



Modeling Short Time Series

What “Including Prior Information” really looks like

Tim Radtke - satRday Berlin - June 15, 2019

Tim Radtke

- B.Sc. in Economics, University of Mannheim
- M.Sc. in Statistics, Humboldt University Berlin
- Data Scientist, Amorelie
- Data Scientist, Bayer

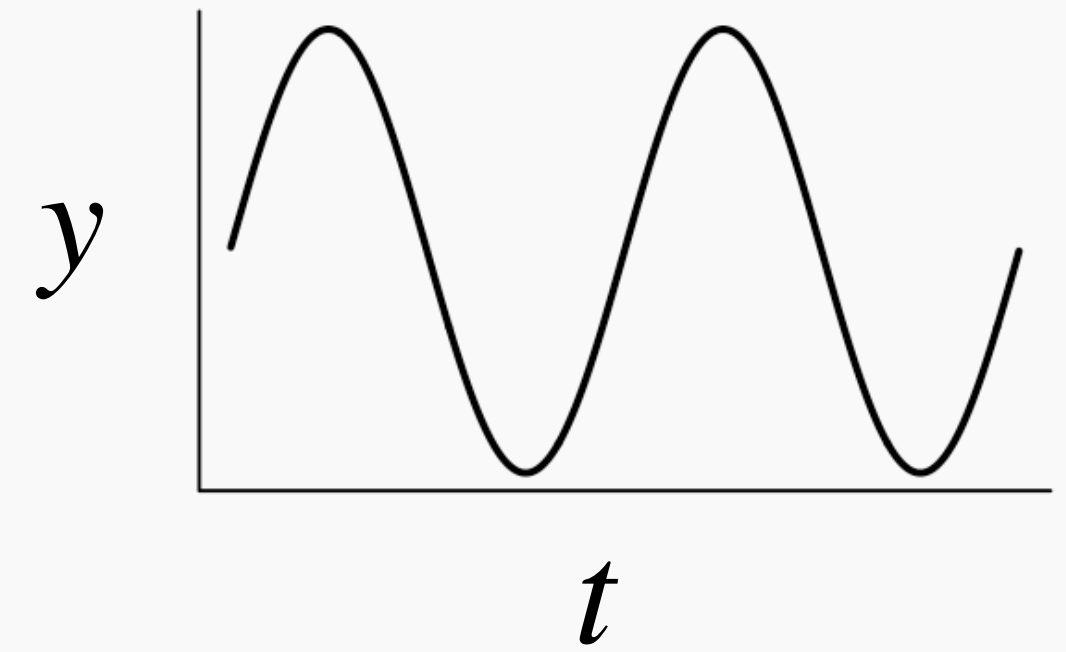
tim@timradtke.net

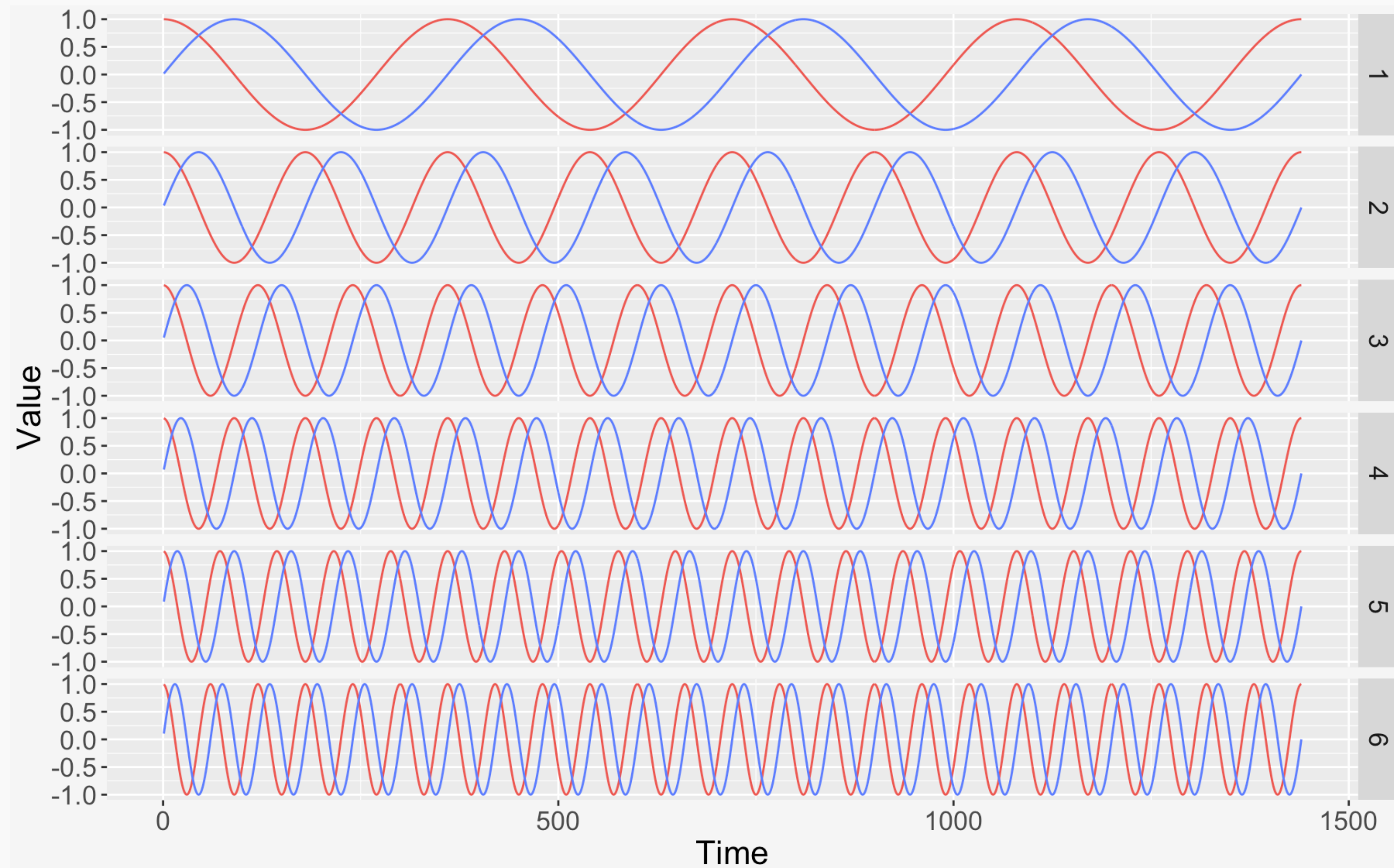
minimizeregret.com

[@timradtke](https://twitter.com/timradtke)

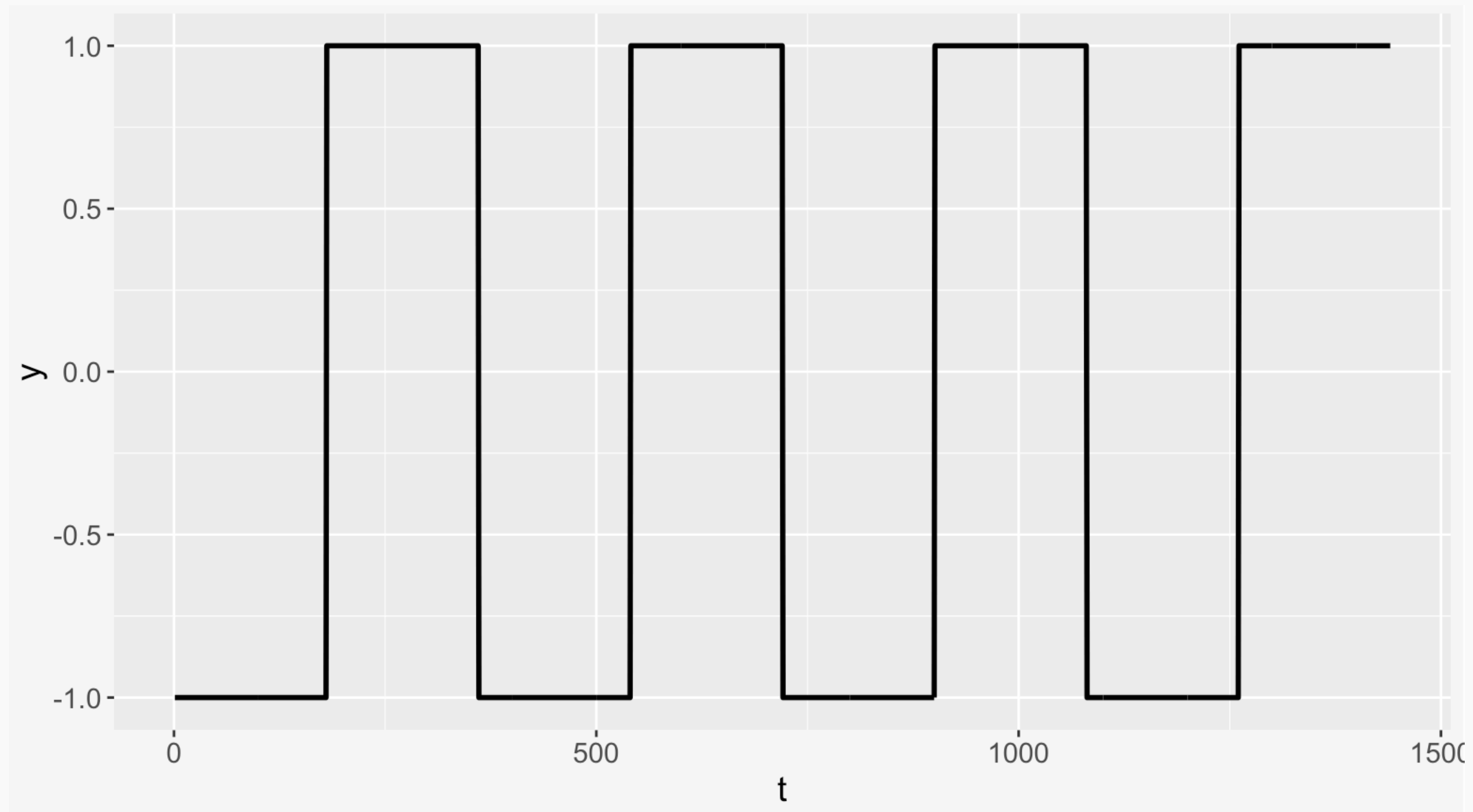
Fourier Terms

$$\sin(2\pi t) \quad \cos(2\pi t)$$





Fourier terms at multiples of a period length of 360.



Step function with period length 360.

Fourier Terms

$$y(t) = \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

Fourier Terms

with $K = 6$, these are 6 sine and 6 cosine curves

$$y(t) = \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

Fourier Terms

with $K = 6$, these are 6 sine and 6 cosine curves

$$y(t) = \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

β_k and γ_k are coefficients in a linear model

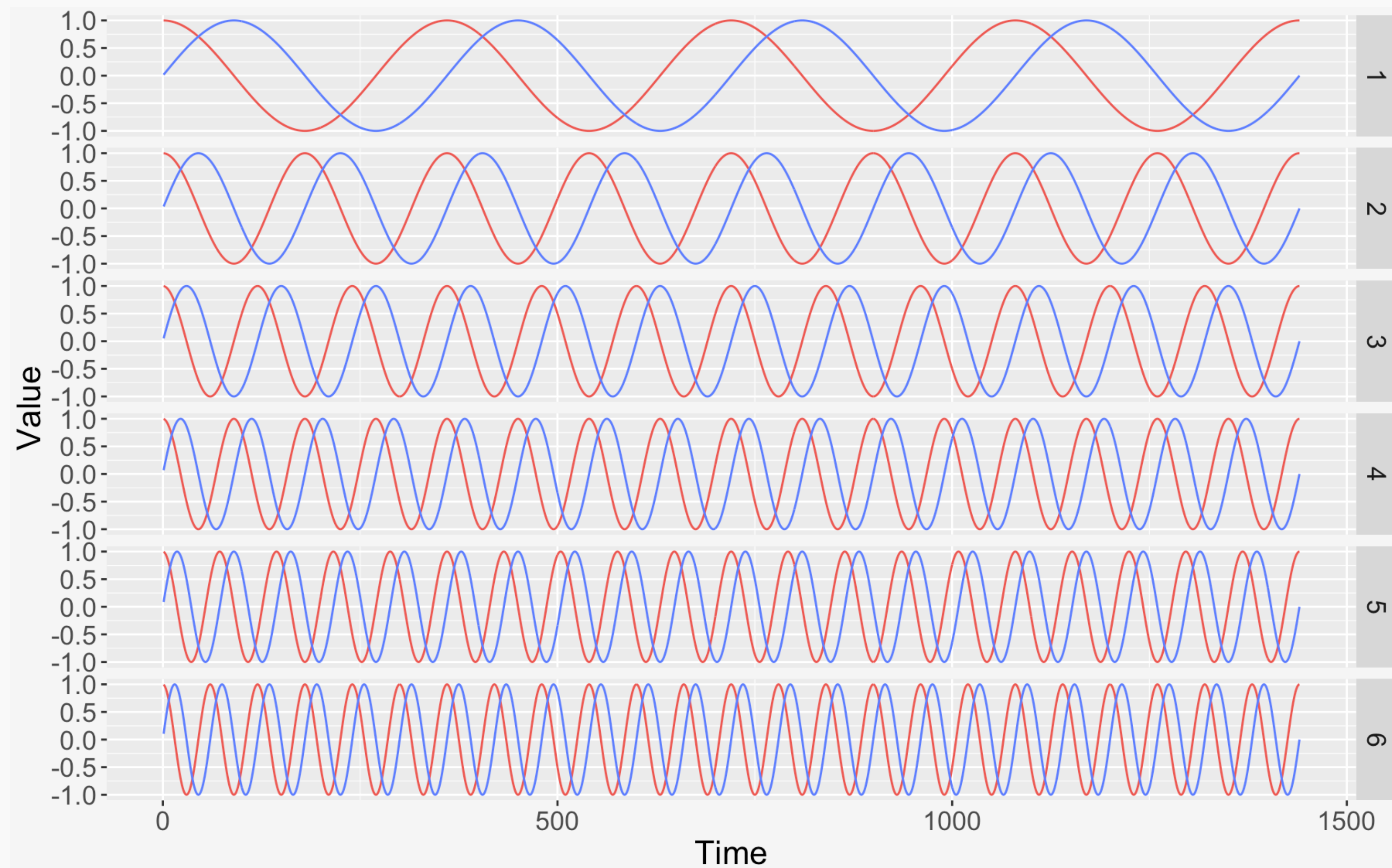


```
> fouriers <- forecast::fourier(ts(1:(4*360),  
+                               frequency = 360), K = 6)
```

```
>
```

```
> fouriers[1:5, 1:4]
```

	S1-360	C1-360	S2-360	C2-360
[1,]	0.01745241	0.9998477	0.03489950	0.9993908
[2,]	0.03489950	0.9993908	0.06975647	0.9975641
[3,]	0.05233596	0.9986295	0.10452846	0.9945219
[4,]	0.06975647	0.9975641	0.13917310	0.9902681
[5,]	0.08715574	0.9961947	0.17364818	0.9848078



Fourier terms at multiples of a period length of 360.

