

# Package ‘VCFArray’

July 3, 2025

**Title** Representing on-disk / remote VCF files as array-like objects

**Version** 1.24.0

**Description** VCFArray extends the DelayedArray to represent VCF data entries as array-like objects with on-disk / remote VCF file as backend. Data entries from VCF files, including info fields, FORMAT fields, and the fixed columns (REF, ALT, QUAL, FILTER) could be converted into VCFArray instances with different dimensions.

**biocViews** Infrastructure, DataRepresentation, Sequencing,  
VariantAnnotation

**Depends** R (>= 3.6), methods, BiocGenerics, DelayedArray (>= 0.7.28)

**License** GPL-3

**Encoding** UTF-8

**URL** <https://github.com/Liubuntu/VCFArray>

**BugReports** <https://github.com/Liubuntu/VCFArray/issues>

**Imports** tools, GenomicRanges, VariantAnnotation (>= 1.29.3),  
GenomicFiles (>= 1.17.3), S4Vectors (>= 0.19.19), Rsamtools

**RoxygenNote** 6.1.1

**VignetteBuilder** knitr

**Suggests** SeqArray, BiocStyle, BiocManager, testthat, knitr, rmarkdown

**git\_url** <https://git.bioconductor.org/packages/VCFArray>

**git\_branch** RELEASE\_3\_21

**git\_last\_commit** b36ef19

**git\_last\_commit\_date** 2025-04-15

**Repository** Bioconductor 3.21

**Date/Publication** 2025-07-02

**Author** Qian Liu [aut, cre],  
Martin Morgan [aut]

**Maintainer** Qian Liu <qliu7@buffalo.edu>

## Contents

|                          |   |
|--------------------------|---|
| dim, VCFArraySeed-method | 2 |
| extract_array            | 3 |

|              |          |
|--------------|----------|
| <b>Index</b> | <b>5</b> |
|--------------|----------|

---

|                          |                                                                            |
|--------------------------|----------------------------------------------------------------------------|
| dim, VCFArraySeed-method | <i>VCFArraySeed or VCFArray related methods, slot getters and setters.</i> |
|--------------------------|----------------------------------------------------------------------------|

---

## Description

dim, dimnames: dimension and dimnames of object contained in the VCF file.

vcffile: extract the VcfFile object corresponding to the backend VCF file.

rowRanges: extract the rowRanges information from the backend VCF file.

## Usage

```
## S4 method for signature 'VCFArraySeed'
dim(x)

## S4 method for signature 'VCFArraySeed'
dimnames(x)

vcffile(x)

## S4 method for signature 'VCFArraySeed'
vcffile(x)

## S4 method for signature 'VCFArraySeed'
rowRanges(x)

## S4 method for signature 'VCFArraySeed'
show(object)

## S4 method for signature 'VCFArray'
vcffile(x)

## S4 method for signature 'VCFArray'
rowRanges(x)
```

## Arguments

x                    the VCFArray or VCFArraySeed objects.

object              the VCFArraySeed object.

**Value**

`dim`: the integer vector of dimensions for `VCFArray` or `VCFArraySeed` objects.

`dimnames`: the unnamed list of dimension names for `VCFArray` and `VCFArraySeed` objects.

`vcffile`: the `VcfFile` object corresponding to the backend VCF file.

**Examples**

```
f1 <- system.file("extdata", "chr22.vcf.gz",
                  package="VariantAnnotation")
va <- VCFArray(f1, name = "GT")
dim(va)
dimnames(va)
vcffile(va)
seed(va)
dim(seed(va))
DelayedArray::type(va)
```

---

|               |                                                   |
|---------------|---------------------------------------------------|
| extract_array | <i>VCFArray constructor and coercion methods.</i> |
|---------------|---------------------------------------------------|

---

**Description**

`extract_array`: the function to extract data from a VCF file, by taking `VCFArraySeed` as input. This function is required by the `DelayedArray` for the seed contract.

`VCFArray`: The function to convert data entries inside VCF file into the `VCFArray` instance.

