
ODVBA

Release v2.1.0 (r386)

Tianhao Zhang

Andreas Schuh

July 18, 2012

Contents

1	Download	iii
2	Changes	iii
2.1	Release 1.0.0 (02/09/2011)	iii
2.2	Release 2.0.0 (02/28/2012)	iii
2.3	Release 2.0.1 (02/28/2012)	iii
2.4	Release 2.1.0 (07/17/2012)	iii
3	Installation	iv
3.1	Prerequisites	iv
	Build Dependencies	iv
	Runtime Requirements	iv
3.2	Configure	iv
3.3	Build	vi
3.4	Test	vi
3.5	Install	vi
4	Documentation	vi
4.1	General Questions	vi
4.2	Software Manual	vi
4.3	API Documentation	vii
5	Example Results	vii
6	Publications	viii

Contact Christos Davatzikos,

People Tianhao Zhang, Christos Davatzikos,

Optimally-Discriminative Voxel-Based Analysis (ODVBA) ¹ can be applied to determine the optimal spatially adaptive smoothing of images, followed by a voxel-based group analysis.

Voxel-based Analysis and Statistical Parametric Mapping (VBA-SPM) ² of imaging data have offered the potential to analyze structural and functional data in great spatial detail, without the need to define a priori regions of interest (ROIs) and assumptions. Gaussian smoothing of images is an important step in VBA-SPM; it accounts for registration errors and integrates imaging signals from a region around each voxel being analyzed. However, it has also become a limitation of VBA-SPM based methods, since it is often chosen empirically, non-optimally, and lacks spatial adaptivity to the shape and spatial extent of the region of interest.

ODVBA provides a mathematically rigorous framework for determining the optimal spatial smoothing of structural and functional images, prior to applying voxel-based group analysis. In order to determine the optimal smoothing kernel, a local discriminative analysis, restricted by appropriate nonnegativity constraints, is applied to a spatial neighborhood around each voxel, aiming to find the direction best highlights the difference between two groups in that neighborhood. Since each voxel belongs to a large number of such neighborhoods, each centered on one of its neighboring voxels, the group difference at each voxel is determined by a composition of all these optimal smoothing directions. Permutation tests are used to obtain the statistical significance of the resulting Optimally-Discriminative VBM (ODVBA) maps.

¹ T. Zhang, C. Davatzikos, "Optimally-Discriminative Voxel-Based Analysis", Proceeding of International Conference on Medical Image Computing and Computer-Assisted Intervention, vol. 13, no.2, pp: 257-265 (2010)

² J. Ashburner, K.J. Friston, "Voxel-based morphometry - The methods", Neuroimage, 11(6) 805-821 (2000)

1 Download

The ODVBA software is freely available under a BSD-style open source license that is compatible with the Open Source Definition by [The Open Source Initiative](#) and contains no restrictions on use of the software.

The full [license](#) text is included with the distribution package and available online.

Please fill [this online form](#) with the appropriate information to receive an email with the download links.

2 Changes

2.1 Release 1.0.0 (02/09/2011)

- Finished first stable SBIA internal release version.

2.2 Release 2.0.0 (02/28/2012)

- Migrated project to [BASIS](#), the new lab standard on software packages.
- Revised command-line tools including a change of their name and CLI.
- Changed names of some C functions.
- Use fixed seed for random number generator if added `--srand-time` option is not given to generate reproducible results.

2.3 Release 2.0.1 (02/28/2012)

- First public release of this software package.
- Reduced size of package by removing unnecessary example data.
- Added Doxygen group for public API functions.
- Moved public API functions used for initialization from `utilities.h` to `odvba.h`.

2.4 Release 2.1.0 (07/17/2012)

- Fixed problem with `odvba-mpi` where each process would create their own random neighborhood if `--srand-time` option is given and neighborhood is created by each process on-the-fly.
- Modified `odvba-mpi` such that only master reads in the data and then sends it to the slaves. This not only fixes before mentioned issue, but also reduces the disk I/O.
- Published new web site based on BASIS support for [Sphinx](#) generated documentation.

3 Installation

An exhaustive list of minimum build dependencies, including the build tools along detailed step-by-step build, test, and installation instructions can be found in the [BASIS how-to guide on software installation](#).

Please refer to this guide first if you are uncertain about the steps summarized here or if you have problems to build, test, or install the software on your system. If this guide does not help you to resolve the issue, please contact us at `sbia-software` at `uphs.upenn.edu`.