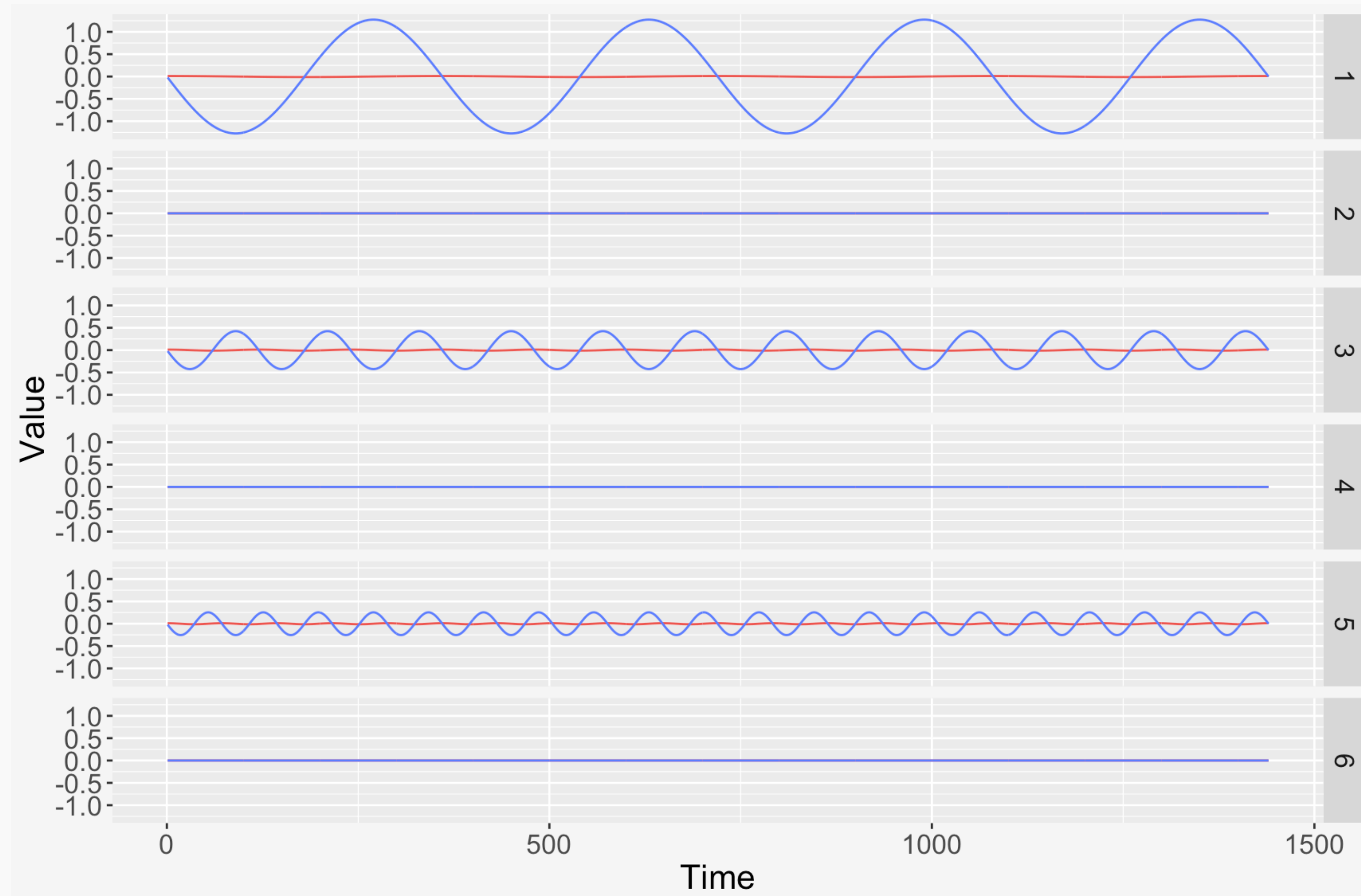


```
> step_function <- rep(rep(c(-1,1), each = 180), 4)
>
> fouriers <- forecast::fourier(ts(step_function,
+                                frequency = 360), K = 6)
>
> fourier_fit <- lm(step_function ~ -1 + fouriers)
```

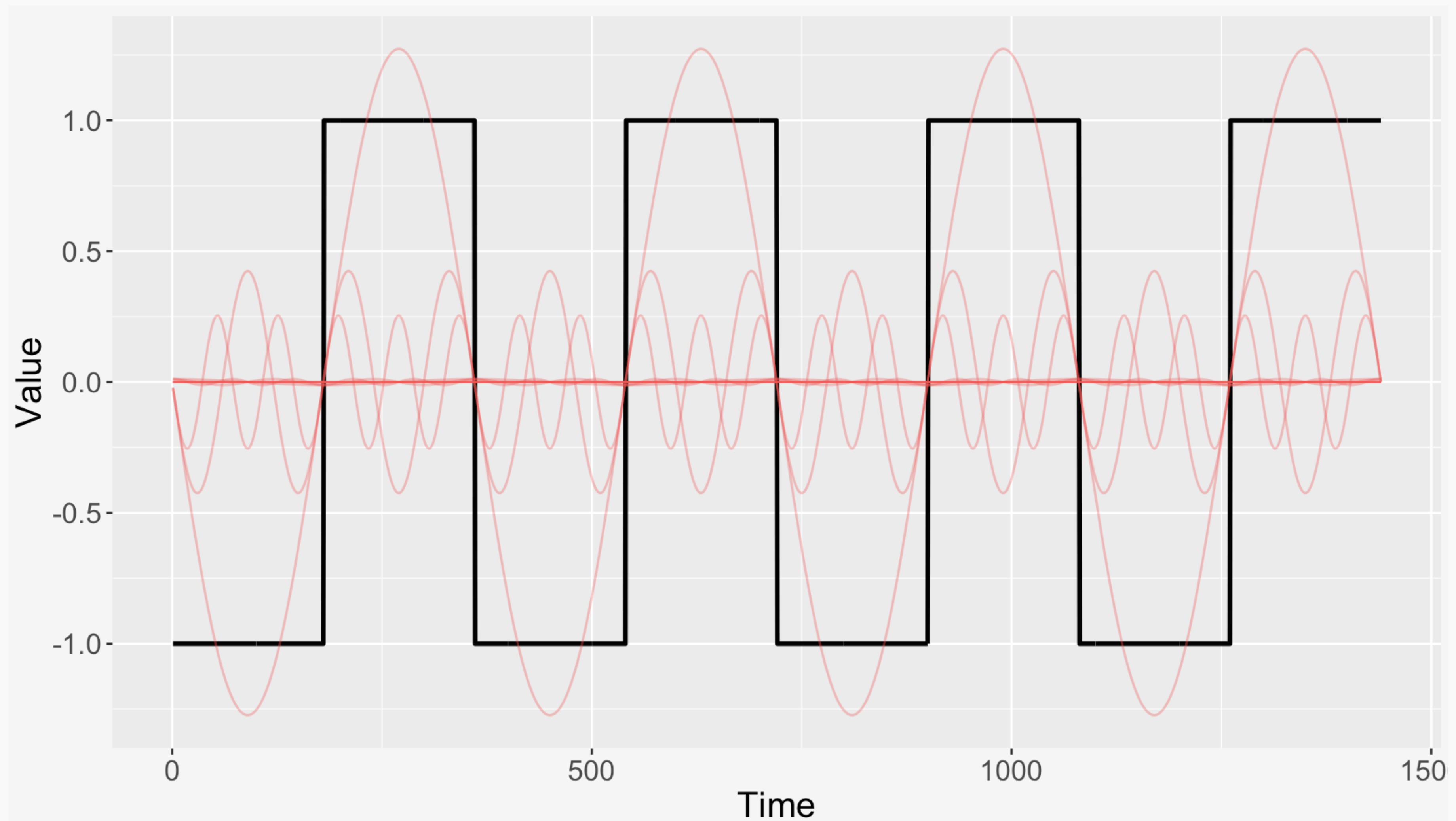


```
> step_function <- rep(rep(c(-1,1), each = 180), 4)
>
> fouriers <- forecast::fourier(ts(step_function,
+                                frequency = 360), K = 6)
>
> fourier_fit <- lm(step_function ~ -1 + fouriers)
```

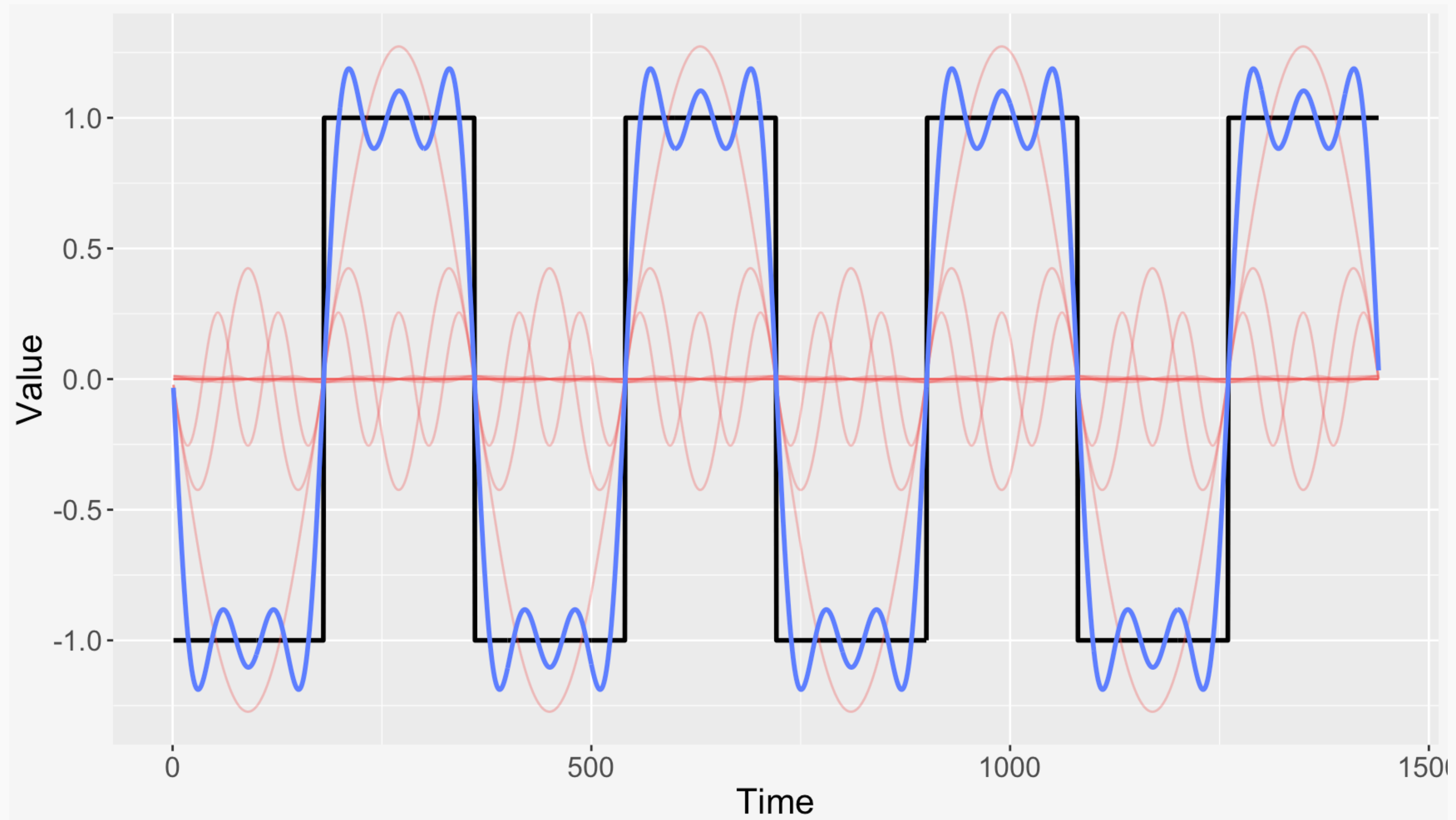
$$y(t) = \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$



