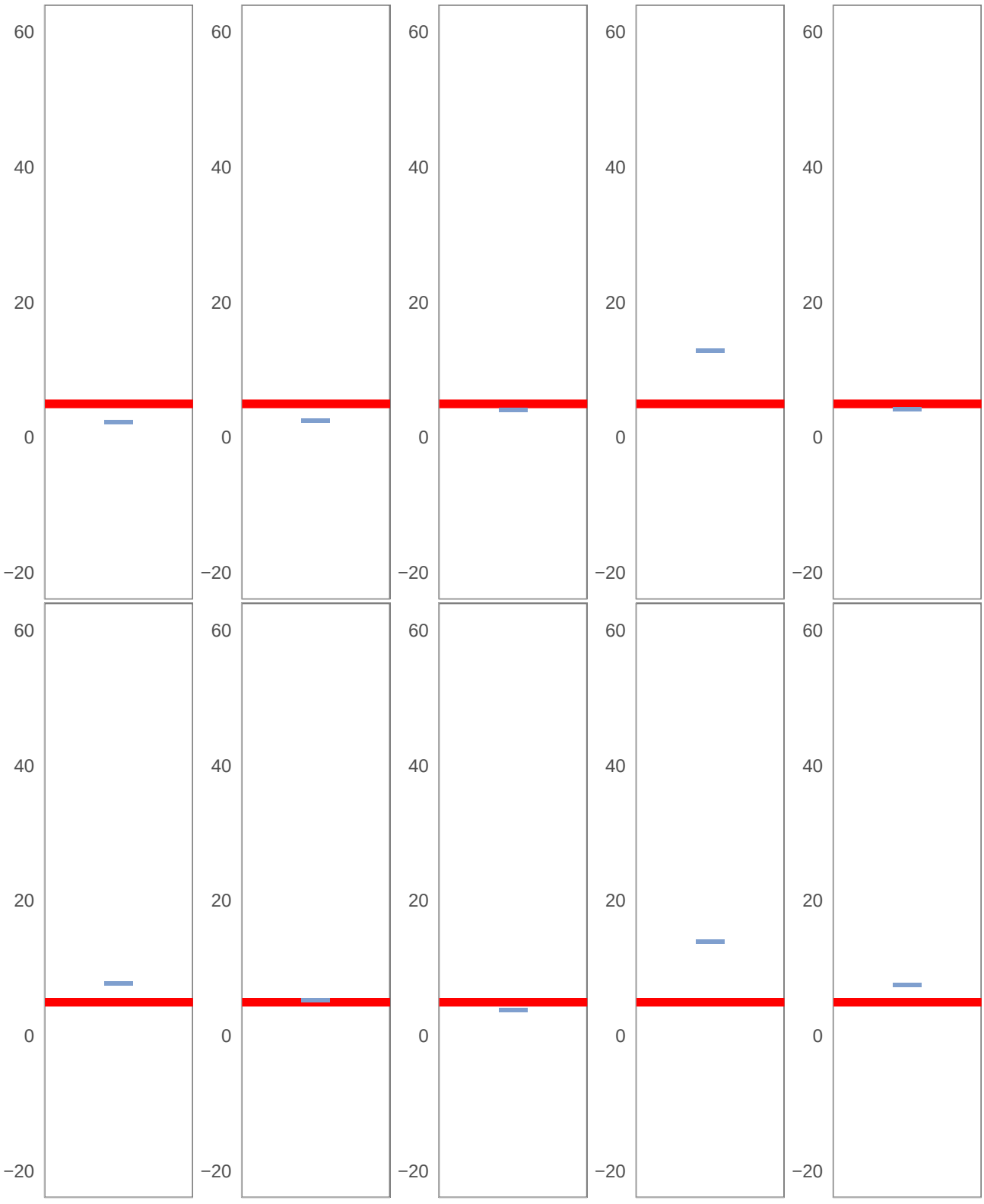

```
n_hop_sample = 50

hops_low_temp = low_temp %>%
  sample_n(n_hop_sample) %>%
  mutate(sample_id = 1:n_hop_sample) %>%
  select(hops_temp = temp, everything())

p = hops_low_temp %>%
  ggplot() +
    geom_hline(yintercept = mean(low_temp$temp),
               size=2, color="red") +
    geom_point(aes(x = 0, y = hops_temp), shape = '-', size=10, color = '#7F9FCE') +
    labs(x = "", y = "") +
    theme(axis.text.x = element_blank(),
          axis.ticks = element_blank()) +
    ylim(-20, 60) +
    transition_manual(sample_id) + theme(aspect.ratio=4) +
    theme(panel.background = element_rect(fill = "white", colour = "grey50"))

p
```

