RUN

#!/bin/sh

#{{{ FEEDS - FSL Evaluation and Example Data Suite

# Stephen Smith and Matthew Webster, FMRIB Image Analysis Group

#

# Copyright (C) 2001-2011 University of Oxford

#

# Part of FSL - FMRIB's Software Library

# http://www.fmrib.ox.ac.uk/fsl

# fsl@fmrib.ox.ac.uk

#

# Developed at FMRIB (Oxford Centre for Functional Magnetic Resonance

# Imaging of the Brain), Department of Clinical Neurology, Oxford

# University, Oxford, UK

#

#

# LICENCE

#

# FMRIB Software Library, Release 4.0 (c) 2007, The University of Oxford

# (the "Software")

#

# The Software remains the property of the University of Oxford ("the

# University").

#

# The Software is distributed "AS IS" under this Licence solely for

# non-commercial use in the hope that it will be useful, but in order

# that the University as a charitable foundation protects its assets for

# the benefit of its educational and research purposes, the University

# makes clear that no condition is made or to be implied, nor is any

# warranty given or to be implied, as to the accuracy of the Software,

# or that it will be suitable for any particular purpose or for use

# under any specific conditions. Furthermore, the University disclaims

# all responsibility for the use which is made of the Software. It

# further disclaims any liability for the outcomes arising from using

# the Software.

#

# The Licensee agrees to indemnify the University and hold the

# University harmless from and against any and all claims, damages and

# liabilities asserted by third parties (including claims for

# negligence) which arise directly or indirectly from the use of the

# Software or the sale of any products based on the Software.

#

# No part of the Software may be reproduced, modified, transmitted or

# transferred in any form or by any means, electronic or mechanical,

# without the express permission of the University. The permission of

# the University is not required if the said reproduction, modification,

# transmission or transference is done without financial return, the

# conditions of this Licence are imposed upon the receiver of the

# product, and all original and amended source code is included in any

# transmitted product. You may be held legally responsible for any

# copyright infringement that is caused or encouraged by your failure to

# abide by these terms and conditions.

#

# You are not permitted under this Licence to use this Software

# commercially. Use for which any financial return is received shall be

# defined as commercial use, and includes (1) integration of all or part

# of the source code or the Software into a product for sale or license

# by or on behalf of Licensee to third parties or (2) use of the

# Software or any derivative of it for research with the final aim of

# developing software products for sale or license to a third party or

# (3) use of the Software or any derivative of it for research with the

# final aim of developing non-software products for sale or license to a

# third party, or (4) use of the Software to provide any service to an

# external organisation for which payment is received. If you are

# interested in using the Software commercially, please contact Isis

# Innovation Limited ("Isis"), the technology transfer company of the

# University, to negotiate a licence. Contact details are:

# innovation@isis.ox.ac.uk quoting reference DE/1112.

#}}}

# the next line restarts using wish \

SGE\_ROOT=""; export SGE\_ROOT; exec $FSLTCLSH "$0" "$@"

#{{{ perror

proc perror { testimage scale } {

global FSLDIR FEEDSDIR PTHRESH MAXPERROR

if { ! [ imtest $FEEDSDIR/results/$testimage ] } {

set PERROR 100

puts "No output image created"

} else {

exec sh -c "${FSLDIR}/bin/fslmaths $FEEDSDIR/data/$testimage -sub $FEEDSDIR/results/$testimage -sqr $FEEDSDIR/results/errsq -odt float"

exec sh -c "${FSLDIR}/bin/fslmaths $FEEDSDIR/data/$testimage -sqr $FEEDSDIR/results/meansq -odt float"

set PERROR [ expr int ( $scale \* 10000.0 \* sqrt ( \

[ exec ${FSLDIR}/bin/fslstats $FEEDSDIR/results/errsq -m ] / \

[ exec ${FSLDIR}/bin/fslstats $FEEDSDIR/results/meansq -m ] ) ) / 100.00 ]

puts "% error = $PERROR"

}

if { $PERROR > $PTHRESH } {

puts "Warning - test failed!"

