Geochemical Data Toolkit *(GCDkit.Mineral)* written in R language

Vojtech Janousek, Czech Geological Survey;
Vojtech Erban, ex-Czech Geological Survey;
Colin M. Farrow, ex-University of Glasgow

http://www.gcdkit.org

User's Guide to version 1.0 (23 May 2023)

GCDkit.Mineral is a system for handling and recalculation of mineral analyses, nowadays mostly obtained by electron microprobe. It is fully menu driven but, at the same time, can be used in an interactive regime (power users only).

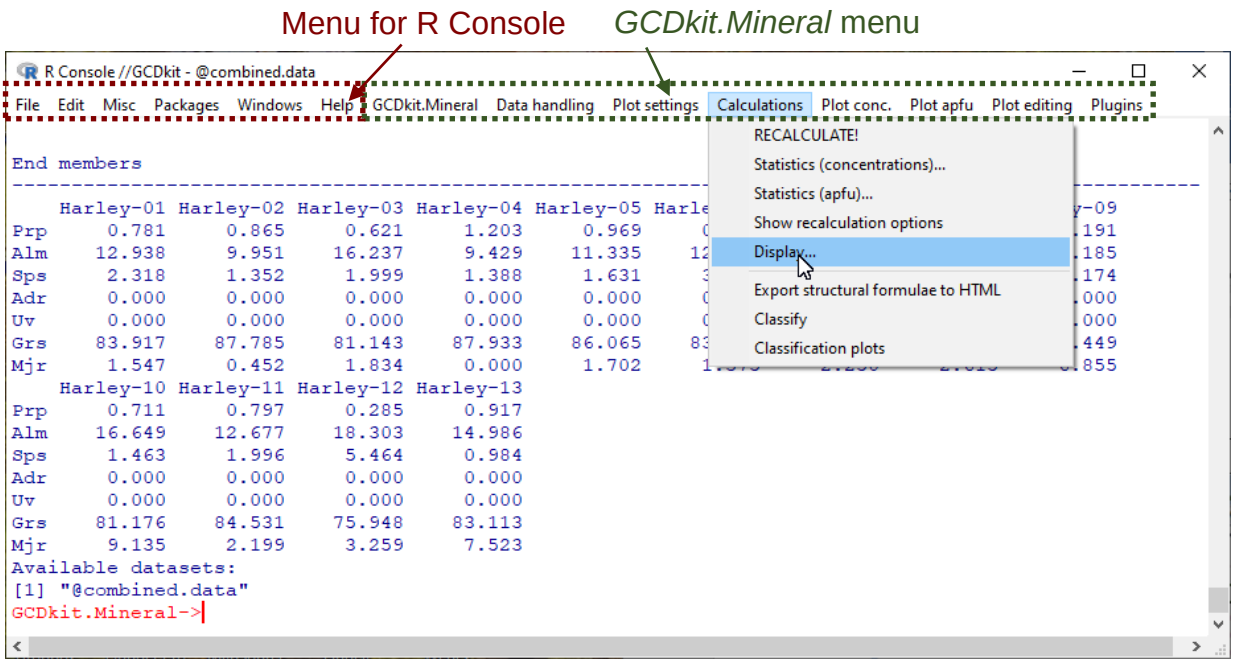
Using this system, the data can be loaded, individual analyses grouped into coherent groups or searched according to various criteria. The analyses for individual mineral species can be recalculated to atoms per formula unit (apfu) on the basis of fixed O equivalents, sum of charges or atoms; the Fe^{II}/Fe^{III} balance can be assessed by a number of models. The calculated atoms can be then allocated to individual crystallographic sites, or used to compute user-defined parameters and end-member proportions. The mineral analyses, both rough in wt. % and recalculated in apfu, can be plotted, and the results exported to the clipboard, TXT, CSV, XLS, XLSX, HTML, DBF or other file formats.

The system is fully customizable; the standard database of recalculation options is stored externally, in an easy-to-understand text file. The programme is expandable by means of the so-called plugins that provide a simple method of adding effortlessly new items to the menus of *GCDkit.Mineral*.

Important!

This *GCDkit.Mineral* was built in **R version 4.1.3** for Windows. Also this manual applies in its entirety only to the Windows (GUI) version of *GCDkit.Mineral*. Due to the quick developments in the R project, it is blocked in any other versions!

The platform-independent (i.e. non-Windows) *GCDkit.Mineral* is still experimental. Some of the features may not be available, others may work unpredictably. Please use sensibly, at your own risk. For further information, see *http://blog.gcdkit.org*.



