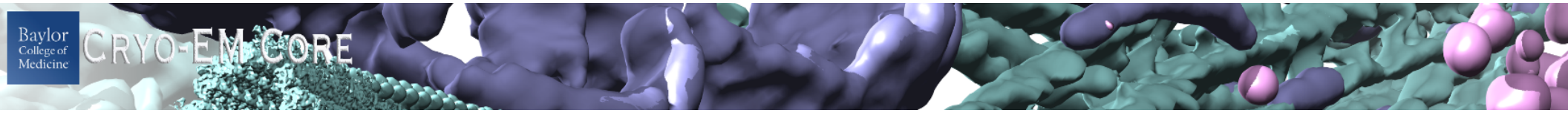


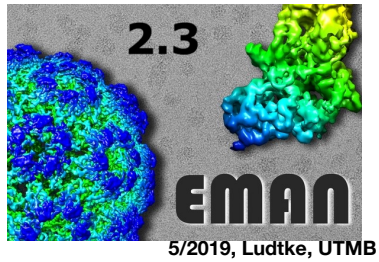
Python and EMAN2

Steve Ludtke

Charles C. Bell Professor
Biochemistry and Molecular Biology
Director, CryoEM/CryoET Core
Co-director CIBR Center
Baylor College of Medicine

Baylor College of Medicine

VERNA & MARRS McLEAN
DEPARTMENT OF
BIOCHEMISTRY AND
MOLECULAR BIOLOGY



Resources

- http://eman2.org/Library - Docs for the C++/Python library
- http://eman2.org/Eman2Metadata - header parameters
- use help()
- e2.py - interactive python interface for EMAN2
- examples/ - lots of small programs, some useful, some just simple code

Key Classes

- EMData - an image in 1-3D, this is the primary class in EMAN2
- Transform - 2/3D Transformation with translation, many conventions
- Symmetry - specifies symmetry transforms
- Processor - an image processing operation, often EMData.process
- Aligner - 2D or 3D registration, often EMData.align
- Averager - Various averaging-like operations
- Reconstructor - 3D reconstruction
- LSXFile - To safely manipulate .lst files

Key Functions

- display(object,[single]) - interactive in e2.py, blocks in regular python
- js_open_dict(filename) - Open a JSON file as a dictionary-like object
- to_numpy(EMData) - Convert an EMData object to a NumPy array
- from_numpy(array) - Convert a NumPy array to EMData
- test_image(size=<n>,type=<i>) - Generate various 2-D test images
- os.listdir(<path>) - useful standard python function
- good_size(boxsize) - next good box size larger than specified
- base_name(filename) - returns unique portion of image filename
- info_name(filename) - returns json file associated with filename in project
- EMUtil.get_image_count(filename) - number of images in file

EMData

- img=EMData(nx,ny,nz)
- img=EMData(filename,n,[hdr_only],[region])
- img.read_image(filename,n,[hdr_only],[region]) - replaces current image
- list=EMData.read_images(filename,[list of #s],[header only])
- img.write_image(fsp,image #,[filefmt],[hdr_only],[region],[numtype],[endian])
- img[x,y,z] - returns (or sets with assignment) value at coordinates
- img["name"] - returns header value (if exists), or sets with assignment
- img.get_attr_dict() - gets all header items as a dictionary
- img.to_zero() - all pixels to 0.0
- img.to_one() - all pixels to 1.0
- img2=img.get_clip(Region(x0,y0,[z0],nx,ny,[nz])) - extract subimage
- img.insert_clip(target,(x,y,z)) - insert subimage into another image

EMData

- fft=img.do_fft() - compute FFT
- img=fft.do_ift() - compute inverse FFT img2=img.process("processor",{parms})
- img.process_inplace("processor",{parms})
- img=img1.align("aligner",img2,{params})
- similarity=img1.cmp("cmp",img2,{params})
- img2=img+img1
- img.add(img2)
- img.mult(img2)
- img.calc_fourier_shell_correlation(img2) - see help()
- locs=img.calc_n_highest_locations(<n>)
- calc_radial_dist(<n>,<r0>,<dr>,[inten]) - radial curve, real or FFT

Transform

- xf=Transform() - identity matrix with no translation
- xf=Transform({"type":"eman","az":5,"alt":10,"phi":15})
- xf.inverse() - inverse of the Transformation including translation
- xf.get_rotation("eman") - e2help.py rotationtypes
- xf.printme() - print the actual matrix
- xf*xf2 - multiply matrices
- Transform.icos_5_to_2() - transform to go from icos 5-fold on Z to 2-2-2
- Transform.tet_3_to_2() - tetrahedral 3-fold on Z to 2-fold on Z

Symmetry

- sym=Symmetries.get("icos") - e2help.py symmetries
- n=sym.get_nsym() - number of asymmetric units
- list=sym.get_syms() - list of all Transforms for this symmetry
- list=sym.gen_orientations("name",{parms}) - make a set Transforms within an asymmetric unit