shorts: An R Package for Modeling Short Sprints

Mladen Jovanović[[1]](#footnote-20)

Jason D. Vescovi[[2]](#footnote-22)

2021-11-20

Short sprint performance is one of the most distinguishable and admired physical traits in sports. Short sprints have been modeled using the mono-exponential equation that involves two parameters: (1) maximum sprinting speed (MSS) and (2) relative acceleration (TAU). The most common methods to assess short sprint performance are with a radar gun or timing gates. In this paper, we: 1) provide the **shorts** package that can model sprint timing data from these two sources; 2) discuss potential issues with assessing sprint time (synchronization and flying start, respectively); and 3) provide model definitions within the **shorts** package to help alleviate errors within the subsequent parameter outcomes.

# Introduction

Short sprint performance is one of the most distinguishable and admired physical traits in sports. Short sprints, commonly performed in most team sports (e.g., soccer, field hockey, handball, football, etc.), are defined as maximal running from a stand still position over a distance that doesn’t result in deceleration at the end. Peak anaerobic power is achieved within the first few seconds (<5 s) of maximal efforts ([Mangine et al. 2014](#ref-mangineSpeedForcePower2014)), whereas the ability to achieve maximal sprint speed varies based on the athlete and their sport. For example, track and field sprinters are trained to achieve maximal speed later in a race (i.e., 50-60 m) ([Ward-Smith 2001](#X28e7fd1011561c0e7fc0149d47b31e35b209202)), but team sport athletes have sport-specific attributes and reach it much sooner (i.e., 30-40 m)([Brown, Vescovi, and Vanheest 2004](#ref-brownAssessmentLinearSprinting2004)). Regardless of the differences in kinematics between athletes, evaluating short sprint performance is routinely included within a battery of fitness tests for a wide range of sports

The use of force plates is considered the gold standard for assessing mechanical properties of sprinting; however, there are logistical and financial challenges to capturing the profile of an entire sprint ([Jean-Benoit Morin et al. 2019](#ref-morinSimpleMethodComputing2019); [Samozino et al. 2016](#ref-samozinoSimpleMethodMeasuring2016)). Radar and laser technology are frequently used laboratory-grade methods ([Buchheit et al. 2014](#X96d817de5db35d8b2a12971d06d48ef5b28c985); [Edwards et al. 2020](#X3fb8cfdfda78a3ccd2e624ec1d800b92a0972bc); [Jiménez-Reyes et al. 2018](#X58c269338f0a45969118b82f26d4598952ff480); [Marcote-Pequeño et al. 2019](#Xf9750e022297e1bf2109ff15a97f4ef96b0601c)) but not normally accessible to practitioners working in sports. Undoubtedly, the most common method available and used to evaluate sprint performance are timing gates. Often multiple gates are positioned at varying distances to capture split times (e.g., 5, 10, 20 m), which can now be incorporated into the method for determining sprint mechanical properties ([Jean-Benoit Morin et al. 2019](#ref-morinSimpleMethodComputing2019); [Samozino et al. 2016](#ref-samozinoSimpleMethodMeasuring2016)). This approach presents an advantage to practitioners who can use the outcomes to describe individual differences, quantify the effects of training interventions, and better understanding the limiting factors of performance.

The **shorts** package ([Jovanović 2021](#ref-R-shorts)), written in the R language ([R Core Team 2021](#ref-R-base)), represents an open-source tool to help sports scientists translate raw timing data into detailed mechanical outcomes through mathematical modeling ([Jean-Benoit Morin et al. 2019](#ref-morinSimpleMethodComputing2019); [Samozino et al. 2016](#ref-samozinoSimpleMethodMeasuring2016)). To best of our knowledge, scientist, researchers, and coaches have been performing short sprints modeling using the built-in solver function of Excel (Microsoft Corporation, Redmond, Washington, United States) ([J. B. Morin 2017](#X0c4cbe9e8656dd48ad922f5803c496b199df4e9); [Jean-Benoit Morin and Samozino 2019](#X41618b79c21e72ab2531fa2b596b4d4af58820d); [Stenroth, Vartiainen, and Karjalainen 2020](#Xf1011dded519ee0de75fb97a2b8a911dfc18ddc); [Stenroth and Vartiainen 2020](#X636a68a11da56e24e74db551b02c2420e7d92cd); [Samozino et al. 2016](#ref-samozinoSimpleMethodMeasuring2016); [Clark et al. 2017](#ref-clarkNFLCombine40Yard2017); [Jean-Benoit Morin et al. 2019](#ref-morinSimpleMethodComputing2019)), which makes the **shorts** package a major improvement in ease-of-use, speed, transparency, reproducibility, and more feature-rich model fitting.

In the current paper, we will provide an explanation of one commonly used mathematical equation to model short sprints, modeling applications using the **shorts** package, issues that can arise during measurement and estimation, and potential solutions to those problems.

# Mathematical model

Short sprints have been modeled using the mono-exponential equation (1) originally proposed by [Furusawa, Hill, and Parkinson](#ref-doi:10.1098/rspb.1927.0035) ([1927](#ref-doi:10.1098/rspb.1927.0035)), and more recently popularized by [Clark et al.](#ref-clarkNFLCombine40Yard2017) ([2017](#ref-clarkNFLCombine40Yard2017)), and [Samozino et al.](#ref-samozinoSimpleMethodMeasuring2016) ([2016](#ref-samozinoSimpleMethodMeasuring2016)). Equation (1) represents the function for instantaneous horizontal velocity given the time and two model parameters:

The parameters of the equation (1) are *maximum sprinting speed* (MSS; expressed in ) and *relative acceleration* (TAU). Mathematically, TAU represents the ratio of MSS to initial acceleration (MAC; *maximal acceleration*, expressed in ) (2).

Although TAU is used in the equations, and later estimated, we prefer to use MAC instead since it is easier to grasp, particularly for less math inclined coaches.

By derivating equation (1), we can get the equation for horizontal acceleration (3).

By integrating equation (1), we can get the equation for distance covered (4).

Let’s consider four athletes with different levels of MSS (high versus low maximal sprinting speed) and MAC (high versus low maximal acceleration; as mentioned previously, using MAC is preferred over using TAU) (Table 1).

Table 1: Four athletes with different MSS and MAC parameters.

| Athlete | MSS | MAC | TAU |
| --- | --- | --- | --- |
| Athlete A | 12 | 10 | 1.20 |
| Athlete B | 12 | 6 | 2.00 |
| Athlete C | 8 | 10 | 0.80 |
| Athlete D | 8 | 6 | 1.33 |

Figure 1 depicts distance, velocity, and acceleration over time (from 0 to 6 s).

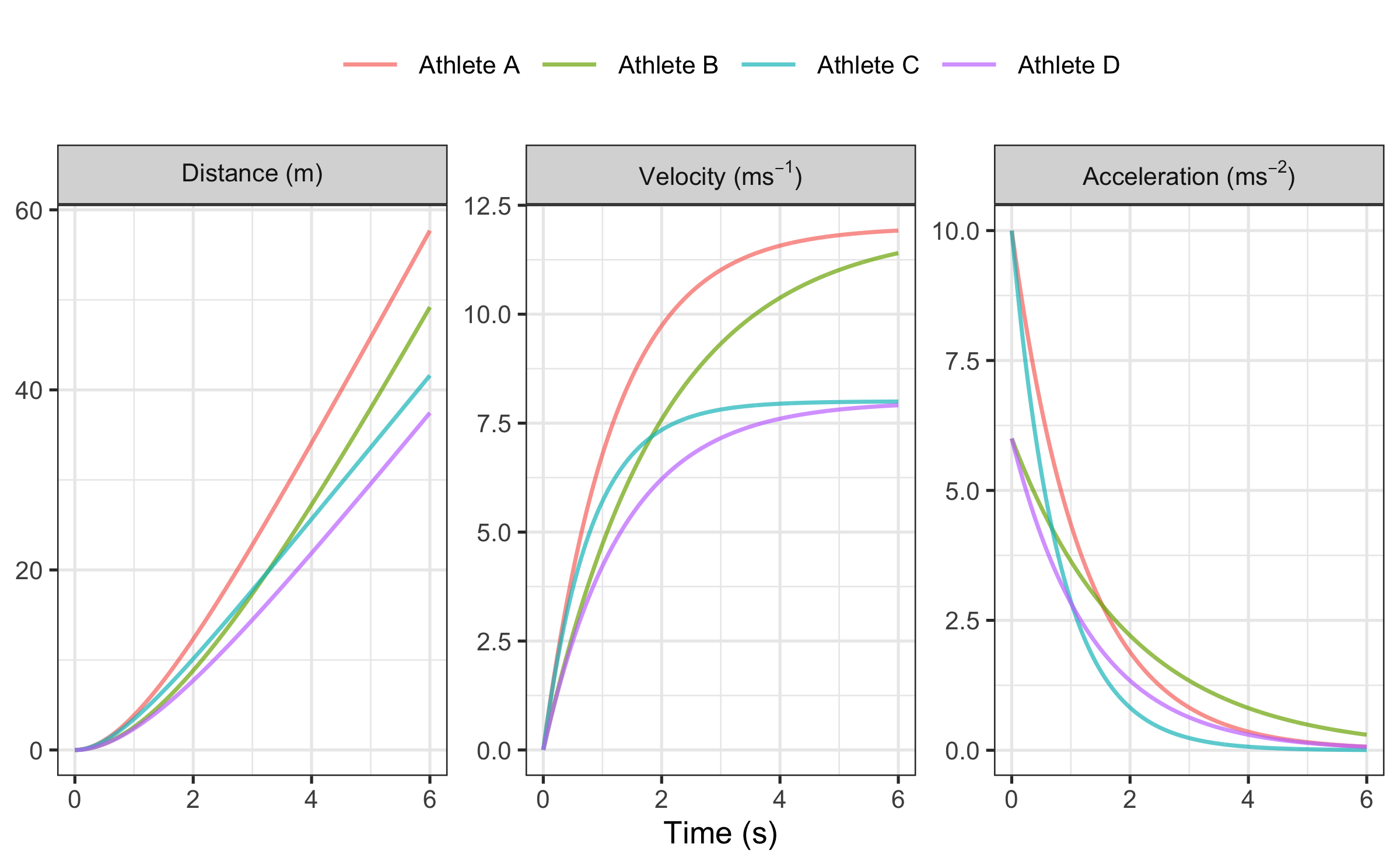


Figure 1: Kinematic characteristic of four athletes with different MSS and MAC parameters over a period of 0 to 6 seconds.

Plotting acceleration against velocity (Figure 2), we will get *Acceleration-Velocity Profile*, which is linear, according to the mathematical model. If the athlete’s body mass (kg) is known, as well as additional air resistance parameters (see [Air resistance and the calculation of force and mechanical power](#X4d450aa5f3b7623b53d936d668d28a709b32989) section of this paper), *Force-Velocity Profile* can be estimated (see [Force-Velocity profile](#force-velocity-profile) section of this paper).

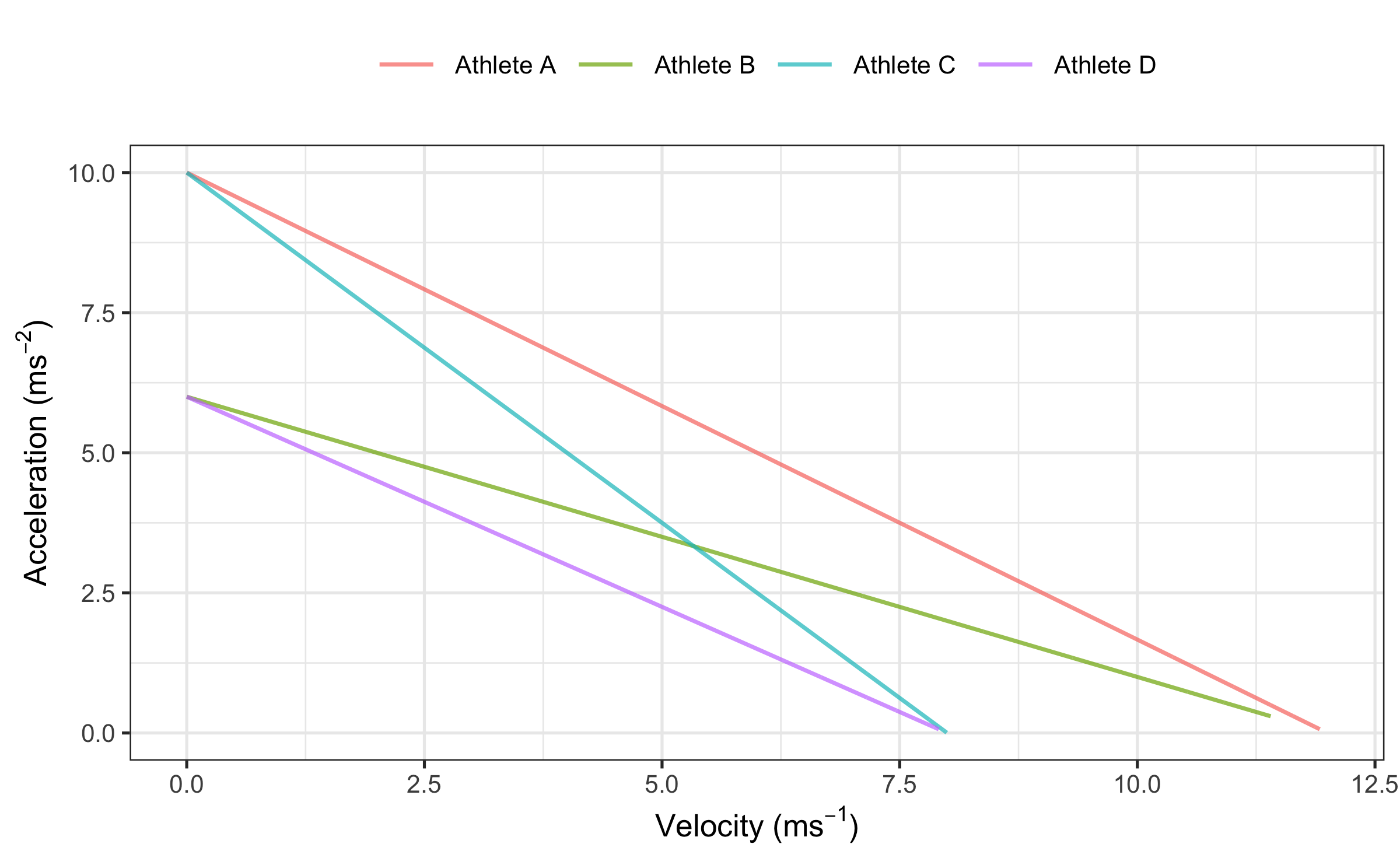


Figure 2: Acceleration-Velocity profile of four athletes with different MSS and MAC parameters.

# Estimation using shorts package

Short sprints profiling is usually performed by: (1) measuring split times using timing gates (i.e., positioned at various distances, e.g., 5, 10, 20, 30, 40 m), or (2) getting a velocity trace using a radar gun. Estimation of MSS and TAU parameters from equation (1) is performed in **shorts** package using non-linear least squares regression implemented in the nls() function ([D. M. Bates and Chambers 1992](#ref-batesNonlinearModels1992); [Douglas M. Bates and Watts 2007](#ref-batesNonlinearRegressionAnalysis2007)) in the **base R** ([R Core Team 2021](#ref-R-base)) and nlme() function in the **nlme** package ([Pinheiro, Bates, and R-core 2021](#ref-R-nlme)) for the mixed-effect models.

## Estimating short sprint parameters using timing gates split times

Let’s consider an example of an athlete with MSS equal to 9 , TAU equal to 1.3, and MAC equal to 6.92 performing 40m sprint with timing gates positioned at each 10m split. For split times, distance is a predictor, and time is the outcome variable, thus the equation (4) becomes:

in equation (5) represents Lambert’s W function ([Goerg 2020](#ref-R-LambertW)). Researchers often incorrectly use the equation (4) ([J. B. Morin 2017](#X0c4cbe9e8656dd48ad922f5803c496b199df4e9); [Jean-Benoit Morin and Samozino 2019](#X41618b79c21e72ab2531fa2b596b4d4af58820d); [Stenroth and Vartiainen 2020](#X636a68a11da56e24e74db551b02c2420e7d92cd)), in which the time is the predictor and distance is the outcome variable, instead of statistically correct equation (5). This practice should be avoided because switching of the predictor and outcome variables in the regression model might produce biased estimated parameters ([Motulsky 2018, 341](#X1975a95524d680e840cacf8c10f7854d2bf6a97)). This might not necessarily be the case with the short sprints profiling, but it is nevertheless a bad statistical practice.