3.1 Prerequisites

See the [Build Dependencies](#) for a summary of the prerequisites of this software. These build dependencies have to be installed before ODVBA can be build. The built executables have no further runtime requirements.

For instructions on how to build or install these prerequisites, please refer to the documentation of the respective software package.

Build Dependencies

Before ODVBA can be build from sources, the following software libraries have to be installed.

Package	Version	Description
BASIS	1.3	A meta-project developed at SBIA to standardize and simplify the software development.
ATLAS	3.8.3	Automatically Tuned Linear Algebra Software.
Boost	1.33	The Boost library.
boost-numeric-bindings	20081116	The ATLAS bindings are used in particular.
nifticlib	1.1	The C library for the NIfTI-1 image format.
OpenCV	1.0 or 1.1pre1	Note that with OpenCV 2 (we tried versions 2.0.0, 2.2.0 and 2.3.1), the runtime of ODVBA suffers quite some. It is strongly recommended to build this software with either OpenCV 1.0.0 or OpenCV 1.1pre1.
Open MPI	1.3.3	If an Open MPI library is found on your system, the <code>odvba-mpi</code> executable is build next to the serial <code>odvba</code> executable. The <code>odvba-mpi</code> executable is suitable for execution on either one machine with multiple cores or, in particular, on a cluster with multiple compute nodes. A subset of the total number of permutation tests is then performed by separate MPI processes. The results are assembled in the end by the main process to write the final results.

Runtime Requirements

This software package does not make use of any third-party commands.

3.2 Configure

This software is based on [BASIS](#), a meta-project developed at [SBIA](#) to ease and standardize the software development, build, testing, and packaging. In particular, [CMake](#), a cross-platform, open-source build system, is used for the configuration of the software build. For detailed instructions on how to build a software based on CMake, please refer to the [BASIS how-to guide on software installation](#).

In summary, the steps to configure the so-called build tree are:

1. Extract source files:

```
tar xzf odvba-2.1.0-source.tar.gz
```

2. Create build directory:

```
mkdir odvba-2.1.0-build
```

3. Change to build directory:

```
cd odvba-2.1.0-build
```

4. Run CMake to configure the build tree:

```
cmake ../odvba-2.1.0-source
```

- Press `c` to configure the build system and `e` to ignore warnings.
- Set `INSTALL_PREFIX` and other CMake variables and options.
- Continue pressing `c` until the option `g` is available.
- Then press `g` to generate the configuration files for the selected build tool.

See the BASIS how-to guide on software installation for a documentation of the [default configuration options](#) provided by BASIS. Additionally, the following CMake options are considered.

NiftiCLib_DIR <dir>

Directory where the [nifticlib](#) library is installed.

BoostNumericBindings_DIR <dir>

Directory where the files of the [boost-numeric-bindings](#) were extracted to.

NiftiCLib_DIR <dir>

Installation directory of the [nifticlib](#) library, e.g., `/usr/local`.

OpenCV_DIR <dir>

Installation directory of the [OpenCV](#) library.

BLAS_atlas_LIBRARY <file>

Path of the ATLAS library.

BLAS_cblas_LIBRARY <file>

Path of the C BLAS library.

BLAS_f77blas_LIBRARY <file>

Path of the Fortran BLAS library.

Boost_INCLUDE_DIR <dir>

Include directory of the [Boost](#) library, e.g., `/usr/include`.

Boost_LIBRARY_DIRS <dir>

Directory where the [Boost](#) libraries are installed, e.g., `/usr/lib`.

Note: Some of these options are marked as advanced and only visible if you enable the view of advanced options.

3.3 Build

After the configuration of the build tree, ODVBA can be build using [GNU Make](#):

```
make
```

3.4 Test

After the build of the software, optionally run the tests using the command:

```
make test
```

In case of failing tests, re-run the tests, but this time by executing [CTest](#) directly with the `-V` option to enable verbose output and redirect the output to a text file:

```
ctest -V >& odvba-test.log
```

And send the file `odvba-test.log` as attachment of the issue report to `sbia-software at uphs.upenn.edu`.

3.5 Install

The final installation copies the built files and additional data and documentation files to the installation directory specified using the `INSTALL_PREFIX` option during the configuration of the build tree:

```
make install
```

After the successful installation, the build directory can be removed again.

Note: The software can be deinstalled again by running the installed `uninstall-odvba` script.

4 Documentation

A [PDF](#) Version of this documentation is available online and included with the distribution package.

