Package 'smoke'

December 8, 2019

Type Package

Title Small Molecule Octet/BLI Kinetics Experiment

Version 1.0.0

Date 2019-12-08

Author Qingan Sun, Xiaojun Li, James C Sacchettini

Maintainer Qingan Sun <quinsun@gmail.com>

Description Bio-Layer Interferometry (BLI) is a technology to determine the binding kinetics between biomolecule. BLI signals are small and noisy when small molecule ligands are investigated. We develop this package to process and analyze the BLI data acquired on Octet Red96 (Fortebio) more accurately.

License GPL-2 | GPL-3

Depends R (>= 3.6.0),methods,graphics, grDevices, stats, utils

LazyData yes

NeedsCompilation no

R topics documented:

`smoke-package` *Small Molecule Octet/BLI Kinetics Experiment*

Description

Bio-Layer Interferometry (BLI) is a technology to determine the binding kinetics between biomolecule. BLI signals are small and noisy when small molecule ligands are investigated. We develop this package to process and analyze the BLI data acquired on Octet Red96 (Fortebio) more accurately.

Details

The DESCRIPTION file:

Package:	smoke
Type:	Package
Title:	Small Molecule Octet/BLI Kinetics Experiment
Version:	1.0.0
Date:	2019-12-08
Author:	Qingan Sun, Xiaojun Li, James C Sacchettini
Maintainer:	Qingan Sun <quinsun@gmail.com>
Description:	Bio-Layer Interferometry (BLI) is a technology to determine the binding kinetics between biomolecule. BI
License:	GPL-2 | GPL-3
Depends:	R (>= 3.6.0),methods,graphics, grDevices, stats, utils
LazyData:	yes

Index of help topics:

```
data.raw                example raw data for BLI traces
doubleRef               substract the double references
estKinetics             estimate initial values of kOn and kOff
fitKinetics             non-linear regression of binding kinetics from
                        BLI traces
readTraces              read BLI traces from csv file
scaLoad                 Scaling BLI traces with protein-Loadings
smoke-package           Small Molecule Octet/BLI Kinetics Experiment
```

~~ An overview of how to use the package, including the most important ~~ ~~ functions ~~

Author(s)

Qingan Sun, Xiaojun Li, James C Sacchettini

Maintainer: Qingan Sun <quinsun@gmail.com>

References

Qingan Sun, Xiaojun Li, et al. The molecular basis of Pyrazinamide activity on Mycobacterium tuberculosis PanD. Nat Commun (in press)

Examples

```
data.raw=readTraces("traces.csv")
data.scaled=scaLoad(data.raw)
data.doubleRefed=doubleRef(data.scaled)
rate0=estKinetics(data.doubleRefed)
kinetics=fitKinetics(data.doubleRefed)
kinetics=fitKinetics(data.doubleRefed,plotResidual=TRUE)
```

`data.raw` *example raw data for BLI traces*

Description

example BLI traces read from csv file

Usage

```
data("data.raw")
```

Format

A data frame with 12290 observations on the following 17 variables.

`t` a numeric vector

`X1` a numeric vector

`X2` a numeric vector

`X3` a numeric vector

`X4` a numeric vector

`X5` a numeric vector

`X6` a numeric vector

`X7` a numeric vector

`X8` a numeric vector

`X9` a numeric vector

`X10` a numeric vector

`X11` a numeric vector

`X12` a numeric vector

`X13` a numeric vector

`X14` a numeric vector

`X15` a numeric vector

`X16` a numeric vector

Details

The BLI traces exported from Octet Red96 as SMPFormat, a specific text file. It is converted with python script into csv file first, then imported with readTraces function. The first column is time, the next 16 columns are time courses of BLI response.

Source

Qingan Sun, Xiaojun Li, et al. The molecular basis of Pyrazinamide activity on Mycobacterium tuberculosis PanD. Nat Commun (in press)

Examples

```
data(data.raw)
## maybe str(data.raw) ; plot(data.raw) ...
```

`doubleRef` *substract the double references*

Description

substract the double references from the BLI traces, align with the baselines before the association step; new dataframe returned as invisible and ploted

Usage

```
doubleRef(data, tA0 = 1260, tD0 = 1860)
```

Arguments

`data`	the BLI traces, first column is time in second
`tA0`	start-Time of Association step
`tD0`	start-Time of dissociation step

Details

the input data may be raw data or the traces scaled with the protein-loading

Value

dataframe: the first column is the time of association-then-dissociation segment; the next seven columns are the processed BLI traces