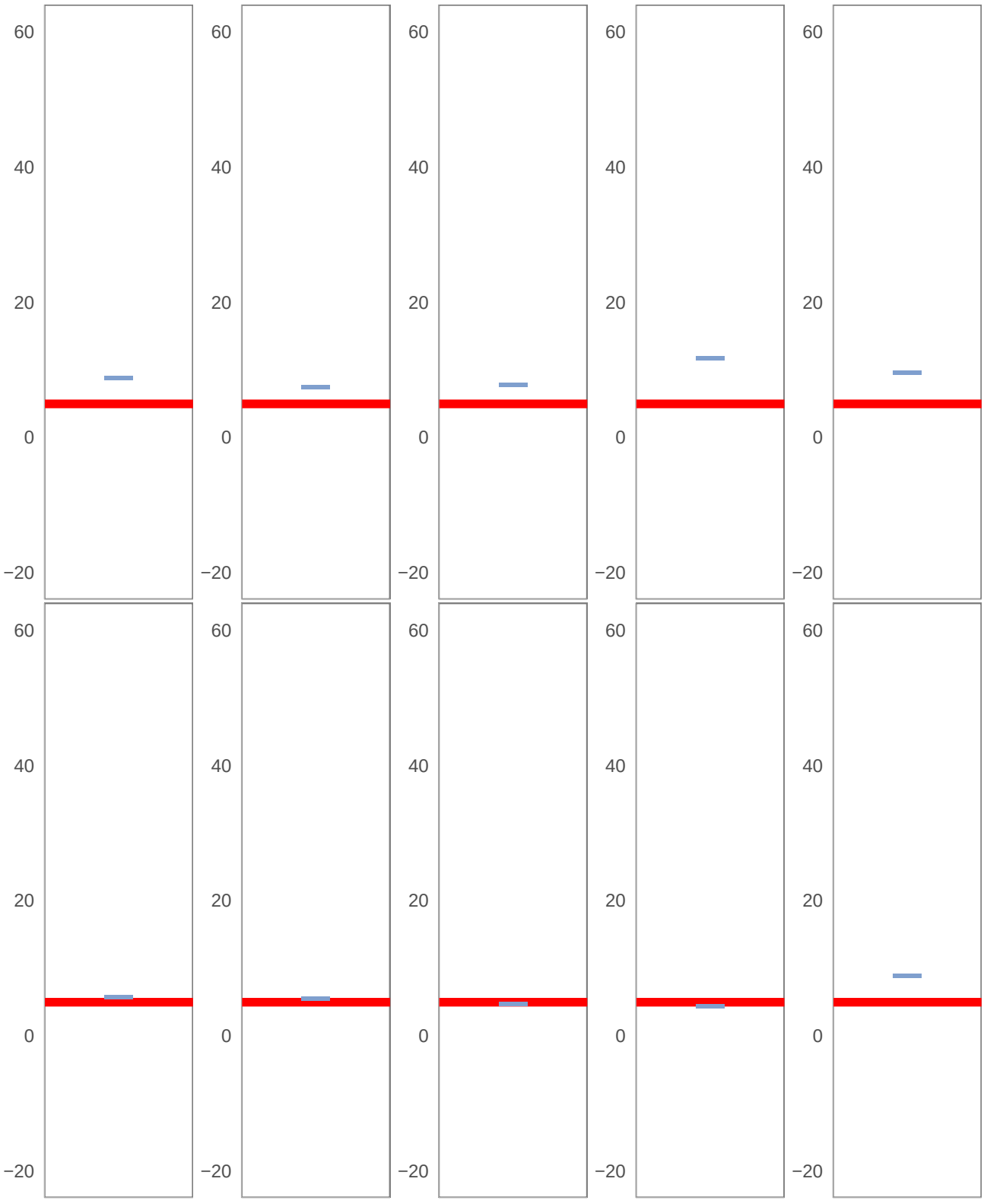
```
## nframes and fps adjusted to match transition
```


