iMKT Pipeline

Brief intro about why we developed the package. The importance of detecting adaptation, and the many MKT derived methodologies. Take from paper.

Explain also that iMKT faces this variety of methodologies, and is able to compute several tests in order to detect adaptation in diverse situations.

In addition, it also provides a way to quantify negative selection, which may overlap with signals of positive selection in the target region, causing the failing of certain MK tests.

This vignette contains three different sections:

- Loading the package and checking test data

- Performing MKT derived methodologies calculation

- Conclusion: comparison of the different MK estimates

Loading the package and checking test data

First of all, install (if this is not done yet) and load the package. Notice that iMKT package includes two sample dataframes named myDafData and myDivergenceData which are the ones used in this tutorial and correspond to XXXXXXXX (Chr2L RAL and simulans??). Therefore, it is possible to replicate the vignettes in order to better understand all the package functionalities.

```
## Load package
# install.packages("devtools")
# devtools::install_github("sergihervas/iMKT")
library(iMKT)

## Sample daf data
head(myDafData)
#>      daf    Pi    P0
#> 1 0.025 22490 17189
#> 2 0.075  3217  4780
#> 3 0.125  1616  2874
#> 4 0.175   999  2088
#> 5 0.225   754  1685
#> 6 0.275   679  1443

## Sample divergence data
myDivergenceData
#>        mi    Di     m0    D0
#> 1 2598805 54641 620019 52537
```

The iMKT package includes several funcions, classified as follows:

- MKT derived methodologies calculation

    - standardMK(): Standard MKT

    - FWW(): FWW correction

- DGRP(): DGRP correction
- asymptoticMK(): Asymptotic MKT
- iMK(): integrative MKT
- completeMKT(): perform all previous tests
- iMK using PopFly and PopHuman data
    - loadPopFly(): load PopFlyData
    - loadPopHuman(): load PopHumanData
    - PopFlyAnalisys(): perform any test using PopFlyData
    - PopHumanAnalysis(): perform any test using PopHumanData
- Miscelanious
    - checkInput(): check data before performing analyses
    - themePublication(): output plots and tables styling

Each function has an associated help page with its description, details about its parameters, usage, examples and so on. Rembember you can access it writting ?? and the function name (or ??library::function) in your console (*example*: ??iMKT::standardMK).

This vignette focuses on the first category of functions: "MKT derived methodologies calculation". Specifically, it contains examples of each function using the sample data described above. For details about functions from "iMKT using PopFly and PopHuman data" category check the corresponding vignette. The functions from the third category are used within other functions and do not produce analyses output.

Performing MKT derived methodologies calculation

The diverse functions from this category have two common input parameters which are required to perform the corresponding test:

- daf: data frame containing DAF, Pi and P0 values (myDafData)
- divergence: data frame containing divergent and analyzed sites for selected (i) and neutral (0) classes (myDivergenceData)

The output of each function always contains the corresponding alpha estimate, together with specific values and details of the selected methodology.

Standard MKT

Brief theoretical description about MKT

The **standardMK()** function uses daf and divergence input parameters and returns as output a list containing:

- alpha symbol: estimate of alpha using the standard MKT
- Fisher exact test P-value: p-value obtained using the Fisher exact test on a 2x2 contingency table (MKT table)
- MKT table: table containing the number of polymorphic and divergent sites for neutral and selected classes.

- Divergence metrics: table containing estimates of Ka, Ks, omega, omegaA, omegaD.

```
standardMK(daf=myDafData, divergence=myDivergenceData)
#> $alpha.symbol
#> [1] 0.2364499
#>
#> $`Fishers exact test P-value`
#> [1] 1.480943e-183
#>
#> $`MKT table`
#>
#>
#>                    Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class            45101         52537
#> Selected class           35816         54641
#>
#> $`Divergence metrics`
#>
#>
#>         Ka          Ks        omega      omegaA      omegaD
#> ----------   ---------   ----------   ---------   ---------
#>  0.0210254   0.0847345   0.2481331   0.058671   0.189462
```

FWW correction

Alpha estimates can be biased by the segregation of slighlty deleterious substitutions. One method to partially controll this effect is to remove low frequency polymorphisms from the analysis, as proposed by Fay et al. (2001). Using this correction, all polymorphic sites from both neutral and selected classes which have a derived allele frequency lower than the pre-defined cutoff are removed for the calculation of alpha.