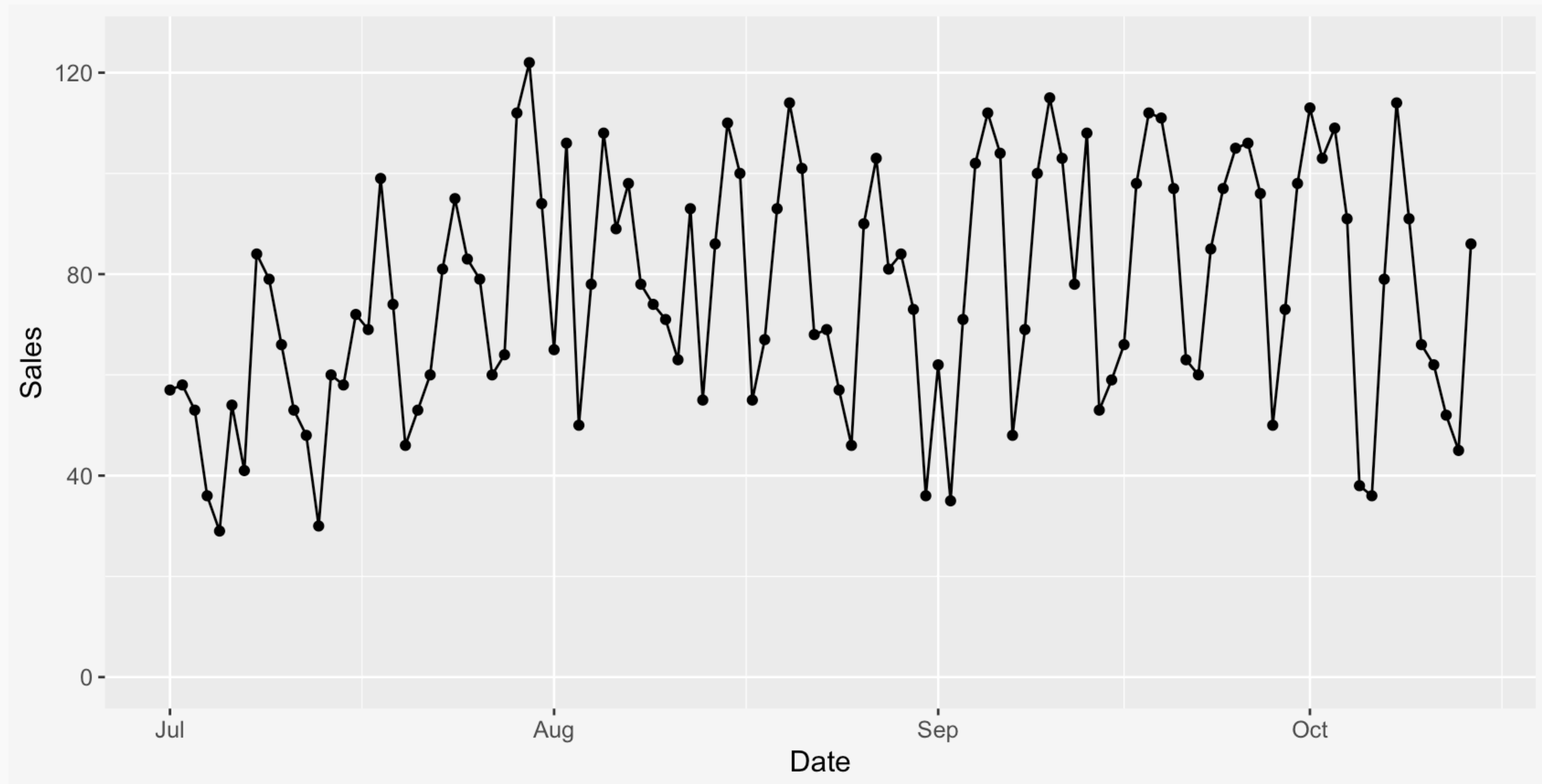
Each Fourier term weighted by its coefficient in linear model.



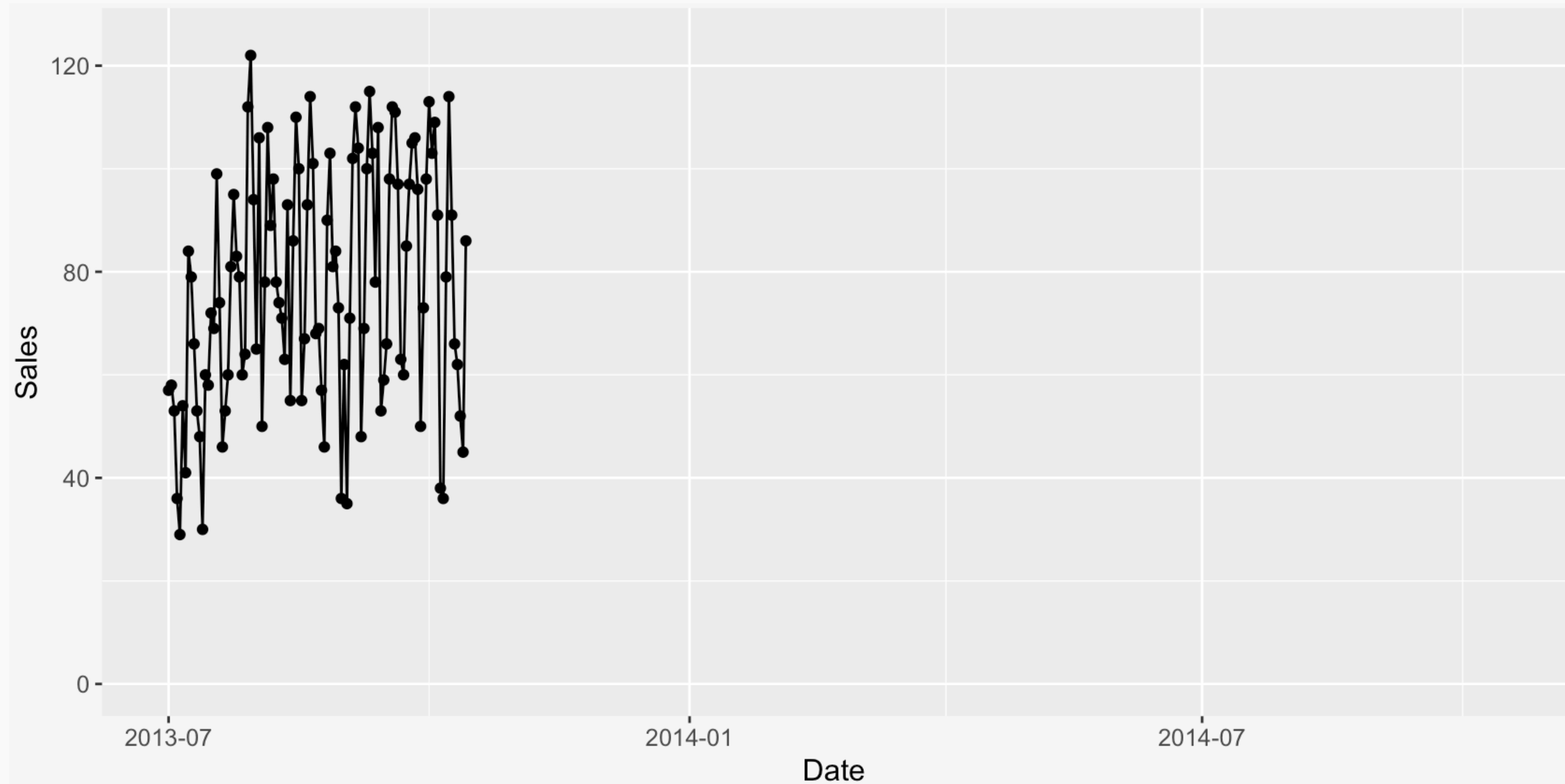
Step function and Fourier terms as weighted by their coefficient.



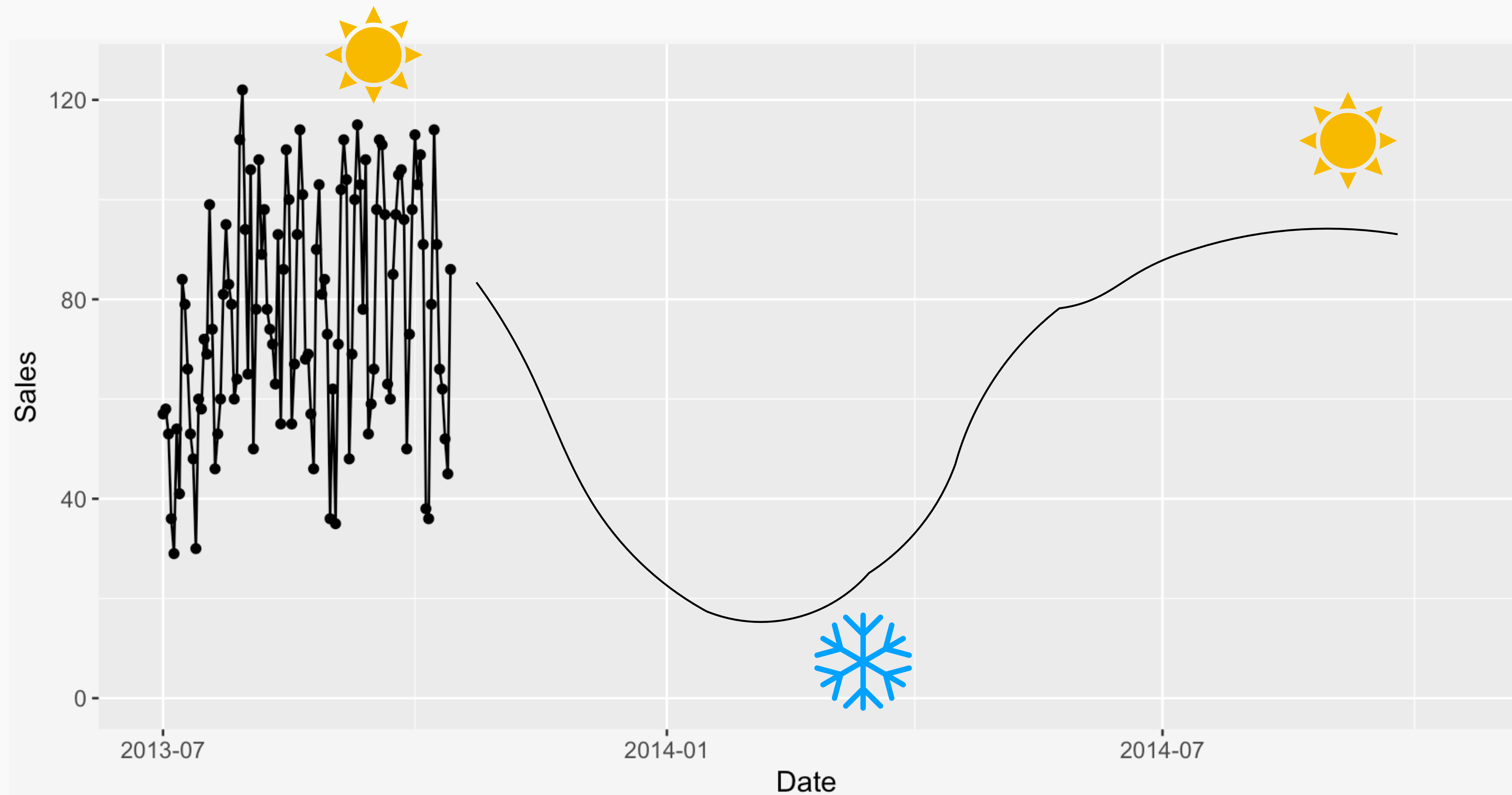
Sum of weighted Fourier terms gives approximation to step function.



