PLAY/LEGO report

Rick Gilmore

2018-10-14 18:51:27

# PLAY pilot summary data

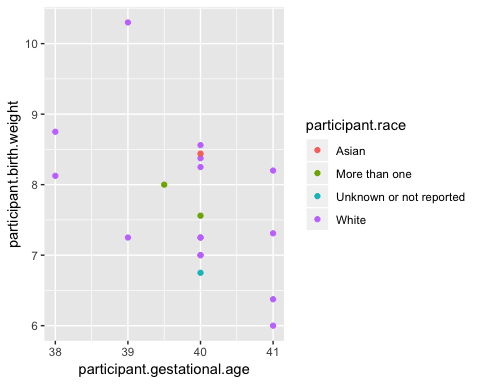
<https://nyu.databrary.org/volume/444>

<https://nyu.databrary.org/volume/254>

## Demographic summary

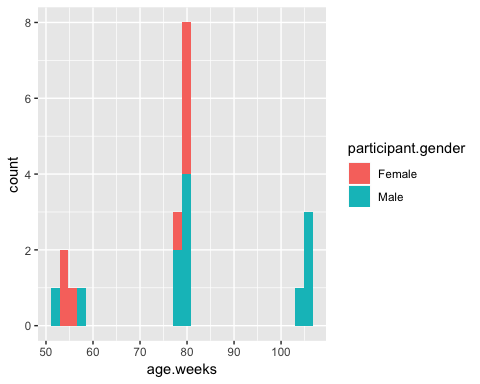
pilot\_demog <- summarize\_demog(vol.id = pilot.vol.id, return.df = TRUE)  
  
pilot\_demog %>%   
 ggplot(.) +  
 aes(x = participant.gestational.age, y = participant.birth.weight, color = participant.race) +  
 geom\_point()

## Warning: Removed 1 rows containing missing values (geom\_point).



pilot\_demog %>%  
 ggplot(.) +  
 aes(x = age.weeks, fill = participant.gender) +  
 geom\_histogram()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



## Parent-report variables

survey.session.id <- 26295  
list\_assets\_in\_session(vol.id = pilot.vol.id, session.id = survey.session.id) %>%  
 select(asset.name, asset.type, asset.id)

## asset.name asset.type asset.id  
## 1 locomotion Comma-separated values 117092  
## 2 child-birth Comma-separated values 116791  
## 3 family Comma-separated values 116790  
## 4 sleep Comma-separated values 116789  
## 5 childcare Comma-separated values 116788  
## 6 language-exposure Comma-separated values 116787

(csv\_fl <- list.files('csv', '\\.csv$', full.names = TRUE))

## [1] "csv/databrary444-Adolph-Tamis-LeMonda-Gilmore-PLAY-pilot-data-materials-26295-Parent\_report\_data-116787-language-exposure.csv"  
## [2] "csv/databrary444-Adolph-Tamis-LeMonda-Gilmore-PLAY-pilot-data-materials-26295-Parent\_report\_data-116790-family.csv"   
## [3] "csv/databrary444-Adolph-Tamis-LeMonda-Gilmore-PLAY-pilot-data-materials-26295-Parent\_report\_data-117092-locomotion.csv"

lang\_exp <- read\_csv(csv\_fl[1])

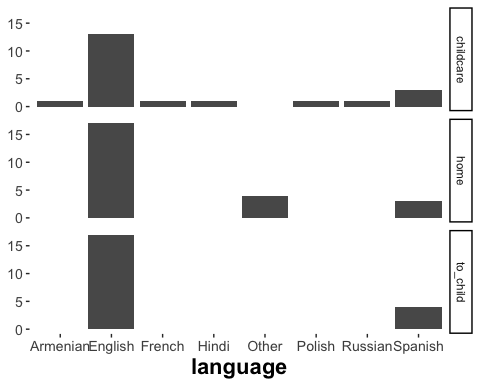
## Parsed with column specification:  
## cols(  
## id = col\_integer(),  
## language = col\_character(),  
## exposure\_context = col\_character()  
## )

names(lang\_exp)

## [1] "id" "language" "exposure\_context"

play.palette <- scale\_fill\_manual(values=c("blue2", "firebrick2", "chartreuse2", "darkorchid2"))  
  
play.theme <-   
 theme\_classic() +  
 theme(legend.position = "bottom",  
 legend.title = element\_blank(),  
 legend.text = element\_text(size = rel(1.2)),  
 axis.title = element\_text(size = rel(1.5), face ="bold"),  
 axis.text.x = element\_text(size = rel(1.2)),  
 axis.text.y = element\_text(size = rel(1.2)),  
 axis.line = element\_blank(),  
 axis.title.y = element\_blank())  
  
lang\_exp %>%  
 ggplot(.) +  
 aes(x = language) +  
 facet\_grid(exposure\_context ~ .) +  
 geom\_histogram(stat='count') +  
 play.theme