The **FWW()** function uses daf and divergence input parameters, along with a default list of cutoffs (0, 0.05, 0.1) and returns as output a list containing:

- Results: alpha estimates (and their associated Fisher exact test P-value) for each cutoff.
- Divergence metrics: global metrics (Ka, Ks, omega) and estimates by cutoff (omegaA, omegaD)
- MKT tables: tables containing the number of polymorphic and divergent sites for neutral and selected classes for each cutoff.

```
FWW(daf=myDafData, divergence=myDivergenceData)
#> $Results
#>                 alpha.symbol Fishers exact test P-value
#> Cutoff =  0        0.2364499             1.480943e-183
#> Cutoff =  0.05    0.5409548               0.000000e+00
#> Cutoff =  0.1     0.5798139               0.000000e+00
#>
#> $`Divergence metrics`
#> $`Divergence metrics`$`Global metrics`
#>          Ka        Ks      omega
#> 1 0.02102543 0.0847345 0.2481331
#>
```

```
#> $`Divergence metrics`$`Estimates by cutoff`
#>               omegaA.symbol omegaD.symbol
#> Cutoff =  0       0.05867104     0.1894620
#> Cutoff =  0.05    0.13422877     0.1139043
#> Cutoff =  0.1     0.14387102     0.1042621
#>
#>
#> $`MKT tables`
#> $`MKT tables`$`Cutoff =  0`
#>
#>
#> Table: cutoff
#>
#>                     Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class           45101         52537
#> Selected class          35816         54641
#>
#> $`MKT tables`$`Cutoff =  0.05`
#>
#>
#> Table: cutoff
#>
#>                     Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class           27912         52537
#> Selected class          13326         54641
#>
#> $`MKT tables`$`Cutoff =  0.1`
#>
#>
#> Table: cutoff
#>
#>                     Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class           23132         52537
#> Selected class          10109         54641
```