Author(s)

Qingan Sun, Xiaojun Li, James C Sacchettini

References

Qingan Sun, Xiaojun Li, et al. The molecular basis of Pyrazinamide activity on Mycobacterium tuberculosis PanD. Nat Commun (in press)

Examples

```
data.scaled=scaLoad(data.raw)
data.doubleRefed=doubleRef(data.scaled)

## The function is currently defined as
function (data, tA0 = 1260, tD0 = 1860)
{
    t = data[, 1]
    raw = data[, 2:17]
    bkCor = raw[, 2 * (1:8) - 1] - raw[, 2 * (1:8)]
    dbCor = bkCor - bkCor[, 8]
    baseRange = which(t < tA0 & t > (tA0 - 180))
    baseT = t[baseRange]
    baseline = numeric()
    for (i in 1:8) {
        y = dbCor[baseRange, i]
```

```
        mod = lm(y ~ baseT)
        baseline = c(baseline, predict(mod, list(baseT = 1260)))
    }
    baseline = rep(baseline, each = length(t))
    dbNorm = dbCor - baseline
    totRange = which(t >= tA0)
    t = t[totRange] - tA0
    dTot = dbNorm[totRange, 1:7]
    data = data.frame(t, dTot)
    colFun = colorRampPalette(c("cyan", "magenta"))
    palet = colFun(7)
    color1 = c(palet, "black")
    color = adjustcolor(color1, alpha.f = 0.5)
    plot(t, dTot[, 1], type = "l", ylim = range(dTot), col = color1[1],
        xlab = "Time (s)", ylab = "Residuals (nm)")
    for (i in 2:7) lines(t, dTot[, i], col = color1[i])
    abline(v = c(tD0 - tA0), lty = 2)
    abline(h = 0)
    invisible(data)
}
```

`estKinetics` *estimate initial values of kOn and kOff*

Description

A robust and rough way to estimate initial values of kON and kOff from data only. There is no manual input for the initial values

Usage

```
estKinetics(data, tD0 = 600, lig = c(16/2^(0:6)))
```

Arguments

`data`	the BLI traces with double-references corrected and aligned at start-time of association
`tD0`	start-Time of dissociation step
`lig`	concentrations of small-molecule analyte

Details

estimates of the association and dissociation with two steps: from single BLI trace, first log transform the first 50s of the association/dissociation step, linear regression for the slope; secondly use that slope as start value for non-linear fitting for the association/dissociation rate. The mean values from all traces are returned. The fitting curves are ploted.

Value

numeric vector of estimated values of kOff and kOn

Note

Due to the small SNR of small-molecule BLI, the function is not intended to get accurate kinetics parameter, but to for the inital value to feed 'fitKinetics'

Author(s)

Qingan Sun, Xiaojun Li, James C Sacchettini

References

Qingan Sun, Xiaojun Li, et al. The molecular basis of Pyrazinamide activity on Mycobacterium tuberculosis PanD. Nat Commun (in press)

See Also

fitKinetics

Examples

```
data.scaled=scaLoad(data.raw)
data.doubleRefed=doubleRef(data.scaled)
rate0=estKinetics(data.doubleRefed)


## The function is currently defined as
function (data, tD0 = 600, lig = c(16/2^(0:6)))
{
    t = data[, 1]
    dTot = data[, 2:8]
    tA = t[which(t < tD0)]
    tD = t[which(t > tD0)]
    dA = dTot[which(t < tD0), ]
    dD = dTot[which(t > tD0), ]
    rMax0 = min(dTot)
    kOn = kOff = rep(NA, 7)
    colFun = colorRampPalette(c("cyan", "magenta"))
    palet = colFun(7)
    color1 = c(palet, "black")
    color = adjustcolor(color1, alpha.f = 0.5)
    plot(1, xlim = range(t), ylim = range(dTot), type = "n",
        xlab = "Time (s)", ylab = "Response (nm)", col = color[1],
        lwd = 6)
    for (i in 1:7) {
        y = dA[, i]
        aE = y[length(y)]
        lines(tA, dA[, i], col = color[i], lwd = 6)
        yP = 1 - dA[, i]/aE
        tP = tA[which(yP > 0 & tA < 50)]
        yP = yP[which(yP > 0 & tA < 50)]
        yP = log(yP)
        modA = lm(yP ~ tP + 0)
        k0 = -coef(modA)
        modA = nls(y ~ A * (1 - exp(-k * tA)), start = list(k = k0,
            A = aE), trace = TRUE)
        lines(tA, predict(modA), col = color1[i], lwd = 6)
        kOn[i] = coef(modA)[1]
```

```
        y = dD[, i]
        d0 = predict(modA)[length(tA)]
        dE = y[length(y)]
        lines(tD, dD[, i], col = color[i], lwd = 6)
        yP = (dD[, i] - dE)/(d0 - dE)
        tP = tD[which(yP > 0 & tD < (tD0 + 50))] - tD0
        yP = yP[which(yP > 0 & tD < (tD0 + 50))]
        yP = log(yP)
        modD = lm(yP ~ tP + 0)
        k0 = -coef(modD)
        modD = nls(y ~ yE + (d0 - yE) * exp(-k * (tD - tD0)),
            start = list(k = k0, yE = dE))
        lines(tD, predict(modD), col = color1[i], lwd = 6)
        kOff[i] = coef(modD)[1]
    }
    abline(v = tD0, lwd = 6, lty = 2)
    kd0 = mean(kOff)
    ka0 = mean(kOn/lig)
    rate0 = c(kd0, ka0)
    names(rate0) = c("kd0", "ka0")
    invisible(rate0)
  }
```