## Warning: Ignoring unknown parameters: binwidth, bins, pad



loco\_onset <- read\_csv(csv\_fl[3])

## Parsed with column specification:  
## cols(  
## id = col\_integer(),  
## hkcrawl\_onset\_date = col\_character(),  
## hkcrawl\_onset\_mos = col\_double(),  
## walk\_onset\_date = col\_character(),  
## walk\_onset\_mos = col\_double(),  
## walk\_onset\_src = col\_character(),  
## `Interview Comments` = col\_character()  
## )

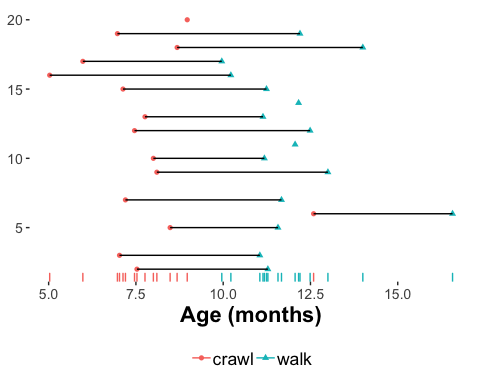
names(loco\_onset)

## [1] "id" "hkcrawl\_onset\_date" "hkcrawl\_onset\_mos"   
## [4] "walk\_onset\_date" "walk\_onset\_mos" "walk\_onset\_src"   
## [7] "Interview Comments"

loco\_onset %>%  
 gather(key = milestone, value = age.mos, hkcrawl\_onset\_mos, walk\_onset\_mos) %>%  
 mutate(milestone = factor(milestone, labels = c("crawl", "walk"))) %>%  
 # arrange(desc(age.mos)) %>%  
 ggplot() +  
 aes(x = age.mos, y = id) +  
 geom\_point(aes(shape = milestone, color = milestone)) +  
 geom\_line(aes(group = id)) +  
 xlab("Age (months)") +  
 geom\_rug(aes(x = age.mos, group = milestone, color = milestone),  
 sides = "b") +  
 play.theme

## Warning: Removed 3 rows containing missing values (geom\_point).

## Warning: Removed 3 rows containing missing values (geom\_path).



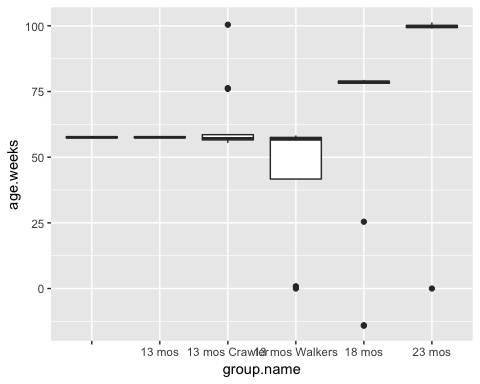
# LEGO

<https://nyu.databrary.org/volume/563>

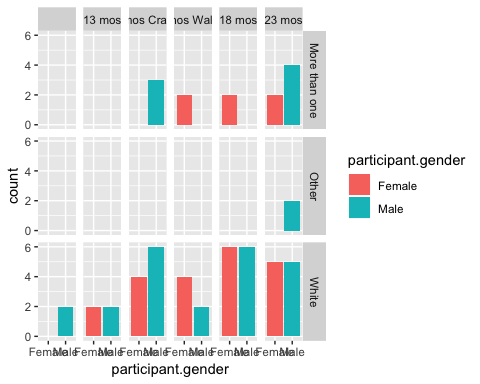
lego\_demog <- databraryapi::summarize\_demog(vol.id = lego.vol.id, return.df = TRUE)

## Warning in `[<-.factor`(`\*tmp\*`, unreported, value = "Unknown or not  
## reported"): invalid factor level, NA generated

lego\_demog %>%  
 ggplot(.) +  
 aes(y = age.weeks, x = group.name) +  
 geom\_boxplot()



lego\_demog %>%  
 ggplot(.) +  
 aes(participant.gender, fill = participant.gender) +  
 facet\_grid(participant.race ~ group.name) +  
 geom\_bar()



# Individual-specific reports from PLAY

Let’s focus on [NYU\_019 in volume 254](https://nyu.databrary.org/volume/254/slot/14514/-)

## Videos

### Child view

<https://nyu.databrary.org/slot/14514/69908000,73576800/asset/61054/download?inline=true>

### Experimenter view

<https://nyu.databrary.org/slot/14514/69908000,73623840/asset/61086/download?inline=true>

## Assets in session

vol\_254\_assets <- list\_assets\_in\_session(254, 14514)  
vol\_254\_assets %>%  
 select(., asset.name, asset.type, asset.id)