MSS and TAU parameters are estimated using model\_using\_splits() function:

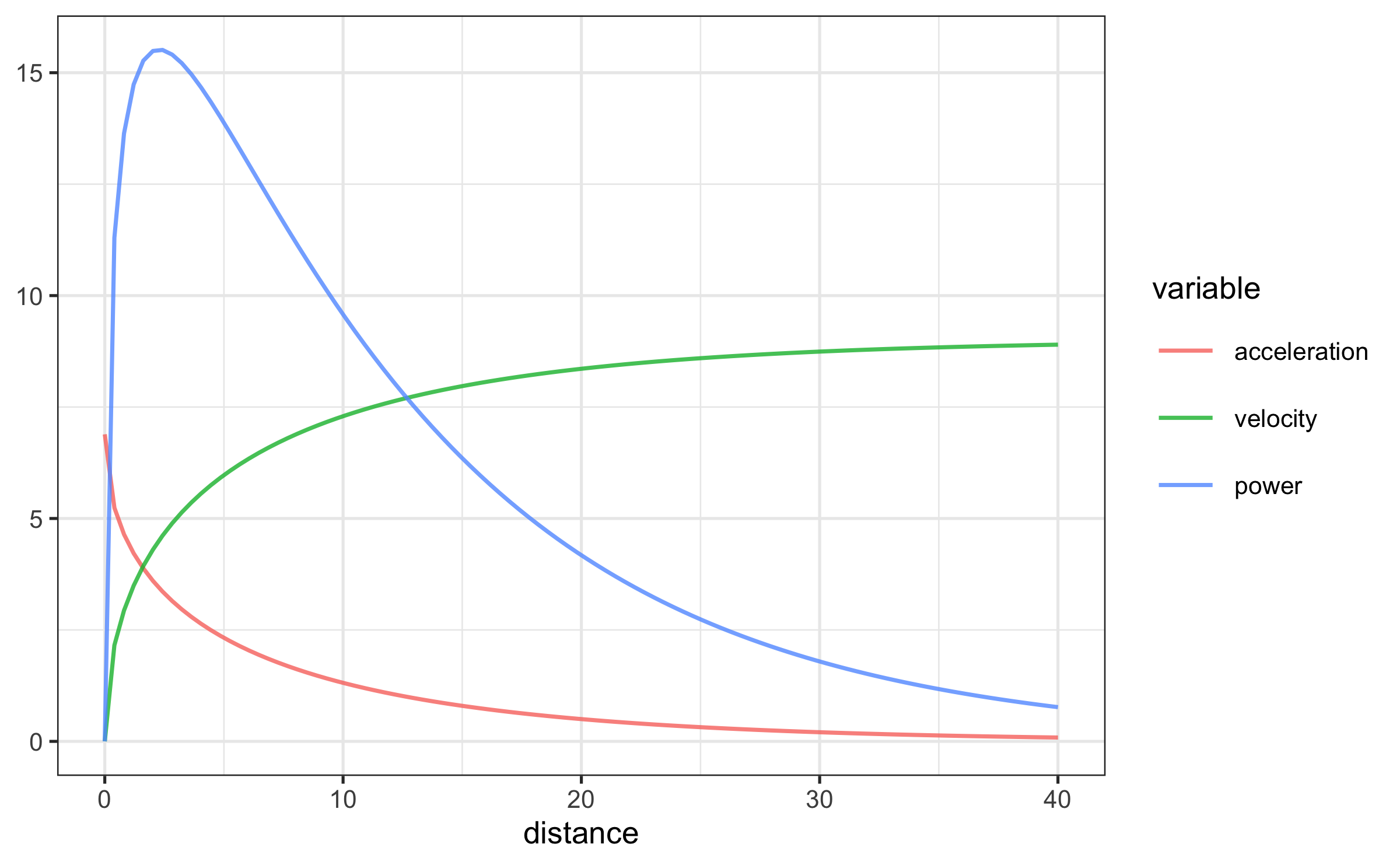
require(shorts)  
  
split\_distance <- c(10, 20, 30, 40)  
  
split\_time <- c(2.17, 3.43, 4.60, 5.73)  
  
m1 <- model\_using\_splits(  
 distance = split\_distance,  
 time = split\_time  
)  
  
m1  
#> Estimated model parameters  
#> --------------------------  
#> MSS TAU MAC   
#> 9.01 1.31 6.89   
#> PMAX time\_correction distance\_correction   
#> 15.52 0.00 0.00   
#>   
#> Model fit estimators  
#> --------------------  
#> RSE R\_squared minErr maxErr maxAbsErr RMSE   
#> 0.00249 1.00000 -0.00265 0.00178 0.00265 0.00176   
#> MAE MAPE   
#> 0.00157 0.04742

Maximal relative power (PMAX) from the output is estimated using , which disregards the air resistance. time\_correction and distance\_corection parameters will be covered later in the paper.

Besides providing *residual standard error* (RSE), **shorts** functions provide additional model fit estimators. Additional information can be gained by exploring the returned object, particularly object returned from the nls() function (i.e., by using the S3 summary() method). To extract estimated model parameters, use S3 coef() method.

To create a simple plot of the model, use S3 plot() method, which returns **ggplot2** ([Wickham, Chang, et al. 2021](#ref-R-ggplot2)) object:

plot(m1) + theme\_bw(8)



Once we have estimated MSS and TAU, we can use predict\_XXX() family of functions to predict various relationships (i.e., time at distance, acceleration at distance, velocity at time, etc.):

# Predict time at distance  
predict\_time\_at\_distance(  
 distance = split\_distance,  
 MSS = m1$parameters$MSS,  
 TAU = m1$parameters$TAU  
)  
#> [1] 2.17 3.43 4.60 5.73

### Air resistance and the calculation of force and mechanical power

To estimate force production at distance or time (using predict\_force\_at\_distance() and predict\_force\_at\_time() functions), as well as power production (using predict\_power\_at\_distance() and predict\_power\_at\_time() functions), one needs to take into account the air resistance. Air resistance (N) is estimated using get\_air\_resistance() function, which takes velocity, body mass (kg), body height (m), barometric pressure (Torr), air temperature (), and wind velocity () as parameters (please refer to [Arsac and Locatelli](#ref-arsacModelingEnergetics100m2002) ([2002](#ref-arsacModelingEnergetics100m2002)), [Samozino et al.](#ref-samozinoSimpleMethodMeasuring2016) ([2016](#ref-samozinoSimpleMethodMeasuring2016)), and [van Ingen Schenau, Jacobs, and de Koning](#ref-vaningenschenauCanCyclePower1991) ([1991](#ref-vaningenschenauCanCyclePower1991)) for more information):

get\_air\_resistance(  
 velocity = 5,  
 bodymass = 80,  
 bodyheight = 1.85,  
 barometric\_pressure = 780,  
 air\_temperature = 20,  
 wind\_velocity = 0.5  
)  
#> [1] 6.1

When estimating force and power, the air resistance parameters can be set using "...", which are forwarded to the get\_air\_resistance():

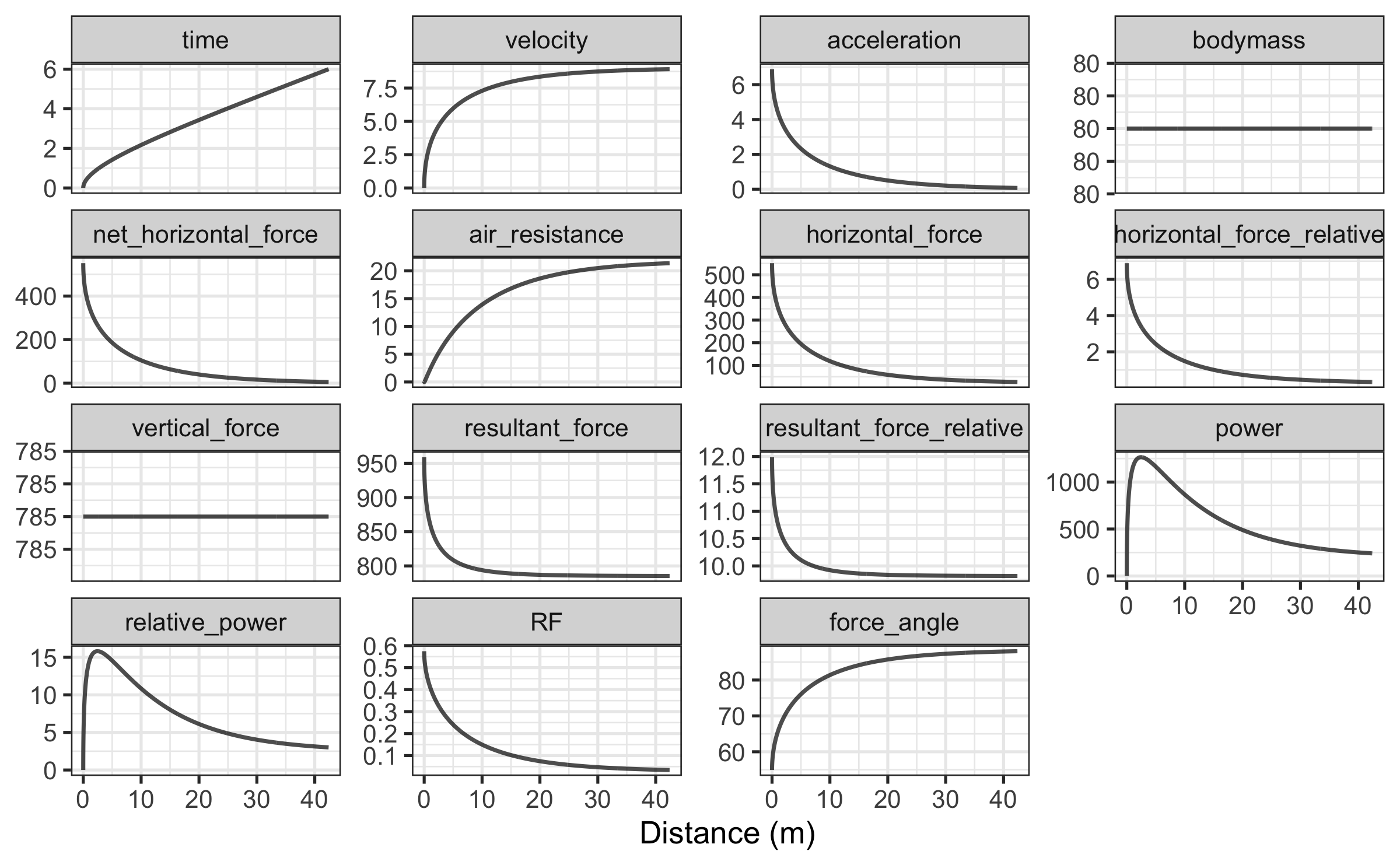
# To calculate horizontal force produced  
predict\_force\_at\_distance(  
 distance = split\_distance,  
 MSS = m1$parameters$MSS,  
 TAU = m1$parameters$TAU,  
 # Additional parameters forwarded to get\_air\_resistance  
 # Otherwise, defaults are used  
 bodymass = 80,  
 bodyheight = 1.85,  
 barometric\_pressure = 780,  
 air\_temperature = 20,  
 wind\_velocity = 0.5  
)  
#> [1] 119.0 58.6 36.9 28.2

The easiest way to get all kinematics and kinetics for short sprints is to use predict\_kinematics() function:

df <- predict\_kinematics(  
 m1,  
 max\_time = 6,  
 frequency = 100,  
 # Additional parameters forwarded to get\_air\_resistance  
 # Otherwise, defaults are used  
 bodymass = 80,  
 bodyheight = 1.85,  
 barometric\_pressure = 780,  
 air\_temperature = 20,  
 wind\_velocity = 0.5  
)

Plotting the model predictions can be done once we convert data from wide to long with the help of **ggplot2** ([Wickham, Chang, et al. 2021](#ref-R-ggplot2)), **dplyr** ([Wickham, François, et al. 2021](#ref-R-dplyr)), **tidyr** ([Wickham 2021a](#ref-R-tidyr)), and **tidyverse** ([Wickham 2021b](#ref-R-tidyverse)) packages:

require(tidyverse)  
  
variable\_names <- colnames(df)  
  
df <- pivot\_longer(data = df, cols = -2) %>%  
 mutate(name = factor(name, levels = variable\_names))  
  
ggplot(df, aes(x = distance, y = value)) +  
 theme\_bw(8) +  
 facet\_wrap(~name, scales = "free\_y") +  
 geom\_line(alpha = 0.7) +  
 ylab(NULL) +  
 xlab("Distance (m)")



These kinematic and kinetic variables are utilized in [Force-Velocity profile](#force-velocity-profile) estimation, which is covered later in this paper.

### Utility functions

Another valuable addition for sport scientists and coaches is the ability to determine the distances and times where 90% of maximum sprinting speed is reached, or where peak power is within 90% range. To identify these values, **shorts** package comes with find\_XXX() family of functions:

# Finds distance where 90% of maximum sprinting speed is reached  
find\_velocity\_critical\_distance(  
 MSS = m1$parameters$MSS,  
 TAU = m1$parameters$TAU,  
 percent = 0.9  
)  
#> [1] 16.5  
  
# Finds maximal power and distance (this time using air resistance)  
find\_max\_power\_distance(  
 MSS = m1$parameters$MSS,  
 TAU = m1$parameters$TAU,  
 # Additional parameters forwarded to get\_air\_resistance  
 # Otherwise, defaults are used  
 bodymass = 80,  
 bodyheight = 1.85,  
 barometric\_pressure = 780,  
 air\_temperature = 20,  
 wind\_velocity = 0.5  
)  
#> $max\_power  
#> [1] 1264  
#>   
#> $distance  
#> [1] 2.46  
  
# Finds distance over 90% power range  
find\_power\_critical\_distance(  
 MSS = m1$parameters$MSS,  
 TAU = m1$parameters$TAU,  
 # Additional parameters forwarded to get\_air\_resistance  
 # Otherwise, defaults are used  
 bodymass = 80,  
 bodyheight = 1.85,  
 barometric\_pressure = 780,  
 air\_temperature = 20,  
 wind\_velocity = 0.5  
)  
#> $lower  
#> [1] 0.959  
#>   
#> $upper  
#> [1] 5.44

### Mixed-effects model

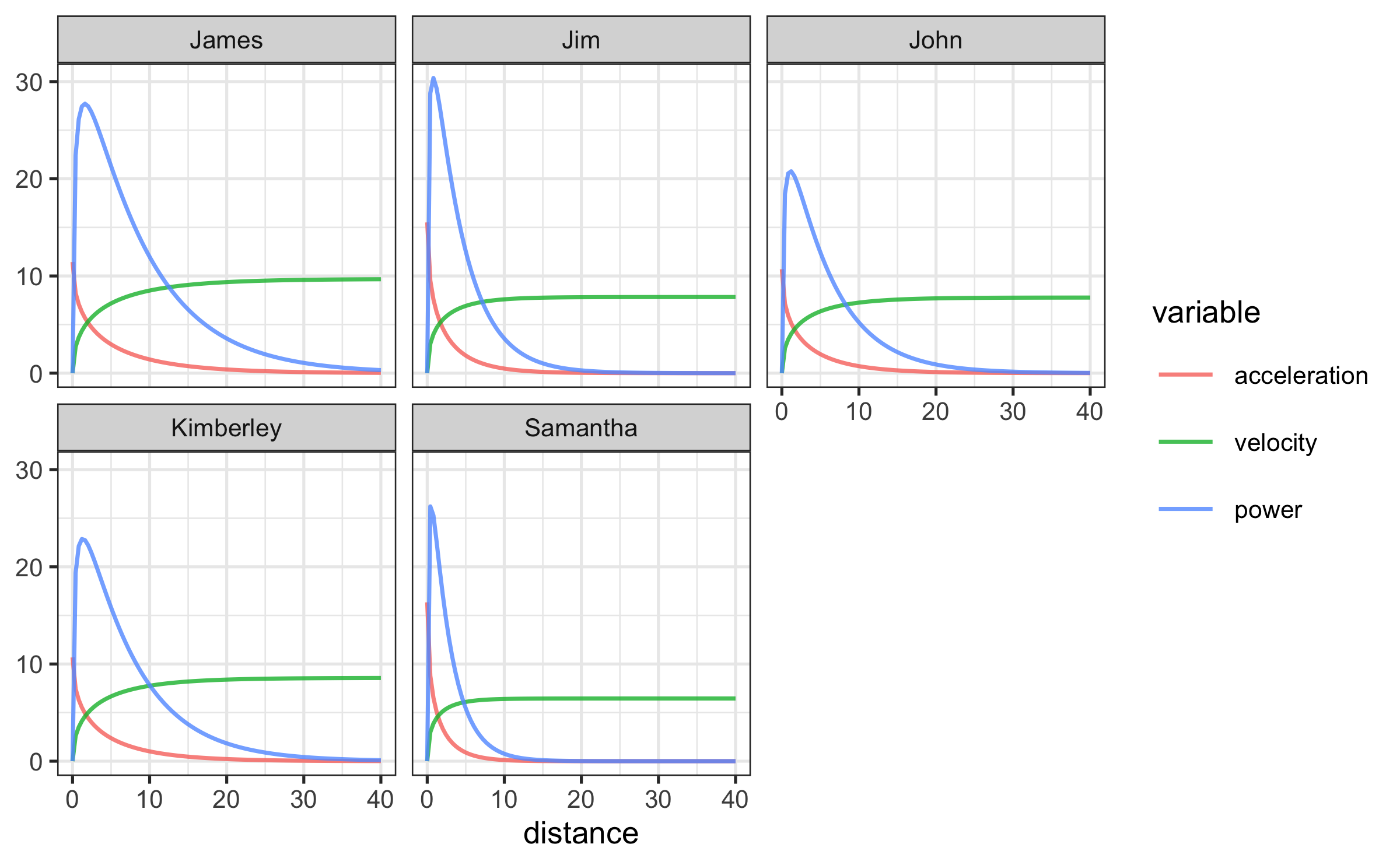
Sprint performance is often evaluated with a group of athletes (e.g., soccer club) representing a single strata of interest. Sports scientists can estimate individual profiles, or utilize mixed-effects models. To perform mixed-effects models in **shorts** for split times, one can use mixed\_model\_using\_splits() function. To demonstrate this functionality, we load the split\_times dataset provided in the **shorts** package:

data(split\_times)  
  
# Mixed model  
m2 <- mixed\_model\_using\_splits(  
 data = split\_times,  
 distance = "distance",  
 time = "time",  
 athlete = "athlete",  
  
 # Select random effects  
 # Default is MSS and TAU  
 random = MSS + TAU ~ 1  
)  
  
m2  
#> Estimated fixed model parameters  
#> --------------------------------  
#> MSS TAU MAC   
#> 8.065 0.655 12.309   
#> PMAX time\_correction distance\_correction   
#> 24.818 0.000 0.000   
#>   
#> Estimated random model parameters  
#> ---------------------------------  
#> athlete MSS TAU MAC PMAX time\_correction  
#> 1 James 9.69 0.847 11.4 27.7 0  
#> 2 Jim 7.83 0.505 15.5 30.4 0  
#> 3 John 7.78 0.727 10.7 20.8 0  
#> 4 Kimberley 8.57 0.802 10.7 22.9 0  
#> 5 Samantha 6.45 0.395 16.3 26.4 0  
#> distance\_correction  
#> 1 0  
#> 2 0  
#> 3 0  
#> 4 0  
#> 5 0  
#>   
#> Model fit estimators  
#> --------------------  
#> RSE R\_squared minErr maxErr maxAbsErr RMSE   
#> 0.0260 0.9998 -0.0496 0.0293 0.0496 0.0214   
#> MAE MAPE   
#> 0.0172 0.9019

Additional information about mixed-effects model performed using the nlme package ([Pinheiro, Bates, and R-core 2021](#ref-R-nlme)) can be obtained using summary() function. S3 method coef() when applied on mixed-model result will return both the fixed and random effects.