60
40
20
0
−20


60
40
20
0
−20


60
40
20
0
−20


60
40
20
0
−20


60
40
20
0
−20

Scoring rule and expected score

The setting of scoring rule

We design the scoring rule by action and whether it's freezing. We give no salt but freezing a large penalty (-100) while salt but not freezing a smaller one (-10). We hold a zero payoff (no penalty) for no salt without freezing and salt with freezing.

```
# the scoring rule
no_salt_not_freezing_payoff = 0
no_salt_freezing_payoff = -100
salt_not_freezing_payoff = -10
salt_freezing_payoff = 0
```

Rational baseline and benchmark

- Baseline (rational agent with prior knowledge)

```
# The rational agent's prior belief is the expected possibility of freezing without knowledge about whi
prior_belief = Reduce("+", sapply(sigma_choices, function(m) {pnorm(0, mu, m)})) / 4

# The payoff of salt or no salt
payoff_salt = salt_freezing_payoff * prior_belief + salt_not_freezing_payoff * (1 - prior_belief)
payoff_no_salt = no_salt_freezing_payoff * prior_belief + no_salt_not_freezing_payoff * (1 - prior_beli

# rational agent's best action and payoff
rprior_action = ifelse(payoff_no_salt > payoff_salt, "No salt", "Salt")
rprior = max(payoff_no_salt, payoff_salt)

rprior
```

```
## [1] -7.957626
```

- Benchmark (rational agent with visualization)