**Usage**

```
## S4 method for signature 'VCFArraySeed'
extract_array(x, index)

VCFArray(file, vindex = character(), name = NA, pfix = NULL)
```

**Arguments**

|                     |                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code>      | the <code>VCFArraySeed</code> object                                                                                                                                                                                                                                                                                                                                          |
| <code>index</code>  | in <code>extract_array()</code> , an unnamed list of subscripts as positive integer vectors, one vector per dimension in <code>x</code> . Empty and missing subscripts (represented by <code>integer(0)</code> and <code>NULL</code> list elements, respectively) are allowed. The subscripts can contain duplicated indices. They cannot contain NAs or non-positive values. |
| <code>file</code>   | takes values for character string (specifying the VCF file path), <code>VcfFile</code> object, and <code>RangedVcfStack</code> object.                                                                                                                                                                                                                                        |
| <code>vindex</code> | in <code>VCFArray()</code> , the character string specifying the index file path. This argument is required if a remote VCF file is used for the <code>file</code> argument.                                                                                                                                                                                                  |
| <code>name</code>   | the data entry from VCF file to be read into <code>VCFArraySeed</code> / <code>VCFArray</code> . For <code>VCFArray</code> . This argument should always be specified.                                                                                                                                                                                                        |

**prefix** the category that the name belongs to. Available values are fixed, info, and info. Can also Check `vcfFields(file)` for matching name and prefix.

### Value

VCFArray class object.

### Examples

```
f1 <- system.file("extdata", "chr22.vcf.gz",
                  package="VariantAnnotation")
va <- VCFArray(f1, name = "GT")
va
vcf <- VariantAnnotation::VcfFile(f1)
va1 <- VCFArray(vcf, name = "GT")
va1
all.equal(va, va1)
## Not run:
## RangedVcfStack class
extdata <- system.file(package = "GenomicFiles", "extdata")
files <- dir(extdata, pattern="^CEUtrio.*bgz$", full=TRUE)[1:2]
names(files) <- sub(".*_([0-9XY]+).*", "\\1", basename(files))
seqinfo <- as(readRDS(file.path(extdata, "seqinfo.rds")), "Seqinfo")
stack <- GenomicFiles::VcfStack(files, seqinfo)
gr <- as(GenomicFiles::seqinfo(stack)[rownames(stack)], "GRanges")
## RangedVcfStack
rgstack <- GenomicFiles::RangedVcfStack(stack, rowRanges = gr)
rgstack
va2 <- VCFArray(rgstack, name = "SB")
va2

## End(Not run)
## coercion
as(va[1:10, ], "array")
```

# Index

`coerce (extract_array)`, [3](#)  
`coerce, ANY, VCFMatrix-method`  
    `(extract_array)`, [3](#)  
`coerce, VCFArray, VCFMatrix-method`  
    `(extract_array)`, [3](#)  
`coerce, VCFMatrix, VCFArray-method`  
    `(extract_array)`, [3](#)  
  
`dim, VCFArraySeed-method`, [2](#)  
`dimnames, VCFArraySeed-method`  
    `(dim, VCFArraySeed-method)`, [2](#)  
  
`extract_array`, [3](#)  
`extract_array, VCFArraySeed-method`  
    `(extract_array)`, [3](#)  
  
`matrixClass, VCFArray-method`  
    `(extract_array)`, [3](#)  
  
`rowRanges, VCFArray-method`  
    `(dim, VCFArraySeed-method)`, [2](#)  
`rowRanges, VCFArraySeed-method`  
    `(dim, VCFArraySeed-method)`, [2](#)  
  
`show, VCFArraySeed-method`  
    `(dim, VCFArraySeed-method)`, [2](#)  
  
`VCFArray (extract_array)`, [3](#)  
`VCFArray-class (extract_array)`, [3](#)  
`VCFArray-method (extract_array)`, [3](#)  
`vcffile (dim, VCFArraySeed-method)`, [2](#)  
`vcffile, VCFArray`  
    `(dim, VCFArraySeed-method)`, [2](#)  
`vcffile, VCFArray-method`  
    `(dim, VCFArraySeed-method)`, [2](#)  
`vcffile, VCFArraySeed`  
    `(dim, VCFArraySeed-method)`, [2](#)  
`vcffile, VCFArraySeed-method`  
    `(dim, VCFArraySeed-method)`, [2](#)  
`VCFMatrix (extract_array)`, [3](#)  
`VCFMatrix-class (extract_array)`, [3](#)