By default the argument **list_cutoff** uses a list of cutoffs with the following values: 0, 0.05, 0.1. Moreover, the function has an optional argument, **plot**, which is set to **FALSE** by default. This parameters can be customized, like in the following example, where we use a list of 4 cutoffs (0.05, 0.15, 0.25, 0.35) and set the plot argument to **TRUE**.
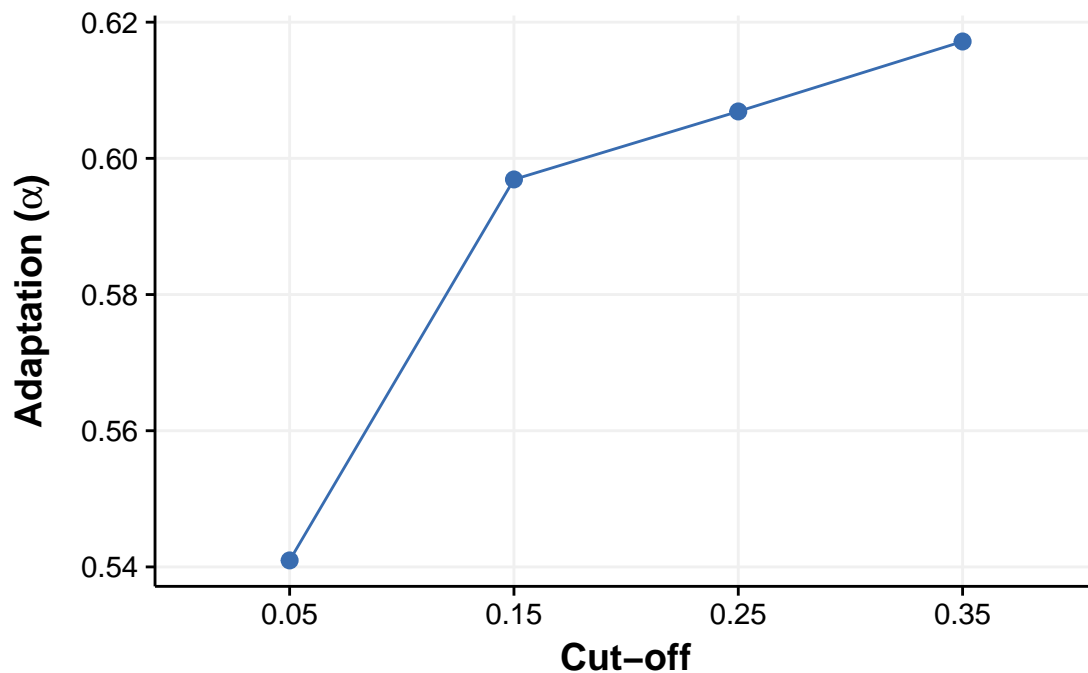
The output in this case contains a **Graph** which shows the adaptation value (alpha) obtained using each cutoff.

```
FWW(daf=myDafData, divergence=myDivergenceData, list_cutoff=c(0.05, 0.15,0.25,0.35), plot=TRUE)
#> $Results
#>               alpha.symbol Fishers exact test P-value
#> Cutoff =  0.05    0.5409548                          0
#> Cutoff =  0.15    0.5969015                          0
#> Cutoff =  0.25    0.6068868                          0
#> Cutoff =  0.35    0.6171609                          0
```

```
#>
#> $Graph
```



```
#>
#> $`Divergence metrics`
#> $`Divergence metrics`$`Global metrics`
#>           Ka        Ks      omega
#> 1 0.02102543 0.0847345 0.2481331
#>
#> $`Divergence metrics`$`Estimates by cutoff`
#>                 omegaA.symbol omegaD.symbol
#> Cutoff =  0.05     0.1342288    0.11390431
#> Cutoff =  0.15     0.1481110    0.10002208
#> Cutoff =  0.25     0.1505887    0.09754438
#> Cutoff =  0.35     0.1531380    0.09499504
#>
#>
#> $`MKT tables`
#> $`MKT tables`$`Cutoff =  0.05`
#>
#>
#> Table: cutoff
#>
#>                   Polymorphism   Divergence
#> ----------------  -------------  -----------
#> Neutral class            27912        52537
#> Selected class           13326        54641
#>
#> $`MKT tables`$`Cutoff =  0.15`
#>
#>
#> Table: cutoff
#>
```

```
#>                    Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class            20258         52537
#> Selected class            8493         54641
#>
#> $`MKT tables`$`Cutoff =  0.25`
#>
#>
#> Table: cutoff
#>
#>                    Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class            16485         52537
#> Selected class            6740         54641
#>
#> $`MKT tables`$`Cutoff =  0.35`
#>
#>
#> Table: cutoff
#>
#>                    Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class            13778         52537
#> Selected class            5486         54641
```

DGRP correction

To take adaptive and slightly deleterious mutation mutually into account, Pn , the count of segregating sites in the non-synonymous class, should be separated into the number of neutral variants and the number of weakly deleterious variants, Pn = Pn(neutral) + Pn(weakly del.). If both numbers are estimated, adaptive and weakly deleterious selection can be evaluated independently.

Consider a pair of 2×2 contingency tables. The first one corresponds to the standard MKT table with the theoretical counts of segregating sites and divergent sites for each cell.

The second table contains the count of Pn and Ps for two-frequency categories: below and over a threshold cutoff.

Add brief explanation about 2nd table!

To estimate alpha from the standard MKT table correcting by the segregation of weakly deleterious variants, we have to substitute the Pn by the expected number of neutral segregating sites, Pn(neutral). The correct estimate of alpha is then alpha = 1 - (Pn (neutral)/Ps)(Ds/Dn).

