BATCH TUTORIAL

==========

Adressed topics

==========

PREREQUISITES:

- the ANT gui is open
- a project is defined and the proper data structure is set (see tutorial_atlasRegistration.doc)
- to apply functions over all or some specific mice you have to select the respective mouse folders in the left listbox of the ANT main gui in advance

How to batch a specific function

- The help of most functions contains examples which can be copied and customized
- Most of the functions, when applied successfully, should return some lines of code stored in the variable 'anth' (i.e. ant-history) in the Matlab workspace. You may type '**char(anth)**' in the command window to see the parameter settings and the function name and function's input arguments. At this point you may copy and paste the respective lines of codes for the previous function to a Matlab m-file. You can also type **uhelp(anth)** to view the code in a help window. In the uhelp-window, you can also select and re-run some lines of codes (first select the lines in the uhelp window than use context menu/evaluate selection to rerun the specific function).
- Most functions have 2, some have 3 input arguments: The first input argument asks whether to open [1] the gui/pop up window with function-specific parameters or to work in silent mode [0]. In most cases [0], i.e. no parameter window is the preferred choice. A call of a function without input argument will most likely result in opening the function-specific parameter setting window.

BRUKER IMPORT

→ currently, there is no proper batch mode for the Bruker import (I work on this)
→ workaround: import all files from all mice and remove unneeded files via batch later on

SELECT MOUSE FOLDERS

- manually, select all/some mice folders from the left list box from the ANT gui
- Alternatively, use **antcb('selectdirs')** function to select all mouse folders or mouse folders that contain or not contain a specific file/image or via index or via list → for help type **antcb('selectdirs?')**

Example:

```
antcb('selectdirs','all')                 % select all mouse folders
antcb('selectdirs','x_t2.nii','find')     % : select only those mouse folders containing the file 'x_t2.nii'
antcb('selectdirs','x_t2.nii','not')      %: select only those mouse folders not containing the file 'x_t2.nii'
```

RENAMIMG

- [xrename] this function allows to RENAME/DELETE/EXTRACT/EXPAND/COPY file(s) → for help type **help xrename**

Examples:

This example shows how to delete the files '1_Localizer_1.nii' and 'B0Map-ADJ_B0MAP_1.nii' and to rename 'T2_TurboRARE_1.nii' to 't2.nii' in one step

```
z.files={ '1_Localizer_1.nii'       '##'   ''      % 1_Localizer_1.nii' is deleted (use ## to delete files)
          'B0Map-ADJ_B0MAP_1.nii'   '##'   ''      % 'B0Map-ADJ_B0MAP_1.nii' is deleted
          'T2_TurboRARE_1.nii'      't2'   '' };   % 'T2_TurboRARE' is renamed to 't2.nii'
xrename(0 ,z.files(:,1),z.files(:,2),z.files(:,3));  % run function without popup window (1st input is 0)
```

Make copy of 'T2_TurboRARE_1.nii' and name it 't2.nii'

```
xrename(0, 'T2_TurboRARE_1.nii',  't2.nii'   ,     ':' ) % use ':' to copy files
```

Make extract the 9th 3D volume from a 4D file ('MSME-T2-map_20slices_1.nii') and store it as 'test.nii'

```
xrename(0, 'MSME-T2-map_20slices_1.nii' , 'test.nii' ,9)  ;
```

DISTRIBUTE FILES

- [xdistributefiles] this function copies file(s) (any format) from outside into pre-selected mouse-folders → for help type **help xdistributefiles**

Examples:

This example copies the 'HippocampusMask.nii' from another directory into each a priory selected mouse folder

```
z.files={ 'O:\data2\longt\templates\HippocampusMask.nii' ,  'HippocampusMask.nii' };
xdistributefiles(0,z)              % run function without popup window (1st input is 0)
```

IMAGE CALCULATION

- [xcalc] this function allows image calculations (fuse images, make masks, mask images, threshold images, average images…) within or across mouse folders → for help type **help xcalc**

Examples:

This example calculates the mean over the 4th dimension of the 4D-volume 'epi_mre_2.nii', and stores the resulting 3D volume as 'avg_epi_mre_2.nii'

```
z=[];
z.niftis     = { 'epi_mre_2.nii' };     % IMAGE(S) to manipulate
z.evalstring = 'mean(@i,4)';            % string to evaluate->see help
z.outName    = '@avg_';                 % output file name
z.outDir     = 'local';                 % output directory:  ["local"] refers to local mouse folder
xcalc(0,z);                             % run function without popup window (1st input is 0)
```