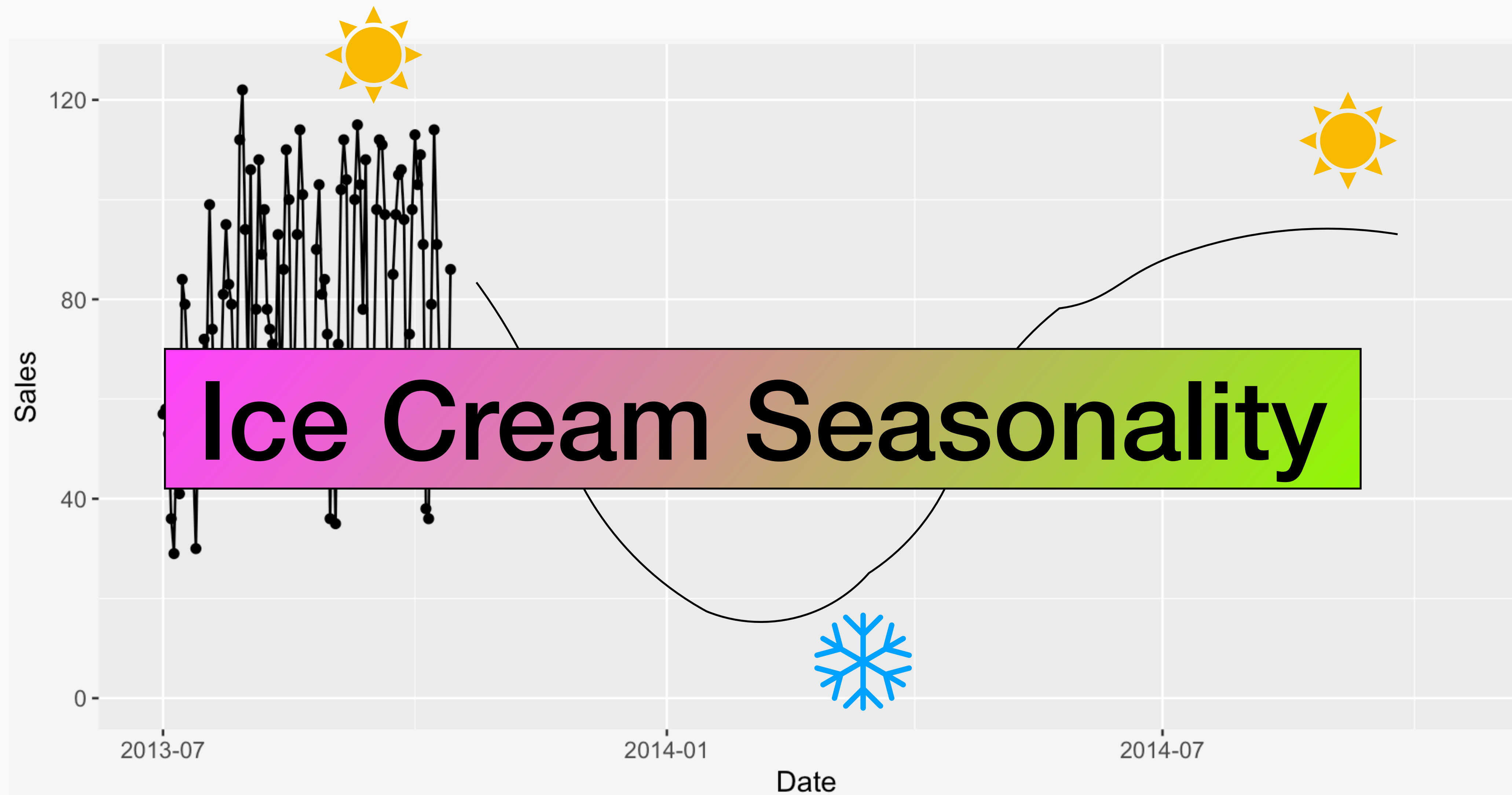
Daily bike sales observed over 3.5 months in 2013 in a NYC shop.



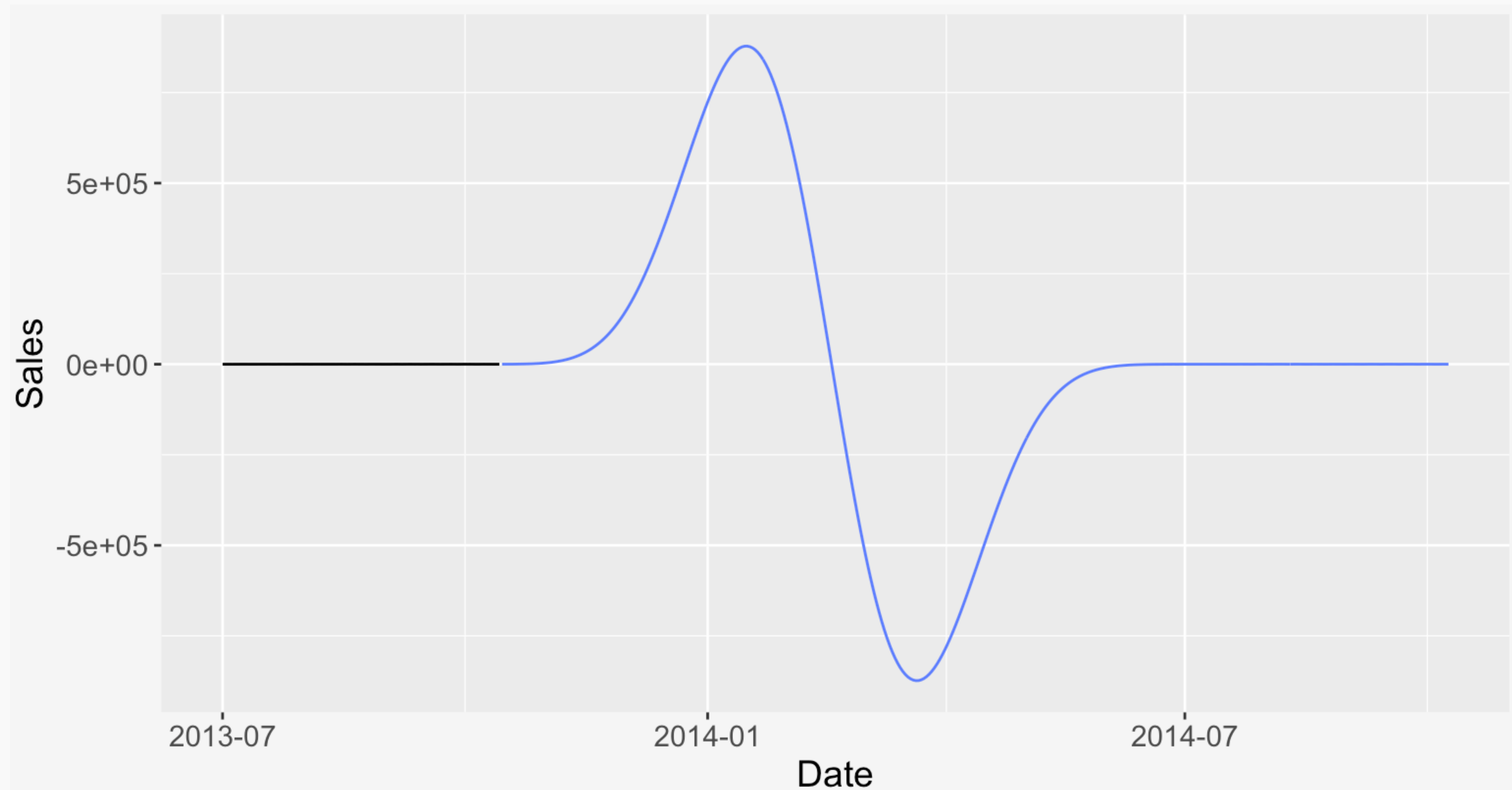
Manager needs forecast for the upcoming year to order new bikes.



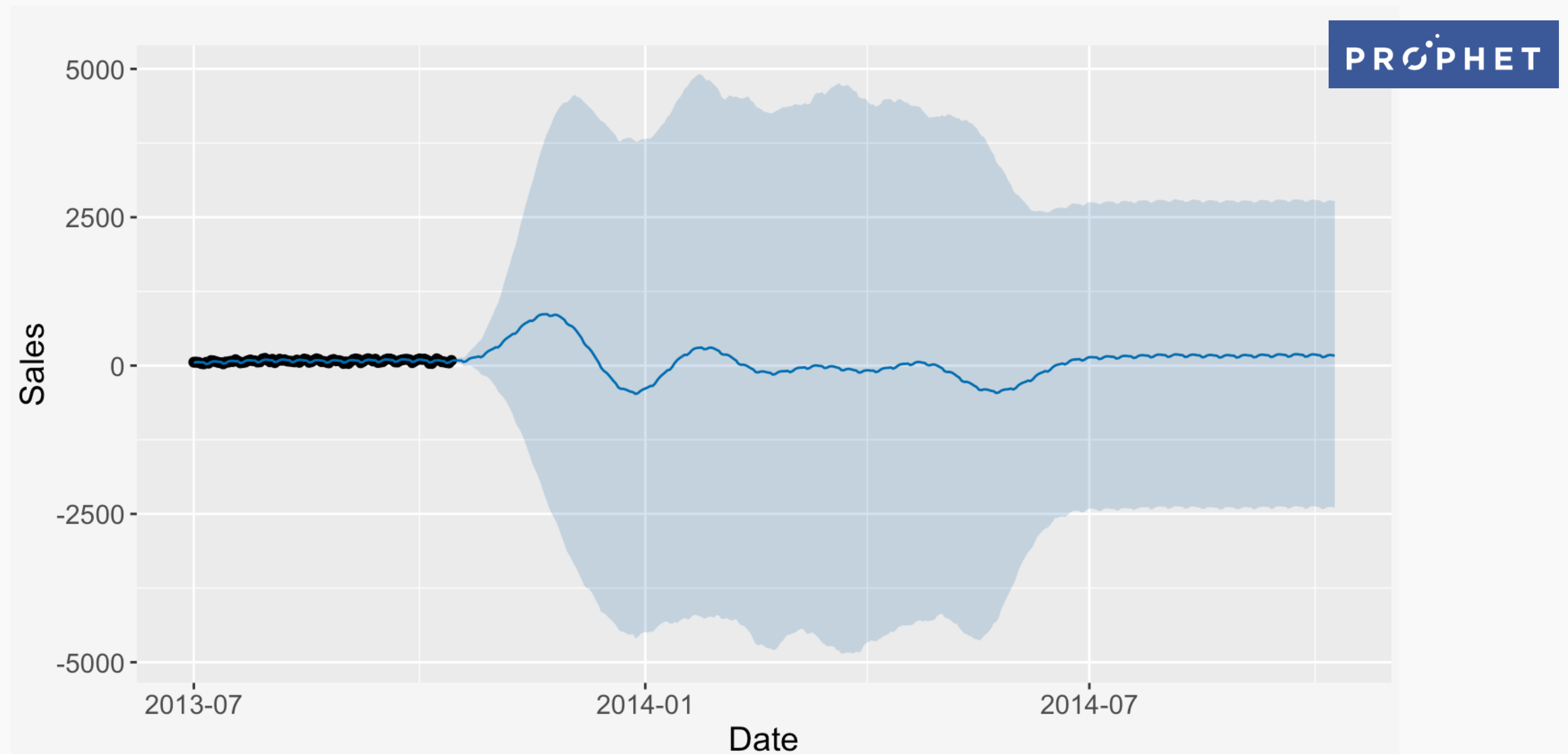
Manager needs forecast for the upcoming year to order new bikes.



Manager needs forecast for the upcoming year to order new bikes.



Forecast from `forecast::auto.arima()` with Fourier terms for yearly seasonality.



Forecast from default prophet model with Fourier terms for yearly seasonality.

Data Generating Process

$$Y_t \sim \mathbf{NegBin}(\lambda_t, \phi) \quad \text{where} \quad \lambda_t, \phi > 0, \quad E[Y_t] = \lambda_t$$

Data Generating Process

$$Y_t \sim \mathbf{NegBin}(\lambda_t, \phi) \quad \text{where} \quad \lambda_t, \phi > 0, \quad E[Y_t] = \lambda_t$$

$$\lambda_t = (1 - \delta - \gamma) \cdot \mu_t + \delta \cdot \lambda_{t-1} + \gamma \cdot y_{t-1}$$

$$\text{where} \quad 0 \leq \delta \leq 1 \quad \text{and} \quad 0 \leq \gamma \leq 1 - \delta$$

Data Generating Process

$$Y_t \sim \mathbf{NegBin}(\lambda_t, \phi) \quad \text{where} \quad \lambda_t, \phi > 0, \quad E[Y_t] = \lambda_t$$

$$\lambda_t = (1 - \delta - \gamma) \cdot \mu_t + \delta \cdot \lambda_{t-1} + \gamma \cdot y_{t-1}$$

$$\text{where} \quad 0 \leq \delta \leq 1 \quad \text{and} \quad 0 \leq \gamma \leq 1 - \delta$$

$$\log(\mu_t) = \alpha_0 + \sum_{i=1}^6 \alpha_i D_i + \tau \frac{t}{m} + \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

Data Generating Process

$$Y_t \sim \text{NegBin}(\lambda_t, \phi) \quad \text{where} \quad \lambda_t, \phi > 0$$

$$\lambda_t = (1 - \delta - \gamma) \cdot \mu_t + \delta \cdot \lambda_{t-1} + \gamma \cdot y_{t-1}$$

$$\text{where} \quad 0 \leq \delta \leq 1 \quad \text{and} \quad 0 \leq \gamma \leq 1 - \delta$$

$$\log(\mu_t) = \alpha_0 + \sum_{i=1}^6 \alpha_i D_i + \tau \frac{t}{m} + \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

