Package ‘fourierin’  
December 8, 2023

Type Package
Title Computes Numeric Fourier Integrals
Version 0.2.5
Date 2023-12-08
Author Guillermo Basulto-Elias
Maintainer Guillermo Basulto-Elias <guillermobasulto@gmail.com>
Description Computes Fourier integrals of functions of one and two variables using the Fast Fourier transform. The Fourier transforms must be evaluated on a regular grid for fast evaluation.
License MIT + file LICENSE
LinkingTo RcppArmadillo, Rcpp
Imports Rcpp (>= 1.0.1),
Suggests MASS, knitr, rmarkdown, dplyr, purrr, ggplot2,
        lattice, rbenchmark, testthat (>= 3.1.0), covr, spelling
RoxygenNote 7.2.3
URL https://github.com/gbasulto/fourierin
BugReports https://github.com/gbasulto/fourierin/issues
VignetteBuilder knitr
Encoding UTF-8
Language en-US
NeedsCompilation yes
Repository CRAN
Date/Publication 2023-12-08 20:40:02 UTC

R topics documented:

fourierin ................................................................. 2
fourierin_1d ............................................................ 5
fourierin_2d ............................................................ 7

Index 11
fourierin

Compute Fourier integrals

Description

It computes Fourier integrals for functions of one and two variables.

Usage

fourierin(
  f,
  lower_int,
  upper_int,
  lower_eval = NULL,
  upper_eval = NULL,
  const_adj,
  freq_adj,
  resolution = NULL,
  eval_grid = NULL,
  use_fft = TRUE
)

Arguments

- **f**: function or a vector of size m. If a function is provided, it must be able to be evaluated at vectors. If a vector of values is provided, such evaluations must have been obtained on a regular grid and the Fourier integral is faster if m is a power of 2.
- **lower_int**: Lower integration limit(s).
- **upper_int**: Upper integration limit(s).
- **lower_eval**: Lower evaluation limit(s). It can be NULL if an evaluation grid is provided.
- **upper_eval**: Upper evaluation limit(s). It can be NULL if an evaluation grid is provided.
- **const_adj**: Factor related to adjust definition of Fourier transform. It is usually equal to 0, -1 or 1.
- **freq_adj**: Constant to adjust the exponent on the definition of the Fourier transform. It is usually equal to 1, -1, 2pi or -2pi.
- **resolution**: A vector of integers (faster if powers of two) determining the resolution of the evaluation grid. Not required if f is a vector.
- **eval_grid**: Optional matrix with d columns with the points where the Fourier integral will be evaluated. If it is provided, the FFT will not be used.
- **use_fft**: Logical value specifying whether the FFT will be used.

Details

See plenty of detailed examples in the vignette.
Value

A list with the elements n-dimensional array and n vectors with their corresponding resolution. Specifically,

values A n-dimensional (resol_1 x resol_2 x ... x resol_n) complex array with the values.

w1 A vector of size resol_1

... 

wn A vector of size resol_n

Examples

### Example 1 ------------------------------------------
### Recovering std. normal from its characteristic function -----
library(fourierin)

## Function to be used in the integrand
myfnc <- function(t) exp(-t^2/2)

## Compute integral
out <- fourierin(f = myfnc, lower_int = -5, upper_int = 5,
    lower_eval = -3, upper_eval = 3, const_adj = -1,
    freq_adj = -1, resolution = 64)

## Extract grid and values
grid <- out$w
values <- Re(out$values)

## Compare with true values of Fourier transform
plot(grid, values, type = "l", col = 3)
lines(grid, dnorm(grid), col = 4)

### Example 2 ------------------------------------------
### Computing characteristic function of a gamma r. v. -------
library(fourierin)

## Function to be used in integrand
myfnc <- function(t) dgamma(t, shape, rate)

## Compute integral
shape <- 5
rate <- 3
out <- fourierin(f = myfnc, lower_int = 0, upper_int = 6,
    lower_eval = -4, upper_eval = 4,
    const_adj = 1, freq_adj = 1, resolution = 64)

## Extract values
grid <- out$w
re_values <- Re(out$values)
im_values <- Im(out$values)  # Imag values

## Now compute the real and imaginary true values of the
## characteristic function.
true_cf <- function(t, shape, rate) (1 - 1i*t/rate)^-shape
true_re <- Re(true_cf(grid, shape, rate))
true_im <- Im(true_cf(grid, shape, rate))