The **DGRP()** function behaves similar to the **FWW()** function. It takes the same input argument and returns the same output but containing also estimates on the **fractions of negative selection** (d: strongly deleterious, f: neutral and b: weakly deleterious).

```
DGRP(daf=myDafData, divergence=myDivergenceData)
#> $Results
#>                alpha.symbol Fishers exact test P-value
#> Cutoff =  0       0.2364499              1.480943e-183
#> Cutoff =  0.05    0.4249071               0.000000e+00
#> Cutoff =  0.1     0.4125636               0.000000e+00
```

```
#>
#> $`Divergence metrics`
#> $`Divergence metrics`$`Global metrics`
#>           Ka        Ks      omega
#> 1 0.02102543 0.0847345 0.2481331
#>
#> $`Divergence metrics`$`Estimates by cutoff`
#>               omegaA.symbol omegaD.symbol
#> Cutoff =  0        0.05867104      0.1894620
#> Cutoff =  0.05     0.10543351      0.1426996
#> Cutoff =  0.1      0.10237067      0.1457624
#>
#>
#> $`MKT tables`
#> $`MKT tables`$`Number of segregating sites by DAF category - Cutoff =  0`
#>
#>
#> Table: cutoff
#>
#>                     DAF.below.cutoff   DAF.above.cutoff
#> ---------------   -----------------   -----------------
#> Neutral class                    0               45101
#> Selected class                   0               35816
#>
#> $`MKT tables`$`Number of segregating sites by DAF category - Cutoff =  0.05`
#>
#>
#> Table: cutoff
#>
#>                     DAF.below.cutoff   DAF.above.cutoff
#> ---------------   -----------------   -----------------
#> Neutral class                17189               27912
#> Selected class               22490               13326
#>
#> $`MKT tables`$`Number of segregating sites by DAF category - Cutoff =  0.1`
#>
#>
#> Table: cutoff
#>
#>                     DAF.below.cutoff   DAF.above.cutoff
#> ---------------   -----------------   -----------------
#> Neutral class                21969               23132
#> Selected class               25707               10109
#>
#> $`MKT tables`$`MKT standard table`
#>
#>
#>                     Polymorphism   Divergence
#> ---------------   --------------   -----------
#> Neutral class               45101        52537
#> Selected class              35816        54641
#>
#>
#> $Fractions
```

```
#>             0       0.05        0.1
#> d 0.810538 0.81053943 0.8105391
#> f 0.189462 0.14269958 0.1457624
#> b 0.000000 0.04676099 0.0436985
```

Again, by default the argument **list_cutoff** uses a list of cutoffs with the following values: 0, 0.05, 0.1, and the argument **plot** is set to **FALSE**. This parameters can be customized, like in the following example, where we use a list of 4 cutoffs (0.05, 0.15, 0.25, 0.35) and set the plot argument to **TRUE**.

The output in this case contains two **Graphs** which show the adaptation value (alpha) and the negative selection fractions obtained using each cutoff.

```
DGRP(daf=myDafData, divergence=myDivergenceData, list_cutoff=c(0.05, 0.15,0.25,0.35), plot=TRUE)
#> $Results
#>                 alpha.symbol Fishers exact test P-value
#> Cutoff =  0.05     0.4249071                          0
#> Cutoff =  0.15     0.3983440                          0
#> Cutoff =  0.25     0.3718449                          0
#> Cutoff =  0.35     0.3527647                          0
#>
#> $Graph
```
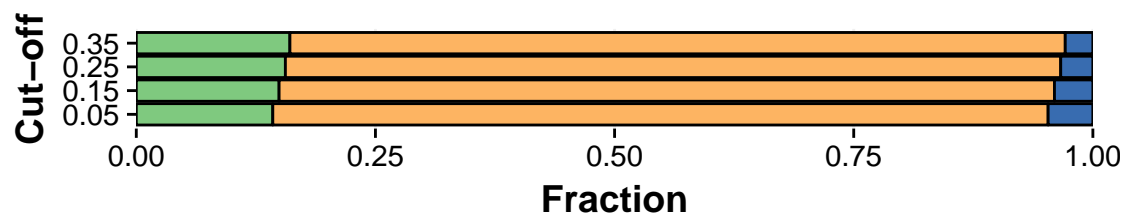
A
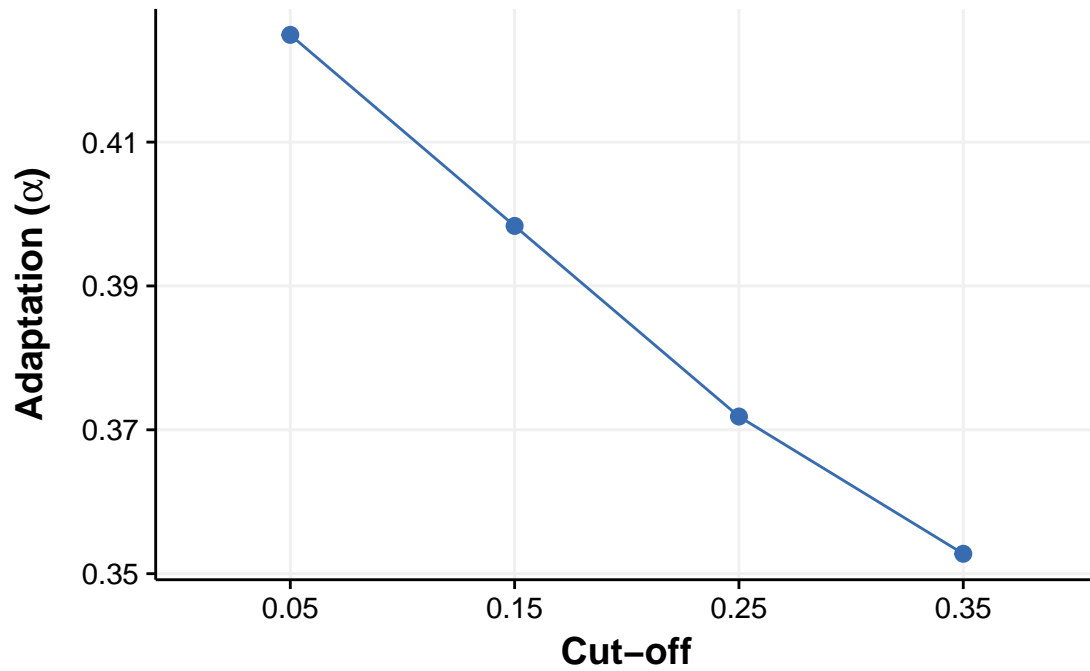
B

