



information-dynamics-toolkit

JIDT: Java Information Dynamics Toolkit for studying information-theoretic measures of computation in complex systems

[Project Home](#) [Downloads](#) [Wiki](#) [Issues](#) [Source](#) [Administer](#)[Export to GitHub](#)[Home](#)[Getting started](#)[Implemented Measures](#)[Demos](#)[Tutorial](#)[Non-Java environments](#)[Matlab/Octave](#)[Python](#)[R](#)[Julia](#)[Clojure](#)[FAQs](#)[Miscellaneous](#)[For serious developers!](#)[Publications resulting](#)[Edit](#) <<**UseInJulia***How to use the toolkit in Julia*

julia

Updated Sep 9, 2014 by [joseph.lizier](#)

Introduction

The Java code from this toolkit can easily be used in [Julia](#).

Here we give only a brief overview of calling Java code from [Julia](#); several longer examples of using the JIDT toolkit in Julia can be viewed at [JuliaExamples](#).

Using Java objects in Julia

First, you need to install the [JavaCall](#) package in Julia; this is done inside a Julia session by calling: `Pkg.add("JavaCall")`.

You can then run your Java code in Julia as follows:

1. Include the JavaCall package with command: `using JavaCall`;
2. Initialise the JavaCall package and tell it where our `infodynamics.jar` file is, e.g.: `JavaCall.init(["-Xmx128M", "-Djava.class.path=$(jarLocation)"]);`
3. Import the classes you wish to use, e.g. `teClass = @jimport infodynamics.measures.discrete.TransferEntropyCalculatorDiscrete`;
4. Create an instance of the calculator you wish to use, e.g. `teCalc = teClass((jint,jint), 2, 1)`
5. Call methods on the object, passing in the return type of the method and a tuple of argument types and the arguments themselves, e.g. or `jcall(teCalc, "getPastCount", jint,(jint,))` or `jcall(teCalc, "initialise", Void,())` for a void return type and no arguments. See how to specify array types below.

Array conversion -- *single dimensional* arrays can be passed directly back and forth. Take care to indicate their type correctly when passing into a Java method (see [JavaCall docs](#)), e.g. `jcall(teCalc, "addObservations", Void, (Array{jint,1}, Array{jint,1})), sourceArray, destArray);` *Multi-dimensional* arrays however are not yet supported in JavaCall (see [here](#)). We're looking into whether we can make a conversion script to do this manually ...

[Terms](#) - [Privacy](#) - [Project Hosting Help](#)Powered by [Google Project Hosting](#)