`fitKinetics` *non-linear regression of binding kinetics from BLI traces*

Description

fit the whole BLI dataset with one association-then-dissociation equation

Usage

```
fitKinetics(data, tD0 = 600, kd0 = 0.08, ka0 = 0.013, lig = c(16/2^(0:6)),
plotResidual = FALSE)
```

Arguments

`data`	the BLI traces with double-references corrected and aligned at start-time of association
`tD0`	start-Time of dissociation step
`kd0`	intial value of kOff, recommend to estimate with estKinetics function
`ka0`	intial value of kOn, recommend to estimate with estKinetics function
`lig`	concentrations of small-molecule analyte
`plotResidual`	logical, default as FALSE to plot the sensorGram with fitting; 'TRUE' to plot the residuals from fitting

Details

To overcome the low SNR in the small-molecule BLI exepriment, the whole dataset is fitted with a single equation to minimize the model variability. The inital kOn and kOff values are recommended to estimate with 'estKinetics' function.

Value

dataframe with the kinetics parameters and std.error from the non-linear regression,including KD, rMax, kOff and kOn

Note

The initial kOn and kOff can be estimated with estKinetics function

Author(s)

Qingan Sun, Xiaojun Li, James C Sacchettini

References

Qingan Sun, Xiaojun Li, et al. The molecular basis of Pyrazinamide activity on Mycobacterium tuberculosis PanD. Nat Commun (in press)