Fraction □ *f* □ *d* □ *b*

```
#>
#> $`Divergence metrics`
#> $`Divergence metrics`$`Global metrics`
#>          Ka        Ks      omega
#> 1 0.02102543 0.0847345 0.2481331
#>
#> $`Divergence metrics`$`Estimates by cutoff`
#>               omegaA.symbol omegaD.symbol
#> Cutoff =  0.05    0.10543351     0.1426996
#> Cutoff =  0.15    0.09884233     0.1492908
#> Cutoff =  0.25    0.09226702     0.1558661
#> Cutoff =  0.35    0.08753258     0.1606005
#>
#>
#> $`MKT tables`
#> $`MKT tables`$`Number of segregating sites by DAF category - Cutoff =  0.05`
#>
#>
#> Table: cutoff
```

```
#>
#>                    DAF.below.cutoff   DAF.above.cutoff
#> ---------------   -----------------   -----------------
#> Neutral class                17189               27912
#> Selected class               22490               13326
#>
#> $`MKT tables`$`Number of segregating sites by DAF category - Cutoff =  0.15`
#>
#>
#> Table: cutoff
#>
#>                    DAF.below.cutoff   DAF.above.cutoff
#> ---------------   -----------------   -----------------
#> Neutral class                24843               20258
#> Selected class               27323                8493
#>
#> $`MKT tables`$`Number of segregating sites by DAF category - Cutoff =  0.25`
#>
#>
#> Table: cutoff
#>
#>                    DAF.below.cutoff   DAF.above.cutoff
#> ---------------   -----------------   -----------------
#> Neutral class                28616               16485
#> Selected class               29076                6740
#>
#> $`MKT tables`$`Number of segregating sites by DAF category - Cutoff =  0.35`
#>
#>
#> Table: cutoff
#>
#>                    DAF.below.cutoff   DAF.above.cutoff
#> ---------------   -----------------   -----------------
#> Neutral class                31323               13778
#> Selected class               30330                5486
#>
#> $`MKT tables`$`MKT standard table`
#>
#>
#>                    Polymorphism   Divergence
#> ---------------   -------------   -----------
#> Neutral class             45101         52537
#> Selected class            35816         54641
#>
#>
#> $Fractions
#>          0.05        0.15        0.25        0.35
#> d 0.81053943 0.81053552 0.8105368 0.81054057
#> f 0.14269958 0.14929076 0.1558661 0.16060050
#> b 0.04676099 0.04017372 0.0335971 0.02885892
```

Asymptotic MKT

Petrov reference + explanation

This function is adapted from the code developed in "Haller BC, Messer PW. asymptoticMK: A Web-Based Tool for the Asymptotic McDonald-Kreitman Test. G3 (Bethesda). 2017 May 5;7(5):1569-1575", stored in: http://github.com/MesserLab/asymptoticMK. The main adaptation we did is that the function presented here only fits an exponential model, removing the linear fitting performed initially in the cases where it was not possible to fit an asymptotic curve.