## asset.name  
## 1 NYU-019-surveys-demographics  
## 2 NYU019  
## 3 NYU019\_Decibel Meter Data  
## 4 NYU019\_Phone Questionnaire  
## 5 NYU019\_1-Hour Free Play  
## 6 NYU019\_1-Hour Free Play (View of Experimenter)  
## 7 NYU019\_Solitary Play  
## 8 NYU019\_Dyadic Play  
## 9 NYU019\_Solitary Play & Dyadic Play (View of Experimenter)  
## 10 NYU019\_At-Home Questionnaires (View of Experimenter)  
## 11 NYU019\_House Walkthrough  
## 12 NYU019\_House Walkthrough (View of Experimenter)  
## asset.type asset.id  
## 1 Comma-separated values 87833  
## 2 Datavyu 63635  
## 3 Comma-separated values 63545  
## 4 MPEG-4 video 61427  
## 5 MPEG-4 video 61054  
## 6 MPEG-4 video 61086  
## 7 MPEG-4 video 61052  
## 8 MPEG-4 video 61050  
## 9 MPEG-4 video 61060  
## 10 MPEG-4 video 61088  
## 11 MPEG-4 video 61048  
## 12 MPEG-4 video 61062

## Process Datavyu

### Download

unlink('tmp', recursive = TRUE)  
dv\_info <- filter(vol\_254\_assets, asset.type == "Datavyu")  
dv\_dir <- download\_datavyu(vol.id = 254, session.id = 14514, asset.id = dv\_info$asset.id)

## Creating directory tmp/

## Successful HTML GET query.

## Content-type is application/vnd.datavyu

## File name unspecified. Generating unique name.

## Downloading Datavyu file as:   
## tmp/254-14514-63635-2018-10-14-1851-35.opf

### Extract codes and code definitions

Now, let’s extract the components of the Datavyu file, including the code definitions.

extract\_dv(in.dir = dv\_dir, auto.write.over = TRUE)

## [1] "tmp"

dv\_to\_csv(dv.dir = dv\_dir)  
list.files('tmp')

## [1] "254-14514-63635-2018-10-14-1851-35.csv"  
## [2] "254-14514-63635-2018-10-14-1851-35.opf"  
## [3] "db"   
## [4] "project"

Next, we can load the Datavyu file as a csv.

dv\_fn <- list.files('tmp', '\\.csv$', full.names = TRUE)  
vol\_254\_14514 <- read\_csv(dv\_fn)

## Parsed with column specification:  
## cols(  
## code = col\_character(),  
## onset = col\_character(),  
## offset = col\_character(),  
## code.value = col\_character()  
## )

names(vol\_254\_14514)

## [1] "code" "onset" "offset" "code.value"

vol\_254\_14514

## # A tibble: 6,145 x 4  
## code onset offset code.value   
## <chr> <chr> <chr> <chr>   
## 1 transcribe 00:00:03:460 00:00:03:960 m,hi   
## 2 transcribe 00:00:06:824 00:00:07:324 m,you have to keep that on   
## 3 transcribe 00:00:07:924 00:00:08:424 m,ok?   
## 4 transcribe 00:00:09:738 00:00:10:238 m,you wanna play with mickey?  
## 5 transcribe 00:00:18:959 00:00:19:459 b,what   
## 6 transcribe 00:00:20:277 00:00:20:777 b,b   
## 7 transcribe 00:00:22:709 00:00:23:209 b,b   
## 8 transcribe 00:00:23:718 00:00:24:218 b,b   
## 9 transcribe 00:00:24:681 00:00:25:181 m,thats not mickey   
## 10 transcribe 00:00:25:905 00:00:26:405 m,wheres mickey?   
## # ... with 6,135 more rows

### Extract code definitions

extract\_dv\_code\_defs(in.dir = 'tmp')

## [1] "tmp/254-14514-63635-2018-10-14-1851-35-code-defs.csv"

code\_fn <- list.files('tmp', '-code-defs\\.csv$', full.names = TRUE)  
vol\_254\_14514\_codes <- read\_csv(code\_fn)

## Parsed with column specification:  
## cols(  
## code = col\_character(),  
## code\_vals = col\_character(),  
## code\_type = col\_character()  
## )

vol\_254\_14514\_codes

## # A tibble: 21 x 3  
## code code\_vals code\_type  
## <chr> <chr> <chr>   
## 1 transcribe source NOMINAL   
## 2 momspeech content NOMINAL   
## 3 momutterancetype <NA> NOMINAL   
## 4 momutterancetype\_rel imperativelap NOMINAL   
## 5 babyvoc content NOMINAL   
## 6 babyutterancetype <NA> NOMINAL   
## 7 babyutterancetype\_rel languagesw NOMINAL   
## 8 id site NOMINAL   
## 9 reliability\_blocks x NOMINAL   
## 10 babyobject o NOMINAL   
## # ... with 11 more rows

### Visualize clips from specific codes

Let’s look at the baby’s locomotion (babyloc) code.

