Refit with Saved Parameters

Overview

This vignette walks through using a text file of previously fit model parameters to use in the `bdotsRefit` function. This is convenient if you have already gone through the refitting process and would like to save/load the refitted parameters in a new session.

To demonstate this process, we start with fitting a set of curves to our data

```
library(bdots)

fit <- bdotsFit(data = cohort_unrelated,
                subject = "Subject",
                time = "Time",
                y = "Fixations",
                group = c("Group", "LookType"),
                curveType = doubleGauss(concave = TRUE),
                cor = TRUE,
                numRefits = 2,
                cores = 2,
                verbose = FALSE)

refit <- bdotsRefit(fit, quickRefit = TRUE, fitCode = 5)
```

From this, we can create an appropriate `data.table` that can be used in a later session

```
parDT <- coefWriteout(refit)
head(parDT)
#>    Subject Group LookType     mu      ht   sig1   sig2      base1    base2
#> 1:       1    50   Cohort 429.76 0.19860 159.89 314.64  0.0097098 0.033761
#> 2:       1    65   Cohort 634.93 0.26350 303.81 215.38 -0.0206361 0.028924
#> 3:       2    50   Cohort 647.07 0.25438 518.96 255.99 -0.2130875 0.013682
#> 4:       2    65   Cohort 723.05 0.25821 392.95 252.94 -0.0548262 0.031973
#> 5:       3    50   Cohort 501.48 0.22477 500.85 158.42 -0.3316790 0.025227
#> 6:       3    65   Cohort 460.72 0.30677 382.73 166.08 -0.2433086 0.039922
```

It's important that columns are included that match the unique identifying columns in our `bdotsObj`, and that the parameters match the coefficients used from `bdotsFit`

```
## Subject, Group, and LookType
head(refit)
#>    Subject Group LookType        fit      R2   AR1 fitCode
#> 1:       1    50   Cohort <gnls[18]> 0.96972  TRUE       0
#> 2:       1    65   Cohort <gnls[18]> 0.98049  TRUE       0
#> 3:       2    50   Cohort <gnls[18]> 0.98117  TRUE       0
#> 4:       2    65   Cohort <gnls[18]> 0.96975  TRUE       0
#> 5:       3    50   Cohort <gnls[18]> 0.97619  TRUE       0
#> 6:       3    65   Cohort <gnls[18]> 0.95349 FALSE       3

## doubleGauss pars
```

```
colnames(coef(refit))
#> [1] "mu"    "ht"    "sig1"  "sig2"  "base1" "base2"
```

We can save our parameter `data.table` for later use, or read in any other appropriately formatted `data.frame`

```
## Save this for later using data.table::fwrite
fwrite(parDT, file = "mypars.csv")
parDT <- fread("mypars.csv")
```

Once we have this, we can pass it as an argument to the `bdotsRefit` function. Doing so will ignore the remaining arguments

```
new_refit <- bdotsRefit(refit, paramDT = parDT)
```

We end up with a `bdotsObj` that matches what we had previously. As seeds have not yet been implemented, the resulting parameters may not be exact. It will, however, assist with not having to go through the entire refitting process again manually (although, there is always the option to save the entire object with `save(refit, file = "refit.RData))`

```
head(new_refit)
#>    Subject Group          LookType        fit      R2  AR1 fitCode
#> 1:       1    50            Cohort <gnls[18]> 0.96972 TRUE       0
#> 2:       1    50 Unrelated_Cohort <gnls[18]> 0.97900 TRUE       0
#> 3:       1    65            Cohort <gnls[18]> 0.98049 TRUE       0
#> 4:       1    65 Unrelated_Cohort <gnls[18]> 0.87164 TRUE       1
#> 5:       2    50            Cohort <gnls[18]> 0.98117 TRUE       0
#> 6:       2    50 Unrelated_Cohort <gnls[18]> 0.95612 TRUE       0
```