Data Generating Process

$$Y_t \sim \text{NegBin}(\lambda_t, \phi) \quad \text{where} \quad \lambda_t, \phi > 0$$

$$\lambda_t = (1 - \delta - \gamma) \cdot \mu_t + \delta \cdot \lambda_{t-1} + \gamma \cdot y_{t-1}$$

$$\text{where} \quad 0 \leq \delta \leq 1 \quad \text{and} \quad 0 \leq \gamma \leq 1 - \delta$$

$$\log(\mu_t) = \alpha_0 + \sum_{i=1}^6 \alpha_i D_i + \tau \frac{t}{m} + \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

prior distributions on β_k and γ_k define what seasonality is expected

Data Generating Process

$$\log(\mu_t) = \alpha_0 + \sum_{i=1}^6 \alpha_i D_i + \tau \frac{t}{m} + \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

$$\beta_k, \gamma_k \sim N(0, 0.25) \quad \textbf{for } k = 1, \dots, K$$



+





```
> library(rstan)
> options(mc.cores = 4)
> rstan_options(auto_write = TRUE)
>
> negbin_prior_stan <- stan_model("negbin_prior.stan")
```

```
data {  
  int<lower=1> N;  
  int<lower=1> K;  
  int<lower=1> F;  
  int y[N];  
  matrix[N, K] x;  
  matrix[N, F] fouriers;  
  matrix[N, 1] trend;  
  real trend_loc;  
  real trend_sd;  
  real fourier_loc[F];  
  real fourier_sd[F];  
}
```




```
parameters {  
    real<lower=0,upper=1> phi;  
    real<lower=0,upper=1-phi> alpha;  
    real delta_inv;  
    vector[K] beta;  
    vector[F] beta_fourier;  
    vector[1] beta_trend;  
    real beta_0;  
}
```





```
transformed parameters {  
  real<lower=0> delta;  
  real<lower=0> mu_t[N];  
  vector<lower=0>[N] seasonal;  
  delta = 1/pow(delta_inv, 2);  
  seasonal = exp(x * beta + fouriers * beta_fourier +  
                trend * beta_trend + beta_0);  
  mu_t[1] = y[1];  
  for (n in 2:N) {  
    mu_t[n] = (1 - phi - alpha) * seasonal[n] +  
              phi * mu_t[n-1] + alpha * y[n-1];  
  }  
}
```

```
model {  
  phi ~ gamma(1,10);  
  alpha ~ gamma(0.5,10);  
  delta_inv ~ normal(0,0.5);  
  beta_fourier ~ normal(fourier_loc, fourier_sd);  
  beta_trend ~ normal(trend_loc, trend_sd);  
  beta ~ normal(0,0.5);  
  beta_0 ~ normal(2,1);  
  for (n in 2:N) {  
    target += neg_binomial_2_lpmf(y[n] | mu_t[n], delta);  
  }  
}
```





```
model {  
  phi ~ gamma(1,10);  
  alpha ~ gamma(0.5,10);  
  delta_inv ~ normal(0,0.5);  
  beta_fourier ~ normal(fourier_loc, fourier_sd);  
  beta_trend ~ normal(trend_loc, trend_sd);  
  beta ~ normal(0,0.5);  
  beta_0 ~ normal(2,1);  
  for (n in 2:N) {  
    target += neg_binomial_2_lpmf(y[n] | mu_t[n], delta);  
  }  
}
```

$\beta_k, \gamma_k \sim N(0, 0.25)$ for $k = 1, \dots, K$



```
res_y <- res_mu <- matrix(ncol = 1000, nrow = N_hciti)
for(j in 1:1000) {
  delta_inv <- rnorm(1, 0, 0.5)
  delta <- 1/(delta_inv^2)
  phi <- rgamma(1,1,10)
  alpha <- rgamma(1, 0.5, 10)
  if(phi>1) phi <- 1
  if(alpha > 1-phi) alpha <- 1 - phi

  beta_fouriers <- rnorm(12, 0, 0.25)
  beta_trend <- rnorm(1, 0.03, 0.02)
  beta_0 <- rnorm(1, 4, 0.5)
  beta <- rnorm(6,0,0.1)

  seasonal <- exp(beta_0 + beta_trend * trend_hciti + as.numeric(xreg_hciti %*% beta) + as.numeric(fourier_hciti %*% beta_fouriers))

  mu_t <- y_t <- rep(NA, N_hciti)
  mu_t[1] <- y_hciti[1] #(1-phi-alpha) * seasonal[1]
  y_t[1] <- y_hciti[1] #rnbino(1, mu = mu_t[1], size = delta)

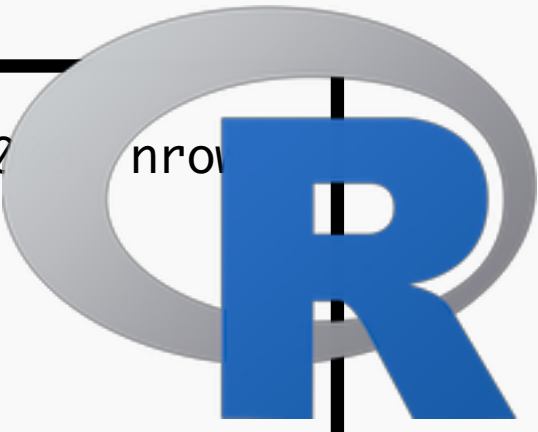
  for(i in 2:N_hciti) {
    mu_t[i] <- (1 - phi - alpha) * seasonal[i] + phi * mu_t[i-1] + alpha * y_t[i-1]
    y_t[i] <- rnbino(1, mu = mu_t[i], size = delta)
  }

  res_mu[,j] <- mu_t
  res_y[,j] <- y_t
}
```

```
res_y <- matrix(ncol = 1000, nrow = N_hciti)
res_mu <- matrix(ncol = 1000, nrow = N_hciti)
```

```
for(j in 1:1000) {
  delta_inv <- rnorm(1, 0, 0.5)
  delta <- 1/(delta_inv^2)
  phi <- rgamma(1, 1, 10)
  alpha <- rgamma(1, 0.5, 10)
  if(phi > 1) phi <- 1
  if(alpha > 1 - phi) alpha <- 1 - phi
```

```
  beta_fouriers <- rnorm(12, 0, 0.25)
  beta_trend <- rnorm(1, 0.03, 0.02)
  beta_0 <- rnorm(1, 4, 0.5)
  beta <- rnorm(6, 0, 0.1)
```



```
res_y <- res_mu <- matrix(ncol = 1000, nrow = N_hciti)
for(j in 1:1000) {
  delta_inv <- rnorm(1, 0, 0.5)
  delta <- 1/(delta_inv^2)
  phi <- rgamma(1,1,10)
  alpha <- rgamma(1, 0.5, 10)
  if(phi>1) phi <- 1
  if(alpha > 1-phi) alpha <- 1 - phi

  beta_fouriers <- rnorm(12,0,0.25)
  beta_trend <- rnorm(1, 0.03, 0.02)
  beta_0 <- rnorm(1, 4, 0.5)
  beta <- rnorm(6,0,0.1)

  seasonal <- exp(beta_0 + beta_trend *
trend_hciti + as.numeric(xreg_hciti %*%
beta) + as.numeric(fourier_hciti %*%
beta_fouriers))

  mu_t <- y_t <- rep(NA, N_hciti)
  mu_t[1] <- y_hciti[1] #(1-phi-alpha) *
seasonal[1]
  y_t[1] <- y_hciti[1] #rnbino(1, mu =
mu_t[1], size = delta)

  for(i in 2:N_hciti) {
    mu_t[i] <- (1 - phi - alpha) *
seasonal[i] + phi * mu_t[i-1] + alpha *
y_t[i-1]
    y_t[i] <- rnbinom(1, mu = mu_t[i], size
= delta)
  }

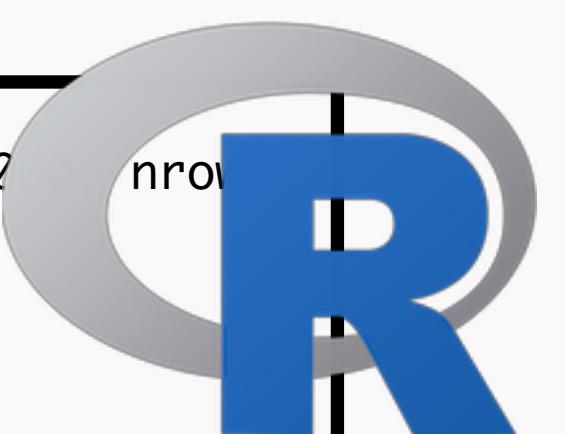
  res_mu[,j] <- mu_t
  res_y[,j] <- y_t
}
```

```
res_y <- matrix(ncol = 1000, nrow = N_hciti)
res_mu <- matrix(ncol = 1000, nrow = N_hciti)
```

```
for(j in 1:1000) {
  delta_inv <- rnorm(1, 0, 0.5)
  delta <- 1/(delta_inv^2)
  phi <- rgamma(1, 1, 10)
  alpha <- rgamma(1, 0.5, 10)
  if(phi > 1) phi <- 1
  if(alpha > 1 - phi) alpha <- 1 - phi
```

```
  beta_fouriers <- rnorm(12, 0, 0.25)
  beta_trend <- rnorm(1, 0.03, 0.02)
  beta_0 <- rnorm(1, 4, 0.5)
  beta <- rnorm(6, 0, 0.1)
```