## Compare them. We can see a slight discrepancy on the tails,
## but that is fixed when resolution is increased.
plot(grid, re_values, type = "l", col = 3)
lines(grid, true_re, col = 4)

# Same here
plot(grid, im_values, type = "l", col = 3)
lines(grid, true_im, col = 4)

###--- Example 3 -------------------------------------------------
###--- Recovering std. normal from its characteristic function ---
library(fourierin)

##-Parameters of bivariate normal distribution
mu <- c(-1, 1)
sig <- matrix(c(3, -1, -1, 2), 2, 2)

##-Multivariate normal density
##-x is n x d
f <- function(x) {
    ##-Auxiliar values
d <- ncol(x)
z <- sweep(x, 2, mu, "+")
    ##-Get numerator and denominator of normal density
    num <- exp(-0.5*rowSums(z * (z %*% solve(sig))))
    denom <- sqrt((2*pi)^d*det(sig))
    return(num/denom)
}

## Characteristic function
## s is n x d
phi <- function(s) {
    complex(modulus = exp(- 0.5*rowSums(s * (s %*% sig))),
           argument = s %*% mu)
}

##-Approximate cf using Fourier integrals
eval <- fourierin(f, lower_int = c(-8, -6), upper_int = c(6, 8),
                 lower_eval = c(-4, -4), upper_eval = c(4, 4),
                 const_adj = 1, freq_adj = 1,
                 resolution = c(128, 128))

## Extract values
t1 <- eval$w1
t2 <- eval$w2
fourierin_1d

Univariate Fourier integrals

Description

It computes Fourier integrals of functions of one and two variables on a regular grid.

Usage

fourierin_1d(
  f,
  lower_int,
  upper_int,
  lower_eval = NULL,
  upper_eval = NULL,
  const_adj,
  freq_adj,
  resolution = NULL,
  eval_grid = NULL,
  use_fft = TRUE
)
Arguments

- **f**: function or a vector of size \( m \). If a function is provided, it must be able to be evaluated at vectors. If a vector of values is provided, such evaluations must have been obtained on a regular grid and the Fourier integral is faster if \( m \) is a power of 2.
- **lower_int**: Lower integration limit(s).
- **upper_int**: Upper integration limit(s).
- **lower_eval**: Lower evaluation limit(s). It can be NULL if an evaluation grid is provided.
- **upper_eval**: Upper evaluation limit(s). It can be NULL if an evaluation grid is provided.
- **const_adj**: Factor related to adjust definition of Fourier transform. It is usually equal to 0, -1 or 1.
- **freq_adj**: Constant to adjust the exponent on the definition of the Fourier transform. It is usually equal to 1, -1, \( 2\pi \) or \(-2\pi\).
- **resolution**: A vector of integers (faster if powers of two) determining the resolution of the evaluation grid. Not required if \( f \) is a vector.
- **eval_grid**: Optional matrix with \( d \) columns with the points where the Fourier integral will be evaluated. If it is provided, the FFT will not be used.
- **use_fft**: Logical value specifying whether the FFT will be used.

Details

See vignette for more detailed examples.

Value

If \( w \) is given, only the values of the Fourier integral are returned, otherwise, a list with the elements

- **w**: A vector of size \( m \) where the integral was computed.
- **values**: A complex vector of size \( m \) with the values of the integral

Examples

```r
##--- Example 1 ---------------------------------------------------
##--- Recovering std. normal from its characteristic function -----
library(fourierin)

# Function to to be used in integrand
myfun <- function(t) exp(-t^2/2)

# Compute Foueien integral
out <- fourierin_1d(f = myfun,
    lower_int = -5, upper_int = 5,
    lower_eval = -3, upper_eval = 3,
    const_adj = -1, freq_adj = -1,
    resolution = 64)

## Extract grid and values
```
```r
grid <- out$w
values <- Re(out$values)

plot(grid, values, type = "l", col = 3)
lines(grid, dnorm(grid), col = 4)

##--- Example 2 -----------------------------------------------
##--- Computing characteristic function of a gamma r. v. ------

library(fourierin)

## Function to to be used in integrand
myfun <- function(t) dgamma(t, shape, rate)

## Compute integral
shape <- 5
rate <- 3
out <- fourierin_1d(f = myfun, lower_int = 0, upper_int = 6,
    lower_eval = -4, upper_eval = 4,
    const_adj = 1, freq_adj = 1, resolution = 64)

grid <- out$w # Extract grid
re_values <- Re(out$values) # Real values
im_values <- Im(out$values) # Imag values

# Now compute the real and # imaginary true values of the # characteristic function.
true_cf <- function(t, shape, rate) (1 - 1i*t/rate)^-shape
ture_re <- Re(true_cf(grid, shape, rate))
ture_im <- Im(true_cf(grid, shape, rate))

# Compare them. We can see a # slight discrepancy on the # tails, but that is fixed # when resolution is # increased.
plot(grid, re_values, type = "l", col = 3)
lines(grid, true_re, col = 4)

# Same here
plot(grid, im_values, type = "l", col = 3)
lines(grid, true_im, col = 4)
```

