

# Package ‘readJDX’

December 22, 2016

**Type** Package

**Title** Import Data in the JCAMP-DX Format

**Version** 0.2.3

**Date** 2016-12-21

**Description** Import data written in the JCAMP-DX format. This is an instrument-independent format used in the field of spectroscopy. Examples include IR, NMR, and Raman spectroscopy. See the vignette for background and supported formats. The official JCAMP-DX site is <<http://www.jcamp-dx.org/>>.

**URL** <https://github.com/bryanhanson/readJDX>

**BugReports** <https://github.com/bryanhanson/readJDX/issues>

**License** GPL (>= 3)

**ByteCompile** TRUE

**Depends** R (>= 3.0)

**Suggests** knitr

**Imports** stringr

**VignetteBuilder** knitr

**RoxygenNote** 5.0.1

**NeedsCompilation** no

**Author** Bryan A. Hanson [aut, cre]

**Maintainer** Bryan A. Hanson <[hanson@depauw.edu](mailto:hanson@depauw.edu)>

**Repository** CRAN

**Date/Publication** 2016-12-22 00:34:17

## R topics documented:

readJDX . . . . .	2
<b>Index</b>	<b>4</b>

---

`readJDX`*Import a File Written in the JCAMP-DX Format*

---

### Description

This is the function users should call, and supervises the entire import process. The JCAMP-DX standard allows quite a bit of latitude and there are many possible formats. Not all possible formats are supported; error messages will generally let you know what's going on. If you have a file that you feel should be supported but gives an error, please file an issue at Github. The standard allows many variations and its impossible to test them all.

### Usage

```
readJDX(file = "", SOFC = TRUE, debug = 0)
```

### Arguments

<code>file</code>	Character. The file name to import.
<code>SOFC</code>	Logical. "Stop on Failed Check" The JCAMP-DX standard requires several checks of the data as it is decompressed. These checks are essential to obtaining the correct results. However, some JCAMP-DX writing programs do not follow the standard to the letter (for instance we have observed that not all writers put FIRSTY into the metadata, even though it is required by the standard). This option is provided for those <b>advanced users</b> who have carefully checked their original files and want to skip the required checks. It may also be useful for troubleshooting. The default is TRUE i.e. stop when something is not right. This ensures that correct data is returned. Change to FALSE at your own risk. NOTE: Only a few checks can be skipped via this option, as there are some parameters that must be available in order to return an answer.
<code>debug</code>	Integer. The level of debug reporting desired. 1 or higher = basic info about each file, and import progress. 2 = detailed info about x values. 3 = detailed info about y values. 4 = details about the DUP expansion process. 5 = details about the calculation of differences when DIF is in use. In cases where an error is about to stop execution, you get additional information regardless of the debug value.

### Value

A list. The first element is the file metadata. Additional elements contain the extracted x, y data in one or more data frames as follows. If the file contains multiple spectra (not currently supported), there will be one data frame for each spectrum. If the file contains the real and imaginary parts of a 1D NMR spectrum, there will be two data frames, one containing the real portion and the other the imaginary portion. If the file contains one non-NMR spectrum, a single data frame will be returned. In all cases the data frame has elements x and y.

### Included Data Files

The examples make use of data files included with the package. File `SBO.jdx` is an IR spectrum of Smart Balance Original spread (a butter substitute). The spectrum is presented in transmission format, and was recorded on a ThermoFisher instrument. The file uses AFFN compression, and was written with the JCAMP-DX 5.01 standard. Note that even though the y-axis is in percent transmission, in the JDX file it is stored on `[0..1]`. File `PCRF.jdx` is a  $^1\text{H}$  NMR spectrum of a hexane extract of a reduced fat potato chip. The spectrum was recorded on a JEOL instrument. The file uses SQZ DIF compression, and was written with the JCAMP-DX 6.00 standard. File `PCRF_line265.jdx` has a deliberate error in it. See the examples.

### Precision

Internally, this package uses a tolerance factor when comparing values during certain checks. This is currently hardwired to `0.0001*diff(range(values))`. This value works fine in the test files. This appears to be necessary because the original values in the files are text strings of varying lengths which get converted to numerical values. Some precision may be lost but it appears trivial with the current settings.

### See Also

Do `browseVignettes("readJCAMPDX")` for background information, references and supported formats.

### Examples

```
sbo <- system.file("extdata", "SBO.jdx", package = "readJDX")
chk <- readJDX(sbo)
plot(chk[[2]]$x, chk[[2]]$y/100, type = "l", main = "Original Smart Balance Spread",
      xlab = "wavenumber", ylab = "Percent Transmission")

pcrf <- system.file("extdata", "PCRF.jdx", package = "readJDX")
chk <- readJDX(pcrf)
plot(chk[[2]]$x, chk[[2]]$y, type = "l", main = "Reduced Fat Potato Chip Extract",
      xlab = "ppm", ylab = "Intensity")

## Not run:
# Line 265 has an N -> G typo. Try with various levels of debug.
# Even with debug = 0 you get useful diagnostic info.
problem <- system.file("extdata", "PCRF_line265.jdx", package = "readJDX")
chk <- readJDX(problem)

## End(Not run)
```

# Index

readJDX, [2](#)