(babyloc <- filter(vol\_254\_14514, code == 'babyloc'))

## # A tibble: 418 x 4  
## code onset offset code.value  
## <chr> <chr> <chr> <chr>   
## 1 babyloc 00:00:09:976 00:00:11:118 l   
## 2 babyloc 00:00:12:342 00:00:15:368 l   
## 3 babyloc 00:00:21:181 00:00:22:304 l   
## 4 babyloc 00:00:23:052 00:00:23:760 l   
## 5 babyloc 00:00:25:636 00:00:29:240 l   
## 6 babyloc 00:00:29:920 00:00:33:007 l   
## 7 babyloc 00:00:36:006 00:00:37:433 l   
## 8 babyloc 00:00:49:844 00:00:50:922 l   
## 9 babyloc 00:01:42:646 00:01:45:570 l   
## 10 babyloc 00:01:48:800 00:01:53:017 l   
## # ... with 408 more rows

# Surrounding assignment w/ parentheses also prints out the assigned value.)

Let’s look at the code values.

unique(babyloc$code.value)

## [1] "l" "h" "."

Focus on holds h because there are fewer of them.

# Select segments for this code  
(babyloc\_segs <- dplyr::filter(babyloc, code.value == 'h'))

## # A tibble: 6 x 4  
## code onset offset code.value  
## <chr> <chr> <chr> <chr>   
## 1 babyloc 00:25:00:031 00:25:00:535 h   
## 2 babyloc 00:32:45:879 00:32:47:206 h   
## 3 babyloc 00:35:01:302 00:35:09:734 h   
## 4 babyloc 00:35:36:288 00:36:24:156 h   
## 5 babyloc 00:40:06:826 00:41:43:793 h   
## 6 babyloc 00:42:15:958 00:42:31:326 h

### Download video segment corresponding to one or more of these codes

Let’s pick the 1st segment just for fun.

# What is the segment range for the video?  
this\_seg <- 1  
seg\_range <- get\_asset\_segment\_range(vol.id = 254, session.id = 14514, asset.id = 61054)  
  
onset\_ms <- HHMMSSmmm\_to\_ms(babyloc\_segs$onset[this\_seg]) + seg\_range[1]  
offset\_ms <- HHMMSSmmm\_to\_ms(babyloc\_segs$offset[this\_seg]) + seg\_range[1]  
  
# babyloc$onset\_ms <- lapply(babyloc$onset, HHMMSSmmm\_to\_ms) + seg\_range[1]  
# babyloc$offset\_ms <- lapply(babyloc$offset, HHMMSSmmm\_to\_ms) + seg\_range[1]  
# babyloc  
# Add cols to babyloc\_segs  
# babyloc\_segs <- babyloc\_segs %>%  
# mutate(., onset\_ms = HHMMSSmmm\_to\_ms(babyloc\_segs$onset) + seg\_range[1],  
# offset\_ms = HHMMSSmmm\_to\_ms(babyloc\_segs$onset) + seg\_range[1])  
  
selected\_seg <- paste0(onset\_ms, ",", offset\_ms)  
message(paste0('Picking segment ', selected\_seg, '.'))

## Picking segment 71408031,71408535.

Now download the video.

download\_video(254, 14514, 61054, out.dir = 'tmp', segment.id = selected\_seg)

## [1] "tmp"

And see if we can display video or if needed, [go to it on Databrary](https://nyu.databrary.org/slot/14514/71873879,71875206/asset/61054/download?inline=true).

v <- list.files('tmp', '\\.mp4$', full.names = TRUE)  
render\_video\_in\_html(v[1], params = ' controls width=600px')

Your browser does not support the video tag.

### Another segment

this\_seg <- 2  
onset\_ms <- HHMMSSmmm\_to\_ms(babyloc\_segs$onset[this\_seg]) + seg\_range[1]  
offset\_ms <- HHMMSSmmm\_to\_ms(babyloc\_segs$offset[this\_seg]) + seg\_range[1]  
selected\_seg <- paste0(onset\_ms, ",", offset\_ms)  
download\_video(254, 14514, 61054, out.dir = 'tmp', segment.id = selected\_seg)

[1] “tmp”

v <- list.files('tmp', '\\.mp4$', full.names = TRUE)  
render\_video\_in\_html(v[this\_seg], params = ' controls width=600px')

Your browser does not support the video tag.

uri\_for\_seg\_slot\_asset(selected\_seg, 14514, 61054)

<https://nyu.databrary.org/slot/14514/71873879,71875206/asset/61054/download?inline=true>