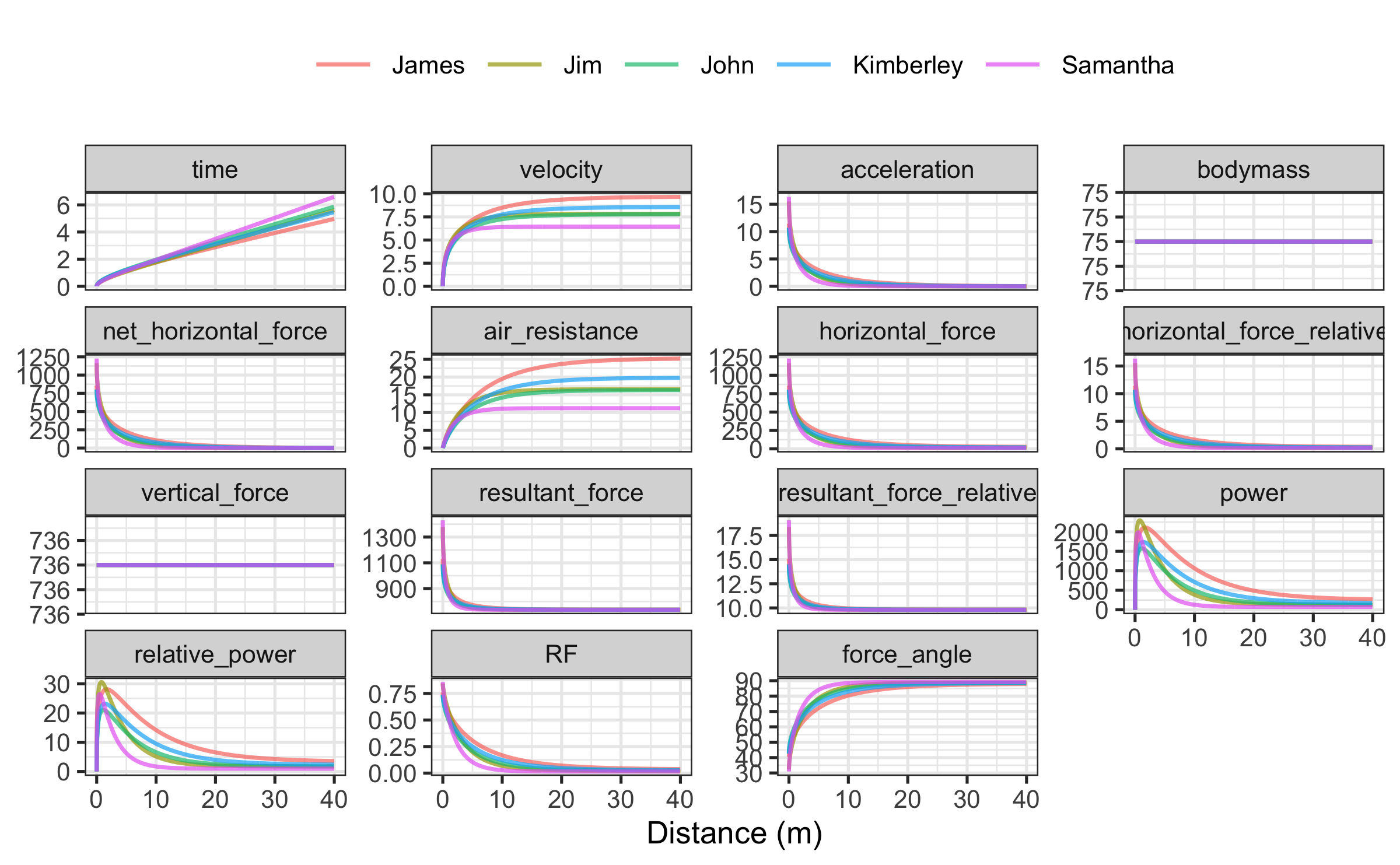
To create a simple plot of the model, use S3 plot() method.

plot(m2) + theme\_bw(8)



The following figure contains kinematics for all athletes in split\_times dataset. Please note that power calculation takes default parameters for each individual:

df <- predict\_kinematics(m2, max\_time = 10)  
  
variable\_names <- colnames(df)  
  
df <- pivot\_longer(df, cols = c(-1, -3)) %>%  
 mutate(name = factor(name, levels = variable\_names))  
  
ggplot(  
 filter(df, distance < 40),  
 aes(x = distance, y = value, group = athlete, color = athlete)  
) +  
 theme\_bw(8) +  
 facet\_wrap(~name, scales = "free\_y") +  
 geom\_line(alpha = 0.7) +  
 ylab(NULL) +  
 xlab("Distance (m)") +  
 theme(  
 legend.position = "top",  
 legend.title = element\_blank()  
 )



## Estimating short sprint parameters using radar gun

Estimation of the short sprint profile using radar gun data takes time as predictor and velocity as the outcome variable. Thus equation (1) is used to estimate MSS and TAU.

Let’s consider the same example of an athlete with MSS equal to 9 , TAU equal to 1.3, and MAC equal to 6.92 performing 40m sprint with velocity estimated using radar run (in this case with 1 Hz sampling rate).

sprint\_time <- seq(0, 6, 1)  
  
sprint\_velocity <- c(0.00, 4.83, 7.07, 8.10, 8.59, 8.81, 8.91)  
  
m3 <- model\_using\_radar(  
 velocity = sprint\_velocity,  
 time = sprint\_time  
)

Both split and radar gun models allow the use of *weighted* non-linear regression. Weighting in regression is utilized when the observations have unequal error variance ([Gelman, Hill, and Vehtari 2020](#ref-gelmanRegressionOtherStories2020)), which can happen due to instrumental error (e.g., a multiplicative error instead of additive error) or due to biological variability (e.g., higher split time variance on shorter distances versus longer distances). In this case, observations with higher error variance get lower weight when fitting the model ([Gelman, Hill, and Vehtari 2020](#ref-gelmanRegressionOtherStories2020)). According to [Gelman, Hill, and Vehtari](#ref-gelmanRegressionOtherStories2020) ([2020](#ref-gelmanRegressionOtherStories2020)), unequal variances are not typically a major issue for the goal of estimating regression parameters, but they can become more important when making predictions about individual observations. Further exploring the topic of unequal error variances is beyond the scope of this paper, but the **shorts** package provides a weighting feature for the potential research of this topic in the future.

Weighted non-linear regression is performed by setting weights parameter. For example, we can give more weight to shorter distances or faster velocities:

m3\_weighted <- model\_using\_radar(  
 velocity = sprint\_velocity,  
 time = sprint\_time,  
 weights = 1 / (sprint\_velocity + 1)  
)

### Mixed-effects model

Mixed-effects model using radar data is done using mixed\_model\_using\_radar() function. To perform mixed model, let’s load data that comes with **shorts** package.

data("radar\_gun\_data")  
  
m4 <- mixed\_model\_using\_radar(  
 radar\_gun\_data,  
 time = "time",  
 velocity = "velocity",  
 athlete = "athlete"  
)

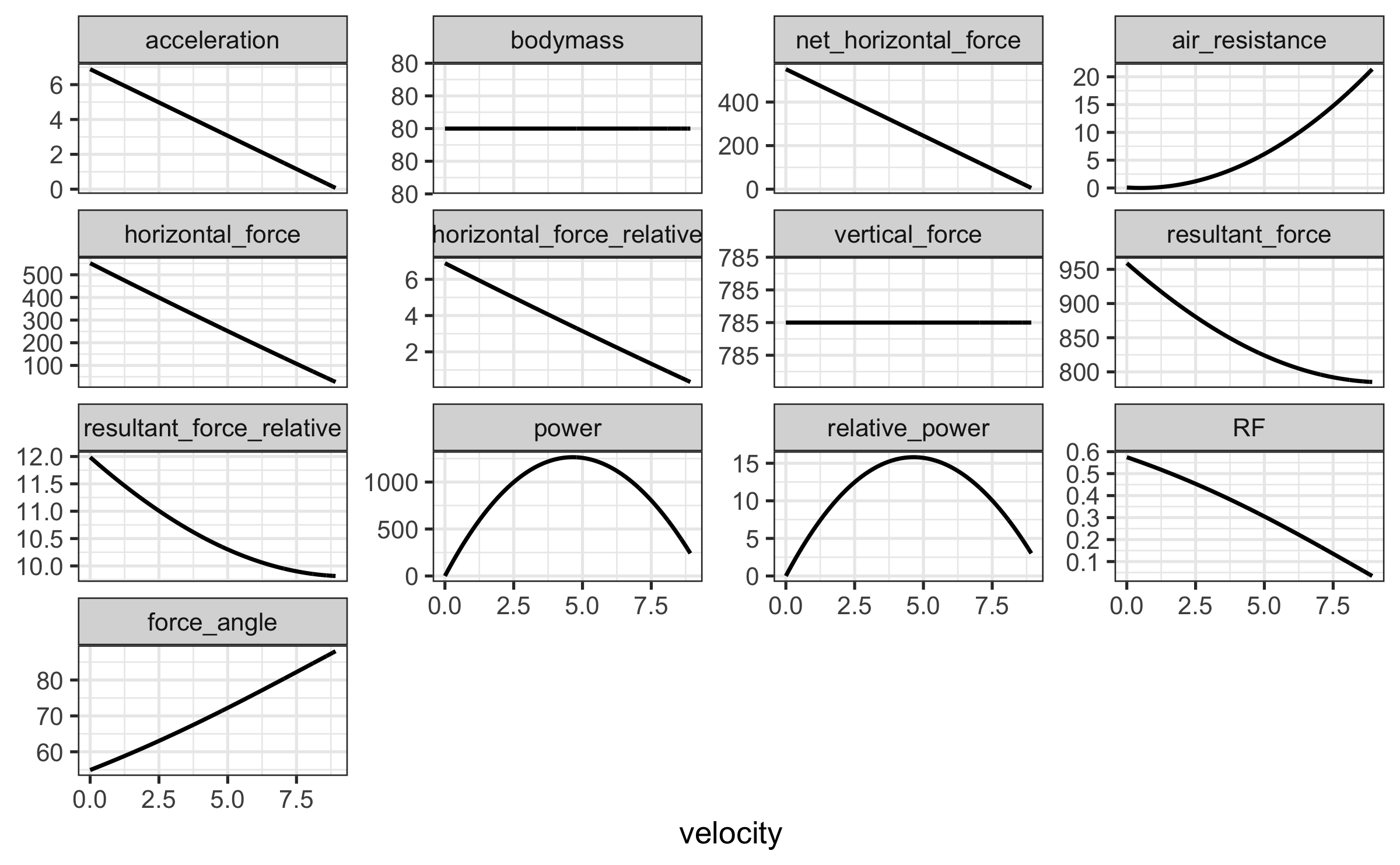
## Force-Velocity profile

To create *Force-Velocity Profile* (FVP) using single athlete estimated sprint model parameters (i.e., TAU and MSS), you can use get\_FV\_profile() function. When estimating FVP, athlete body mass (kg) can be set using bodymass parameter, while the air resistance parameters can be set using "...", which are forwarded to the get\_air\_resistance() function. Details of the FVP method implemented in the **shorts** package, as well as the interpretation from a sprint training perspective, are covered elsewhere ([Thomas A. Haugen, Breitschädel, and Samozino 2020](#X51c34a68a97c6a6d254e5eebdf120de021512e2); [Jean-Benoit Morin and Samozino 2016](#X932832648b20a15319bc6a2634444ef1f3a5625); [Jean-Benoit Morin et al. 2019](#ref-morinSimpleMethodComputing2019); [Samozino et al. 2016](#ref-samozinoSimpleMethodMeasuring2016)).

# To create Force-Velocity Profile  
fvp <- get\_FV\_profile(  
 MSS = m1$parameters$MSS,  
 TAU = m1$parameters$TAU,  
 bodymass = 80,  
 # Additional parameters forwarded to get\_air\_resistance  
 # Otherwise, defaults are used  
 bodyheight = 1.85,  
 barometric\_pressure = 780,  
 air\_temperature = 20,  
 wind\_velocity = 0.5  
)  
  
fvp  
#> Estimated Force-Velocity Profile  
#> --------------------------------  
#> bodymass F0 F0\_rel V0   
#> 80.00000 544.51032 6.80638 9.36184   
#> Pmax Pmax\_relative FV\_slope RFmax\_cutoff   
#> 1274.40402 15.93005 -0.72703 0.30000   
#> RFmax Drf RSE\_FV RSE\_Drf   
#> 0.48779 -0.06676 1.54814 0.00447

To plot FVP kinematics and kinetics (which are exactly the same as generated by the predict\_kinematics() function), use S3 plot() function. By default, FVP estimated kinetics are plotted against velocity (on x-axis).

plot(fvp) + theme\_bw(8)



To plot FVP estimated kinetics against time, use type = "time" parameter:

plot(fvp, "time") + theme\_bw(8)

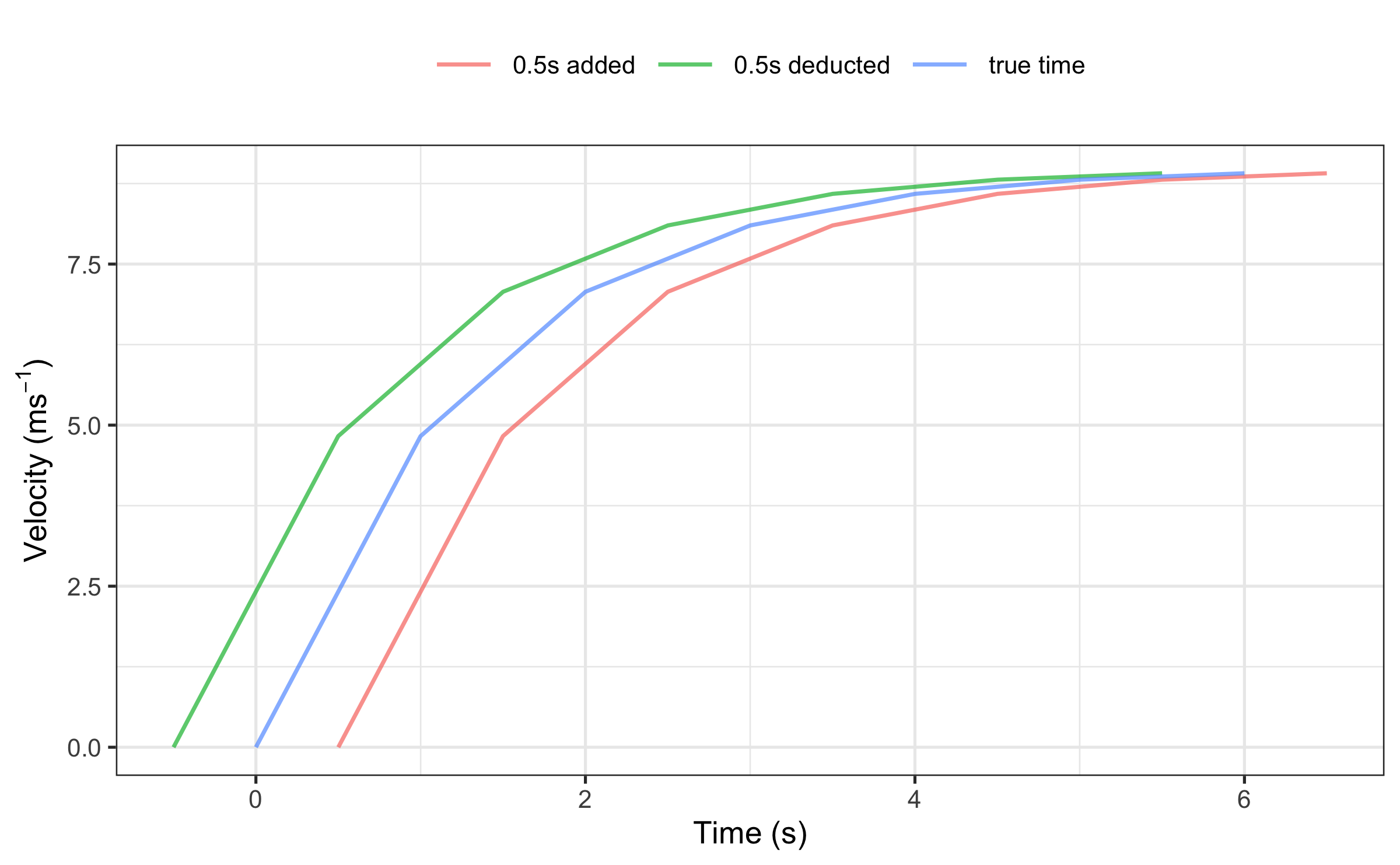
# Problems with estimation

There is a challenge when collecting sprint data that could have a substantial impact on modeled outcomes. To ensure accurate parameter outcomes, the initial force production must be synced with start time ([Thomas A. Haugen, Breitschädel, and Samozino 2020](#X51c34a68a97c6a6d254e5eebdf120de021512e2); [Thomas A. Haugen, Breitschädel, and Seiler 2019](#ref-haugenSprintMechanicalVariables2019)). Below we describe this challenge when using radar guns or timing gates and suggest potential solutions within the **shorts** package.

