Geopointset

Geopointset is the format used by Metview to combine a set of Geopoints variables into a single entity for ease of processing. Thus, a set of observations can be grouped in the same way that fields are grouped into a fieldset variable. For a full list and details of functions and operators on geopoints, see Geopointset Functions.

Creating a geopointset

A geopointset can be created with the `create_geo_set()` function, which takes any number of geopoints variables as arguments, or none. Both geopoints and geopointset variables can be concatenated with a geopointset.

```
set1 = create_geo_set()             # creates an empty set
set2 = create_geo_set(g1, g2, g3)   # assuming that g1,g2,g3 are geopoints variables
set3 = set1 & g1 & g2               # set3 has 2 geopoints
set4 = set2 & set3                  # set4 has 5 geopoints
```

Accessing geopintset elements

The `count()` function returns the number of geopoints variables contained by the set.

Use the indexing operator `[]` to access the geopoints variables contained in a set. For example:

```
print(type(set4))    # geopointset
print(count(set4))   # 5
g1 = set4[1]         # extract the first item
print(type(g1))      # geopoints
print(count(g1))     # 244 (if there are 244 points in this geopoints variable)
```

Operations on geopointsets

As a geopointset is simply a container for geopoints variables, most operations on a geopointset are performed on each of its component geopoints. For example, the following line of code with return a new geopointset where each geopoints variable has had the `cos()` function applied to its values:

```
cgset = cos(gset)
```

Operations between geopointsets and numbers are performed on each geopoints, e.g.

```
gsetplus1 = gset + 1 # add 1 to each value in each geopoints var in gset
```

Operations can be performed between geopointsets and geopointsets, or geopointsets and fieldsets, as long as they both contain the same number of items, or they contain exactly one item. Otherwise, if they contain a different number of items, the computation will fail.

For example, if `gset_5a` and `gset_5b` each contain 5 geopoints variables, the following code will add each pair of geopoints variables, giving a resulting geopointset of size 5:

```
gsetsum_r1 = gset_5a + gset_5b  # gset_5b[n] is added to gset_5a[n]
```

If `gset_1c` contains a single geopoints variable, the following code will produce a geopointset with 5 items, the result of adding `gset_1c[1]` to each item in `gset_5a`:

```
gsetsum_r2 = gset_5a + gset_1c  # gset_1c[1] is added to each gset_5a[n]
```

Likewise, geopointset/fieldset operations work the same way:

```
gsetdiff_r1 = fc_fieldset_5 - gset_5a # gset_5a[n] is subtracted from fc_fieldset_5[n]
gsetdiff_r2 = fc_fieldset_5 - gset_1c # gset_1c[1] is subtracted from each field
```

Filtering a geopointset

Individual geopoints variables can contain meta-data - see Geopoints for details. To select only those geopoints variables with given meta-data, use the `filter()` function as described in Geopointset Functions.

The Geopointset file format

The format for a geopointset file is very simply a header followed by a contatenation of geopoints files - see Geopoints for details of the format. The overall header is this:

```
#GEOPOINTSET
```

The subsequent geopoints structures should all share the same format as each other. Here's an example with 3 geopoints files inside the set:

```
#GEOPOINTSET
#GEO
# lat        lon        height        date                time        value
# Missing values represented by 3e+38 (not user-changeable)
#DATA
 69.6523        18.9057        0        20130512        0        100869.8625
 63.4882        10.8795        0        20130512        0        100282.3392
 63.5657        10.694        0        20130512        0        100241.1666
 61.2928        5.0443        0        20130512        0        99852.18932
#GEO
# lat        lon        height        date                time        value
# Missing values represented by 3e+38 (not user-changeable)
#METADATA
param=geopotential
#DATA
 60.82        23.5        0        20130512        600        101045.8
#GEO
# lat        lon        height        date                time        value
# Missing values represented by 3e+38 (not user-changeable)
#DATA
 55.01        8.41        0        20130513        0        100949.1809
 54.33        8.62        0        20130513        0        101027.9101
 53.71        7.15        0        20130513        0        100846.619
```