BoneJ Module Documentation

The following is a documentation of how to use the BoneJ modules and scripts that have been developed for the Quantitative Bone Texture Project. The BoneJ Module scrips were written in Python with Fiji scripts being written with Jython. To run these scripts there needs to be a working installation of Fiji with BoneJ installed, additionally files must be 3D 8-bit binary files. All functions utilize a macro file that is included with the modules written in Jython that is run within Fiji.

The scripts rely on invoking Fiji from the system’s command line. Fiji is called in headless mode, thus no GUI will appear. The macro files with Fiji scripts written in Jython are run to return the final results.

Trabecular Thickness

Fits spheres into every foreground voxel of a segmented image to determine the thickness of the trabecular microstructure. The diameter of the largest sphere that is able to fit inside the foreground voxel and contains the point for each point is measured by the plugin. The plugin outputs the mean thickness of the sample, standard deviation of the sample, and the max thickness value.

Function

def Thickness(array, voxel\_size, fiji\_path, showMaps, maskArtefacts)

array = Numpy array of the image

voxel\_size = Size of the voxels in the image, ex. [51.2,51.2,51.2]. Thickness module assumes microns.

fiji\_path = Path to the users local Fiji installation

showMaps = True will generate a thickness map which is saved as optional\_dict. False will generate no thickness map.

maskArtefacts = True will remove foreground voxels that are not present in the original image from the final thickness map. Always recommended to select True, the artifacts can cause bias and distortions in the image.

Results

Mean Tb. Th = The mean trabecular thickness value (microns) of the image.

Std Tb. Th = The standard deviation of the trabecular thickness (microns) values.

Max Tb. Th = The max trabecular thickness value (microns) of the image.

If showMaps is set to True a thickness map is generated.

Trabecular Spacing

Fits spheres into every background voxel of a segmented image to determine the thickness of the of the marrow space between trabeculae. The diameter of the largest sphere that is able to fit inside the background voxel and contains the point for each point is measured by the plugin. spacing value.

Function

def Spacing(array, voxel\_size, fiji\_path, showMaps = True , maskArtefacts = False):

array = Numpy array of the image

voxel\_size = Size of the voxels in the image, ex. [51.2,51.2,51.2]. Spacing module assumes microns.

fiji\_path = Path to the users local Fiji installation

showMaps = True will generate a spacing map which is saved as optional\_dict. False will generate no spacing map.

maskArtefacts = True will remove background voxels that are not present in the original image from the final spacing map. Always recommended to select True, the artifacts can cause bias and distortions in the image.

Results

Mean Tb. Sp = The mean trabecular spacing value (microns) of the image.

Std Tb. Sp = The standard deviation of the trabecular spacing (microns) values.

Max Tb. Sp = The max trabecular spacing value (microns) of the image.

If showMaps is set to True a spacing map is generated.

Anisotropy

Assigns a numerical value on a scale of 0-1 to quantify trabecular bone’s directionality. Degree of anisotropy is representative of the microstructure’s orientation. The closer to 0 the more isotropic a bone, the closer to 1 the more anisotropic a bone.

Plugin uses mean intercept length vectors to calculate the degree of anisotropy. Vectors of equal length all emanating from the same random point within the image are drawn throughout. As the vectors change from the foreground to the background this is counted as an intercept for that specific vector. The vector length divided by the number of boundary hits (when foreground changes from background) gives the mean intercept length. A point cloud is generated which is representative of the vectors multiplied by the mean intercept length. The equation of an ellipsoid is solved that fits this point cloud. This gives eigenvalues related to the lengths of the axis of the ellipsoid along with eigenvectors that give the orientation of the ellipsoid axes. The Degree of Anisotropy is.

Plugin finds mean intercept length vectors from n directions where points change from the background to the foreground. Parallel lines over an input image are drawn where each line segment in an image sample points from background to foreground. The MIL vectors are then plotted into a point cloud around the origin. The equation of an ellipsoid is solved that fits the point cloud. The Degree of Anisotropy is measured based on the ellipsoid radii.

Function

def Anisotropy(array, voxel\_size, fiji\_path, NDirs = 2000, nLines = 10000, samplingincrement = 1.73, radii = False, eigens = False, MILvectors = False):

Directions = Number of times the sampling is performed from various directions. Min value is 9. Recommended value is 2000.

Lines per direction = The number of parallel lines drawn in each direction. Recommended value 10000.

Sampling Increment = The distance between sampling points along a line. Minimum, default, and recommended value is 1.73.

radii = True of False. If True is input the radii of the fitted ellipsoid results are output.

eigens = True or False. If True is input the eigenvectors and values of the fitted ellipsoid is output

It is best to run a convergence analysis to determine the best parameters for Anisotropy. Recommended parameters may not give stable results in a reasonable amount of time. An ImageJ macro has been included for this.

Results

Degree of anisotropy = Quantitative value representing the directionality of trabecular bone sample. 0 is isotropic, 1 is anisotropic. The higher the value the more orientation in the microstructure of the bone. (Dimensionless)

Radii of fitted ellipsoid = Radii lengths, a<b<c of the ellipsoid fit to the point cloud. Used to calculate degree of anisotropy. . Only output if radii is set to True. (microns)

Eigenvectors and values = Values of the x,y,,z components of the three eigvenctors of the ellipsoid fit to the point cloud (m00,m01,m02..). Eigenvalues are listed as D1,D2,D3 which correspond , a,b,c are the radii of the ellipsoid fit to the point cloud vector.