See Also

estKinetics

Examples

```
#data.raw=readTraces("traces.csv")
data.scaled=scaLoad(data.raw)
data.doubleRefed=doubleRef(data.scaled)
rate0=estKinetics(data.doubleRefed)
kinetics=fitKinetics(data.doubleRefed)
kinetics=fitKinetics(data.doubleRefed,plotResidual=TRUE)

## The function is currently defined as
function (data, tD0 = 600, kd0 = 0.08, ka0 = 0.013, lig = c(16/2^(0:6)),
    plotResidual = FALSE)
{
    t = tTot = data[, 1]
    dTot = data[, 2:8]
    rMax0 = min(dTot)
    x = rep(lig, each = length(tTot))
    t = rep(t, 7)
    y = unlist(dTot[, 1:7])
    temp = rep(0, 7)
    for (i in 1:7) {
        temp1 = temp
        temp1[i] = 1
        assign(paste("dt", i, sep = ""), rep(temp1, each = length(tTot)))
    }
    mA = mD = rep(0, length(t))
    mA[which(t < tD0)] = 1
    mD[which(t > tD0)] = 1
    dfTot = data.frame(y, t, x, dt1, dt2, dt3, dt4, dt5, dt6,
        dt7, mA, mD)
    startLst = list(rMax = rMax0, kd = kd0, ka = ka0, sa1 = 0,
        sa2 = 0, sa3 = 0, sa4 = 0, sa5 = 0, sa6 = 0, sa7 = 0,
        sd1 = 0, sd2 = 0, sd3 = 0, sd4 = 0, sd5 = 0, sd6 = 0,
        sd7 = 0, ja1 = 0, ja2 = 0, ja3 = 0, ja4 = 0, ja5 = 0,
        ja6 = 0, ja7 = 0, jd1 = 0, jd2 = 0, jd3 = 0, jd4 = 0,
```

```
        jd5 = 0, jd6 = 0, jd7 = 0)
    modTot = nls(y ~ mA * (rMax/(1 + (kd/ka/x)) * (1 - 1/exp((ka *
        x + kd) * t)) + dt1 * (ja1 + sa1 * t) + dt2 * (ja2 +
        sa2 * t) + dt3 * (ja3 + sa3 * t) + dt4 * (ja4 + sa4 *
        t) + dt5 * (ja5 + sa5 * t) + dt6 * (ja6 + sa6 * t) +
        dt7 * (ja7 + sa7 * t)) + mD * (rMax/(1 + (kd/ka/x)) *
        (1 - 1/exp((ka * x + kd) * tD0))/exp(kd * (t - tD0)) +
        dt1 * (jd1 + sd1 * (t - tD0)) + dt2 * (jd2 + sd2 * (t -
        tD0)) + dt3 * (jd3 + sd3 * (t - tD0)) + dt4 * (jd4 +
        sd4 * (t - tD0)) + dt5 * (jd5 + sd5 * (t - tD0)) + dt6 *
        (jd6 + sd6 * (t - tD0)) + dt7 * (jd7 + sd7 * (t - tD0))),
        start = startLst, trace = TRUE)
    dfTotPre = cbind(dfTot, predict(modTot))
    coefTot = coef(summary(modTot))
    kd = coefTot[2, 1]
    ka = coefTot[3, 1]
    kD = kd/ka
    kDsd = sqrt((coefTot[2, 2]/ka)^2 + (kd * coefTot[3, 2]/ka/ka)^2)
    coefTot = rbind(c(kD, kDsd, NA, NA), coefTot)
    rownames(coefTot)[1] = "KD";rownames(coefTot)[3:4]=c("kOff","kOn")
    colFun = colorRampPalette(c("cyan", "magenta"))
    palet = colFun(7)
    color1 = c(palet, "black")
    color = adjustcolor(color1, alpha.f = 0.5)
    if (!plotResidual) {
        plot(tTot, dTot[, 1], ylim = range(y), type = "l", xlab = "Time (s)",
            ylab = "Response (nm)", col = color[1], lwd = 6)
        for (i in 2:7) lines(tTot, dTot[, i], col = color[i],
            lwd = 6)
        for (i in 1:7) lines(tTot, dfTotPre[which(dfTot[, 3] ==
            lig[i]), 13], col = color1[i], lwd = 6)
        abline(v = tD0, lwd = 6, lty = 2)
    }
    else {
        residual = dfTotPre[, 13] - dfTot[, 1]
        bound = (range(y)[2] - range(y)[1])/2
        plot(tTot, residual[which(dfTot[, 3] == lig[1])], ylim = c(-bound,
            bound), type = "l", xlab = "Time (s)", ylab = expression(Delta *
            "Response (nm)"), col = color[1], lwd = 6)
        for (i in 2:7) lines(tTot, residual[which(dfTot[, 3] ==
            lig[i])], col = color[i], lwd = 6)
        abline(h = 0, lwd = 6)
        abline(v = tD0, lwd = 6, lty = 2)
    }
    invisible(coefTot[1:4, ])
}
```

`readTraces` *read BLI traces from csv file*

Description

read BLI traces from csv file. The first column is time in second, the next 16 columns are BLI traces.The time series are put in an dataframe, returned as invisible and plot

Usage

```
readTraces(csvFile = "traces.csv", tLoadS = 180, tLoadF = 780, tBlockS = 840,
tBlockF = 960, tA0 = 1260, tD0 = 1860)
```

Arguments

csvFile	the csv file of BLI traces
tLoadS	start-Time of protein-Loading
tLoadF	end-Time of protein-Loading
tBlockS	start-Time of blocking
tBlockF	end-Time of blocking
tA0	start-Time of Association step
tD0	start-Time of dissociation step

Details

The BLI traces exported from Octet Red96 as SMPFormat, a specific text file. It is converted with python script into csv file first. In this csv file, the first column is time, the next columns are BLI traces in the row-wise order.