## Problems with time sync with radar gun

One source of error in the modeled estimation using a radar gun is the time synchronization. In theory, synchronization is ideal when a sprint is initiated at (i.e., ). In practice, this is often not the case. Let’s use our athlete and add and deduct 0.5 s to simulate an error in synchronization and its effect on estimated MSS and TAU.

df <- tibble(  
 `true time` = sprint\_time,  
 velocity = sprint\_velocity,  
 `0.5s added` = `true time` + 0.5,  
 `0.5s deducted` = `true time` - 0.5  
)  
  
plot\_df <- pivot\_longer(df, cols = -2, names\_to = "Sync issue")  
  
ggplot(  
 plot\_df,  
 aes(x = value, y = velocity, color = `Sync issue`)  
) +  
 theme\_bw(8) +  
 geom\_line(alpha = 0.7) +  
 xlab("Time (s)") +  
 ylab(expression("Velocity (" \* ms^-1 \* ")")) +  
 theme(  
 legend.title = element\_blank(),  
 legend.position = "top"  
 )



The following three models estimate MSS and TAU from the three datasets. Tables are created using the kable() function from the **knitr** package ([Xie 2014](#ref-knitr2014), [2015](#ref-knitr2015), [2021b](#ref-R-knitr)) and kable\_styling() function from the **kableExtra** package ([Zhu 2021](#ref-R-kableExtra)).

require(knitr)  
  
# Without synchronization issues  
m5 <- model\_using\_radar(  
 velocity = df$velocity,  
 time = df$`true time`  
)  
  
# With time added  
m6 <- model\_using\_radar(  
 velocity = df$velocity,  
 time = df$`0.5s added`  
)  
  
# With time deducted  
m7 <- model\_using\_radar(  
 velocity = df$velocity,  
 time = df$`0.5s deducted`  
)  
  
model\_df <- rbind(  
 data.frame(  
 model = "True time",  
 t(coef(m5))  
 ),  
 data.frame(  
 model = "Added 0.5s time",  
 t(coef(m6))  
 ),  
 data.frame(  
 model = "Deducted 0.5s time",  
 t(coef(m7))  
 )  
)  
  
kable(model\_df, digits = 2, booktabs = TRUE)

| model | MSS | TAU | MAC | PMAX | time\_correction | distance\_correction |
| --- | --- | --- | --- | --- | --- | --- |
| True time | 9.00 | 1.30 | 6.92 | 15.6 | 0 | 0 |
| Added 0.5s time | 9.91 | 2.34 | 4.23 | 10.5 | 0 | 0 |
| Deducted 0.5s time | 10.08 | 1.86 | 5.43 | 13.7 | 0 | 0 |

As can be seen from the example, all estimated parameters are affected by an error in synchronization of time with velocity (with MSS being the least affected in this example). The potential solution incorporated into the **shorts** package involves estimation of the *time correction* parameter using the following equation:

This model is incorporated in the model\_using\_radar\_with\_time\_correction() function:

# With time added  
m8 <- model\_using\_radar\_with\_time\_correction(  
 velocity = df$velocity,  
 time = df$`0.5s added`  
)  
coef(m8)  
#> MSS TAU MAC   
#> 9.00 1.30 6.92   
#> PMAX time\_correction distance\_correction   
#> 15.58 -0.50 0.00  
  
# With time deducted  
m9 <- model\_using\_radar\_with\_time\_correction(  
 velocity = df$velocity,  
 time = df$`0.5s deducted`  
)  
coef(m9)  
#> MSS TAU MAC   
#> 9.00 1.30 6.92   
#> PMAX time\_correction distance\_correction   
#> 15.58 0.50 0.00

When using predict\_XXX() family of functions, one can provide estimated time correction to get predictions at original time scale.

# Using the true time  
predict\_velocity\_at\_time(  
 time = df$`true time`,  
 MSS = m5$parameters$MSS,  
 TAU = m5$parameters$TAU  
)  
#> [1] 0.00 4.83 7.07 8.11 8.59 8.81 8.91  
  
# Using time with sync issues  
predict\_velocity\_at\_time(  
 time = df$`0.5s added`,  
 MSS = m8$parameters$MSS,  
 TAU = m8$parameters$TAU,  
 time\_correction = m8$parameters$time\_correction  
)  
#> [1] 0.0000782 4.8299729 7.0681475 8.1053182 8.5859434 8.8086652  
#> [7] 8.9118746

### Mixed-model approach

When it comes to mixed-model approach, time correction can be modeled as a fixed effect or random effect using the mixed\_model\_using\_radar\_with\_time\_correction() function.

# Adding 0.5s to radar\_gun\_data  
radar\_gun\_data$time <- radar\_gun\_data$time + 0.5  
  
# Mixed model with time correction being fixed effect  
m10 <- mixed\_model\_using\_radar\_with\_time\_correction(  
 radar\_gun\_data,  
 time = "time",  
 velocity = "velocity",  
 athlete = "athlete",  
 random = MSS + TAU ~ 1  
)  
  
# Mixed model with time correction being random effect  
m11 <- mixed\_model\_using\_radar\_with\_time\_correction(  
 radar\_gun\_data,  
 time = "time",  
 velocity = "velocity",  
 athlete = "athlete",  
 random = MSS + TAU + time\_correction ~ 1  
)

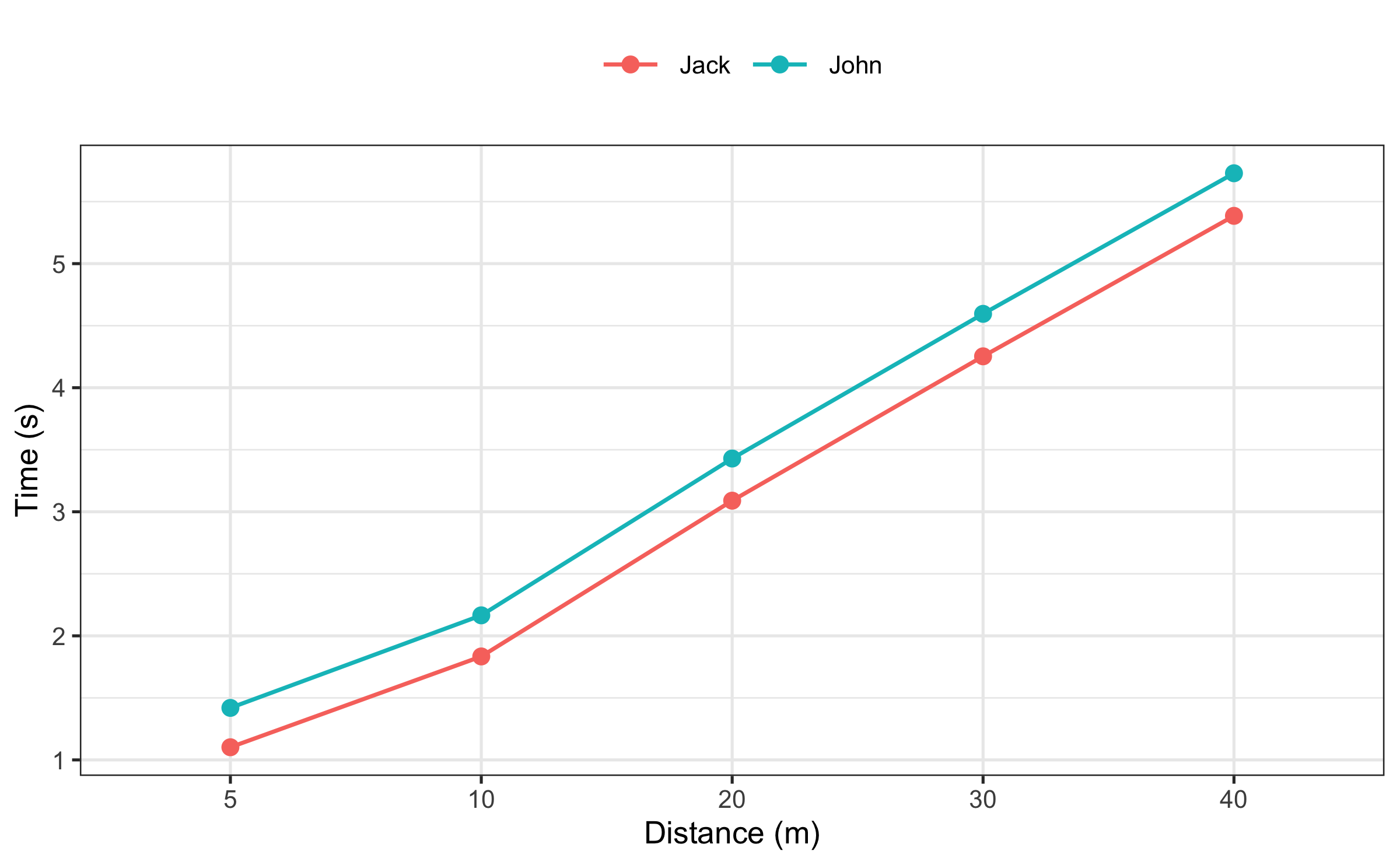
## Problems at the start when using timing gates

Let’s imagine we have two twin brothers with same short sprint characteristics: MSS equal to 9 , TAU equal to 1.3, and MAC equal to 6.92 . Let’s call them John and Jack. They both perform 40m sprint using timing gates set at 5, 10, 20, 30, and 40 m. The initial timing gate at the start (i.e., m) serves to activate the timing system (i.e., when they cross the beam).

John represents the *theoretical model*, in which we assume that the initial force production and the timing initiation are perfectly synchronized. Jack, on the other hand, represents a *practical model*, and decides to move slightly behind the initial timing gate (i.e. for 0.5 m) and use body rocking to initiate the sprint start. In other words, Jack is using a *flying start*, a common scenario when testing field sports athletes. Flying start distance is often recommended to avoid premature triggering of the timing system by lifted knees or swinging arms ([Altmann et al. 2018](#ref-altmannAccuracySingleBeam2018), [2015](#X1e9453b82425d54fd450da95e3e109c3caee0a3), [2017](#ref-altmannValiditySingleBeamTiming2017); [Thomas A. Haugen, Breitschädel, and Samozino 2020](#X51c34a68a97c6a6d254e5eebdf120de021512e2); [T. Haugen and Buchheit 2016](#ref-haugenSprintRunningPerformance2016)).

MSS <- 9  
TAU <- 1.3  
MAC <- MSS / TAU  
  
split\_times <- tibble(  
 distance = c(5, 10, 20, 30, 40),  
 john\_time = predict\_time\_at\_distance(distance, MSS, TAU),  
  
 # Jack's performance  
 jack\_distance = distance + 0.5,  
 jack\_true\_time = predict\_time\_at\_distance(jack\_distance, MSS, TAU),  
 time\_05m = predict\_time\_at\_distance(0.5, MSS, TAU),  
 jack\_time = jack\_true\_time - time\_05m  
)

Here is a graphical representation of the sprint splits (please refer to the [Supplemental material](#supplemental-material) for the R code):



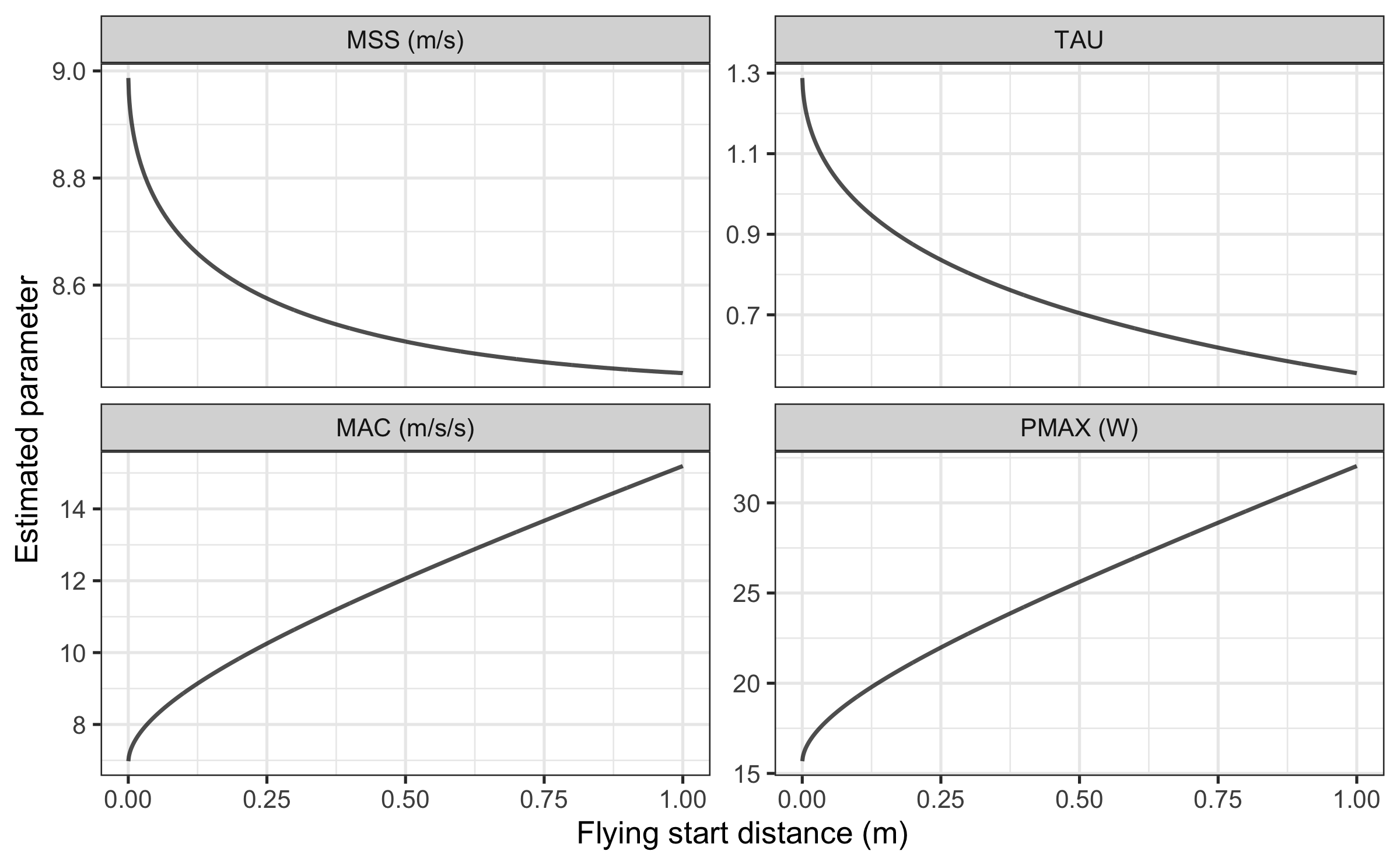
Using the following code, we can see the differences in estimated MSS and TAU parameters:

# Since this is a perfect simulation and stats::nls will complain  
# we need to add very small noise, or measurement error to the times  
set.seed(1667)  
rand\_noise <- rnorm(nrow(split\_times), 0, 10^-5)  
split\_times$john\_time <- split\_times$john\_time + rand\_noise  
split\_times$jack\_time <- split\_times$jack\_time + rand\_noise  
  
john\_profile <- model\_using\_splits(  
 distance = split\_times$distance,  
 time = split\_times$john\_time  
)  
  
jack\_profile <- model\_using\_splits(  
 distance = split\_times$distance,  
 time = split\_times$jack\_time  
)  
  
sprint\_parameters <- rbind(  
 coef(john\_profile),  
 coef(jack\_profile)  
)  
  
rownames(sprint\_parameters) <- c("John", "Jack")  
  
kable(sprint\_parameters, digits = 2, booktabs = TRUE)

|  | MSS | TAU | MAC | PMAX | time\_correction | distance\_correction |
| --- | --- | --- | --- | --- | --- | --- |
| John | 9.00 | 1.3 | 6.92 | 15.6 | 0 | 0 |
| Jack | 8.49 | 0.7 | 12.06 | 25.6 | 0 | 0 |

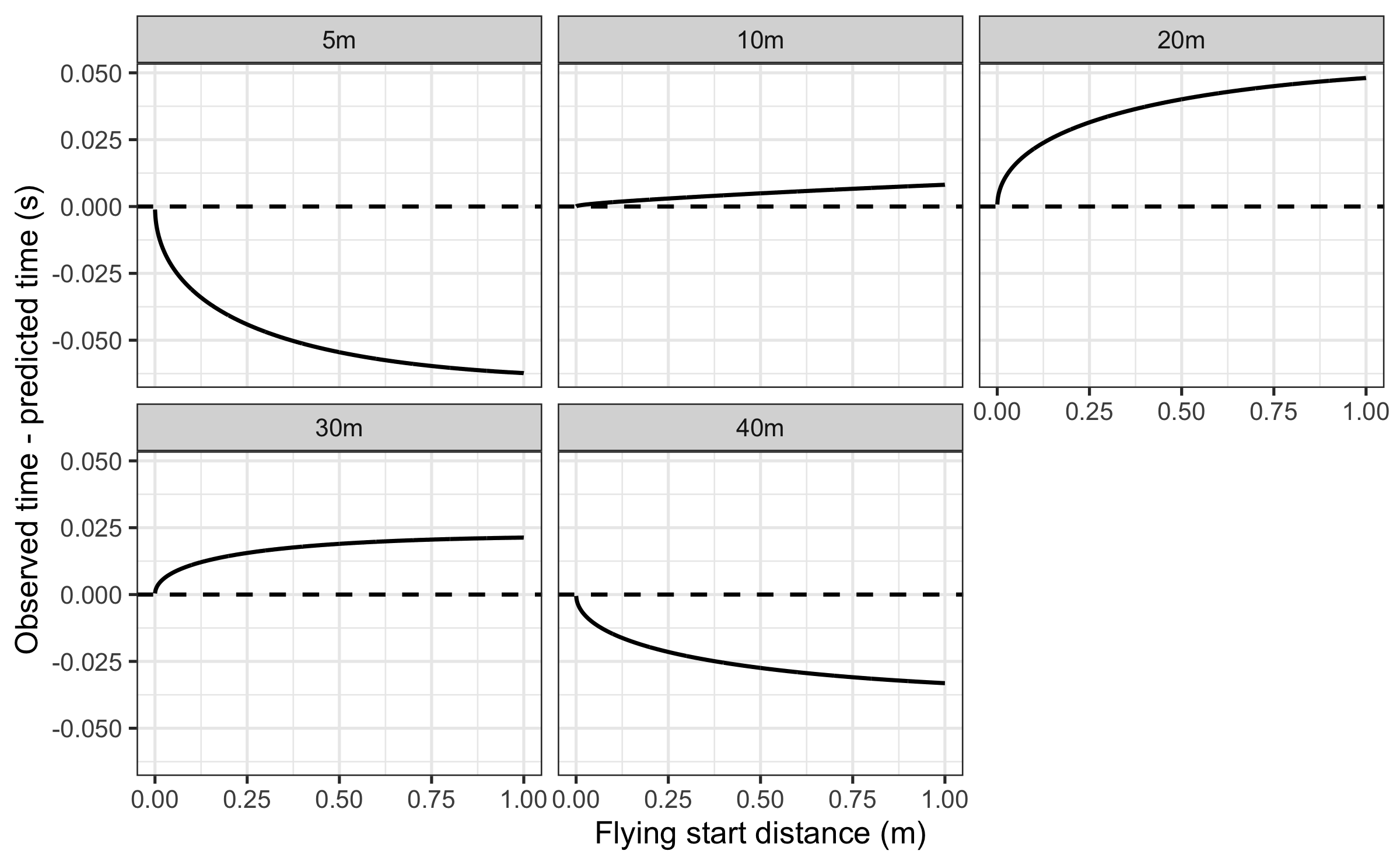
As can be seen from the results, a flying start yields biased estimates, particularly for the TAU, MAC and PMAX.