The **asymptoticMK()** function uses the common daf and divergence parameters along with two arguments which define the lower and higher limit for the asymptotic alpha fit (xlow and xhigh). These two optional parameters are set to 0 and 1 by default, although it is recommended to use a higher limit of 0.9 in order to remove possible biases due to polarization error.

The function's output is a table with: the model type (exponential) along with the fitted function values (a, b, c), the asymptotic alpha estimate with its corresponding lower and higher confidence interval values, and the original alpha estimate (using the standard MKT methodology and the polymorphic sites within the xlow and xhigh cutoffs).

```
asymptoticMK(daf=myDafData, divergence=myDivergenceData, xlow=0, xhigh=0.9)
#>        model         a         b         c alpha_asymptotic     CI_low
#> 1 exponential 0.6258904 -1.395108 18.96187        0.6258904 0.6044232
#>     CI_high alpha_original
#> 1 0.6476488      0.2157308
```

iMK

Asymptotic explanation + Sergi slightly deleterious approach

The **iMK()** function takes the default input parameters (daf and divergence), the xlow and xhigh arguments (presented in the **asymptoticMK()** function) and it also has the optional argument **plot**, set as **FALSE** by default. However, in this example we use **plot=TRUE** to display the graphical results.

The output of the function contains:

- Asymptotic MK table: table corresponding to the asymptoticMK() function output.

- Fractions of sites: negative selection fractions (d: strongly deleterious, f: neutral and b: weakly deleterious).

- Graphs: 3 plots showing: (A) the distribution of alleles frequencies for neutral and selected sites, (B) the asymptotic alpha estimate with xlow, xhigh, original alpha and asymptotic alpha marks, and (C) the negative selection fractions.

```
iMK(daf=myDafData, divergence=myDivergenceData, xlow=0, xhigh=0.9, plot=TRUE)
#> $`Asymptotic MK table`
#>        model      a       b       c alpha_asymptotic CI_low CI_high
#> 1 exponential 0.6259 -1.3951 18.9619           0.6259 0.6043  0.6476
#>   alpha_original
#> 1         0.2157
#>
#> $`Fractions of sites`
#>   Type   Fraction
#> 1    d 0.81053796
```

```
#> 2    f 0.06232362
#> 3    b 0.12713842
#>
#> $Graphs
```