Value

dataframe: the first column is time; the next 16 columns are the raw BLI traces

Author(s)

Qingan Sun, Xiaojun Li, James C Sacchettini

References

Qingan Sun, Xiaojun Li, et al. The molecular basis of Pyrazinamide activity on Mycobacterium tuberculosis PanD. Nat Commun (in press)

See Also

data.raw

Examples

```
data.raw=readTraces("traces.csv")
## The function is currently defined as
function (csvFile = "traces.csv", tLoadS = 180, tLoadF = 780,
    tBlockS = 840, tBlockF = 960, tA0 = 1260, tD0 = 1860)
{
    data = read.csv("traces.csv", header = FALSE)
    t = data[, 1]
    raw = data[, 1 + (1:16)]
    colnames(raw) = 1:16
    data = data.frame(t, raw)
    colFun = colorRampPalette(c("cyan", "magenta"))
    palet = colFun(7)
    color1 = c(palet, "black")
    color = adjustcolor(color1, alpha.f = 0.3)
```

```
  plot(t, raw[, 1], type = "l", ylim = range(raw), col = color1[1],
      xlab = "Time (s)", ylab = "Response (nm)", lwd = 8)
  lines(t, raw[, 2], col = color1[1], lwd = 8)
  for (i in 2:8) {
      lines(t, raw[, i * 2 - 1], col = color1[i], lwd = 8)
      lines(t, raw[, i * 2], col = color1[i], lwd = 8)
  }
  abline(v = c(tLoadS, tLoadF, tBlockS, tBlockF, tA0, tD0),
      lty = 2, lwd = 6)
  invisible(data)
}
```

`scaLoad` *Scaling BLI traces with protein-Loadings*

Description

An optional step in data processing

Usage

```
scaLoad(data, tLoadS = 180, tLoadF = 780)
```

Arguments

`data`	the BLI traces, first column is time in second
`tLoadS`	start-Time of protein-Loading step
`tLoadF`	end-Time of protein-Loading step

Details

This is an optional step in data processing. The BLI traces are scaled with protein-Loadings, the new dataframe returned as invisible and ploted.

Value

dataframe: the first column is time; the next 16 columns are BLI traces

Author(s)

Qingan Sun, Xiaojun Li, James C Sacchettini

References

Qingan Sun, Xiaojun Li, et al. The molecular basis of Pyrazinamide activity on Mycobacterium tuberculosis PanD. Nat Commun (in press)

Examples

```
data.scaled=scaLoad(data.raw)

## The function is currently defined as
function (data, tLoadS = 180, tLoadF = 780)
{
    t = data[, 1]
    raw = data[, 2:17]
    load0Range = which(t < tLoadS & t > (tLoadS - 60))
    loadRange = which(t < tLoadF & t > (tLoadF - 60))
    load0 = load = numeric()
    for (i in 1:8) {
        tT = t[load0Range]
        y = raw[load0Range, i * 2 - 1]
        mod = lm(y ~ tT)
        load0 = c(load0, predict(mod, list(tT = tLoadS)))
        y = raw[load0Range, i * 2]
        mod = lm(y ~ tT)
        load0 = c(load0, predict(mod, list(tT = tLoadS)))
        tT = t[loadRange]
        y = raw[loadRange, i * 2 - 1]
        mod = lm(y ~ tT)
        load = c(load, predict(mod, list(tT = tLoadF)))
        y = raw[loadRange, i * 2]
        mod = lm(y ~ tT)
        load = c(load, predict(mod, list(tT = tLoadS)))
    }
    load0[(1:8) * 2] = 0
    load[(1:8) * 2] = 1
    raw = raw - rep(load0, each = length(t))
    load = load - load0
    dim(load) = c(2, 8)
    meanLoad = apply(load, 1, mean)
    loadFactor = load/meanLoad
    loadFactor = c(loadFactor)
    raw = raw/rep(loadFactor, each = length(t))
    colFun = colorRampPalette(c("cyan", "magenta"))
    palet = colFun(7)
    color1 = c(palet, "black")
    color = adjustcolor(color1, alpha.f = 0.3)
    plot(t, raw[, 1], type = "l", ylim = range(raw), col = color1[1],
        xlab = "Time (s)", ylab = "Response (nm)", lwd = 8)
    lines(t, raw[, 2], col = color1[1], lwd = 8)
    for (i in 2:8) {
        lines(t, raw[, i * 2 - 1], col = color1[i], lwd = 8)
        lines(t, raw[, i * 2], col = color1[i], lwd = 8)
    }
    abline(v = c(tLoadS, tLoadF), lty = 2, lwd = 6)
    data = data.frame(t, raw)
    invisible(data)
  }
```

Index