

# Package ‘NFWdist’

October 12, 2022

**Type** Package

**Title** The Standard Distribution Functions for the 3D NFW Profile

**Version** 0.1.0

**Author** Aaron Robotham

**Maintainer** Aaron Robotham <aaron.robotham@uwa.edu.au>

**Description** Density, distribution function, quantile function and random generation for the 3D Navarro, Frenk & White (NFW) profile. For details see Robotham & Howlett (2018) <[arXiv:1805.09550](#)>.

**License** GPL-3

**LazyData** true

**Depends** R (>= 3.00)

**Suggests** knitr, gsl, lamW

**VignetteBuilder** knitr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-05-29 08:31:03 UTC

## R topics documented:

nfw	1
<b>Index</b>	<b>4</b>

---

nfw *The Standard Distribution Functions for the 3D NFW Profile*

---

## Description

Density, distribution function, quantile function and random generation for the 3D NFW profile

**Usage**

```
dnfw(x, con = 5, log = FALSE)
pnfw(q, con = 5, log.p = FALSE)
qnfw(p, con = 5, log.p = FALSE)
rnfw(n, con = 5)
```

**Arguments**

<code>x, q</code>	Vector of quantiles. This is scaled such that ‘x’ and ‘q’ are equal to $R/R_{\text{vir}}$ for NFW. This means the PDF is only defined between 0 and 1.
<code>p</code>	Vector of probabilities.
<code>n</code>	Number of observations. If $\text{length}(n) > 1$ , the length is taken to be the number required.
<code>con</code>	The NFW profile concentration parameter, where $c=R_{\text{vir}}/R_s$ .
<code>log, log.p</code>	Logical; if TRUE, probabilities/densities p are returned as $\log(p)$ .

**Details**

The novel part of this package is the general solution for the CDF inversion (i.e. `qnfw`). As far as I can see this has not been published anywhere, and it is a useful function for populating halos in something like an HOD.

One of `lamW` (fastest) or `gsl` (easier to install) must be installed to use the `qnfw` and `rnfw` functions!. Try to install `lamW` first (since it is about four times faster), but if that is tricky due to `Rcpp` dependencies then use `gsl` instead.

**Value**

`dnfw` gives the density, `pnfw` gives the distribution function, `qnfw` gives the quantile function, and `rnfw` generates random deviates.

**Note**

This seems to work at least as efficiently as `accept reject`, but it is ultimately much more elegant code in any case.

**Author(s)**

Aaron Robotham

**References**

Robotham & Howlett, 2018, arXiv 1805.09550

**See Also**

`lambert_W0` (`gsl`) or `lambertW0` (`lamW`).

**Examples**

```
#Both the PDF (dnfw) integrated up to x, and CDF at q (pnfw) should be the same:  
#0.373, 0.562, 0.644, 0.712
```

```
for(con in c(1,5,10,20)){  
  print(integrate(dnfw, lower=0, upper=0.5, con=con)$value)  
  print(pnfw(0.5, con=con))  
}
```

```
#The qnfw should invert the pnfw, returning the input vector (1:9)/10:  
for(con in c(1,5,10,20)){  
  print(qnfw(p=pnfw(q=(1:9)/10,con=con), con=con))  
}
```

```
#The sampling from rnfw should recreate the expected PDF from dnfw:
```

```
for(con in c(1,5,10,20)){  
  plot(density(rnfw(1e6,con=con), bw=0.01))  
  lines(seq(0,1,len=1e3), dnfw(seq(0,1,len=1e3),con=con),col='red')  
  legend('topright',legend=paste('con =',con))  
}
```

# Index

## \* **nfw**

nfw, 1

dnfw (nfw), 1

NFW (nfw), 1

nfw, 1

NFWdist (nfw), 1

pnfw (nfw), 1

qnfw (nfw), 1

rnfw (nfw), 1