$\beta_k, \gamma_k \sim N(0, 0.25)$ for $k = 1, \dots, K$



```
res_y <- res_mu <- matrix(ncol = 1000, nrow = N_hciti)
for(j in 1:1000) {
  delta_inv <- rnorm(1, 0, 0.5)
  delta <- 1/(delta_inv^2)
  phi <- rgamma(1,1,10)
  alpha <- rgamma(1, 0.5, 10)
  if(phi>1) phi <- 1
  if(alpha > 1-phi) alpha <- 1 - phi

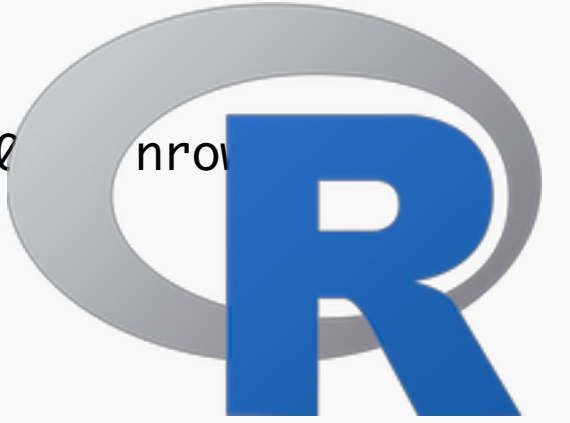
  beta_fouriers <- rnorm(12,0,0.25)
  beta_trend <- rnorm(1, 0.03, 0.02)
  beta_0 <- rnorm(1, 4, 0.5)
  beta <- rnorm(6,0,0.1)

  seasonal <- exp(beta_0 + beta_trend *
trend_hciti + as.numeric(xreg_hciti %*%
beta) + as.numeric(fourier_hciti %*%
beta_fouriers))

  mu_t <- y_t <- rep(NA, N_hciti)
  mu_t[1] <- y_hciti[1] #(1-phi-alpha) *
seasonal[1]
  y_t[1] <- y_hciti[1] #rnbino(1, mu =
mu_t[1], size = delta)

  for(i in 2:N_hciti) {
    mu_t[i] <- (1 - phi - alpha) *
seasonal[i] + phi * mu_t[i-1] + alpha *
y_t[i-1]
    y_t[i] <- rnbino(1, mu = mu_t[i], size
= delta)
  }

  res_mu[,j] <- mu_t
  res_y[,j] <- y_t
}
```

```
seasonal <- exp(beta_0 +
               beta_trend * trend_hciti +
               as.numeric(xreg_hciti %*% beta) +
               as.numeric(fourier_hciti %*% beta_fouriers))

mu_t <- y_t <- rep(NA, N_hciti)

mu_t[1] <- y_hciti[1]
y_t[1] <- y_hciti[1]

for(i in 2:N_hciti) {
  mu_t[i] <- (1 - phi - alpha) * seasonal[i] +
             phi * mu_t[i-1] + alpha * y_t[i-1]
  y_t[i] <- rbinom(1, mu = mu_t[i], size = delta)
}

res_mu[,j] <- mu_t
res_y[,j] <- y_t
}
```

```
res_y <- res_mu <- matrix(ncol = 10, nrow
= N_hciti)
for(j in 1:1000) {
  delta_inv <- rnorm(1, 0, 0.5)
  delta <- 1/(delta_inv^2)
  phi <- rgamma(1,1,10)
  alpha <- rgamma(1, 0.5, 10)
  if(phi>1) phi <- 1
  if(alpha > 1-phi) alpha <- 1 - phi

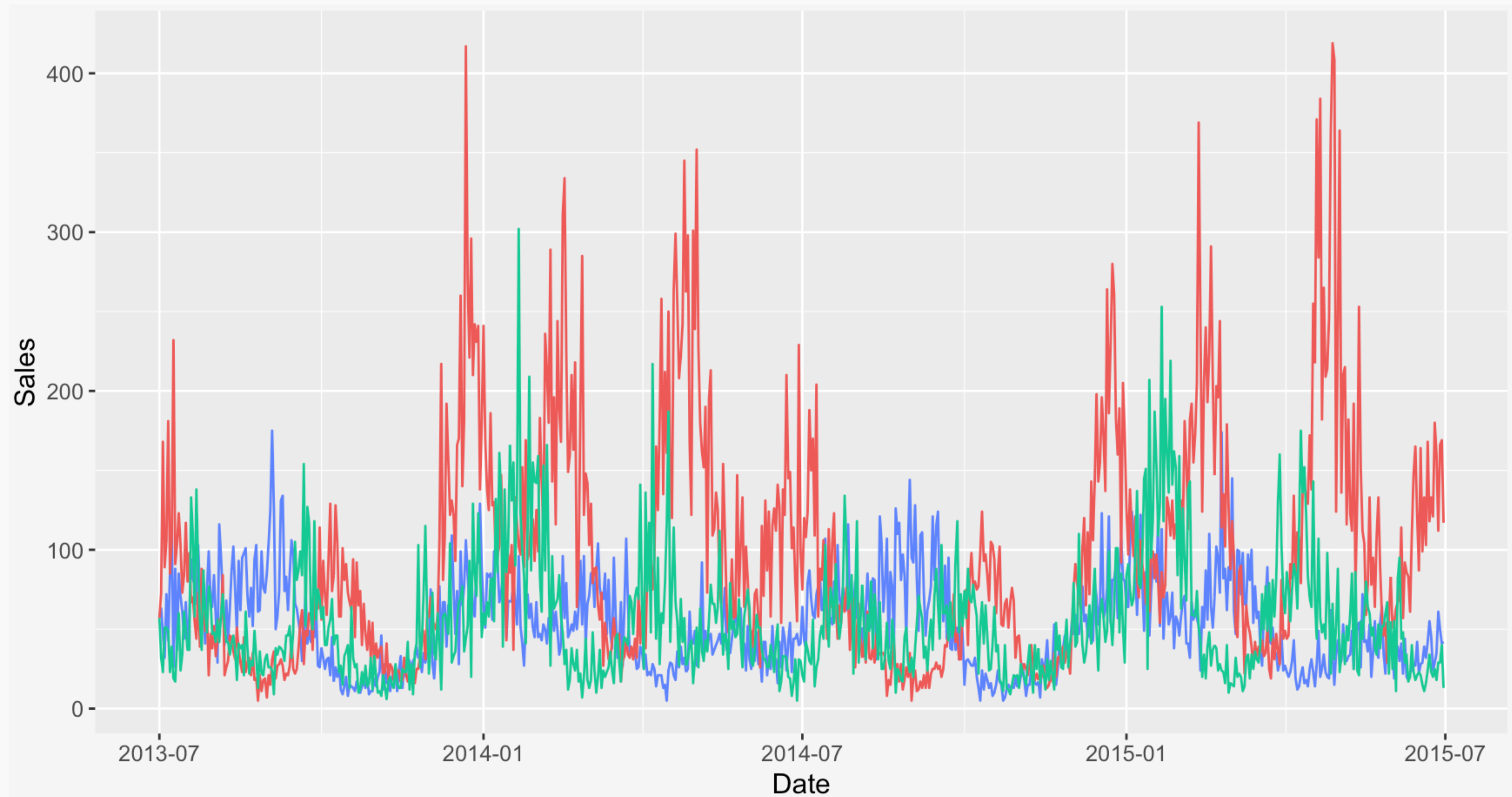
  beta_fouriers <- rnorm(12,0,0.25)
  beta_trend <- rnorm(1, 0.03, 0.02)
  beta_0 <- rnorm(1, 4, 0.5)
  beta <- rnorm(6,0,0.1)
```

```
  seasonal <- exp(beta_0 + beta_trend *
trend_hciti + as.numeric(xreg_hciti %*%
beta) + as.numeric(fourier_hciti %*%
beta_fouriers))

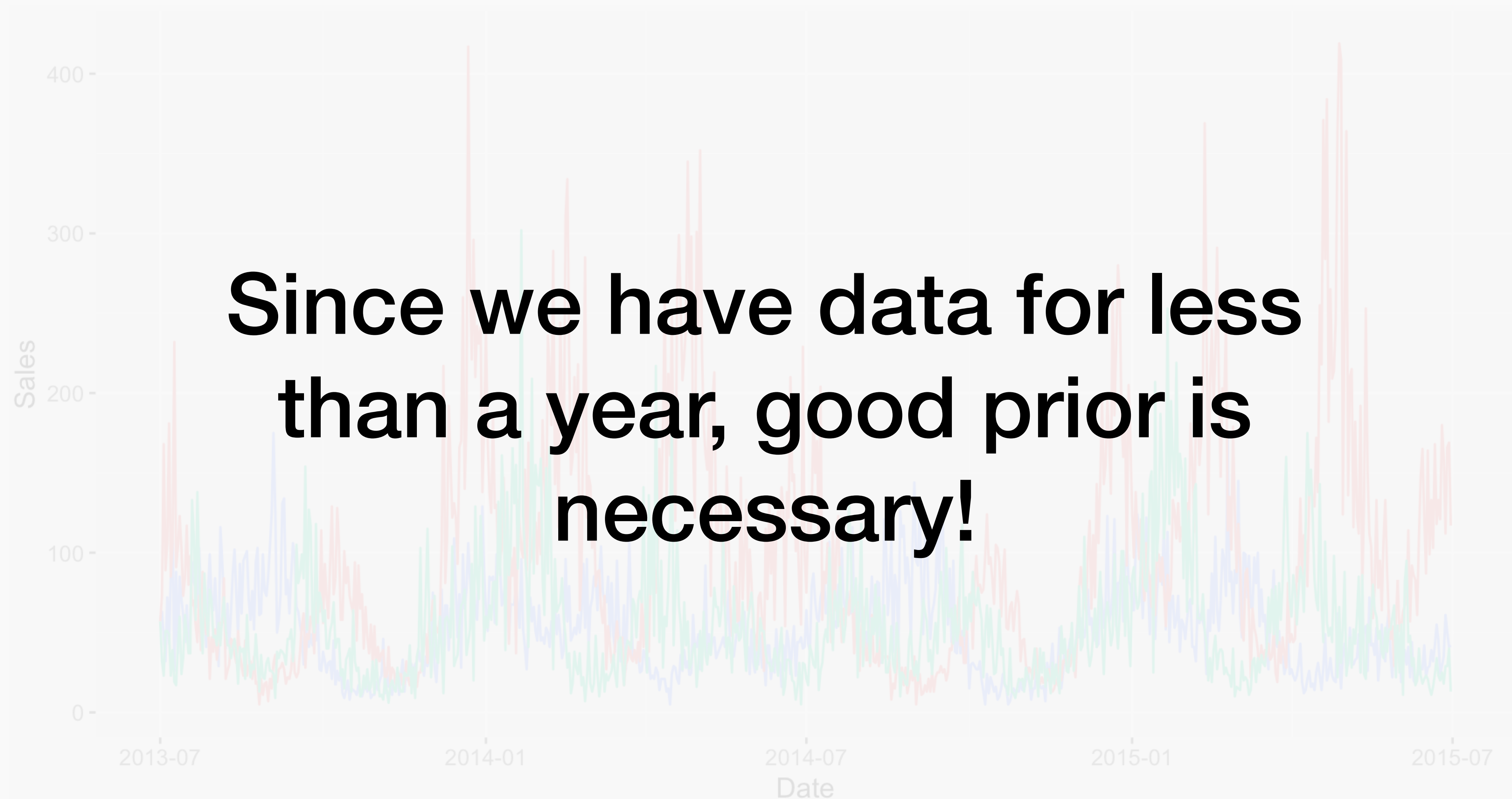
  mu_t <- y_t <- rep(NA, N_hciti)
  mu_t[1] <- y_hciti[1] #(1-phi-alpha) *
seasonal[1]
  y_t[1] <- y_hciti[1] #rbinom(1, mu =
mu_t[1], size = delta)

  for(i in 2:N_hciti) {
    mu_t[i] <- (1 - phi - alpha) *
seasonal[i] + phi * mu_t[i-1] + alpha *
y_t[i-1]
    y_t[i] <- rbinom(1, mu = mu_t[i], size
= delta)
  }

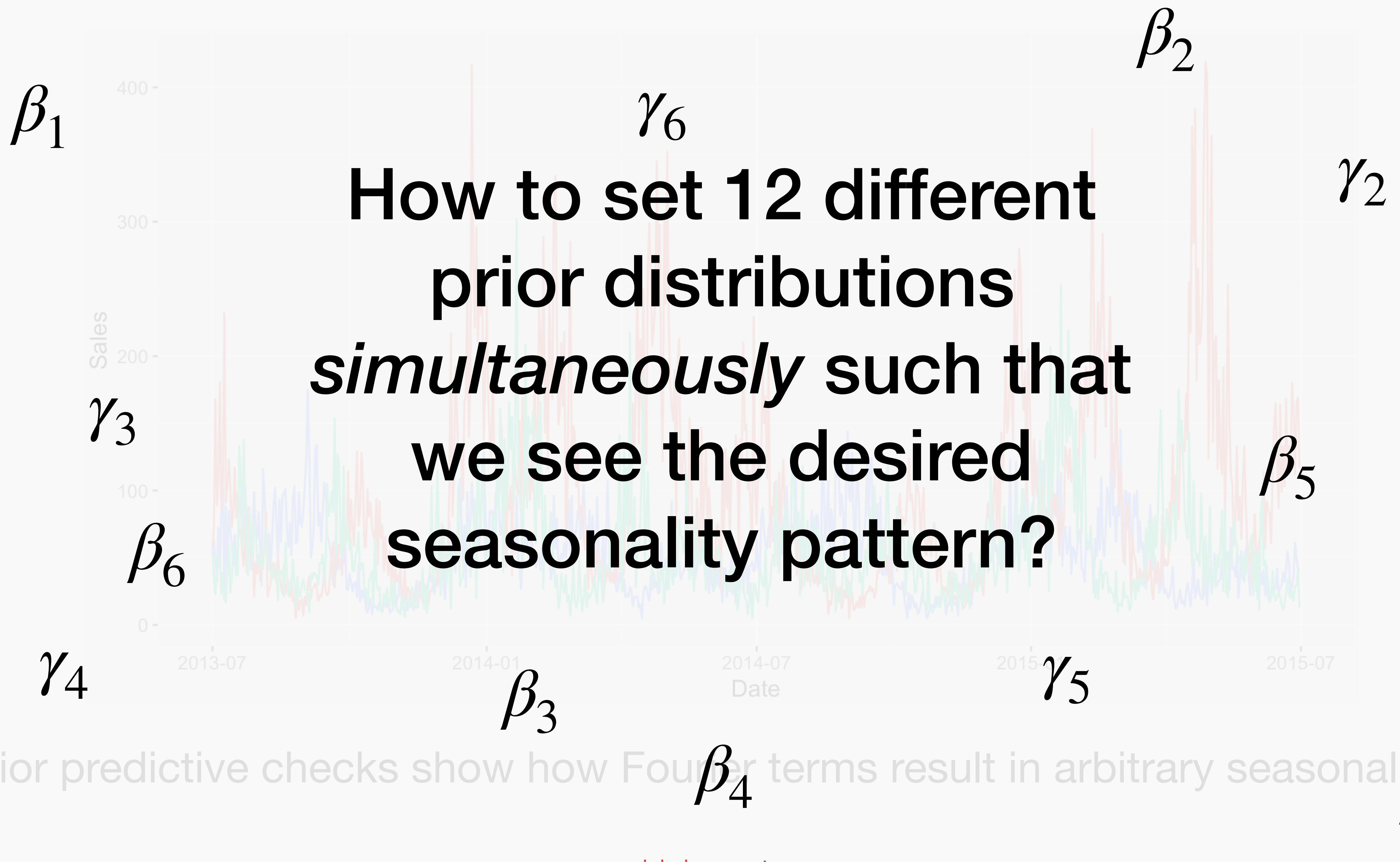
  res_mu[,j] <- mu_t
  res_y[,j] <- y_t
}
```

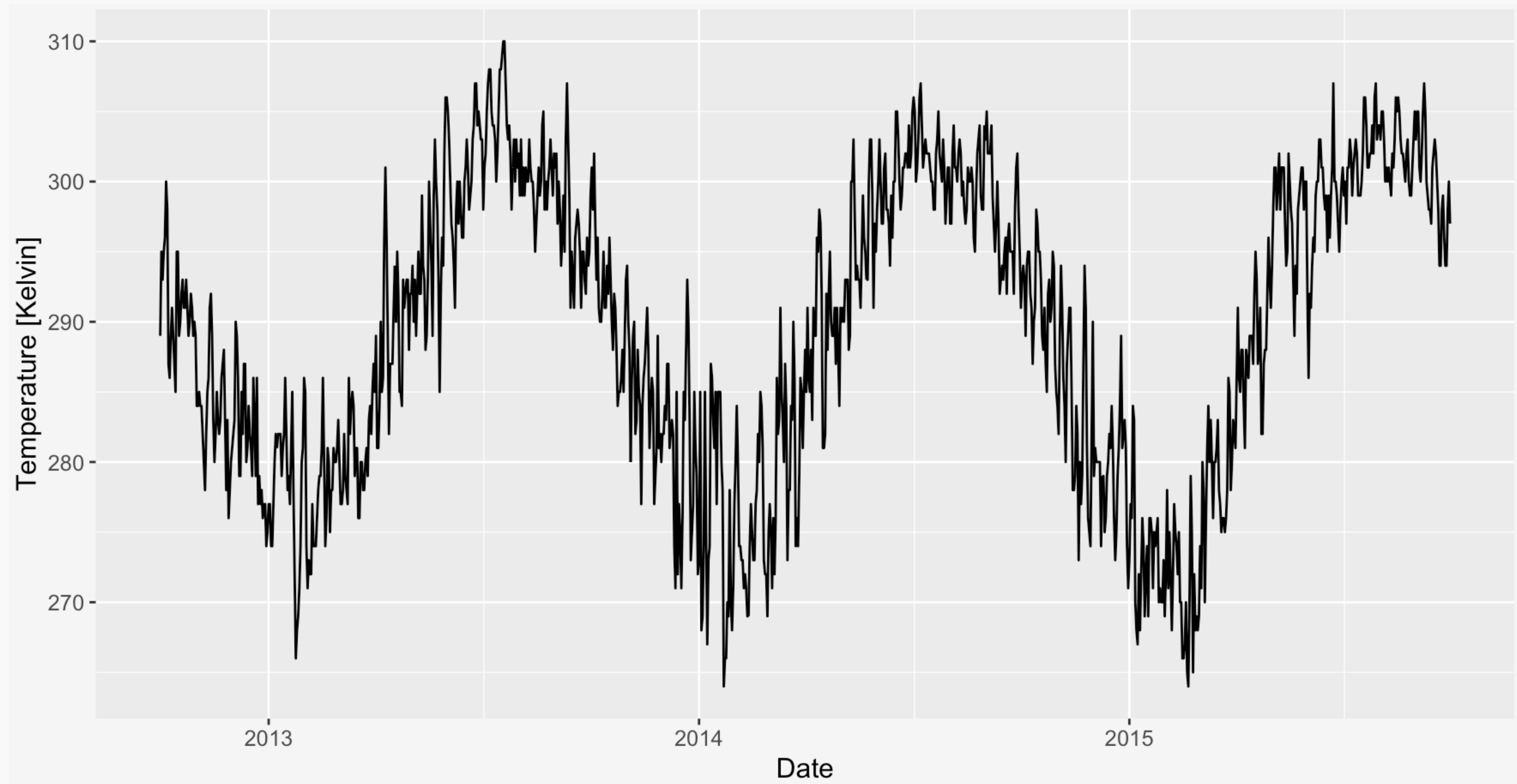
Prior predictive checks show how Fourier terms result in arbitrary seasonalities.