To explore this further, we have run a simple simulation by increasing Jack’s flying start distance from 0 to 1 m and depicting the estimated MSS, TAU, MAC, and PMAX parameters (please refer to the [Supplemental material](#supplemental-material) for the R code).

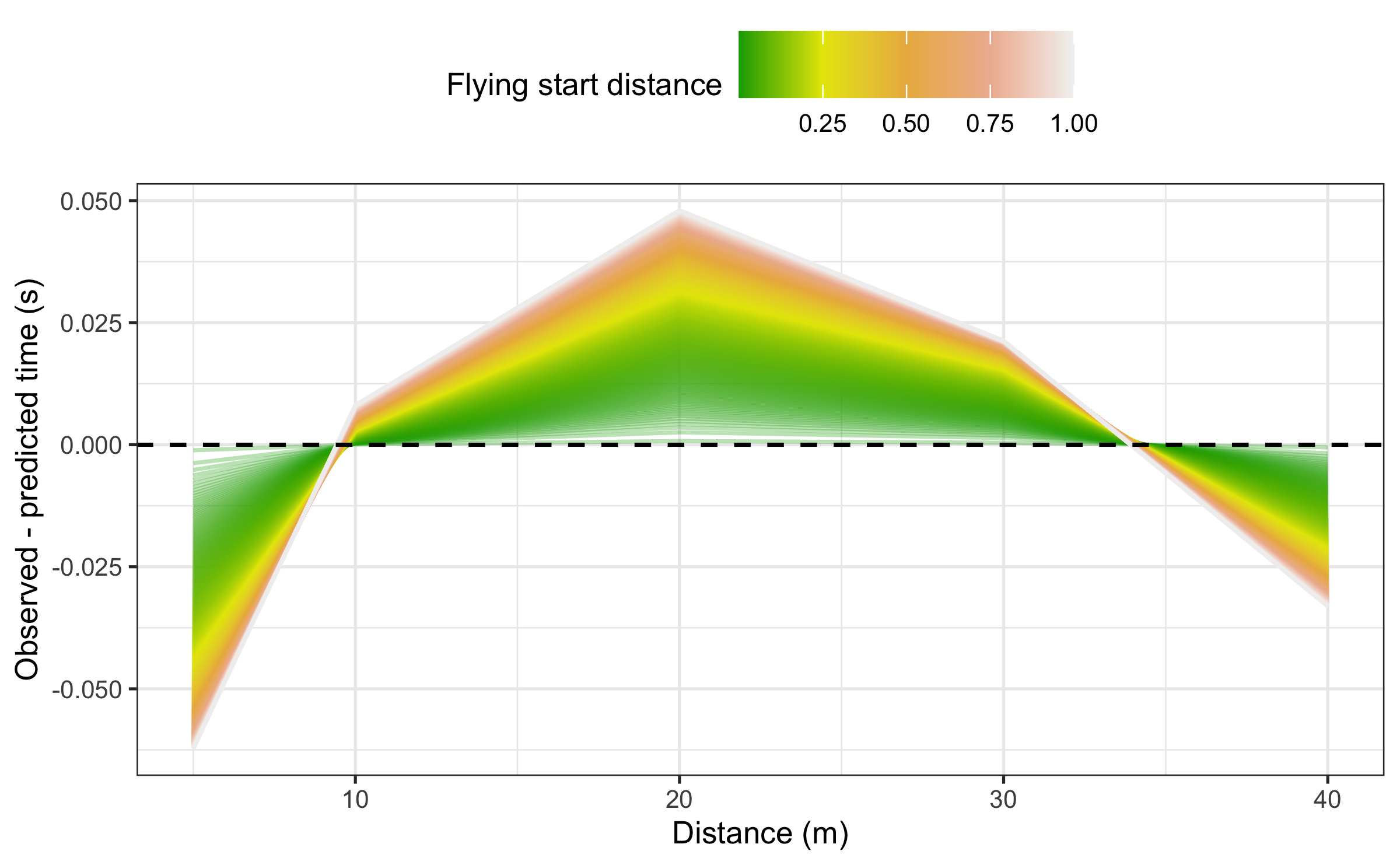


As can be seen from the figure, MSS is underestimated as flying start distance increases. MAC (and also TAU) are highly affected by the flying start distance, and from the figure we can notice that MAC is overestimated as flying start distance increases. Estimated PMAX is also overestimated as flying start distance increases.

Model residuals are also affected by the flying start distance. The shape of residuals distribution depends on number and splits utilized (e.g., 10, 20, 30, 40 m versus 5, 15, 30 m), but here we can see the effect of the flying start distance on the model residuals per split distance utilized in our simple simulation:



Another way to visualize the effect of the flying start distance on split distance residuals can be found on the next figure:



Clearly, any type of flying start where there is a difference between initial force production and start time can result in biased parameters and predictions. Since maximal sprint speed is difficult to improve, the effects of start inconsistencies can mask effects of the training intervention. It is thus crucial to standardize the start when testing and implementing the following techniques when using the **shorts** package.

### How to overcome missing the initial force production when using timing gates?

A potential solution is to use a correction factor - the recommendation in the literature is +0.5 s ([Thomas A. Haugen, Breitschädel, and Seiler 2020](#ref-haugenSprintMechanicalProperties2020), [2019](#ref-haugenSprintMechanicalVariables2019)). Interestingly, the average difference between using timing gates and a block start for 40 m sprint time was 0.27 s ([Thomas A. Haugen, Tønnessen, and Seiler 2012](#ref-haugenDifferenceStartImpact2012)). So, while a timing correction factor is warranted to avoid subsequent errors in estimates of kinetic variables (e.g., overestimate power), a correction factor that is too large will have the opposite effect (e.g., underestimate power).

Rather than providing *apriori* time correction from the literature, **shorts** package provides an estimation of this parameter from the data provided, together with MSS and TAU. Exactly the same method is suggested by [Stenroth, Vartiainen, and Karjalainen](#Xf1011dded519ee0de75fb97a2b8a911dfc18ddc) ([2020](#Xf1011dded519ee0de75fb97a2b8a911dfc18ddc)), named *time shift method*, and the estimated parameter named *time shift parameter*. We have named this parameter *time correction* to be in agreement with the parameter introduced in [Problems with time sync with radar gun](#problems-with-time-sync-with-radar-gun) section of this paper, as well as the available literature.

When implementing time correction, equation (5) becomes:

To estimate time correction parameter, we use model\_using\_splits\_with\_time\_correction() function. Here is how we can estimate Jack parameters using either provided time correction (e.g., +0.3 and +0.5 s) or estimated time correction:

jack\_profile\_fixed\_time\_short <- model\_using\_splits(  
 distance = split\_times$distance,  
 time = split\_times$jack\_time,  
 time\_correction = 0.3  
)  
  
jack\_profile\_fixed\_time\_long <- model\_using\_splits(  
 distance = split\_times$distance,  
 time = split\_times$jack\_time,  
 time\_correction = 0.5  
)  
  
jack\_profile\_time\_estimated <- model\_using\_splits\_with\_time\_correction(  
 distance = split\_times$distance,  
 time = split\_times$jack\_time  
)  
  
jack\_parameters <- rbind(  
 coef(john\_profile),  
 coef(jack\_profile),  
 coef(jack\_profile\_fixed\_time\_short),  
 coef(jack\_profile\_fixed\_time\_long),  
 coef(jack\_profile\_time\_estimated)  
)  
  
rownames(jack\_parameters) <- c(  
 "John",  
 "Jack - No corrections",  
 "Jack - Fixed time correction (+0.3s)",  
 "Jack - Fixed time correction (+0.5s)",  
 "Jack - Estimated time correction"  
)  
  
kable(jack\_parameters, digits = 2, booktabs = TRUE)

|  | MSS | TAU | MAC | PMAX | time\_correction | distance\_correction |
| --- | --- | --- | --- | --- | --- | --- |
| John | 9.00 | 1.30 | 6.92 | 15.6 | 0.00 | 0 |
| Jack - No corrections | 8.49 | 0.70 | 12.06 | 25.6 | 0.00 | 0 |
| Jack - Fixed time correction (+0.3s) | 9.00 | 1.25 | 7.19 | 16.2 | 0.30 | 0 |
| Jack - Fixed time correction (+0.5s) | 9.62 | 1.77 | 5.43 | 13.1 | 0.50 | 0 |
| Jack - Estimated time correction | 8.96 | 1.22 | 7.37 | 16.5 | 0.28 | 0 |

In Jack’s case, both +0.3 s fixed time correction and time correction estimation yield parameters closer to John’s (i.e. true parameters). We have used these two model definitions in retrospective pilot study ([Vescovi and Jovanović 2021](#Xc4200044e1419d2ae905af7be5d26d53e3f416c)), demonstrating statistically significant differences in estimated FVP parameters.

Instead of using time correction as simple intercept in the equation (7), we can estimate the *distance correction* (i.e., the flying start distance) using the following equation:

Equation (8) was used to generate the data for John and Jack (please refer to the provided R code), as well as for the simple simulation, and can be used as model definition to control for the flying start distance.

To estimate distance correction parameter, we use model\_using\_splits\_with\_distance\_correction() function. Here is how we can estimate Jack parameters using distance correction:

jack\_profile\_distance\_correction <- model\_using\_splits\_with\_distance\_correction(  
 distance = split\_times$distance,  
 time = split\_times$jack\_time  
)  
  
jack\_parameters <- rbind(  
 jack\_parameters,  
 coef(jack\_profile\_distance\_correction)  
)  
  
rownames(jack\_parameters) [6] <- "Jack - Estimated distance correction"  
  
kable(jack\_parameters, digits = 2, booktabs = TRUE)

|  | MSS | TAU | MAC | PMAX | time\_correction | distance\_correction |
| --- | --- | --- | --- | --- | --- | --- |
| John | 9.00 | 1.30 | 6.92 | 15.6 | 0.00 | 0.0 |
| Jack - No corrections | 8.49 | 0.70 | 12.06 | 25.6 | 0.00 | 0.0 |
| Jack - Fixed time correction (+0.3s) | 9.00 | 1.25 | 7.19 | 16.2 | 0.30 | 0.0 |
| Jack - Fixed time correction (+0.5s) | 9.62 | 1.77 | 5.43 | 13.1 | 0.50 | 0.0 |
| Jack - Estimated time correction | 8.96 | 1.22 | 7.37 | 16.5 | 0.28 | 0.0 |
| Jack - Estimated distance correction | 9.00 | 1.30 | 6.92 | 15.6 | 0.00 | 0.5 |

Another model definition, which also represents a novel approach implemented in the **shorts** package, is to utilize both time and distance corrections. Thus, equation (7) becomes:

This model is implemented in model\_using\_splits\_with\_corrections() function. Below are the model estimates:

jack\_profile\_time\_distance\_correction <- model\_using\_splits\_with\_corrections(  
 distance = split\_times$distance,  
 time = split\_times$jack\_time  
)  
  
jack\_parameters <- rbind(  
 jack\_parameters,  
 coef(jack\_profile\_time\_distance\_correction)  
)  
  
rownames(jack\_parameters) [7] <-   
 "Jack - Estimated time & distance corrections"  
  
kable(jack\_parameters, digits = 2, booktabs = TRUE)

|  | MSS | TAU | MAC | PMAX | time\_correction | distance\_correction |
| --- | --- | --- | --- | --- | --- | --- |
| John | 9.00 | 1.30 | 6.92 | 15.6 | 0.00 | 0.0 |
| Jack - No corrections | 8.49 | 0.70 | 12.06 | 25.6 | 0.00 | 0.0 |
| Jack - Fixed time correction (+0.3s) | 9.00 | 1.25 | 7.19 | 16.2 | 0.30 | 0.0 |
| Jack - Fixed time correction (+0.5s) | 9.62 | 1.77 | 5.43 | 13.1 | 0.50 | 0.0 |
| Jack - Estimated time correction | 8.96 | 1.22 | 7.37 | 16.5 | 0.28 | 0.0 |
| Jack - Estimated distance correction | 9.00 | 1.30 | 6.92 | 15.6 | 0.00 | 0.5 |
| Jack - Estimated time & distance corrections | 9.00 | 1.30 | 6.92 | 15.6 | 0.40 | 0.5 |

As can be seen from the results, adding distance correction results in correctly estimating Jack’s sprint parameters. There are a few issues with this model definition. Besides being novel and still not validated with actual data, distance correction model has four parameters to estimate, which implies that at least four sprint splits are needed. This imposes practical limitations, since acquiring five timing gates (one for the start and four for splits) might be practically troublesome. One strategy that is sometimes implemented is adding zeros to the sample (i.e., and ), which increase the number of observations. Unfortunately, this strategy should not be implemented since it will bias the estimated parameters:

jack\_profile\_zero <- model\_using\_splits(  
 distance = c(0, split\_times$distance),  
 time = c(0, split\_times$jack\_time)  
)  
  
jack\_profile\_time\_estimated\_zero <- model\_using\_splits\_with\_time\_correction(  
 distance = c(0, split\_times$distance),  
 time = c(0, split\_times$jack\_time)  
)  
  
adding\_zero\_issue <- rbind(  
 coef(jack\_profile),  
 coef(jack\_profile\_zero),  
 coef(jack\_profile\_time\_estimated),  
 coef(jack\_profile\_time\_estimated\_zero)  
)  
  
rownames(adding\_zero\_issue) <- c(  
 "Jack - No corrections",  
 "Jack - No corrections with zeros added",  
 "Jack - Estimated time correction",  
 "Jack profile - Estimated time correction with added zeros")  
  
kable(adding\_zero\_issue, digits = 2, booktabs = TRUE)

|  | MSS | TAU | MAC | PMAX | time\_correction | distance\_correction |
| --- | --- | --- | --- | --- | --- | --- |
| Jack - No corrections | 8.49 | 0.70 | 12.06 | 25.6 | 0.00 | 0 |
| Jack - No corrections with zeros added | 8.49 | 0.70 | 12.06 | 25.6 | 0.00 | 0 |
| Jack - Estimated time correction | 8.96 | 1.22 | 7.37 | 16.5 | 0.28 | 0 |
| Jack profile - Estimated time correction with added zeros | 8.51 | 0.73 | 11.65 | 24.8 | 0.02 | 0 |

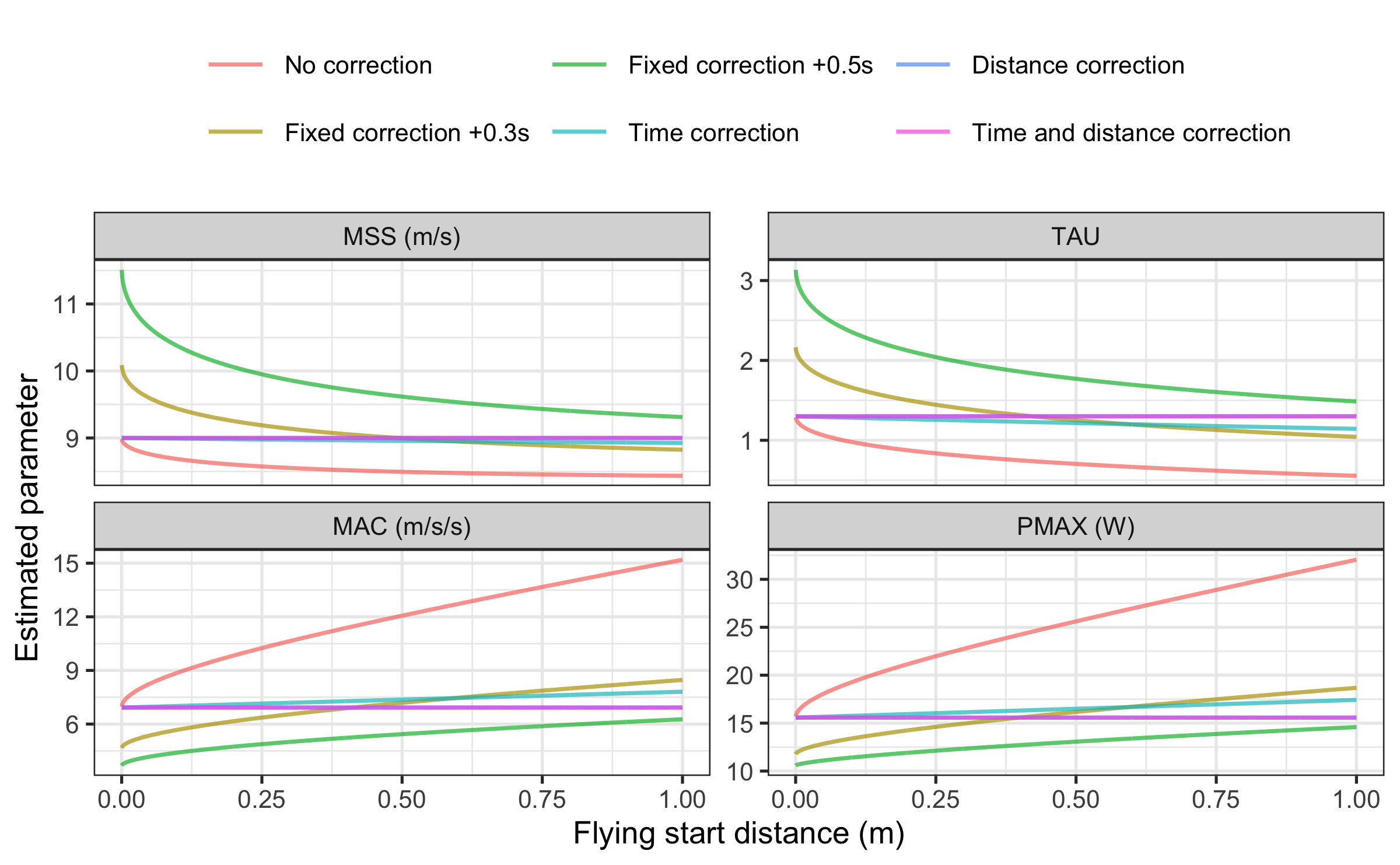
Adding zeros to the sample nullifies the potential benefits of using time correction model and should be avoided in practice.

