compute_geopotential_on_ml.ksh

This script has been created to help users to get the required data for compute_geopotential_on_ml.py. You have to place both scripts in the same folder.

Examples

See: ./compute_geopotential_on_ml.ksh -h

```
./compute_geopotential_on_ml.ksh -d 20150101
./compute_geopotential_on_ml.ksh -c ei -g 0.75/0.75 -d 20150701
./compute_geopotential_on_ml.ksh -d 20150101 -g 1.5/1.5
./compute_geopotential_on_ml.ksh -t 00/12 -d 20150110/to/20150131
./compute_geopotential_on_ml.ksh -t 00 -g 128 -y fc -d 20150101
```

```
#!/bin/bash
#
# Copyright 2023 ECMWF.
#
# This software is licensed under the terms of the Apache Licence Version 2.0
# which can be obtained at http://www.apache.org/licenses/LICENSE-2.0
#
# In applying this licence, ECMWF does not waive the privileges and immunities granted to it by
# virtue of its status as an intergovernmental organisation nor does it submit to any
jurisdiction.
#
# ******************************************************************************
# Function      : compute_geopotential_on_ml
#
# Author (date) : Cristian Simarro (01/07/2015)
#
# Category      : COMPUTATION
#
# OneLineDesc   : Retrieves the files from mars and computes geopotential on model levels
#
# Description   : This script will retrieve the necessary fields (t,q,z,lnsp) from MARS. Then
it will
#                 call the Python script compute_geopotential_on_ml.py to calculate the
geopotential
#                 on models levels
#
# Parameters    :   -d|--date <date>              MARS formatted keyword. Default Yesterday
(20151012)
#                                 Examples: 20100501, 20100501/20100502/20100503, 20100501/to
/20100503
#                   -t|--time <time>              MARS formatted keyword time. Default 0
#                                 Examples: 00, 00/12
#                   -s|--step <step>              MARS formatted keyword step (only valid if
type=fc). Default 0
#                                 Examples: 00, 00/03/06/09
#                   -c|--class <class>            MARS formatted keyword class. Default od
#                                 Examples: od, ei
#                   -g|--grid <grid>              MARS formatted keyword grid. Default 1.5/1.5
#                                 Examples: 0.75/0.75, 128, 640
#                   -y|--type <type>              MARS formatted keyword type. Default an
#                                 Examples: fc, an
#                   -r|--stream <stream>          MARS formatted keyword stream. Default oper
#                                 Examples: oper, enfo
#                   -a|--area <area>              MARS formatted keyword area. Default global
#                                 Examples: europe, 75/60/10/-20
```

```
#
# Return Value  : tq_ml_[date]_[time].grib     - with all the levelist of t and q
#                 zlnsp_ml__[date]_[time].grib - levelist 1 for params z and lnsp
#                 compute_geopotential_on_ml.ksh.log - the logs will be stored here unless (-v)
#
# Dependencies  : compute_geopotential_on_ml.py to calculate the geopotential
#
# Example Usage :
#                 ./compute_geopotential_on_ml.ksh 20150601
#                 ./compute_geopotential_on_ml.ksh -t 00/12 -d 20150101/to/20150110 -g 128 -a
europe
#

usage(){
        print "  ./compute_geopotential_on_ml.ksh - Retrieves the files from mars and computes
geopotential on model levels"
    print
    print "    If you want to get more information about the format of the MARS keywords see:"
    print "       https://software.ecmwf.int/wiki/display/UDOC/MARS"
    print
        print "Usage: ./compute_geopotential_on_ml.ksh [ options ]"
    print
    print "options"
    print "  -v                            Enable verbose mode"
    print "  -k                            Do not clean after retrieve"
    print
    print "  -d|--date <date>              MARS formatted keyword. Default Yesterday ($DATE)"
    print "                                 Examples: 20100501, 20100501/20100502/20100503,
20100501/to/20100503"
    print "  -t|--time <time>              MARS formatted keyword time. Default $TIME"
    print "                                 Examples: 00, 00/12"
    print "  -s|--step <step>              MARS formatted keyword step (only valid if
type=fc). Default $STEP"