Prior predictive checks show how Fourier terms result in arbitrary seasonalities.



Prior predictive checks show how Fourier terms result in arbitrary seasonalities.

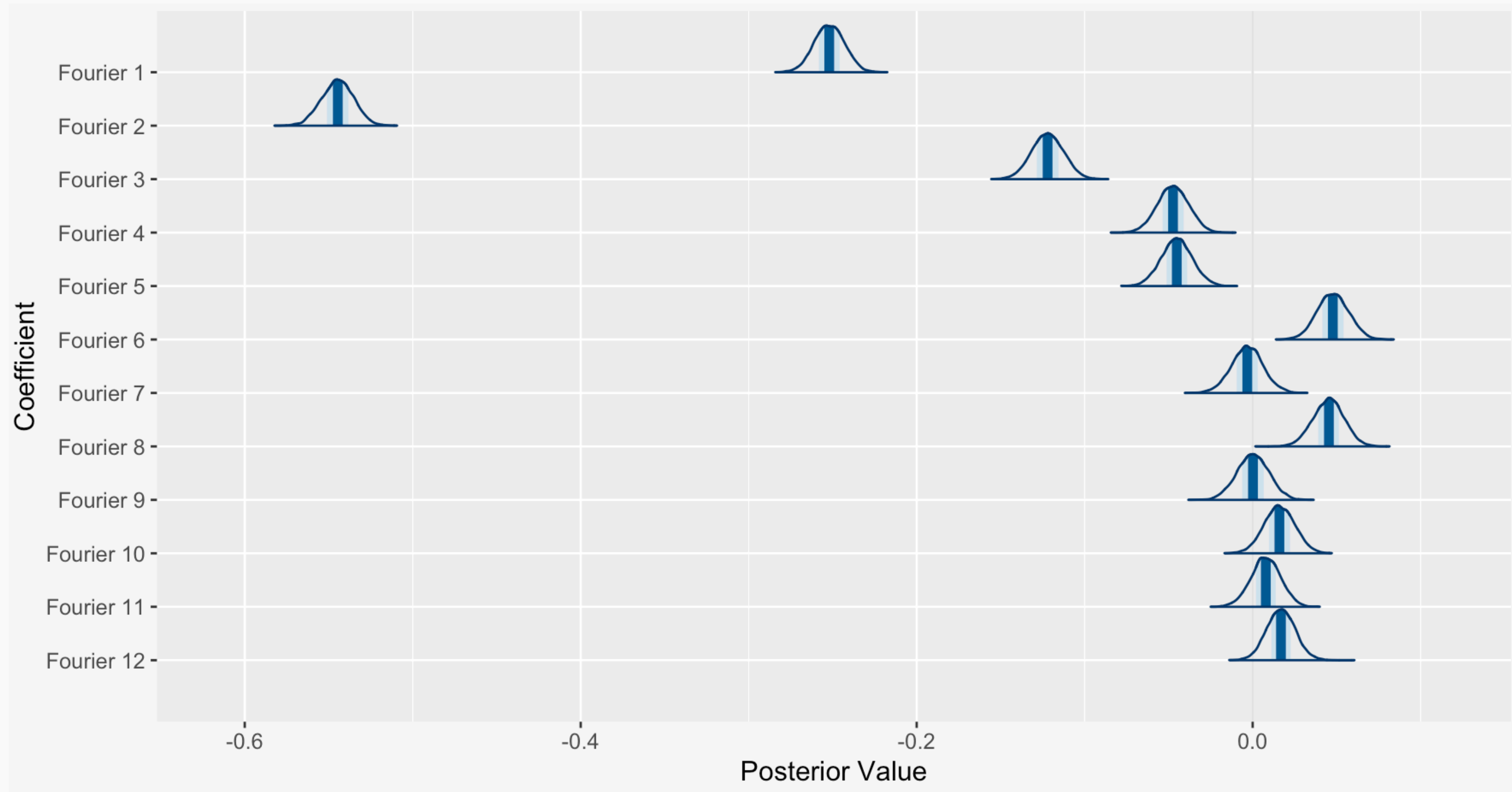


Daily temperature in New York City (available on [Kaggle](#)).

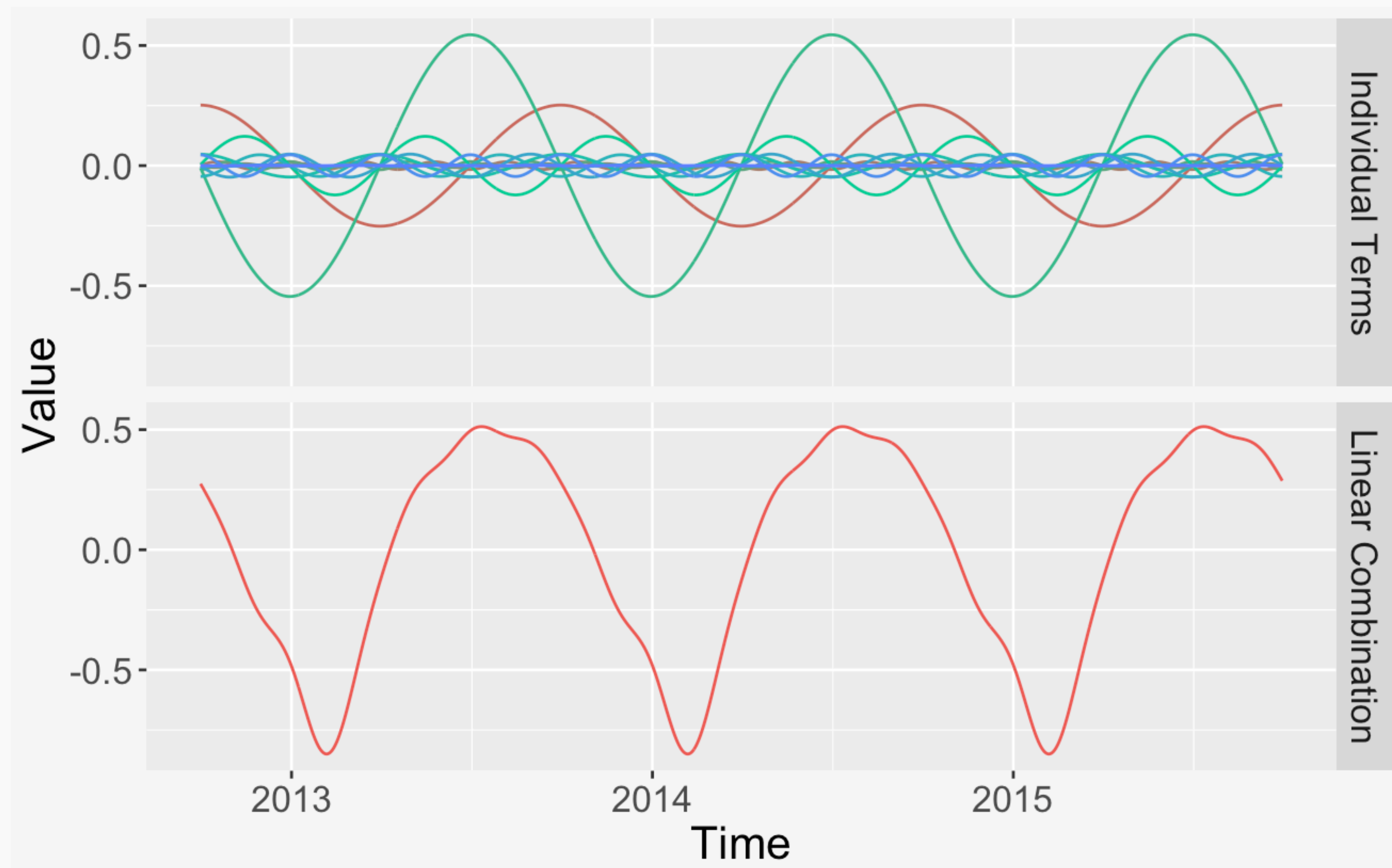
Transfer Posterior as Prior

Learn coefficients in model for which we have enough data.

$$P(\lambda \mid \tilde{T}) \propto P(\tilde{T} \mid \lambda) \cdot P(\lambda)$$



Posterior distributions of Fourier coefficients in temperature model.



Linear combination of Fourier terms weighted by the posterior mean coefficients.

Transfer Posterior as Prior

Learn coefficients in model for which we have enough data.

$$P(\lambda \mid \tilde{T}) \propto P(\tilde{T} \mid \lambda) \cdot P(\lambda)$$



$$P(\lambda \mid \tilde{Y}) \propto P(\tilde{Y} \mid \lambda) \cdot P(\lambda)$$

Transfer learned posterior as new prior distribution into original model where only few data is available.

Data Generating Process

$$\log(\mu_t) = \alpha_0 + \sum_{i=1}^6 \alpha_i D_i + \tau \frac{t}{m} + \sum_{k=1}^K \left(\beta_k \sin\left(\frac{2\pi kt}{m}\right) + \gamma_k \cos\left(\frac{2\pi kt}{m}\right) \right)$$