It is important to note that using predict\_XXX() family of functions with distance correction model and using estimated distance correction parameter will yield wrong predictions on the original scale (but not on the *theoretical* scale in which force production starts at d=0m and t=0s). This is because predict\_XXX() family of functions utilize equation (9) and its derivates and integrals, which works for the uncorrected model (time and distance correction parameters are equal to zero), time correction models (distance correction parameter is equal to zero), and time and distance corrections model. You can notice that equations (7) and (9) are equivalent to the (5) equation when time and distance correction parameters are equal to zero. Unfortunately, this doesn’t work for the distance correction model when the estimated distance correction parameter is used since model definition outlined in the equation (8) is different that the equation (9) when the estimated distance correction parameter is different than zero. The solution is to use generic S3 predict() function, but this only predicts split times at split distances.

# Jack's observed times  
split\_times$jack\_time  
#> [1] 1.10 1.83 3.09 4.25 5.39  
  
# To predict at "theoretical" distance scale (where force application starts at d=0m)  
predict\_time\_at\_distance(  
 distance = split\_times$distance,   
 MSS = jack\_profile\_time\_estimated$parameters$MSS,  
 TAU = jack\_profile\_time\_estimated$parameters$TAU)  
#> [1] 1.38 2.12 3.37 4.54 5.67  
  
predict\_time\_at\_distance(  
 distance = split\_times$distance,   
 MSS = jack\_profile\_distance\_correction$parameters$MSS,  
 TAU = jack\_profile\_distance\_correction$parameters$TAU)  
#> [1] 1.42 2.17 3.43 4.60 5.73  
  
# To predict at original distance scale, use all estimated parameters  
predict\_time\_at\_distance(  
 distance = split\_times$distance,   
 MSS = jack\_profile\_time\_estimated$parameters$MSS,  
 TAU = jack\_profile\_time\_estimated$parameters$TAU,  
 time\_correction = jack\_profile\_time\_estimated$parameters$time\_correction)  
#> [1] 1.10 1.84 3.09 4.25 5.39  
  
# Unfortunately, this doesn't work for the distance correction model  
predict\_time\_at\_distance(  
 distance = split\_times$distance,   
 MSS = jack\_profile\_distance\_correction$parameters$MSS,  
 TAU = jack\_profile\_distance\_correction$parameters$TAU,  
 distance\_correction = jack\_profile\_distance\_correction$parameters$distance\_correction)  
#> [1] 1.50 2.23 3.49 4.65 5.78  
  
# Instead use generic S3 predict() function  
predict(jack\_profile\_distance\_correction)  
#> [1] 1.10 1.83 3.09 4.25 5.39  
  
predict(  
 jack\_profile\_distance\_correction$model,  
 newdata = data.frame(distance = split\_times$distance))  
#> [1] 1.10 1.83 3.09 4.25 5.39

In the next section we examine how these models (no correction, fixed time correction, estimated time correction, estimated distance correction, and estimated time and distance correction) perform using simulated data with varying flying start distance (please refer to the [Supplemental material](#supplemental-material) for the R code).

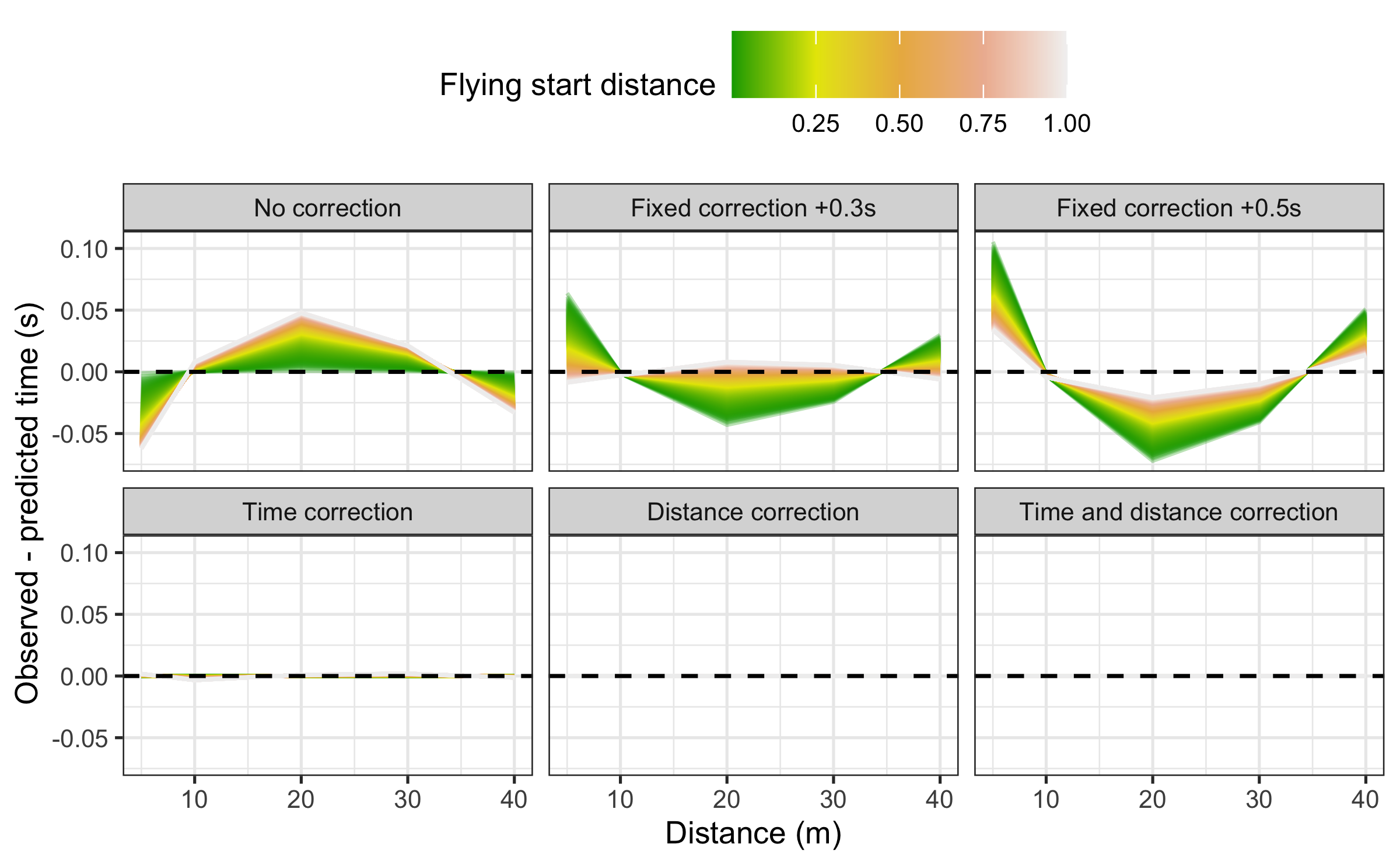
As can be seen from the next figure, the estimated time correction model estimates MSS almost perfectly, while the estimated distance and estimate time and distance correction models estimates MSS perfectly.



Similar outcomes are observed for the MAC parameter. The distance correction model and time and distance corrections models performs perfectly, while the time correction model performs almost as good. PMAX demonstrates the same properties as MSS and MAC.

The next figure depicts residuals across split distances for each simulated flying start distance. As can be seen, time correction, distance correction, and time and distance correction models perform much better than the other models.

# Residuals  
ggplot(  
 model\_df,  
 aes(  
 y = residuals,  
 x = distance,  
 color = flying\_start\_distance,  
 group = flying\_start\_distance  
 )  
) +  
 theme\_bw(8) +  
 geom\_line(alpha = 0.3) +  
 geom\_hline(yintercept = 0, linetype = "dashed") +  
 scale\_color\_gradientn(colours = terrain.colors(5, rev = FALSE)) +  
 facet\_wrap(~model) +  
 xlab("Distance (m)") +  
 ylab("Observed - predicted time (s)") +  
 theme(legend.position = "top") +  
 labs(color = "Flying start distance")



The outcomes from this simple simulation data clearly demonstrates that the time correction, distance correction, and time and distance correction models represent sound improvements in parameter estimation and model fit compared to no corrections model and fixed correction model when attempting to overcome the flying start issues. Since the time correction and distance correction models are simpler and requires three parameters to be estimated, they might be practically more useful than the time and distance correction model, which requires four parameters estimation and thus more than four timing gates and sprint splits. More elaborate simulation study is currently ongoing and the results will be reported in another paper.

Time correction, distance correction, and time and distance corrections are also implemented in the mixed-models using mixed\_model\_using\_splits\_with\_time\_correction(), mixed\_model\_using\_splits\_with\_distance\_correction(), and mixed\_model\_using\_splits\_with\_corrections() functions. We will showcase their use at the end of this paper.

# Leave-one-out Cross-Validation

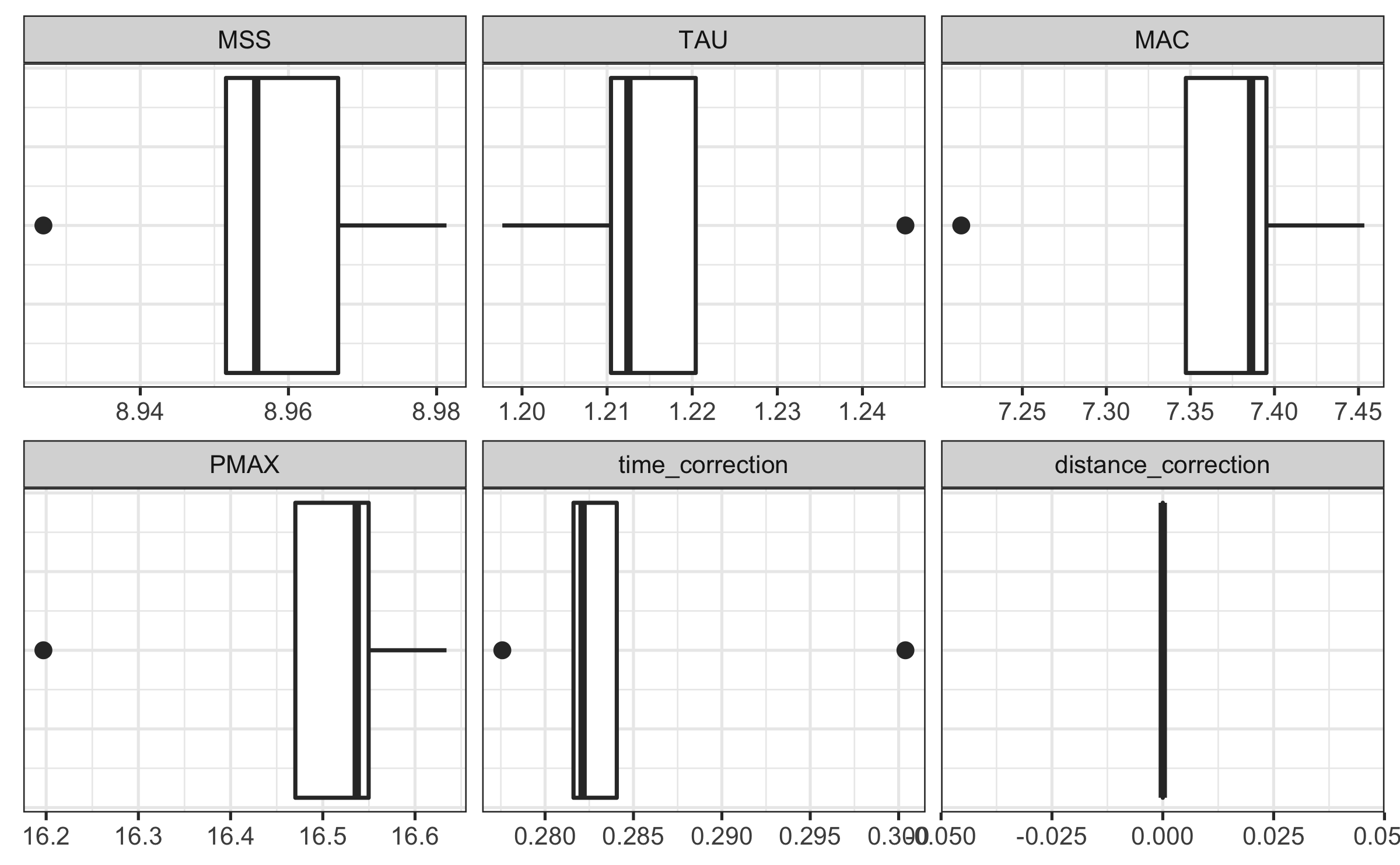
To estimate parameter stability, model over-fitting, and performance on unseen data, **shorts** model function comes with implemented *leave-one-out cross validation* (LOOCV) ([James et al. 2017](#X61f23a4de41b0471e8dae622089ec4fbb2f330f); [Jovanović 2020](#X87b6e3e0ab837a9dc3f187ffd4719394205cfac); [Kuhn and Johnson 2018](#ref-kuhnAppliedPredictiveModeling2018)). LOOCV involves a simple, yet powerful procedure, of removing each observation, rebuilding the model, and making predictions for that removed observation. This process is repeated for each observation in the model dataset. LOOCV allows one to check estimated parameters stability, and model performance on unseen data.

Let’s perform LOOCV using Jack’s data and the time correction model:

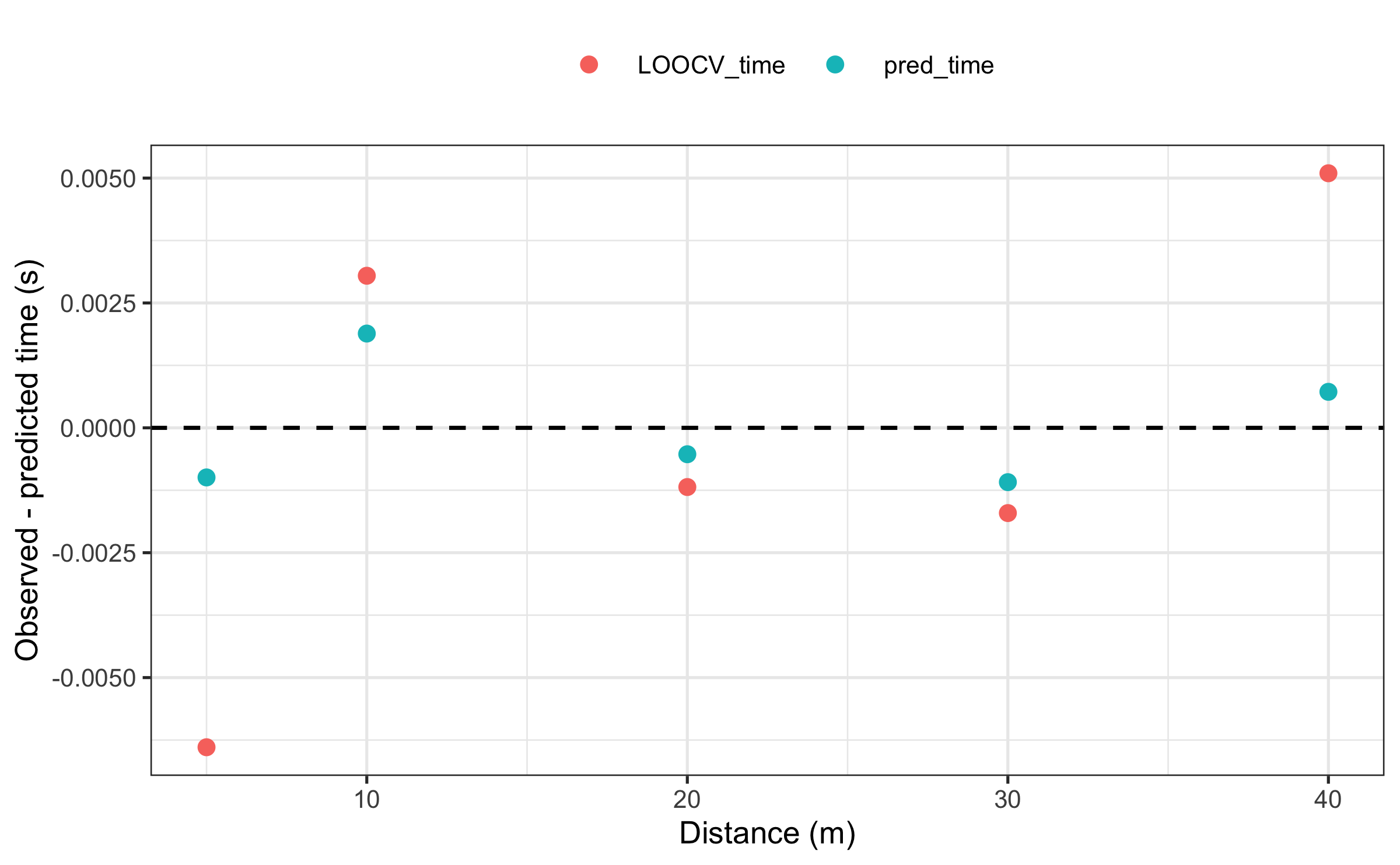
jack\_LOOCV <- model\_using\_splits\_with\_time\_correction(  
 distance = split\_times$distance,  
 time = split\_times$jack\_time,  
 LOOCV = TRUE  
)

The model print output provides training dataset estimates and model performance, as well as LOOCV estimates and model performance.

Next we plot estimated parameters across LOOCV folds (please refer to the [Supplemental material](#supplemental-material) for the R code):



Here is the plot of the training and LOOCV residuals:



As expected, the model has more issues predicting unseen split times for both short or long distances. Please note, that since LOOCV removes one observation, if the model estimates three parameters, then at least four observations are needed, since we need to make sure the model can be estimated once a single observation is removed. LOOCV can also be implemented with the mixed-effects models in the **shorts** package.

# Example analysis

Let’s utilize demonstrated functionalities of the **shorts** package using real-world data. The first dataset comes from Usain Bolt’s performance from IAAF World Championship held in London, 2017, and the second dataset involve Jason Vescovi’s sample data for 52 female soccer and field hockey athletes which comes with the **shorts** package (see the dataset documentation by executing ?vescovi command in the R console).

## Usain Bolt’s run from London 2017

The following dataset represents Usain Bolt’s race in the finals at the IAAF World Championship held in London, 2017. Since reaction time enters the splits, we want to see how that will affect the model estimates, and particularly, if the estimated time correction model will pick-up reaction time.