4.1 General Questions

If you have any questions regarding ODVBA, please contact us at `sbia-software at uphs.upenn.edu`. Emails sent to this address will be forwarded to the right people at SBIA who will then reply to you and try to answer your questions.

4.2 Software Manual

For guidance on the use of the command-line programs, please refer to the [Software Manual \(ref\)](#) included with the distribution package in PDF, as well as the help output of the programs:

```
odvba-index --help
odvba-ni --help
odvba --help
odvba-mpi --help
```

4.3 API Documentation

The API documentation is generated from in-source comments using Doxygen.

- API Functions v2.1 v2.0

5 Example Results

It is demonstrated in *Figure 1* that ODVBA is much more powerful to detect the true atrophy hidden in the data than the traditional methods, i.e., SPM³ and SnPM⁴. Moreover, ODVBA possesses the spatial adaptivity to the shape and spatial extent of the region of interest, while SPM and SnPM do not.

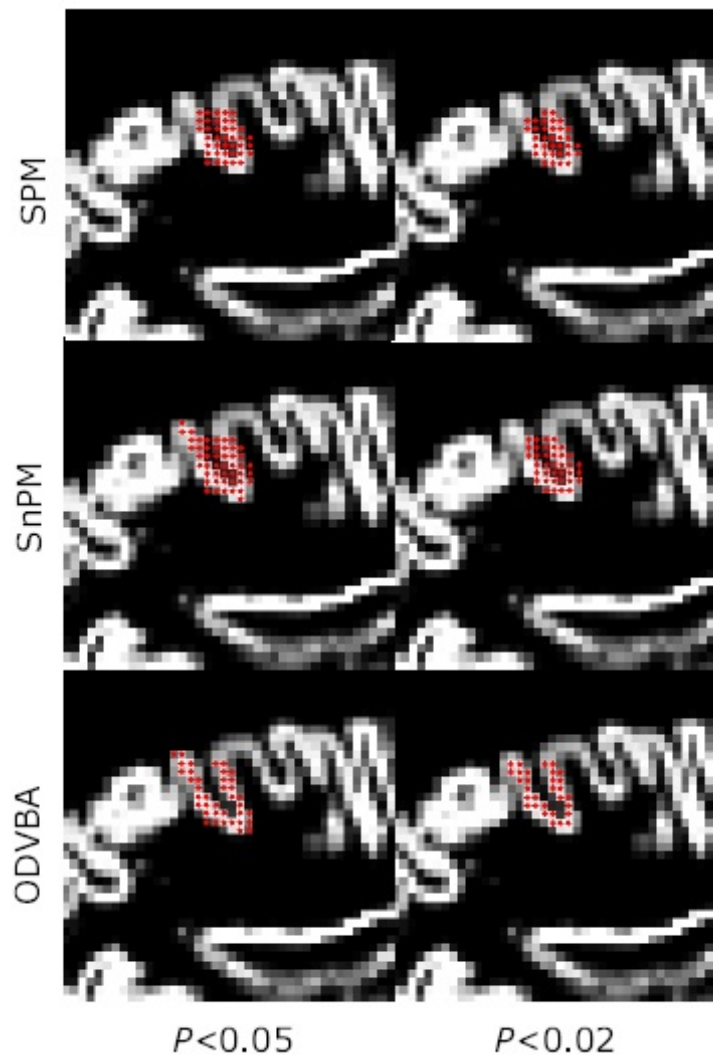


Figure 1: Comparison of simulated atrophy detection of ODVBA to SPM and SnPM.

The ODVBA method precisely describes the “U”-like shape of the atrophy (which is created in the simulated data), while SPM and SnPM do not. In SPM and SnPM, the images are blurred so that the results are not accurate.

³ J. Ashburner, K.J. Friston, “Voxel-based morphometry - The methods”, *Neuroimage*, 11(6) 805–821 (2000)

⁴ T.E. Nichols, A.P. Holmes, “Nonparametric permutation tests for functional neuroimaging: a primer with examples”, *Human Brain Mapping*, vol. 15, no. 1, pp: 1-25 (2002)

6 Publications

Please cite the following paper when any results were produced with the help of ODVBA:

- T. Zhang, C. Davatzikos, “Optimally-Discriminative Voxel-Based Analysis”, Proceeding of International Conference on Medical Image Computing and Computer-Assisted Intervention, vol. 13, no.2, pp: 257-265 (2010)