Anisotropy Convergence Script

*//number of directions to draw probes*

nDirsMax **=** 32768**;** *//<- edit to suit your needs*

*//number of lines per direction*

nLinesMax **=** 1024**;** *//<- edit to suit your needs*

*// --- No need to edit the rest*

row **=** 0**;**

setBatchMode**(true);**

**for** **(**nDirs **=** 16**;** nDirs **<=** nDirsMax**;** nDirs **\*=** 2**){**

**for** **(**nLines **=** 1**;** nLines **<=** nLinesMax**;** nLines **\*=** 2**){**

startTime **=** getTime**();**

run**(**"Anisotropy"**,** "inputimage=net.imagej.ImgPlus@73956688 directions="**+**nDirs**+**" lines="**+**nLines**+**" samplingincrement=1.73 recommendedmin=true printradii=true printeigens=true displaymilvectors=false instruction=\"\""**);**

endTime **=** getTime**();**

duration **=** endTime **-** startTime**;**

setResult**(**"nDirs"**,** row**,** nDirs**);**

setResult**(**"nLines"**,** row**,** nLines**);**

setResult**(**"Duration"**,** row**,** duration**);**

updateResults**();**

row**++;**

**}**

**}**

**(Probably should turn this into Python script as well)**

Results

Table with anisotropy results and a second table with a list of nDirs, nLines, and the duration of the run are generated.

Connectivity

Plugin determines the number of connected structures in the image. The connected structures are representative of trabeculae in a trabecular network. Connectivity is determined from measuring the Euler characteristic denoted χ. The Euler characteristic is a topologically invariant value meant to describe a shape or structure regardless of how it is bent. It is defined as χ = objects – handles + cavities. A handle is analogous to a hole through an object, while a cavity hole enclosed inside of an object.

Before Connectivity is run the plugin Purify is run within the script. Purify is a preprocessing step that filters an image by removing all particles but the largest foreground and background particles. Once purify is run there is a single connected bone phase and a single connected marrow phase. From there the Euler characteristic is calculated for every bone voxel in the image. The intersection of voxels and stack edges is checked to calculate the bone’s contribution to the Euler characteristic of the bone it is connected to. Connectivity is 1- Δχ, connectivity density is defined as Connectivity/stack volume.

Function

def Connectivity(array,voxel\_size,fiji\_path):

array = Numpy array of the image

voxel\_size = Size of the voxels in the image, ex. [51.2,51.2,51.2]. Spacing module assumes microns.

fiji\_path = Path to the users local Fiji installation

Results

Euler characteristic = Euler characteristic of the sample if it were floating in space

Corrected Euler = The contribution of the bone sample to the Euler characteristic of the bone to which it is connected

Connectivity = Connectivity of the image described as the number of trabeculae (

Connectivity Density = The number of trabeculae per unit volume (

Area Volume Fraction

Calculates Bone Volume/Total Volume, the volume of mineralized bone per unit volume of the sample. Foreground voxels which represent bone are divided by the total number of voxels in the image.

Function

def Area\_VolumeFraction(array,voxel\_size,fiji\_path):

array = Numpy array of the image

voxel\_size = Size of the voxels in the image, ex. [51.2,51.2,51.2]. Spacing module assumes microns.

fiji\_path = Path to the users local Fiji installation

Results

Bone volume: Volume of bone voxels (microns)

Total volume: Volume of entire image (microns)

BV/TV: Ratio of Bone volume to total volume of the image (dimensionless)

References

Domander R, Felder AA, Doube M. 2021 BoneJ2 - refactoring established research software. Wellcome Open Res. **6**. doi:[10.12688/wellcomeopenres.16619.1](https://doi.org/10.12688/wellcomeopenres.16619.1)

Doube M, Kłosowski MM, Arganda-Carreras I, Cordeliéres F, Dougherty RP, Jackson J, Schmid B, Hutchinson JR, Shefelbine SJ. BoneJ: free and extensible bone image analysis in ImageJ. Bone 47 1076-1079 (2010). doi: 10.1016/j.bone.2010.08.023

Dougherty R, Kunzelmann K (2007), Computing local thickness of 3D structures with ImageJ, Microsc. Microanal., 13: 1678-1679, [<doi:10.1017/S1431927607074430>](http://dx.doi.org/10.1017/S1431927607074430)

Hildebrand T, Rüegsegger P (1997), A new method for the model-independent assessment of thickness in three-dimensional images, J. Microsc., 185: 67-75, [<doi:10.1046/j.1365-2818.1997.1340694.x>](http://dx.doi.org/10.1046/j.1365-2818.1997.1340694.x)

Odgaard A (1997), Three-dimensional methods for quantification of cancellous bone architecture, Bone, 20, 315-328, [<doi:10.1016/S8756-3282(97)00007-0>](http://dx.doi.org/10.1016/S8756-3282(97)00007-0)

Harrigan TP, Mann RW (1984), Characterization of microstructural anisotropy in orthotropic materials using a second rank tensor, J Mater Sci, 19, 761-767, [<doi:10.1007/BF00540446>](http://dx.doi.org/10.1007/BF00540446)

Odgaard A, Gundersen HJG (1993), Quantification of connectivity in cancellous bone, with special emphasis on 3-D reconstructions, Bone 14: 173-182, [<doi:10.1016/8756-3282(93)90245-6>.](http://dx.doi.org/10.1016/8756-3282(93)90245-6)

Toriwaki J, Yonekura T (2002), Euler number and connectivity indexes of a three dimensional digital picture, [Forma 17: 183-209](http://www.scipress.org/journals/forma/abstract/1703/17030183.html)

Rasband, W.S., ImageJ, U. S. National Institutes of Health, Bethesda, Maryland, USA, https://imagej.nih.gov/ij/, 1997-2018.

BoneJ2

ImageJ 1.53q