For the sake of this analysis, only 10 m splits over 60 m race distance are used.

bolt\_reaction\_time <- 0.183  
  
bolt\_distance <- c(10, 20, 30, 40, 50, 60)  
bolt\_time <- c(1.963, 2.983, 3.883, 4.763, 5.643, 6.493)  
  
# No corrections model  
bolt\_m1 <- model\_using\_splits(  
 distance = bolt\_distance,  
 time = bolt\_time  
)  
  
# Model with reaction time as fixed time correction  
bolt\_m2 <- model\_using\_splits(  
 distance = bolt\_distance,  
 time = bolt\_time,  
 time\_correction = -bolt\_reaction\_time  
)  
  
# Model with estimated time correction  
bolt\_m3 <- model\_using\_splits\_with\_time\_correction(  
 distance = bolt\_distance,  
 time = bolt\_time  
)  
  
# Model with estimated time correction, but deducted reaction time  
bolt\_m4 <- model\_using\_splits\_with\_time\_correction(  
 distance = bolt\_distance,  
 time = bolt\_time - bolt\_reaction\_time  
)  
  
# Model with estimated time and distance corrections  
bolt\_m5 <- model\_using\_splits\_with\_corrections(  
 distance = bolt\_distance,  
 time = bolt\_time  
)  
  
# Model with estimated time and distance corrections and  
# deducted reaction time  
bolt\_m6 <- model\_using\_splits\_with\_corrections(  
 distance = bolt\_distance,  
 time = bolt\_time - bolt\_reaction\_time  
)  
  
bolt\_model <- rbind(  
 data.frame(  
 model = "No correction",  
 t(coef(bolt\_m1))  
 ),  
 data.frame(  
 model = "No correction - RT",  
 t(coef(bolt\_m2))  
 ),  
 data.frame(  
 model = "Time correction",  
 t(coef(bolt\_m3))  
 ),  
 data.frame(  
 model = "Time correction - RT",  
 t(coef(bolt\_m4))  
 ),  
 data.frame(  
 model = "Distance correction",  
 t(coef(bolt\_m5))  
 ),  
 data.frame(  
 model = "Distance correction - RT",  
 t(coef(bolt\_m6))  
 )  
)  
  
kable(bolt\_model, digits = 2, booktabs = TRUE)

| model | MSS | TAU | MAC | PMAX | time\_correction | distance\_correction |
| --- | --- | --- | --- | --- | --- | --- |
| No correction | 12.1 | 1.56 | 7.77 | 23.6 | 0.00 | 0.00 |
| No correction - RT | 11.7 | 1.21 | 9.74 | 28.6 | -0.18 | 0.00 |
| Time correction | 11.7 | 1.20 | 9.76 | 28.6 | -0.18 | 0.00 |
| Time correction - RT | 11.7 | 1.20 | 9.76 | 28.6 | 0.00 | 0.00 |
| Distance correction | 11.6 | 0.85 | 13.56 | 39.3 | -0.81 | -3.98 |
| Distance correction - RT | 11.6 | 0.85 | 13.56 | 39.3 | -0.63 | -3.98 |

Here is the model estimate of the time and distance it takes for Bolt to reach 99% of MSS. Please note that we are not using distance and time correction parameters, since we want these estimates to be on the time/distance scale aligned with the actual sprint start (i.e., theoretical scale), not the measurement scale.

bolt\_model <- bolt\_model %>%  
 group\_by(model) %>%  
 mutate(  
 dist\_99\_MSS = find\_velocity\_critical\_distance(  
 MSS = MSS, TAU = TAU,  
 # time\_correction = time\_correction,  
 # distance\_correction = distance\_correction,  
 percent = 0.99  
 ),  
 time\_99\_MSS = find\_velocity\_critical\_time(  
 MSS = MSS, TAU = TAU,  
 # time\_correction = time\_correction,  
 percent = 0.99  
 )  
 )  
  
kable(bolt\_model[c(1, 8, 9)], digits = 2, booktabs = TRUE)

| model | dist\_99\_MSS | time\_99\_MSS |
| --- | --- | --- |
| No correction | 68.7 | 7.20 |
| No correction - RT | 51.1 | 5.55 |
| Time correction | 51.0 | 5.54 |
| Time correction - RT | 51.0 | 5.54 |
| Distance correction | 35.8 | 3.94 |
| Distance correction - RT | 35.8 | 3.94 |

## Vescovi data

The data from Vescovi represents a sub-set of data from a total of 220 high-level female athletes (151 soccer players and 69 field hockey players) ([Vescovi and Jovanović 2021](#Xc4200044e1419d2ae905af7be5d26d53e3f416c)). Using a random number generator, a total of 52 players (35 soccer and 17 field hockey) were selected for the sample dataset.

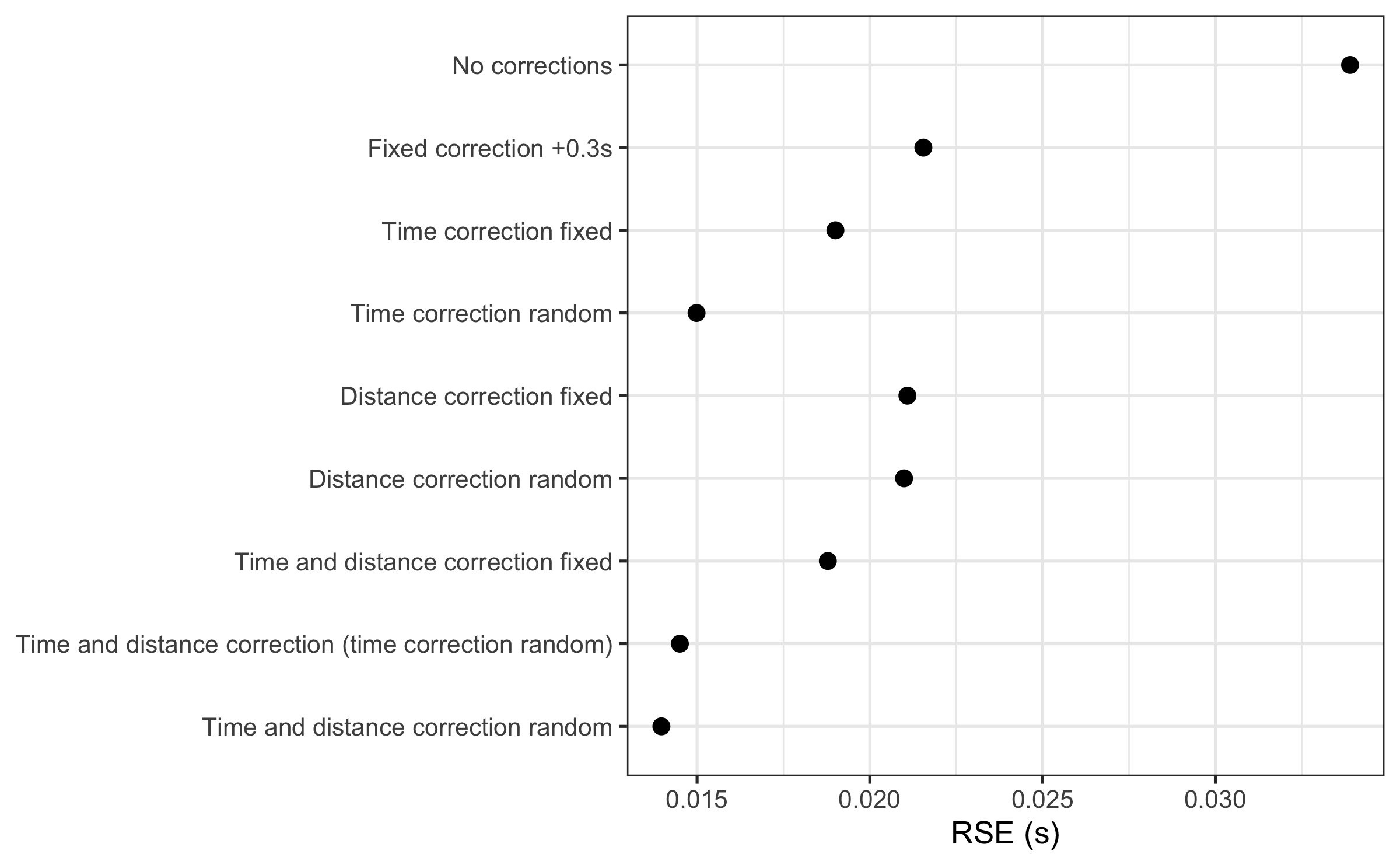
The protocol for assessing linear sprint speed has been described previously ([Vescovi 2014](#ref-vescoviImpactMaximumSpeed2014), [2016](#Xca67888bc977f16214d494c4585ab68c389caa1), [2012](#X307d051ee81aa49cbf164f3eada8ab66872dc43)) and was identical for each cohort. Briefly, all athletes performed a standardized warm-up that included general exercises such as jogging, shuffling, multi-directional movements, and dynamic stretching exercises. Infrared timing gates (Brower Timing, Utah) were positioned at the start line and at 5, 10, 20, 30, and 35 m at a height of approximately 1.0 m. Participants stood with their lead foot positioned approximately 5 cm behind the initial infrared beam (i.e., start line). Only forward movement was permitted (no leaning or rocking backwards) and timing started when the laser of the starting gate was triggered. The best 35 m time, and all associated split times were kept for analysis.

Below is the mixed-effects models analysis of the dataset.

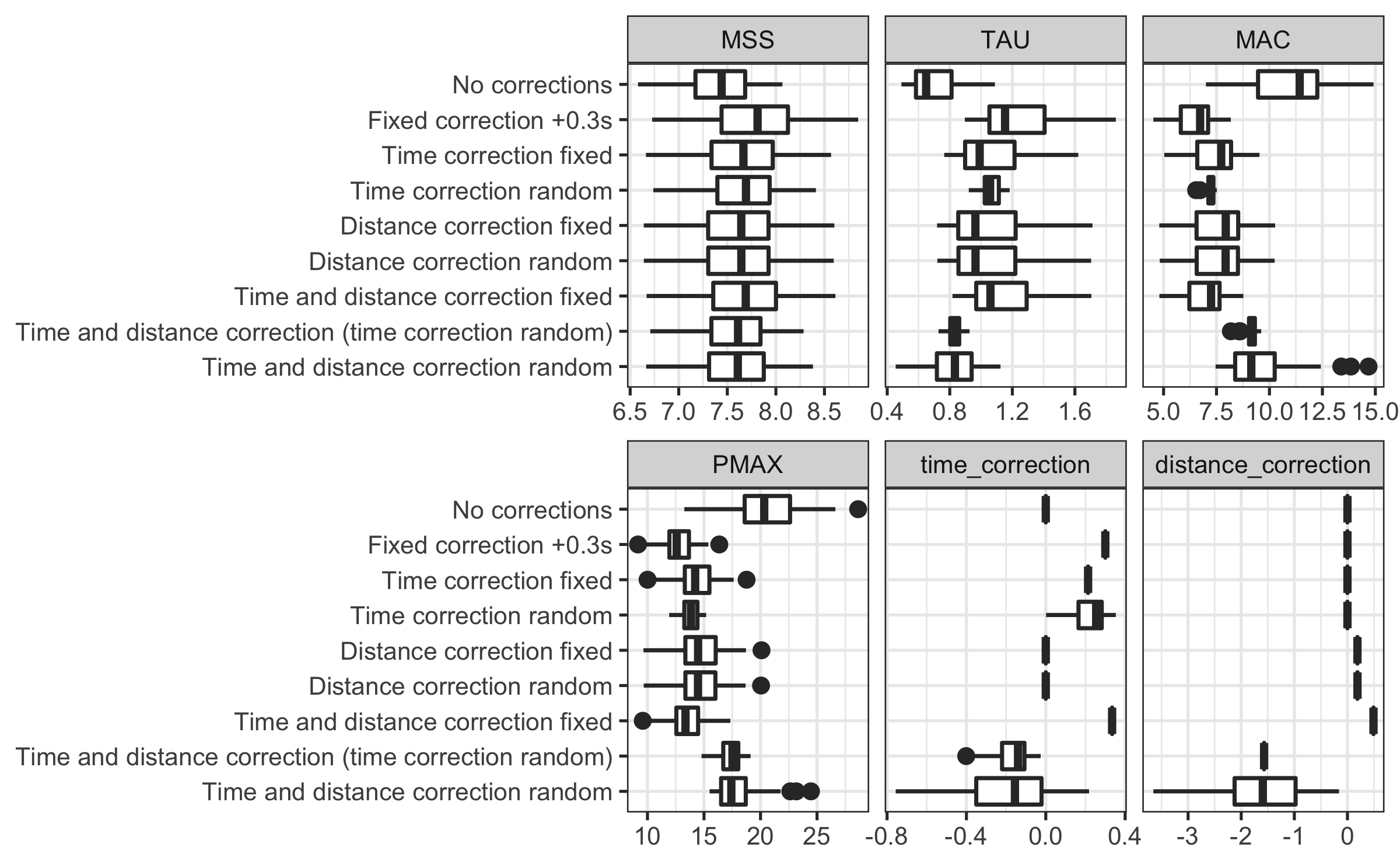
data("vescovi")  
  
# Convert data to long  
df <- vescovi %>%  
 select(1:13) %>%  
 # slice(1:10) %>%  
 pivot\_longer(  
 cols = 9:13,  
 names\_to = "distance",  
 values\_to = "time"  
 ) %>%  
 mutate(  
 distance = as.numeric(str\_extract(distance, "^[0-9]+"))  
 )

The following models were used: (1) no corrections model, (2) fixed time correction model (using +0.3s heuristic rule of thumb), (3) time correction model with time correction as fixed effect, (4) time correction model with time correction as random effect, (5) distance correction model with distance correction as fixed effect, (6) distance correction model with distance correction as random effect, (7) time and distance correction model, (8) time and distance correction model with distance correction as fixed effect (and time correction as random effect), and (9) time and distance correction model with distance and time correction as random effects (please refer to the [Supplemental material](#supplemental-material) for the R code).

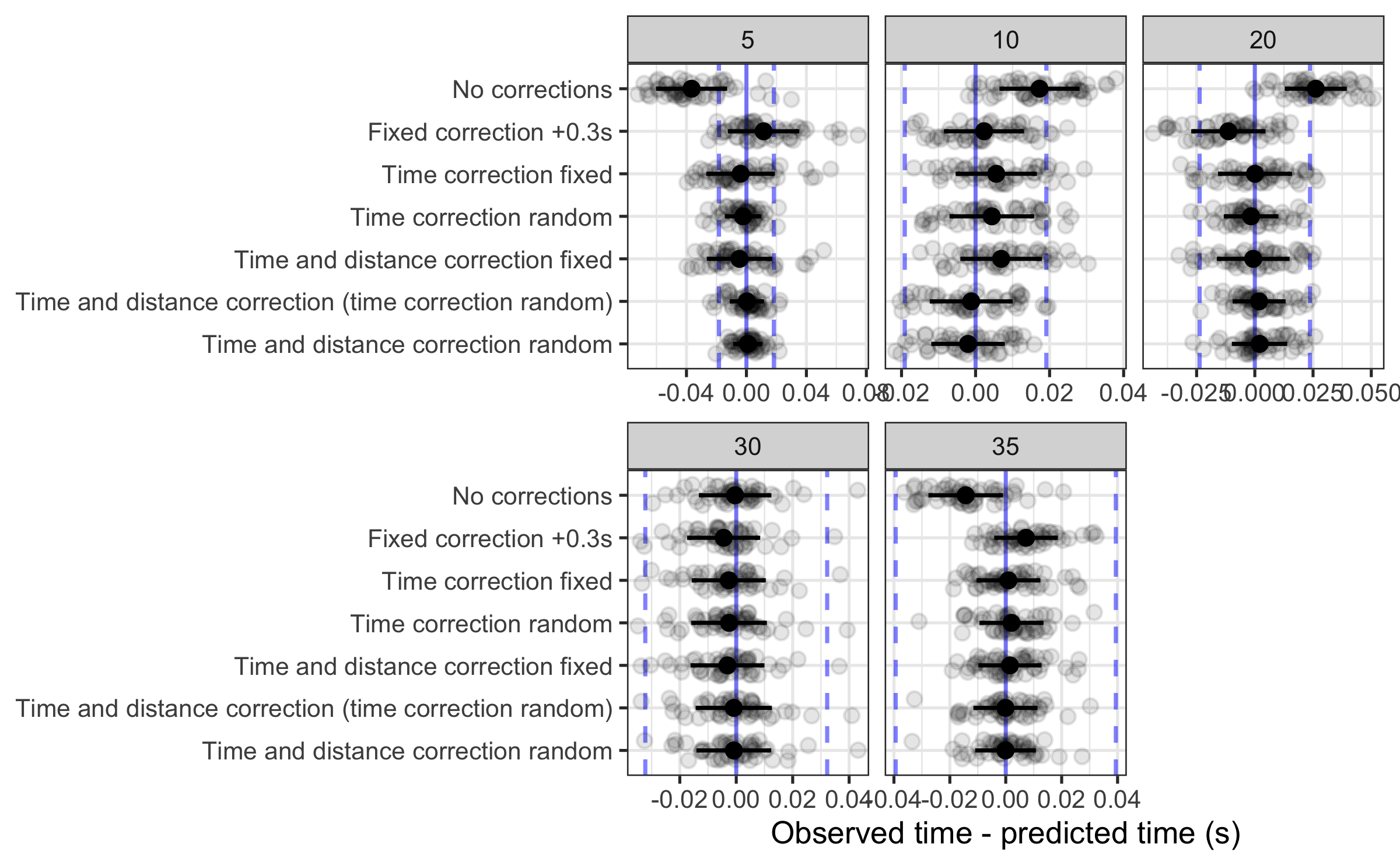
The following image represents model fit estimator RSE for each model. As can be seen, RSE is reduced the more flexible the model.