}

if { $PERROR > $MAXPERROR } {

set MAXPERROR $PERROR

}

return $PERROR

}

#}}}

#{{{ simpleperror

proc simpleperror { a b denom } {

global PTHRESH MAXPERROR

set PERROR [ expr int ( 10000.0 \* ( $a - $b ) / $denom ) / 100.00 ]

if { $PERROR < 0 } {

set PERROR [ expr 0 - $PERROR ]

}

puts "% error = $PERROR"

if { $PERROR > $PTHRESH } {

puts "Warning - test failed!"

}

if { $PERROR > $MAXPERROR } {

set MAXPERROR $PERROR

}

return $PERROR

}

#}}}

#{{{ process command line arguments

set feeds\_list "fdt fugue susan sienax bet2 feat melodic first fnirt"

foreach f $feeds\_list {

set feeds($f) 0

}

if { [ lindex $argv 0 ] == "" } {

puts "Usage: ./RUN all"

puts "or: ./RUN \[packages\]"

puts " package options: $feeds\_list"

exit

}

if { [ lindex $argv 0 ] == "all" } {

foreach f $feeds\_list {

set feeds($f) 1

}

} else {

foreach f $argv {

set feeds($f) 1

}

}

#}}}

#{{{ setup vars and first printouts

# Don't ask!

set S 0.2

catch { [ expr $S ] } errmsg

set FEEDSVERSION 6.0.4

set PTHRESH 1

set MAXPERROR 0

if { [ info exists env(FSLDIR) ] } {

set FSLDIR $env(FSLDIR)

} else {

puts "\nPlease set FSLDIR to the fsl top level directory before running this test\n"

exit

}

set INMEDX 0

set INGUI 0

source ${FSLDIR}/tcl/fslstart.tcl

set FEEDSDIR [ exec pwd ]

puts "\nFSL Evaluation and Example Data Suite v$FEEDSVERSION\n"

set FSLVERSION [ exec cat ${FSLDIR}/etc/fslversion ]

set FSLVERSION [ lindex [ split $FSLVERSION : ] 0 ]

if { [ string compare $FSLVERSION $FEEDSVERSION ] != 0 } {

puts "Warning - FSL version ($FSLVERSION) not the same as this test

suite version ($FEEDSVERSION)! Therefore this test may

not run...maybe you should download the latest version of

the test suite and/or FSL.\n"

}

puts "start time = [ exec date ]"

puts "hostname = [ exec hostname ]"

puts "os = [ exec uname -a ]\n"

fsl:exec "/bin/rm -rf $FEEDSDIR/results ; mkdir $FEEDSDIR/results"

set logout $FEEDSDIR/results/LOG

#}}}

# to add ASAP:

# filmbabe (make\_flobs and filmbabe) / mm / randomise

#{{{ FUGUE

if { $feeds(fugue) } {

puts "\nStarting PRELUDE & FUGUE at [ exec date ]"

fsl:exec "${FSLDIR}/bin/prelude -c $FEEDSDIR/data/fieldmap -o $FEEDSDIR/results/unwrapped\_phase"

perror unwrapped\_phase 0.5

fsl:exec "${FSLDIR}/bin/fugue -i $FEEDSDIR/data/epi -p $FEEDSDIR/results/unwrapped\_phase -d 0.295 -u $FEEDSDIR/results/unwarped\_epi"

perror unwarped\_epi 0.2

}

#}}}

#{{{ SUSAN

if { $feeds(susan) } {

puts "\nStarting SUSAN at [ exec date ]"

fsl:exec "${FSLDIR}/bin/susan $FEEDSDIR/data/structural 2000 2 3 1 0 $FEEDSDIR/results/structural\_susan"

perror structural\_susan 0.25

}

#}}}

#{{{ SIENAX

if { $feeds(sienax) } {

puts "\nStarting SIENAX (including testing BET and FLIRT and FAST) at [ exec date ]"

fsl:exec "${FSLDIR}/bin/imcp $FEEDSDIR/data/structural $FEEDSDIR/results/structural"