---

### fourierin_2d

**Bivariate Fourier integrals**

**Description**

It computes Fourier integrals for functions of one and two variables.
Usage

fourierin_2d(
  f,
  lower_int,
  upper_int,
  lower_eval = NULL,
  upper_eval = NULL,
  const_adj,
  freq_adj,
  resolution = NULL,
  eval_grid = NULL,
  use_fft = TRUE
)

Arguments

f        function or a vector of size m. If a function is provided, it must be able to be evaluated at vectors. If a vector of values is provided, such evaluations must have been obtained on a regular grid and the Fourier integral is faster if m is a power of 2.
lower_int Lower integration limit(s).
upper_int Upper integration limit(s).
lower_eval Lower evaluation limit(s). It can be NULL if an evaluation grid is provided.
upper_eval Upper evaluation limit(s). It can be NULL if an evaluation grid is provided.
const_adj Factor related to adjust definition of Fourier transform. It is usually equal to 0, -1 or 1.
freq_adj Constant to adjust the exponent on the definition of the Fourier transform. It is usually equal to 1, -1, 2pi or -2pi.
resolution A vector of integers (faster if powers of two) determining the resolution of the evaluation grid. Not required if f is a vector.
eval_grid Optional matrix with d columns with the points where the Fourier integral will be evaluated. If it is provided, the FFT will not be used.
use_fft Logical value specifying whether the FFT will be used.

Value

If w is given, only the values of the Fourier integral are returned, otherwise, a list with three elements

w1        Evaluation grid for first entry
w2        Evaluation grid for second entry
values    m1 x m2 matrix of complex numbers, corresponding to the evaluations of the integral
Examples

```r
##--- Recovering std. normal from its characteristic function -----
library(fourierin)

##-Parameters of bivariate normal distribution
mu <- c(-1, 1)
sig <- matrix(c(3, -1, -1, 2), 2, 2)

##-Multivariate normal density
##-x is n x d
f <- function(x) {
    ##-Auxiliar values
d <- ncol(x)
z <- sweep(x, 2, mu, "-")

    ##-Get numerator and denominator of normal density
    num <- exp(-0.5*rowSums(z * (z %*% solve(sig))))
denom <- sqrt((2*pi)^d*det(sig))

    return(num/denom)
}

##-Characteristic function
##-s is n x d
phi <- function(s) {
    complex(modulus = exp(- 0.5*rowSums(s * (s %*% sig))未来的符号未知),
           argument = s %*% mu)
}

##-Approximate cf using Fourier integrals
eval <- fourierin_2d(f, lower_int = c(-8, -6), upper_int = c(6, 8),
                     lower_eval = c(-4, -4), upper_eval = c(4, 4),
                     const_adj = 1, freq_adj = 1,
                     resolution = c(128, 128))

## Extract values
t1 <- eval$w1
t2 <- eval$w2
t <- as.matrix(expand.grid(t1 = t1, t2 = t2))
approx <- eval$values
true <- matrix(phi(t), 128, 128) # Compute true values

##-This is a section of the characteristic functions
i <- 65
plot(t2, Re(approx[i, ]), type = "l", col = 2,
ylab = "",
xlab = expression(t[2]),
main = expression(paste("Real part section at ",
                          t[1], " = 0")))
lines(t2, Re(true[i, ]), col = 3)
legend("topleft", legend = c("true", "approximation"),
col = 3:2, lwd = 1)
```
## Another section, now of the imaginary part

```r
plot(t[1], Im(approx[, i]), type = "l", col = 2,
     ylab = "",
     xlab = expression(t[1]),
     main = expression(paste("Imaginary part section at ",
                              t[2], " = 0")))
lines(t[1], Im(true[, i]), col = 3)
legend("topleft", legend = c("true", "approximation"),
        col = 3:2, lwd = 1)
```
Index

fourierin, 2
fourierin_1d, 5
fourierin_2d, 7