COREGISTRATION

- [xcoreg] - This function allows to (rigid) co-register one image onto another image. You can optionally apply the resulting transformation rule to other images → for help type **help xcoreg**

Example:

Register 'f_avepi_mre_2.nii' onto 't2.nii' and apply the transformation rule also for 'm123.nii' and 'm124.nii' (at the end 'f_avepi_mre_2.nii', 'm123.nii' and 'm124.nii' will be coregistered to 't2.nii'). The resulting images have the prefix 'r'

```
z=[];
z.TASK={ '[2]' };
z.targetImg1={ 't2.nii' };            % TARGE IMAGE , this is the reference image
z.sourceImg1={ 'f_avepi_mre_2.nii' }; % SOURCE IMAGE, image to transform
z.sourceImgNum1=[1];
z.applyImg1={ 'm123.nii' and 'm124.nii' }; % other images to transform
z.cost_fun='nmi';
z.sep=[4  2  1  0.5  0.1  0.05];
z.tol=[0.01  0.01  0.01  0.001  0.001  0.001];
z.fwhm=[7  7];
z.centerering=[0];
z.reslicing=[0];                  % reslice image: [0]no just change hdr.mat, [1]yes
z.interpOrder='auto';
z.prefix='r';
z.warping=[0];                    %nonlinear warp: [0]no, [1]additional notlinear warping
z.warpPrefix='warped';
xcoreg(0,z);                      % run function without popup window (1st input is 0)
```

SLICEWISE RIGID/AFFINE/NONLINEAR REGISTRATION

- [xregister2d] - This function allows a slice-wise 2d registration using rigid and/or affine and/or nonlinear transformation method → for help type **help xregister2d**

Example:

Slicewise register 'f_avepi_mre_2.nii' to 't2.nii', apply the resulting transformation to no other images. Use rigid and nonlinear (bspline) transformation. Because slice-to slice assignment is fine, register pairwise (slice 1 to slice 1, than 2 to 2 …and 9 to 9, as defined in x.sliceAssign), the resulting output is 'pf_avepi_mre_2.nii' (prefix: p)

```
x=[];
x.refIMG=            { 't2.nii' };      % (<<) SELECT REFERENCE IMAGE (example: t2.nii)
x.sourceIMG=         { 'f_avepi_mre_2.nii' };     % (<<) SELECT IMAGE to calculate the transformation
x.applyIMG=          { '' };  % IMAGE(S) to apply transformation (if empty sourceIMG is transformed only
x.prefix=            'p';     % this prefix is used for the output file >>[prefix+"name of applyIMG"]
x.rigid=             [1];     % do rigid transformation [0|1]
x.affine=            [0];     % do affine transformation [0|1]
x.bspline=           [1];     % do b-spline transformation [0|1]
% ========================================================================================================
x.InterpOrder=       [0];     % InterpolationOrder: [0]nearest neighbor, [1]trilinear interpolation, [3]cubic
x.xyresolution=      'ref';   % Final XY-resolution: "ref" from refIMG, "source" from sourceIMG
x.preserveIntensity=[0];      % preserve intensity: [0] no, [1] preserve min-max-range [2] preserve mean+SD
x.sliceAssign=      '[1 1;2 2;3 3;4 4;5 5;6 6;7 7;8 8;9 9]'  % Slice-to-slice assigment of refIMG & sourceIMG: "auto" use
                                image information; otherwise specify as pairwise vector/matrix such as [15 1] or [15 1; 16 2]
x.reslice2refIMG=    [1];     % force to match dimensions of refImage
x.createMask=        [0];     % create binary MaskImage  [ used slices contain "1"-elements; filename: "newfilename+_mask"]
% ========================================================================================================
x.cleanup=           [1];     % remove unnecessary data
x.keepFolder=        [1];     % keeps local 2d-elastix folder
xregister2d(0,x)              % run function without popup window (1st input is 0)
```

NORMALIZE: TRANSFORM T2.NII FROM MOUSE SPACE TO ATLAS SPACE

- [xwarp3] - This function transforms the 't2.nii' from mouse space to Allen Atlas space (rigid+affine+nonlinear steps) → for help type **help xwarp3**

Example:

For all apriory selected mouse folders perform the 4 necessary steps: [1] initialize warping (copy necessary files, skullstripping), [2a] automatically coregister the image (defined by setting 'autoreg' to 1) , [3] segment the image (make 3 tissue compartments) and [4] nonlinear transform the image

```
xwarp3('batch','task',[1:4],'autoreg',1);
```

TRANSFORM OTHER IMAGES TO ATLAS SPACE or MOUSE SPACE

- [doelastix] - This function allows to transform other images to Allen space, once the transformation parameters for 't2.nii' has been calculated (xarp3 has to be run before) → for help type **help doelastix**

-This function is also used to transform images from Allen Space to mouse space

Example:

Transform the 'lesion_64.nii' from this 3 mouse folders to Allen space, using 3rd order interpolation

```
files={'O:\harms1\harms3_lesionfill\dat\s20150505SM02_1_x_x\lesion_64.nii'
       'O:\harms1\harms3_lesionfill\dat\s20150505SM03_1_x_x\lesion_64.nii'
       'O:\harms1\harms3_lesionfill\dat\s20150505SM09_1_x_x\lesion_64.nii'}
doelastix(  1 , [],  files   ,3 ,'local' );
```

-NOTE: This function has to be refurbished. In this state the files have to be addressed in a fullpath mode.

- The 1st input defines the direction [1] to Allen space, [-1] to mouse space,
- The 4th argument defines the interpolation order (use 0 for next neighbour interpolation, for masks or atlasses)
- The 5th argument 'local' is mandatory to apply the necessary initial rigid body transformation for backward transformation

Example:

Transform the atlas 'ANO.nii' from Allen space to mouse space

-note: that the 1st input argument is now '-1', indicating a back projection to mouse space

```
 files={'O:\harms1\harms3_lesionfill\dat\s20150505SM02_1_x_x\ANO.nii'
        'O:\harms1\harms3_lesionfill\dat\s20150505SM03_1_x_x\ANO.nii'
        'O:\harms1\harms3_lesionfill\dat\s20150505SM09_1_x_x\ANO.nii'}
doelastix(  -1 , [],  files   ,3 ,'local' );
```

Extract region-based parameters from an Image

- [xgetlabels3] - This function extracts region-based parameters from an image. The image/volume can be in Allen atlas space or in mouse space. The output is saved in a multi-sheet excel file.

→ **help for xgetlabels3 has to be written**, but see explanation in the parameter settings window of 'xgetlabels3'

Example:

For all Allen atlas regions extract the parameters (frequency (counting), percent overlap, volume (qmm), reference volume (the reference is the Allen atlas image), mean, standard deviation median, minimum and maximum, as indicated by [1]) from the image 'x_pf_avepi_mre_2.nii'. Note that 'x_pf_avepi_mre_2.nii' is the transformed result of ('pf_avepi_mre_2.nii) in Allen space (prefix 'x_' always indicates a projection to Allen space, whereas prefix 'ix' indicates a back projection of an Image from Atlas to mouse space)

```
z=[];
z.files={ 'O:\data2\jing2\dat\MMRE_age_20w_mouse1_cage6_13122017\x_pf_avepi_mre_2.nii'
          'O:\data2\jing2\dat\MMRE_age_20w_mouse2_cage6_13122017\x_pf_avepi_mre_2.nii' };
          'O:\data2\jing2\dat\MMRE_age_20w_mouse3_cage6_13122017\x_pf_avepi_mre_2.nii' };
          'O:\data2\jing2\dat\MMRE_age_20w_mouse4_cage6_13122017\x_pf_avepi_mre_2.nii' };
          'O:\data2\jing2\dat\MMRE_age_20w_mouse5_cage6_13122017\x_pf_avepi_mre_2.nii'
        };
z.masks        ='';      % <optional> use additional mouse-specific mask, example: lesion mask
z.filetag      ='';
z.masktag      ='';
z.hemisphere   ='both';  % aggregate resulting parameters over 'both', 'left' or 'right' hemisphere
z.threshold    =[0];     % lower intensity threshold→ aggregate parameters only for values above threshold
z.space        ='allen'; % indicates the space ['allen' vs 'native'](must match with the space of the input
z.frequency    =[1];     % the next fields indicate which parameters should be aggregated [0|1]
z.percOverlapp =[1];
z.volref       =[1];
z.vol          =[1];
z.mean         =[1];
z.std          =[1];
z.median       =[1];
z.min          =[1];
z.max          =[1];
xgetlabels3(0,z);        % run function without popup window (1st input is 0)
```