cosin
scientific software

cosin scientific software AG

Hans-Stützle-Str. 20
81249 München
GERMANY

info@cosin.eu
www.cosin.eu

cosin/rgr
Cosin Road Modeling
RGR File Format Documentation

Preface

This documentation describes the unencrypted RGR file format. Even if documented here, it is strongly recommended not to generate RGR files with user-written code. Rather, use**cosin/tools** whenever applicable.

File Structure

A binary, optionally compressed RGR file consists of 2 to 4 separate blocks that are lined up sequentially without any separators:

1. **Block 1**: the header with (few) data entries in plain text, followed by an exclamation mark. To enable access to the following binary data directly and efficiently using 'memory mapping' in case of huge files, it is suggested to place the exclamation mark such that the following binary data starts on a double word boundary, that is at byte number 8, 16, 24, 32, .. after the header. Fill additional space in front of the exclamation mark with blanks.

   The header starts with the keyword `$RGR_data`, followed by other keywords (see table below) in arbitrary order. Each keyword optionally is followed by an equal sign and a value (= string or number in the usual number format). Undefined keyword / value pairs are ignored. The header may not be longer than 4096 bytes. In order to parse the header as efficient as possible, no blanks are permitted in any group consisting of a keyword, equal sign and value. The header must not contain any exclamation mark besides the trailing one.

   Example:

   ```
   $RGR_data lu=mm nc=1001 ncd=5 xmin=0 dx=10 nx=1001 xextr=0 ymin=-1700 dy=10 ny=341 yextr=0 !
   ```

Keyword	**mandatory**	**Meaning**
xmin	*	minimum x value of the superimposed RGR grid, relative to the center-line: if xmin = 0, the grid starts at the first node of the center-line. If no center-line is specified, grid starts in global coordinates at x =xmin. Unit as specified by lu.
dx	*	grid spacing in x (= longitudinal) direction. Unit as specified by lu.
nx	*	number of grid lines in x direction
ymin	*	minimum y value of the superimposed RGR grid, relative to the center-line. Unit as specified by lu.
dy	*	grid spacing in y (= lateral) direction. Unit as specified by lu.
ny	*	number of grid lines in y direction
lu		length unit (mm, m, in, ft). Default value is mm
xextr		kind of extrapolation in x direction: 0: constant end value, defined by value in the nearest grid point 1: cyclic repetition (like the 'repeating sequence' in Simulink) 2: alternating cyclic repetition 3: constant end value, defined by zextr. Default value is 0
yextr		kind of extrapolation in y direction: 0: constant end value, defined by value in the nearest grid point 1: cyclic repetition (like the 'repeating sequence' in Simulink) 2: alternating cyclic repetition 3: constant end value, defined by zextr. Default value is 0
zextr		height value to be used outside grid if extrapolation kind is 3
nc		number of center-line data points. If this keyword is not set or its value is 0, no center-line data will be read. Default value is 0
ncd		number of values per center-line data point. Interpretation of data is as follows: ncd=2: x/y coordinates of data point [length unit] ncd=3: x/y coordinates and height of data point [length unit] ncd=4: x/y coordinates and height of data point [length unit], road width at data point [length unit] ncd=5: x/y coordinates and height of data point [length unit], road width at data point [length unit], tangent of road banking angle at data point [-] = banking [%] / 100 (positive if road heigt at right border is smaller than at left border). Default value is 2
filtc		optional low-pass filter path constant to filter center-line x/y data with, given in length units. Default is 'no filtering'

Keyword	mandatory	Meaning
`incc`		optional center-line data increment. If `incc = x` is set, only every x-th value will be used to compute the center-line spline functions. This option is useful if center-line data are noisy or have only small distance, since in these cases the computation of the center-line normal, and thus the rgr grid orientation might become inaccurate. Default value is 1
`closed`		mark the center-line (if specified) as being closed, forcing periodic rather than extrapolated evaluation if travel path is less zero or greater center-line length. only keyword (without equal sign and value)
`xc0`		x-shift of center-line in global coordinates. Unit as specified by `lu`. Default value is 0
`yc0`		y-shift of center-line in global coordinate. Unit as specified by `lu`. Default value is 0
`zc0`		z-shift of center-line in global coordinates. Unit as specified by `lu`. Default value is 0
`phic0`		rotation angle of center-line [deg]. Default value is 0
`nonan`		file does not contain any NaN values; input check can be skipped, thus shortening loading time. only keyword (without equal sign and value). Used to accelerate file loading; maybe omitted without changing file properties
`compressed`		file contains compressed data only keyword (without equal sign and value). Used to accelerate file loading; maybe omitted without changing file properties
`uncompressed`		file does not contain any compressed data only keyword (without equal sign and value). Used to accelerate file loading; maybe omitted without changing file properties

2. **Block 2** (optional): center line data, stored as binary IEEE float numbers, little endian, `nc*ncd` values, meaning as described in the table above. The data directly follows the header's exclamation mark (no blank allowed in between). Memory order is: `ncd` values of the first center-line node, `ncd` values of the second node, and so on. This block is completely omitted if `nc` is not specified or has a value of 0.

3. **Block 3**: the height values of the RGR grid, stored as binary IEEE float numbers, little endian, `nx*ny` values, meaning as described in the table above. The data directly follows the center-line data (no bytes in between); the sequence is: `ny` z-values of the first x-value,`ny` values of the second x-value, and so on.

   The data in this block optionally can be compressed: if k-times the same value w appears, the two numbers $k \cdot 10^{10}$ and w can be used instead. Please note that compressed files can not be accessed using memory mapping. As a matter of convenience, compression is indicated in the header with the additional keyword `compressed` (without an equal sign or value), but is recognized without this keyword as well. With keywords `binary_LE` and `binary_BE`, the endianness can be changed when reading/writing in exotic operating systems. Default storage mode is little-endian, which is used by all variants of Windows, Linux, and macOS.

4. **Block 4** (optional): location-dependent road attributes such as friction factors, soft-soil model data, micro-

texture, etc. The values of these attributes are selected via a location-dependent index. Every index value corresponds to an attribute value which is specified in the header. The indices are stored in the same sequence as the height values in block 3, using one byte each (unsigned char) instead of four bytes as in block 2 and 3. For friction factors, the required additional keywords in the header ar `mu1=`, `mu2=`, ... (`mu1` being the friction factor of index 1, and so on). Up to 99 different indices are possible. In this block, no compression is permitted. The block can be omitted. If it exists, it must contain at least `nx*ny` bytes. Any extra bytes are ignored.