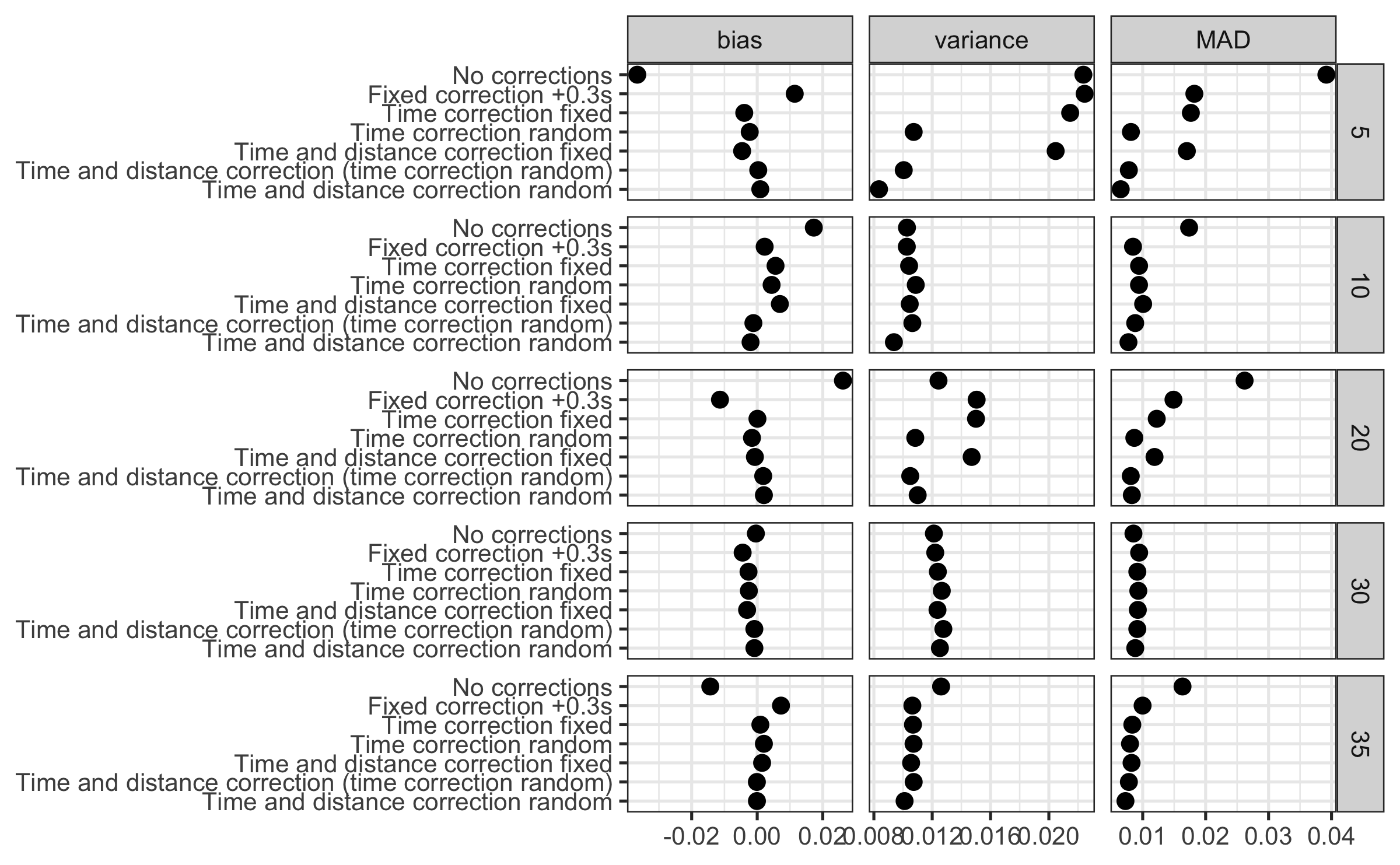
The following image depicts estimated parameters for each model:



The following image depicts model residuals across distance splits. To provide practical magnitude of the residuals, we have used between subject observed time SD multiplied with 0.2 and -0.2. This provides practical anchor for the residual magnitude, often referred to as *smallest worthwhile change* (SWC) or *smallest effect size of interest* (SESOI) ([Jovanović 2020](#X87b6e3e0ab837a9dc3f187ffd4719394205cfac)). If the residuals are within this magnitude band, then the model is good in making practically useful predictions. Error bars represent residual bias 1 SD.



The following figure depicts model residuals estimators (bias, or mean residual; variance, or SD of the residuals, and MAD, or mean absolute difference).



Which model should be used? Although providing a better fit (using RSE as an estimator of model fit), the time and distance correction models often estimate these parameters that are harder to interpret (e.g., negative distance correction). Although providing novel theoretical models in this paper, we acknowledge the need for validating them in practice, against gold-standard methods, assessing their agreement, as well as their power in detecting and adjusting for timing inconsistencies. More thorough theoretical simulation study is currently in development with the aim to explore the behavior of these models under different scenarios.

We are hoping that the **shorts** package will help fellow sports scientists and coaches in exploring short sprint profiles and help in driving research, particularly in devising measuring protocols that are sensitive enough to capture training intervention changes, but also robust enough to take into account potential sprint initiation and timing inconsistencies.

# Supplemental material

The *R Markdown* ([Xie 2021a](#ref-R-bookdown); [Allaire et al. 2021](#ref-R-rmarkdown); [Xie, Allaire, and Grolemund 2018](#ref-rmarkdown2018); [Xie, Dervieux, and Riederer 2020](#ref-rmarkdown2020)) source code for the paper can be found on the GitHub repository: <https://github.com/mladenjovanovic/shorts-paper>.

# References

Allaire, JJ, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, Winston Chang, and Richard Iannone. 2021. *Rmarkdown: Dynamic Documents for r*. <https://CRAN.R-project.org/package=rmarkdown>.

Altmann, Stefan, Marian Hoffmann, Gunther Kurz, Rainer Neumann, Alexander Woll, and Sascha Haertel. 2015. “Different Starting Distances Affect 5-m Sprint Times.” *Journal of Strength and Conditioning Research* 29 (8): 2361–66. <https://doi.org/10.1519/JSC.0000000000000865>.

Altmann, Stefan, Max Spielmann, Florian Azad Engel, Rainer Neumann, Steffen Ringhof, Doris Oriwol, and Sascha Haertel. 2017. “Validity of Single-Beam Timing Lights at Different Heights.” *Journal of Strength and Conditioning Research* 31 (7): 1994–99. <https://doi.org/10.1519/JSC.0000000000001889>.

Altmann, Stefan, Max Spielmann, Florian Azad Engel, Steffen Ringhof, Doris Oriwol, Sascha Härtel, and Rainer Neumann. 2018. “Accuracy of Single Beam Timing Lights for Determining Velocities in a Flying 20-m Sprint: Does Timing Light Height Matter?” *Journal of Human Sport and Exercise* 13 (3). <https://doi.org/10.14198/jhse.2018.133.10>.

Arsac, Laurent M., and Elio Locatelli. 2002. “Modeling the Energetics of 100-m Running by Using Speed Curves of World Champions.” *Journal of Applied Physiology* 92 (5): 1781–88. <https://doi.org/10.1152/japplphysiol.00754.2001>.

Bates, D. M., and J. M. Chambers. 1992. “Nonlinear Models.” In *Statistical Models in S*. Wadsworth & Brooks/Cole.

Bates, Douglas M., and Donald G. Watts. 2007. *Nonlinear Regression Analysis and Its Applications*. Wiley Series in Probability and Mathematical Statistics. New York, NY: Wiley.

Brown, Todd D., Jason D. Vescovi, and Jaci L. Vanheest. 2004. “Assessment of Linear Sprinting Performance: A Theoretical Paradigm.” *Journal of Sports Science & Medicine* 3 (4): 203–10.

Buchheit, Martin, Pierre Samozino, Jonathan Alexander Glynn, Ben Simpson Michael, Hani Al Haddad, Alberto Mendez-Villanueva, and Jean Benoit Morin. 2014. “Mechanical Determinants of Acceleration and Maximal Sprinting Speed in Highly Trained Young Soccer Players.” *Journal of Sports Sciences* 32 (20): 1906–13. <https://doi.org/10.1080/02640414.2014.965191>.

Clark, Kenneth P., Randall H. Rieger, Richard F. Bruno, and David J. Stearne. 2017. “The NFL Combine 40-Yard Dash: How Important Is Maximum Velocity?” *Journal of Strength and Conditioning Research*, June, 1. <https://doi.org/10.1519/JSC.0000000000002081>.

Edwards, Toby, Benjamin Piggott, Harry G. Banyard, G. Gregory Haff, and Christopher Joyce. 2020. “Sprint Acceleration Characteristics Across the Australian Football Participation Pathway.” *Sports Biomechanics*, August, 1–13. <https://doi.org/10.1080/14763141.2020.1790641>.

Furusawa, K., Archibald Vivian Hill, and J. L. Parkinson. 1927. “The Dynamics of "Sprint" Running.” *Proceedings of the Royal Society of London. Series B, Containing Papers of a Biological Character* 102 (713): 29–42. <https://doi.org/10.1098/rspb.1927.0035>.

Gelman, Andrew, Jennifer Hill, and Aki Vehtari. 2020. *Regression and Other Stories*. S.l.: Cambridge University Press.

Goerg, Georg M. 2020. *LambertW: Probabilistic Models to Analyze and Gaussianize Heavy-Tailed, Skewed Data*. <https://CRAN.R-project.org/package=LambertW>.

Haugen, Thomas A., Felix Breitschädel, and Pierre Samozino. 2020. “Power-Force-Velocity Profiling of Sprinting Athletes: Methodological and Practical Considerations When Using Timing Gates.” *Journal of Strength and Conditioning Research* 34 (6): 1769–73. <https://doi.org/10.1519/JSC.0000000000002890>.

Haugen, Thomas A., Felix Breitschädel, and Stephen Seiler. 2019. “Sprint Mechanical Variables in Elite Athletes: Are Force-Velocity Profiles Sport Specific or Individual?” Edited by Leonardo A. Peyré-Tartaruga. *PLOS ONE* 14 (7): e0215551. <https://doi.org/10.1371/journal.pone.0215551>.

———. 2020. “Sprint Mechanical Properties in Soccer Players According to Playing Standard, Position, Age and Sex.” *Journal of Sports Sciences* 38 (9): 1070–76. <https://doi.org/10.1080/02640414.2020.1741955>.

Haugen, Thomas A, Espen Tønnessen, and Stephen K Seiler. 2012. “The Difference Is in the Start: Impact of Timing and Start Procedure on Sprint Running Performance:” *Journal of Strength and Conditioning Research* 26 (2): 473–79. <https://doi.org/10.1519/JSC.0b013e318226030b>.

Haugen, Thomas, and Martin Buchheit. 2016. “Sprint Running Performance Monitoring: Methodological and Practical Considerations.” *Sports Medicine* 46 (5): 641–56. <https://doi.org/10.1007/s40279-015-0446-0>.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2017. *An Introduction to Statistical Learning: With Applications in R*. 1st ed. 2013, Corr. 7th printing 2017 edition. New York: Springer.

Jiménez-Reyes, Pedro, Pierre Samozino, Amador García-Ramos, Víctor Cuadrado-Peñafiel, Matt Brughelli, and Jean-Benoît Morin. 2018. “Relationship Between Vertical and Horizontal Force-Velocity-Power Profiles in Various Sports and Levels of Practice.” *PeerJ* 6 (November): e5937. <https://doi.org/10.7717/peerj.5937>.

Jovanović, Mladen. 2020. *Bmbstats: Bootstrap Magnitude-Based Statistics for Sports Scientists*. Mladen Jovanović.

———. 2021. *Shorts: Short Sprints*. <https://mladenjovanovic.github.io/shorts/>.

Kuhn, Max, and Kjell Johnson. 2018. *Applied Predictive Modeling*. 1st ed. 2013, Corr. 2nd printing 2016 edition. New York: Springer.

Mangine, Gerald T., Jay R. Hoffman, Adam M. Gonzalez, Adam J. Wells, Jeremy R. Townsend, Adam R. Jajtner, William P. McCormack, et al. 2014. “Speed, Force, and Power Values Produced From Nonmotorized Treadmill Test Are Related to Sprinting Performance:” *Journal of Strength and Conditioning Research* 28 (7): 1812–19. <https://doi.org/10.1519/JSC.0000000000000316>.

Marcote-Pequeño, Ramón, Amador García-Ramos, Víctor Cuadrado-Peñafiel, Jorge M. González-Hernández, Miguel Ángel Gómez, and Pedro Jiménez-Reyes. 2019. “Association Between the Force and Performance Variables Obtained in Jumping and Sprinting in Elite Female Soccer Players.” *International Journal of Sports Physiology and Performance* 14 (2): 209–15. <https://doi.org/10.1123/ijspp.2018-0233>.

Morin, J. B. 2017. “A Spreadsheet for Sprint Acceleration Force-Velocity-Power Profiling.” JB Morin, PhD - Sport Science. December 13, 2017. <https://jbmorin.net/2017/12/13/a-spreadsheet-for-sprint-acceleration-force-velocity-power-profiling/>.

Morin, Jean-Benoit, and Pierre Samozino. 2016. “Interpreting Power-Force-Velocity Profiles for Individualized and Specific Training.” *International Journal of Sports Physiology and Performance* 11 (2): 267–72. <https://doi.org/10.1123/ijspp.2015-0638>.

———. 2019. “Spreadsheet for Sprint Acceleration Force-Velocity-Power Profiling.”

Morin, Jean-Benoit, Pierre Samozino, Munenori Murata, Matt R Cross, and Ryu Nagahara. 2019. “A Simple Method for Computing Sprint Acceleration Kinetics from Running Velocity Data: Replication Study with Improved Design.” *Journal of Biomechanics* 94 (September): 82–87. <https://doi.org/10.1016/j.jbiomech.2019.07.020>.

Motulsky, Harvey. 2018. *Intuitive Biostatistics: A Nonmathematical Guide to Statistical Thinking*. Fourth edition. New York: Oxford University Press.

Pinheiro, José, Douglas Bates, and R-core. 2021. *Nlme: Linear and Nonlinear Mixed Effects Models*. <https://svn.r-project.org/R-packages/trunk/nlme/>.

R Core Team. 2021. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.

Samozino, P., G. Rabita, S. Dorel, J. Slawinski, N. Peyrot, E. Saez de Villarreal, and J.-B. Morin. 2016. “A Simple Method for Measuring Power, Force, Velocity Properties, and Mechanical Effectiveness in Sprint Running: Simple Method to Compute Sprint Mechanics.” *Scandinavian Journal of Medicine & Science in Sports* 26 (6): 648–58. <https://doi.org/10.1111/sms.12490>.

Stenroth, Lauri, and Paavo Vartiainen. 2020. “Spreadsheet for Sprint Acceleration Force-Velocity-Power Profiling with Optimization to Correct Start Time.” <https://doi.org/10.13140/RG.2.2.12841.83045>.

Stenroth, Lauri, Paavo Vartiainen, and Pasi A. Karjalainen. 2020. “Force-Velocity Profiling in Ice Hockey Skating: Reliability and Validity of a Simple, Low-Cost Field Method.” *Sports Biomechanics*, June, 1–16. <https://doi.org/10.1080/14763141.2020.1770321>.

van Ingen Schenau, Gerrit Jan, Ron Jacobs, and Jos J. de Koning. 1991. “Can Cycle Power Predict Sprint Running Performance?” *European Journal of Applied Physiology and Occupational Physiology* 63 (3-4): 255–60. <https://doi.org/10.1007/BF00233857>.

Vescovi, Jason D. 2012. “Sprint Speed Characteristics of High-Level American Female Soccer Players: Female Athletes in Motion (FAiM) Study.” *Journal of Science and Medicine in Sport* 15 (5): 474–78. <https://doi.org/10.1016/j.jsams.2012.03.006>.

———. 2014. “Impact of Maximum Speed on Sprint Performance During High-Level Youth Female Field Hockey Matches: Female Athletes in Motion (FAiM) Study.” *International Journal of Sports Physiology and Performance* 9 (4): 621–26. <https://doi.org/10.1123/ijspp.2013-0263>.

———. 2016. “Locomotor, Heart-Rate, and Metabolic Power Characteristics of Youth Women’s Field Hockey: Female Athletes in Motion (FAiM) Study.” *Research Quarterly for Exercise and Sport* 87 (1): 68–77. <https://doi.org/10.1080/02701367.2015.1124972>.

Vescovi, Jason D., and Mladen Jovanović. 2021. “Sprint Mechanical Characteristics of Female Soccer Players: A Retrospective Pilot Study to Examine a Novel Approach for Correction of Timing Gate Starts.” *Frontiers in Sports and Active Living* 3 (May): 629694. <https://doi.org/10.3389/fspor.2021.629694>.

Ward-Smith, A. J. 2001. “Energy Conversion Strategies During 100 m Sprinting.” *Journal of Sports Sciences* 19 (9): 701–10. <https://doi.org/10.1080/02640410152475838>.

Wickham, Hadley. 2021a. *Tidyr: Tidy Messy Data*. <https://CRAN.R-project.org/package=tidyr>.

———. 2021b. *Tidyverse: Easily Install and Load the Tidyverse*. <https://CRAN.R-project.org/package=tidyverse>.

Wickham, Hadley, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. 2021. *Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. <https://CRAN.R-project.org/package=ggplot2>.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2021. *Dplyr: A Grammar of Data Manipulation*. <https://CRAN.R-project.org/package=dplyr>.

Xie, Yihui. 2014. “Knitr: A Comprehensive Tool for Reproducible Research in R.” In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC. <http://www.crcpress.com/product/isbn/9781466561595>.

———. 2015. *Dynamic Documents with R and Knitr*. 2nd ed. Boca Raton, Florida: Chapman; Hall/CRC. <https://yihui.org/knitr/>.

———. 2021a. *Bookdown: Authoring Books and Technical Documents with r Markdown*. <https://CRAN.R-project.org/package=bookdown>.

———. 2021b. *Knitr: A General-Purpose Package for Dynamic Report Generation in r*. <https://yihui.org/knitr/>.

Xie, Yihui, J. J. Allaire, and Garrett Grolemund. 2018. *R Markdown: The Definitive Guide*. Boca Raton, Florida: Chapman; Hall/CRC. <https://bookdown.org/yihui/rmarkdown>.

Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. *R Markdown Cookbook*. Boca Raton, Florida: Chapman; Hall/CRC. <https://bookdown.org/yihui/rmarkdown-cookbook>.

Zhu, Hao. 2021. *kableExtra: Construct Complex Table with Kable and Pipe Syntax*. <https://CRAN.R-project.org/package=kableExtra>.

1. Faculty of Sport and Physical Education, University of Belgrade, Serbia, [coach.mladen.jovanovic@gmail.com](mailto:coach.mladen.jovanovic@gmail.com) [↑](#footnote-ref-20)
2. Faculty of Kinesiology and Physical Education, Graduate School of Exercise Science, Toronto, ON Canada, [vescovij@gmail.com](mailto:vescovij@gmail.com) [↑](#footnote-ref-22)