$$\beta_k \sim P(\beta_k | \tilde{T}) \quad \textbf{for } k = 1, \dots, K$$

$$\gamma_k \sim P(\gamma_k | \tilde{T}) \quad \textbf{for } k = 1, \dots, K$$



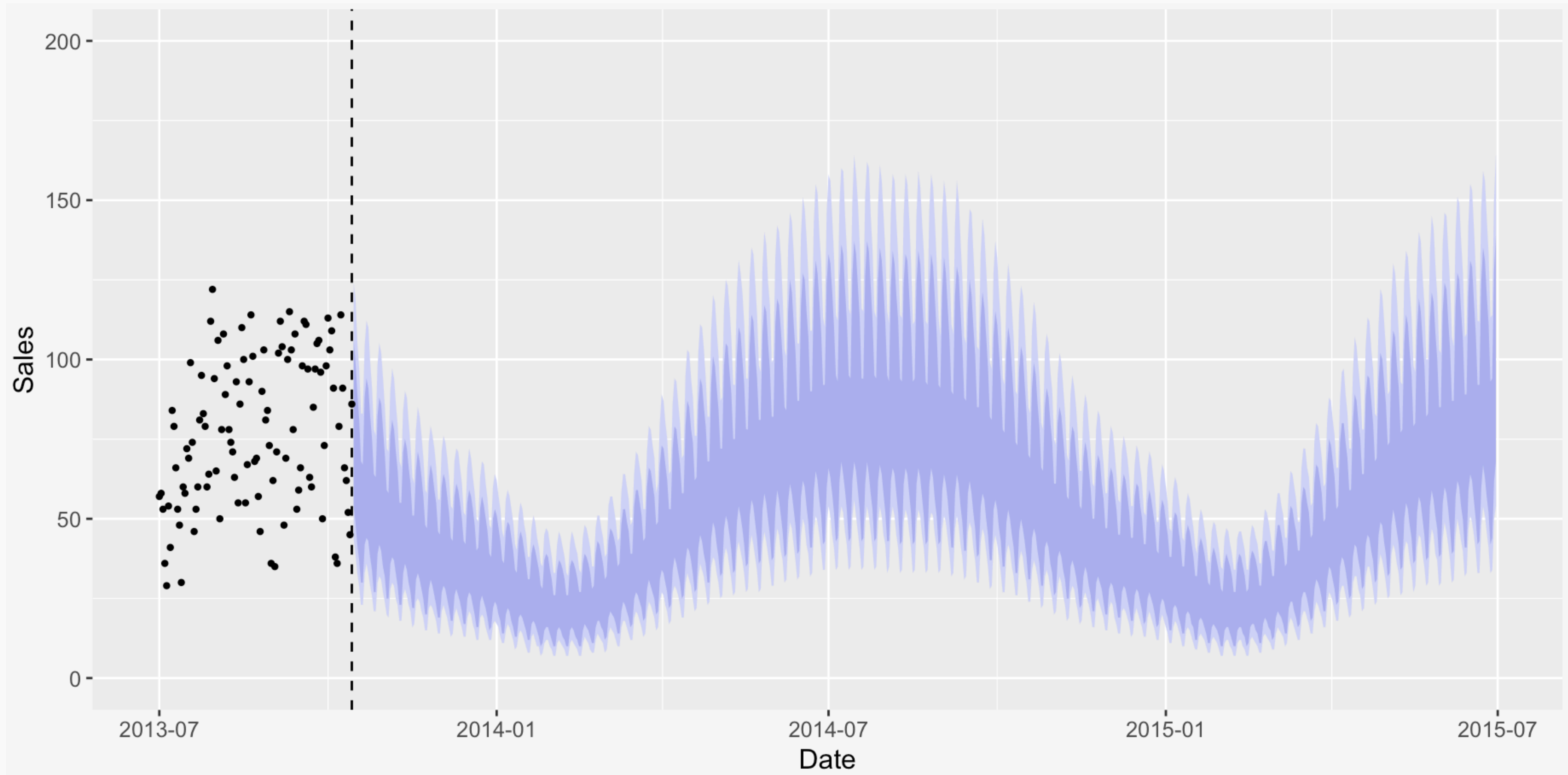
```
> beta_samples <- as.data.frame(temperature_fit)

> beta_means <- as.numeric(colMeans(beta_samples))

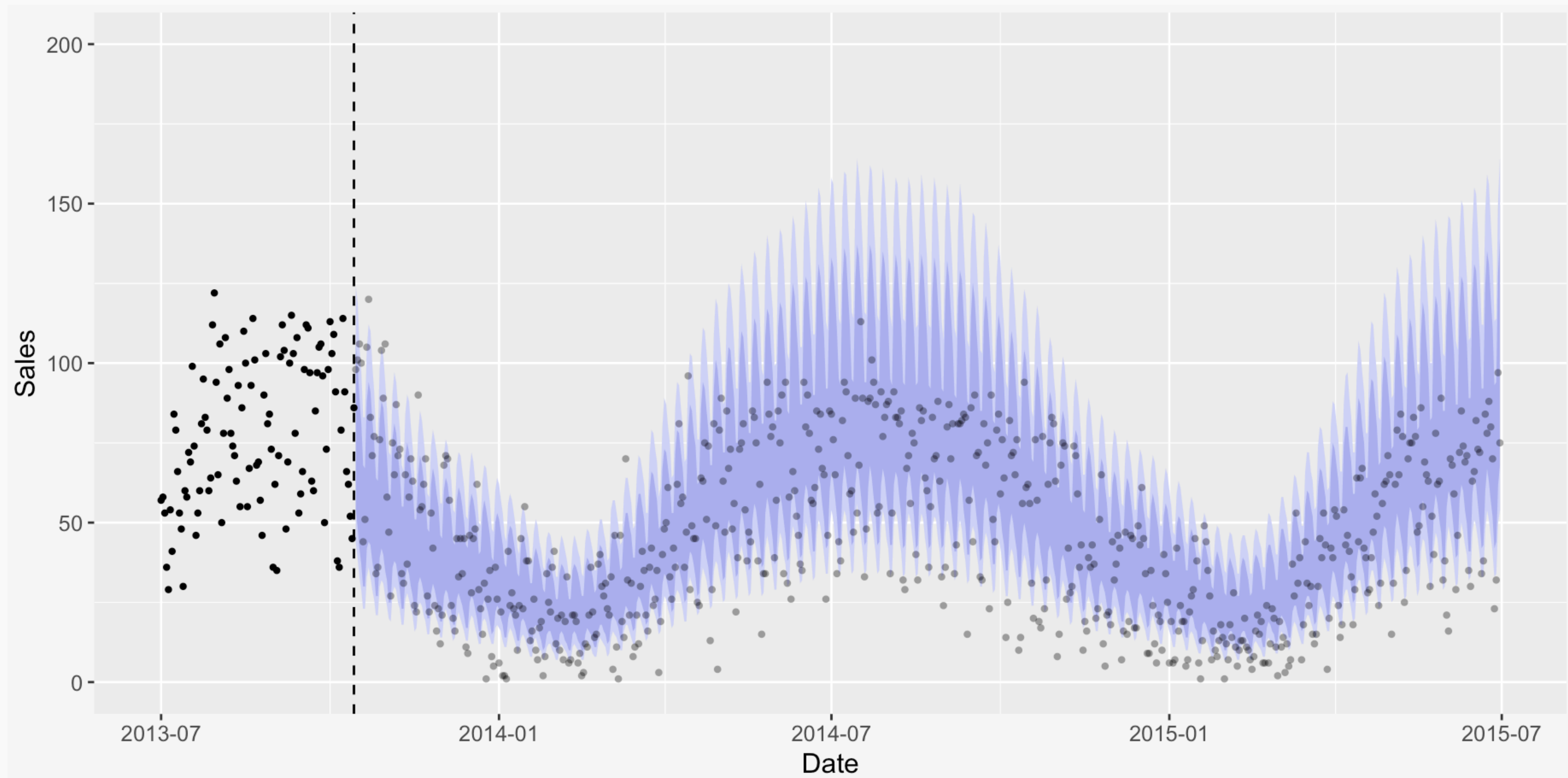
> beta_sds <- as.numeric(apply(beta_samples, 2, sd))
```

```
> sales_temp_fit <- sampling(negbin_prior_stan,  
+                             data = list(y = sales,  
+                                         x = xreg,  
+                                         fouriers = fouriers,  
+                                         trend = trend,  
+                                         trend_loc = 0.03,  
+                                         trend_sd = 0.02,  
+                                         fourier_loc = beta_means,  
+                                         fourier_sd = beta_sds,  
+                                         N = N, K = K,  
+                                         F = 12),  
+                             iter = 4000, algorithm = "NUTS", seed = 512)
```

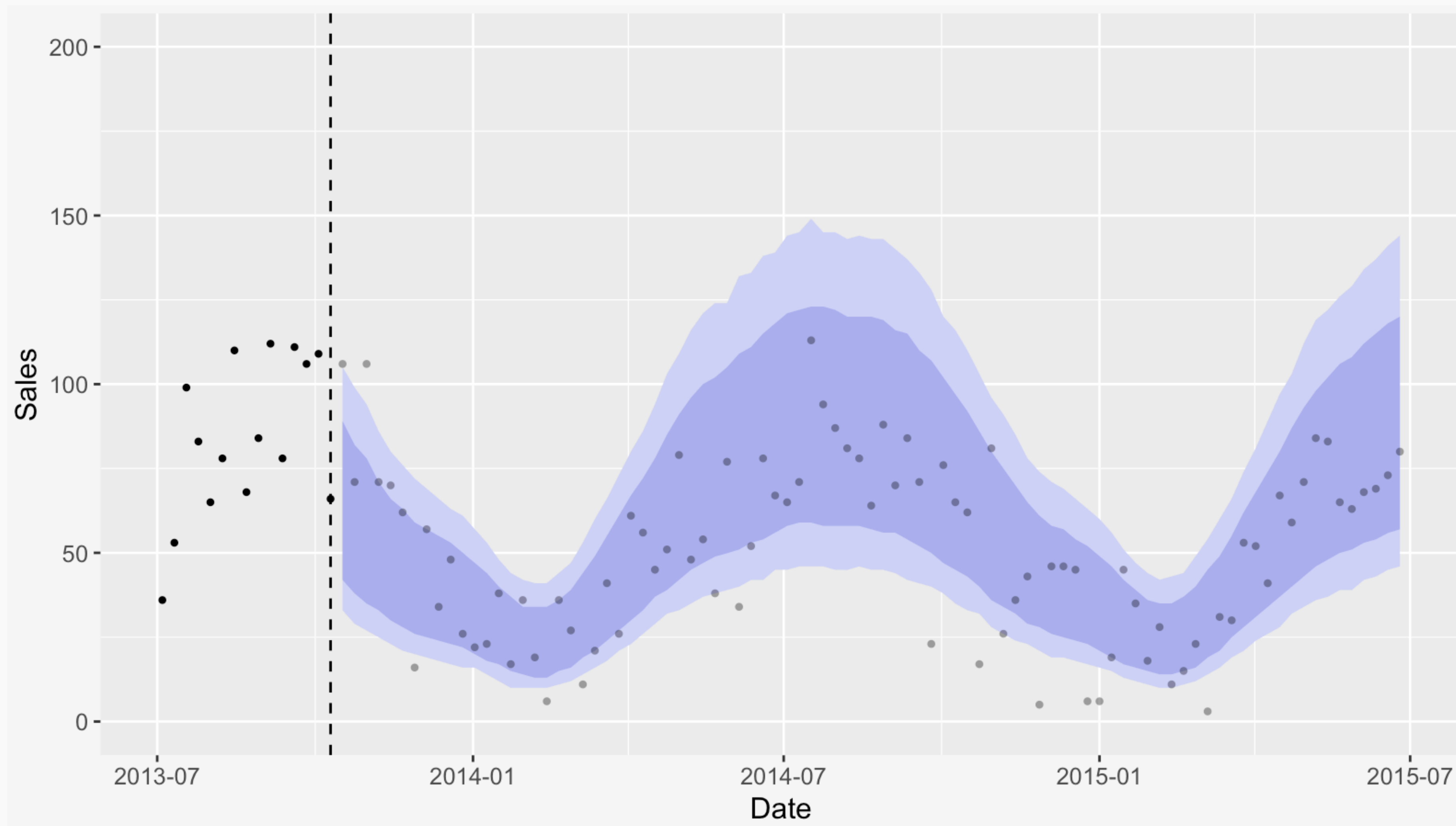




Posterior predictive distribution.



Posterior predictive distribution and actual future observations.

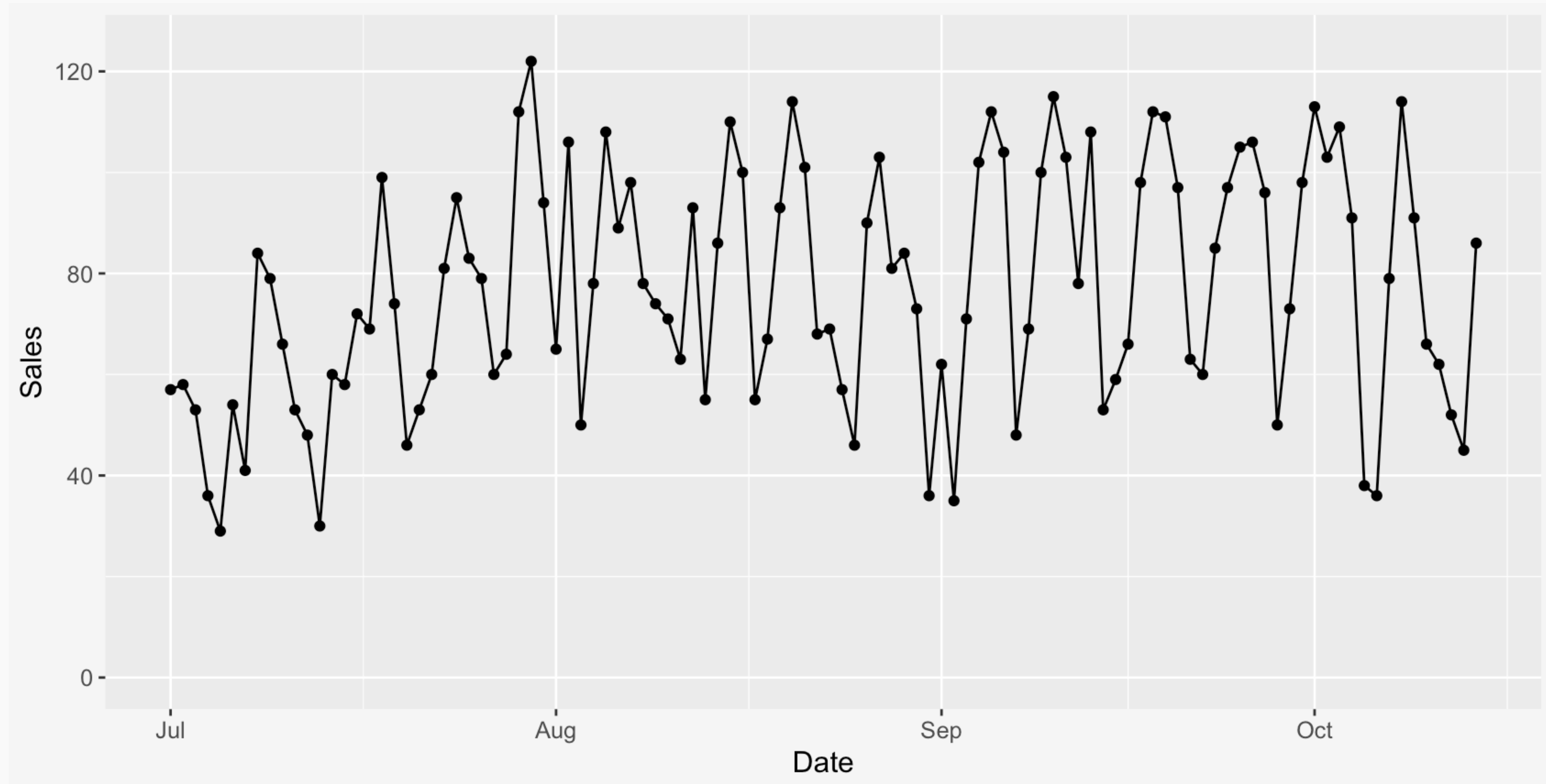


Posterior predictive distribution and future observations for Thursdays only.



github.com/timradtke/short-time-series

citibikenyc.com



s3.amazonaws.com/tripdata/index.html

minimizeregret.com



Thank you!

tim@timradtke.net

minimizeregret.com

[@timradtke](#)