    print "                                 Examples: 00, 00/03/06/09"
    print "  -c|--class <class>            MARS formatted keyword class. Default $CLASS"
    print "                                 Examples: od, ei"
    print "  -g|--grid <grid>              MARS formatted keyword grid. Default $GRID"
    print "                                 Examples: 0.75/0.75, 128, 640"
    print "  -y|--type <type>              MARS formatted keyword type. Default $TYPE"
    print "                                 Examples: fc, an"
    print "  -r|--stream <stream>          MARS formatted keyword stream. Default $STREAM"
    print "                                 Examples: oper, enfo"
    print "  -a|--area <area>              MARS formatted keyword area. Default $AREA"
    print "                                 Examples: europe, 75/60/10/-20"
    print "  -l|--levelist <levelist>      Slash separated list of levels to write in the
output file. Default $LEVELIST"
    print "                                 Examples: 1, 91, 1/2/3/4"
    print "  -e|--extra <extra>            (special) Extra keyword that you want to append to
the MARS request."
    print "                                 Examples: \"expect=3\""
}

VERBOSE=0
DATE=`date +%Y%m%d -d "-1 day"`
TIME=0
CLASS="od"
GRID="1.5/1.5"
STEP=0
CLEAN=1
```

```
STREAM="oper"
TYPE="an"
LEVELIST="all"
AREA="global"
OPTS="h(help)v(verbose)k(keep)d:(date)t:(time)c:(class)g:(grid)y:(type)s:(step)r:(stream)a:
(area)l:(levelist)e:(extra)"
while getopts "$OPTS" optchar
do
       case "$optchar" in
       d)    DATE=$OPTARG;echo "date set to $OPTARG";;
       t)    TIME=$OPTARG;echo "time set to $OPTARG";;
       c)    CLASS=$OPTARG;echo "class set to $OPTARG";;
       g)    GRID=$OPTARG;echo "grid set to $OPTARG";;
       y)    TYPE=$OPTARG;echo "type set to $OPTARG";;
       s)    STEP=$OPTARG;echo "step set to $OPTARG";;
       r)    STREAM=$OPTARG;echo "stream set to $OPTARG";;
       a)    AREA=$OPTARG;echo "area set to $OPTARG";;
       l)    LEVELIST="$OPTARG";echo "levelist set to $OPTARG";;
       e)    EXTRA="$OPTARG,";echo "extra set to $OPTARG";;
       v)    VERBOSE=1;;
       k)    CLEAN=0;;
       h)    usage;
             exit 0;;
       ?)    usage;
             exit 0;;
       esac
done

EXECUTABLE=`basename $0`
IFS="/"
set -A DSTRING $DATE
set -A TSTRING $TIME
IFS=""

FNAME="${DSTRING[0]}_${TSTRING[0]}"

if [[ ! -f compute_geopotential_on_ml.py ]]; then
    echo "[ERROR] you need compute_geopotential_on_ml.py in the same directory"
    exit 1
fi

if [[ $TYPE == "fc" ]]; then
 STEP="step=$STEP,"
else
 STEP=""
fi

cat << EOF1 > tq.req
retrieve,
date= $DATE,
class = $CLASS,
time= $TIME,
stream = $STREAM,
levtype= ml,
grid= $GRID,
$STEP
$EXTRA
type= $TYPE,
area= $AREA,
```

```
param= t/q,
levelist= all,
target="tq_ml_${FNAME}.grib"
EOF1

cat << EOF2 > lnsp.req
retrieve,
date= $DATE,
class = $CLASS,
time= $TIME,
stream = $STREAM,
levtype= ml,
grid= $GRID,
$STEP
$EXTRA
type= $TYPE,
area= $AREA,
param=lnsp,
levelist=1,
target="lnsp_ml_${FNAME}.grib"
EOF2

cat << EOF3 > z.req
retrieve,
date= $DATE,
class = $CLASS,
time= $TIME,
stream = $STREAM,
levtype= ml,
grid= $GRID,
$EXTRA
area= $AREA,
param=z,
levelist=1,
type=an,
target="z_ml_${FNAME}.grib"
EOF3

if [ $VERBOSE -eq 0 ]; then
   rm -f ${EXECUTABLE}.log
fi

for i in tq.req lnsp.req z.req; do
   echo "executing mars request for $i ..."
   if [ $VERBOSE -eq 1 ]; then
            cmnd="`which mars` $i"
   else
     cmnd="`which mars` $i >>${EXECUTABLE}.log"
   fi
     echo $cmnd
     eval $cmnd
     ret_code=$?
     if [ $ret_code != 0 ]; then
        printf "Error : [%d] when executing command: '$cmnd'\n" $ret_code
        exit $ret_code
     fi
done

if [[ -f "lnsp_ml_${FNAME}.grib" && -f "z_ml_${FNAME}.grib" ]]; then
```

```
    cat lnsp_ml_${FNAME}.grib z_ml_${FNAME}.grib > zlnsp_ml_${FNAME}.grib
    rm -f lnsp_ml_${FNAME}.grib z_ml_${FNAME}.grib
    if [ $CLEAN -eq 1 ]; then
      rm -f z.req tq.req lnsp.req
    fi
fi
cmnd="python3 compute_geopotential_on_ml.py tq_ml_${FNAME}.grib zlnsp_ml_${FNAME}.grib -l
\"$LEVELIST\""
echo $cmnd
eval $cmnd
ret_code=$?
if [ $ret_code != 0 ]; then
   printf "Error : [%d] when executing command: '$cmnd'\n" $ret_code
   exit $ret_code
fi
if [ $CLEAN -eq 1 ]; then
   rm -f z.req tq.req lnsp.req zlnsp_ml_${FNAME}.grib tq_ml_${FNAME}.grib
fi
```