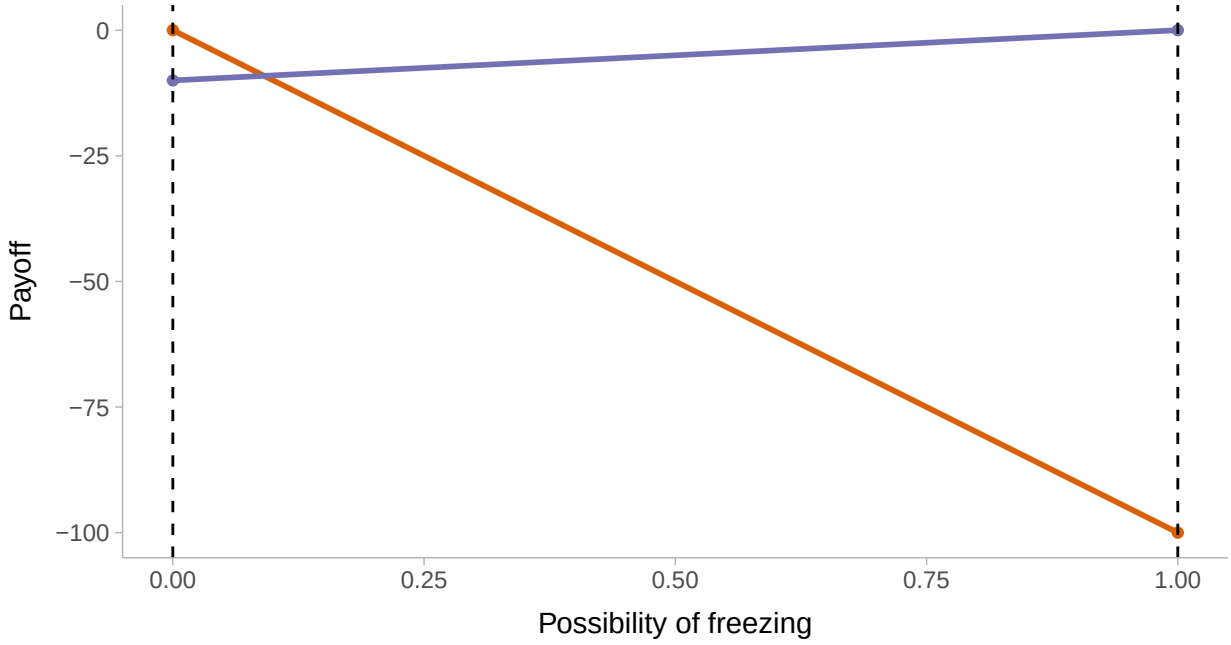
```
# The rational agent's posterior believes vary from visualization condictions, where we assume that CI,
full_information_belief = sapply(sigma_choices, function(m) {pnorm(0, mu, m)})
only_mean_belief = Reduce("+", sapply(sigma_choices, function(m) {pnorm(0, mu, m)})) / 4

# The payoff of full information
full_information_payoff_salt = sapply(full_information_belief, function(belief) {
  salt_freezing_payoff * belief + salt_not_freezing_payoff * (1 - belief)
})
full_information_payoff_no_salt = sapply(full_information_belief, function(belief) {
  no_salt_freezing_payoff * belief + no_salt_not_freezing_payoff * (1 - belief)
})

# The payoff of only mean
only_mean_payoff_salt = salt_freezing_payoff * only_mean_belief +
  salt_not_freezing_payoff * (1 - only_mean_belief)
only_mean_payoff_no_salt = no_salt_freezing_payoff * only_mean_belief +
  no_salt_not_freezing_payoff * (1 - only_mean_belief)

# rational agent's best actions and payoffs
full_information_actions = ifelse(full_information_payoff_salt > full_information_payoff_no_salt,
                                  "Salt",
                                  "No salt")
```

```
only_mean_action = ifelse(only_mean_payoff_salt > only_mean_payoff_no_salt, "Salt", "No salt")

rpos_full_information = Reduce("+", sapply(1:length(full_information_payoff_salt),
                                  function(i) {
                                    max(full_information_payoff_salt[i], full_information_payoff_no_salt
                                  })) / 4

rpos_only_mean = max(only_mean_payoff_salt, only_mean_payoff_no_salt)

list(full_information = rpos_full_information, only_mean = rpos_only_mean)
```

```
## $full_information
## [1] -5.689238
##
## $only_mean
## [1] -7.957626
```

We then draw the expected score of the agent for both no-salt and salt actions as a function of his belief p, where the upper segments represent the rational agent with posterior information's payoff and the lower parts are the theoretical lowerbound of decision's payoff.

```
# The line of no salt
df_no_salt = data.frame(x = c(0, 1), y = c(no_salt_not_freezing_payoff, no_salt_freezing_payoff))
# The line of salt
df_salt = data.frame(x = c(0, 1), y = c(salt_not_freezing_payoff, salt_freezing_payoff))

theme_set(theme_ggdist())
ggplot() +
  geom_line(data = df_no_salt, aes(x, y), color = "#d95f02", size=1) +
  geom_line(data = df_salt, aes(x, y), color = "#7570b3", size=1) +
  geom_point(data = df_no_salt, aes(x, y), color = "#d95f02") +
  geom_point(data = df_salt, aes(x, y), color = "#7570b3") +
  geom_vline(xintercept = 0, linetype="dashed") +
  geom_vline(xintercept = 1, linetype="dashed") +
  theme(aspect.ratio = 0.5) +
  labs(x="Possibility of freezing", y="Payoff")
```


0
−25
−50
−75
−100
Payoff
0.00
0.25
0.50
0.75
1.00
Possibility of freezing