A

B

$\alpha_{asymptotic} = 0.626$

$\alpha_{standard} = 0.216$
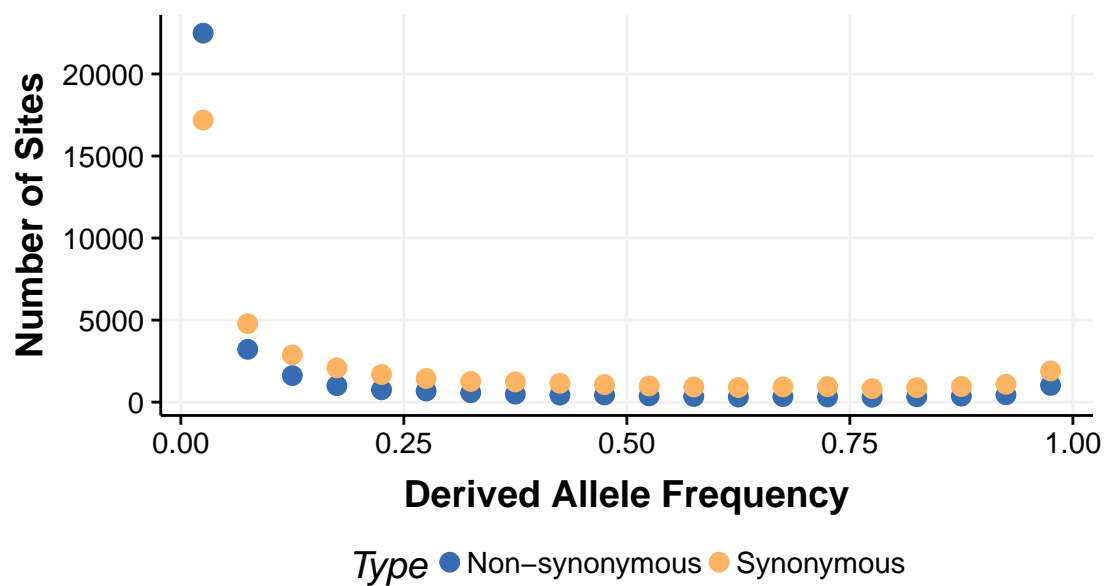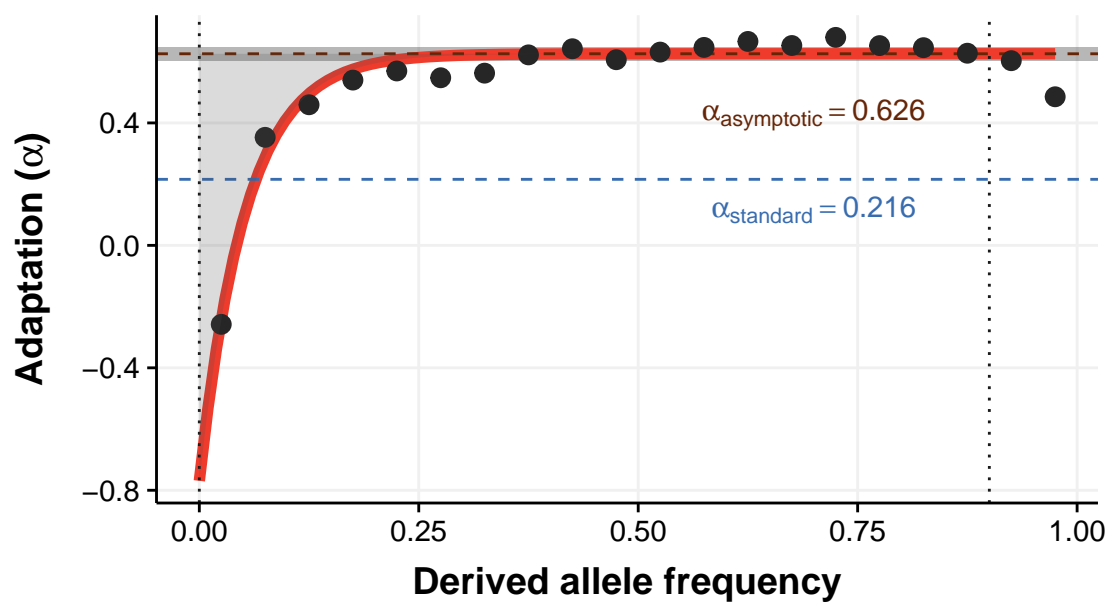
C
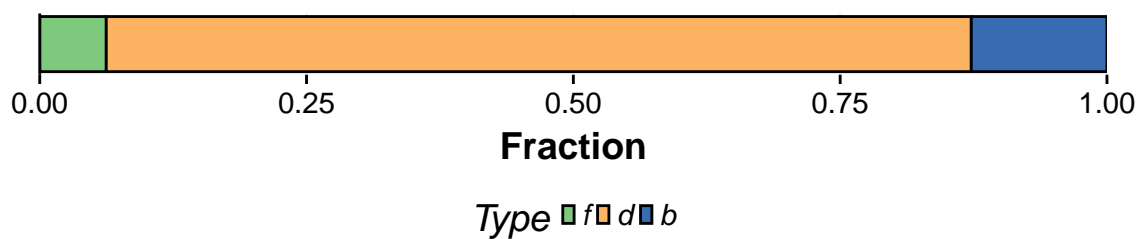
Conclusion: comparison of the different MK estimates

Summary of each alpha estimate:

Standard	FWW_0.05	FWW_0.1	DGRP_0.05	DGRP_0.1	asymptotic_iMK
0.2365	0.5409	0.5798	0.4249	0.4249	0.6259

Comment results and strengths and weakness of each method.