fsl:exec "cd $FEEDSDIR/results ; ${FSLDIR}/bin/sienax structural -d -r"

puts "checking error on BET:"

perror structural\_sienax/I\_brain 0.2

puts "checking error on FLIRT:"

perror structural\_sienax/I\_stdmask 0.01

puts "checking error on FAST:"

puts "checking error on single-image binary segmentation:"

perror structural\_sienax/I\_stdmaskbrain\_seg 0.05

puts "checking error on partial volume images:"

perror structural\_sienax/I\_stdmaskbrain\_pve\_0 0.02

perror structural\_sienax/I\_stdmaskbrain\_pve\_1 0.03

perror structural\_sienax/I\_stdmaskbrain\_pve\_2 0.03

puts "checking error on SIENAX volume outputs:"

foreach sienastats { pgrey vcsf GREY WHITE BRAIN } {

set r [ exec sh -c "grep $sienastats $FEEDSDIR/results/structural\_sienax/report.sienax | awk '{ print \$2 }'" ]

set d [ exec sh -c "grep $sienastats $FEEDSDIR/data/structural\_sienax/report.sienax | awk '{ print \$2 }'" ]

simpleperror $r $d 1600000

}

}

#}}}

#{{{ BET2

if { $feeds(bet2) } {

puts "\nStarting BET2 at [ exec date ]"

fsl:exec "/bin/cp $FEEDSDIR/data/head\_t?.nii.gz $FEEDSDIR/results"

fsl:exec "cd $FEEDSDIR/results ; ${FSLDIR}/bin/bet head\_t1 head\_t1\_brain -A2 head\_t2"

puts "checking error on T1 brain extraction:"

perror head\_t1\_brain 0.05

puts "checking error on skull and scalp surfaces:"

perror head\_t1\_brain\_inskull\_mesh .01

perror head\_t1\_brain\_outskull\_mesh .01

perror head\_t1\_brain\_outskin\_mesh .01

}

#}}}

#{{{ FEAT

if { $feeds(feat) } {

puts "\nStarting FEAT at [ exec date ]"

# fix FEAT setup file to use FEEDSDIR and FSLDIR

fsl:exec "cp ${FEEDSDIR}/data/fmri.feat/design.fsf ${FEEDSDIR}/results/design.fsf"

fsl:echo ${FEEDSDIR}/results/design.fsf "

set fmri(regstandard) ${FSLDIR}/data/standard/MNI152\_T1\_2mm\_brain

set feat\_files(1) ${FEEDSDIR}/data/fmri

set highres\_files(1) ${FEEDSDIR}/data/structural\_brain

set fmri(outputdir) ${FEEDSDIR}/results/fmri.feat"

# run FEAT

fsl:exec "${FSLDIR}/bin/feat ${FEEDSDIR}/results/design.fsf"

puts "checking error on filtered functional data:"

perror fmri.feat/filtered\_func\_data 0.1

puts "checking error on raw Z stat images:"

perror fmri.feat/stats/zstat1 0.02

perror fmri.feat/stats/zstat2 0.02

perror fmri.feat/stats/zfstat1 0.02

puts "checking error on thresholded Z stat images:"

perror fmri.feat/thresh\_zstat1 0.02

perror fmri.feat/thresh\_zstat2 0.02

perror fmri.feat/thresh\_zfstat1 0.02

puts "checking error on registration images:"

perror fmri.feat/reg/example\_func2highres 0.02

perror fmri.feat/reg/example\_func2standard 0.02

#{{{ check error on largest cluster of Talairached zfstat1

puts "checking error on position of largest cluster of Talairached zfstat1:"

set iptr [ open ${FEEDSDIR}/data/fmri.feat/cluster\_zfstat1\_std.txt r ]

gets $iptr line

gets $iptr line

scan $line "%f %f %f %f %f %f %f %f %f" D(1) D(2) D(3) D(4) D(5) D(6) D(7) D(8) D(9)

close $iptr

set iptr [ open ${FEEDSDIR}/results/fmri.feat/cluster\_zfstat1\_std.txt r ]

gets $iptr line

gets $iptr line

scan $line "%f %f %f %f %f %f %f %f %f %f %f" R(1) R(2) R(3) R(4) R(5) R(6) R(7) R(8) R(9) R(10) R(11)

close $iptr

simpleperror $D(4) $R(6) 500

simpleperror $D(5) $R(7) 500

simpleperror $D(6) $R(8) 500

simpleperror $D(7) $R(9) 500

simpleperror $D(8) $R(10) 500

simpleperror $D(9) $R(11) 500

#}}}

}

#}}}

#{{{ MELODIC

if { $feeds(melodic) } {

puts "\nStarting MELODIC at [ exec date ]"

fsl:exec "${FSLDIR}/bin/melodic -i $FEEDSDIR/data/fmri -o $FEEDSDIR/results/fmri.ica --tr=3 --seed=2"

fsl:exec "${FSLDIR}/bin/fslcc $FEEDSDIR/data/fmri.ica/melodic\_IC $FEEDSDIR/results/fmri.ica/melodic\_IC > $FEEDSDIR/results/fmri.ica/fslcc.txt"

set MAXV 0

set MAXA 0

set iptr [ open $FEEDSDIR/results/fmri.ica/fslcc.txt r ]

while { ( [ gets $iptr line ] >= 0 ) } {

scan $line "%d %d %f" A B C

if { $A == 52 } {

if { $C > $MAXV } {

set MAXV $C

}

}

if { $A == 39 } {

if { $C > $MAXA } {

set MAXA $C

}

}

}

close $iptr

if { $MAXA < $MAXV } {

set MAXV $MAXA

}

simpleperror $MAXV 1 60

}

#}}}

#{{{ FIRST

if { $feeds(first) } {

puts "\nStarting FIRST at [ exec date ]"

fsl:exec "${FSLDIR}/bin/first\_flirt data/structural results/structural\_to\_std\_sub"

fsl:exec "${FSLDIR}/bin/run\_first -i data/structural -t results/structural\_to\_std\_sub.mat -n 20 -o results/structural\_first\_L\_Hipp -m ${FSLDIR}/data/first/models\_336\_bin/L\_Hipp\_bin.bmv"

perror structural\_first\_L\_Hipp 0.01

}

#}}}

#{{{ FDT

if { $feeds(fdt) } {

puts "\nStarting FDT (bedpost) at [ exec date ]"

fsl:exec "cp -r $FEEDSDIR/data/fdt\_subj1 $FEEDSDIR/results"

fsl:exec "unset FSLMACHINELIST; ${FSLDIR}/bin/bedpostx $FEEDSDIR/results/fdt\_subj1 -n 1"

puts "checking error on bedpost output:"

perror fdt\_subj1.bedpostX/dyads1 .005

perror fdt\_subj1.bedpostX/mean\_f1samples .005

perror fdt\_subj1.bedpostX/mean\_ph1samples .003

perror fdt\_subj1.bedpostX/mean\_th1samples .002

perror fdt\_subj1.bedpostX/merged\_f1samples .005

perror fdt\_subj1.bedpostX/merged\_ph1samples .003

perror fdt\_subj1.bedpostX/merged\_th1samples .002

}

if { $feeds(fnirt) } {

puts "\nStarting FNIRT at [ exec date ]"

fsl:exec "cp -r $FEEDSDIR/data/feeds\_fnirt\* $FEEDSDIR/results"

fsl:exec "cp -r $FEEDSDIR/data/sad\* $FEEDSDIR/results"

fsl:exec "cp -r $FEEDSDIR/data/happy\* $FEEDSDIR/results"

set cwd [ pwd ]

cd $FEEDSDIR/results

fsl:exec "${FSLDIR}/bin/fnirt --config=feeds\_fnirt"

cd $cwd

perror feeds\_fnirt\_sad2happy 0.1

}

#}}}

#{{{ finish up

puts "\nend time = [ exec date ]\n"

if { $MAXPERROR < $PTHRESH } {

puts "\nAll tests passed"

exit 0

} else {

puts "\nWarning - not all tests passed"

exit 1

}

#}}}