TECHNICAL NOTES

- After starting R opens *"R Console"*, a text window serving for entry of commands as well as display of textual output. In addition, during the session may be opened one or more graphical windows.
- Please note that on the Windows systems, the *GCDkit.Mineral* menu appears right of the menus of the R system itself (see Figure above). On non-Windows systems, please type *library("GCDkit.Mineral")* and then *menuet()* into the R command prompt in order to (re-) start the Tcl/Tk menu system.
- The interactions built in many plots (most commonly data point identification) can be stopped from the menu that appears after pressing the right mouse button.
- If experiencing problems, the computation/plotting can be interrupted anytime by hitting *Escape* key or from the menu (*Misc*|*Stop current computation*).
- The errors encountered while running the *GCDkit.Mineral* system and displayed in the Console are (mostly) not fatal. In most cases it means that the command can be re-run (on Windows, one can simply press the up/down arrows in the Console to scroll through the commands history), entering/modifying the parameters that caused the crash. If the problem persists, record the details (including the version of *GCDkit.Mineral*, R and your operation system) and send the data file to us, so we can track down and fix the bug.
- *GCDkit.Mineral* requires the R packages *base*, *stats*, *tools*, *methods*, *utils*, *graphics*, *MASS*, *grid*, *lattice*, *foreign*, *compiler* and *tcltk* to be installed (but this is normally the case).
- Moreover, the package *sp* has to be always present as it is used for classification purposes, and the package *R2HTML* that provides output to HTML format. The package *XML* provides access needed to read input from online databases. On Windows, highly recommended is also the package *RODBC* enabling import and export into the MS Excel, Access and dBase (DBF) formats. All these can be installed from our distribution. But if need be, any of the above packages can be added via the menu *Packages*|*Install package(s)*... (you need to be online). Offline installation is also possible, though: zip file downloaded from the CRAN site *(http://www.r-project.org)* can be introduced via menu *Packages*|*Install package(s) from local zip files*...

- **Note that recommended systems** to run *GCDkit.Mineral* are Windows 11/10/8/7. The current version should also work (including the Graphical User Interface, GUI) on Mac OS X (release 10.6 and above) and various distributions of Linux (Debian, RedHat, SUSE, Ubuntu).
- It is always a good idea to close the unnecessary graphical windows, for instance using the function *graphicsOff()* or the corresponding item from the menu *GCDkit.Mineral*.
- By the same token, please try to remove periodically all unwanted datasets still stored in the memory. For this purpose serves the function *purgeDatasets()*.
- On Windows, more immediate response is obtained when print buffering (*Misc*|*Buffered Output*) is disabled.
- *PDF* output should be given preference, as it is the most accurate and it also supports semitransparency used in some diagrams. If no semi-transparent colours are desired, *PostScript* is a viable alternative. Using *WMF* (Windows metafiles) or copying via clipboard are to be discouraged, as these two methods lead sometimes to distortions of the graphs. For saving simply right-click the graphical window and select the desired format.
- If semi-transparent colours are used, some printers ignore them when printing directly from the plotting window. In these cases, use PDF export first and then print from some other application, such as Adobe Reader.
- The use of special symbols or accented characters (such as in some East European languages) in the datasets is mostly okay but it may also sometimes cause unexpected problems. So please be sensible, especially in the variable names.

Quick guide to data files

GCDkit.Mineral requires plain text data files, delimited by tabs, commas or semicolons (delimiter is recognised automatically, as is a decimal point/comma). See the testing files that come with the distribution for examples of valid data files with mineral data.

As an (less forgiving) alternative on operation systems with graphical user interface, the data can be pasted, via clipboard, from any well-behaved software, including Excel. Be careful, though, as the precision of the data imported will be that *displayed* in the original Excel file!

If library *RODBC* has been installed, and R is run in **32-bit mode**, the *GCDkit.Mineral* first attempts to establish ODBC connection to the selected file, and open it as a Excel (*.XLS, *.XLSX), comma-separated values (*CSV), dBase III/IV (*.DBF), or Access (*.MDB) format. The DBF files were used to store data by once popular *MinPet* package (Richard, 1995) used for mineral recalculations, that however runs only on, now obsolete, 16-bit Windows.

Without ODBC support, or on inherently 64-bit systems such as Mac OS X, the *GCDkit.Mineral* can always read at least the plain text files.

As a rule of thumb, if encountering any problems with loading data, please use import from text file or *via* clipboard, the methods that are much more robust than any other one.

In text files and spreadsheets, the first row should contain names of the data columns (except for the first column that is automatically assumed to contain sample names). Hence the first line of the text file may (or may not) have one item less than the following ones.

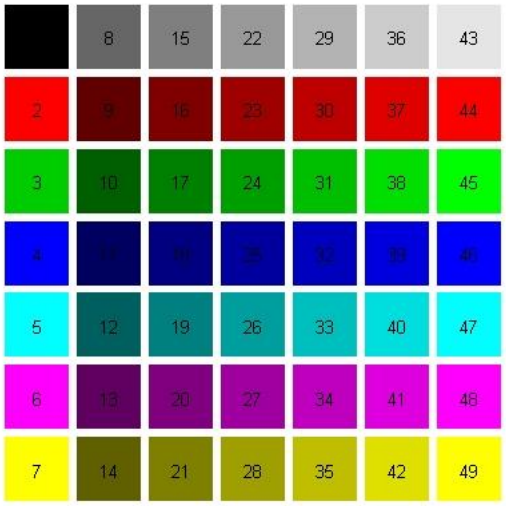
Missing values (NA) are allowed anywhere in the data file body; values equalling to zero, or any of ‘NA’, ‘N.A.’, ‘-’, ‘b.d.’, ‘bd’, ‘b.d.l.’, ‘bdl’ and ‘n.d.’ are also treated as such. While loading Excel files, the values *#WHATEVER!* (Excel error messages) are also replaced by 'NA'.

If desired, negative numbers and values '< x' (used by some authors to indicate items below detection limit of *x*) can be either replaced by their half (i.e. interpreted as a half of the detection limit) or 'NA'. User is prompted which of these options he prefers.

Alternatively, negative values can be imported in the original form if required.

The data rows start with sample name and do not have to be all of the same length (the rest of the row is filled by 'NA' automatically).

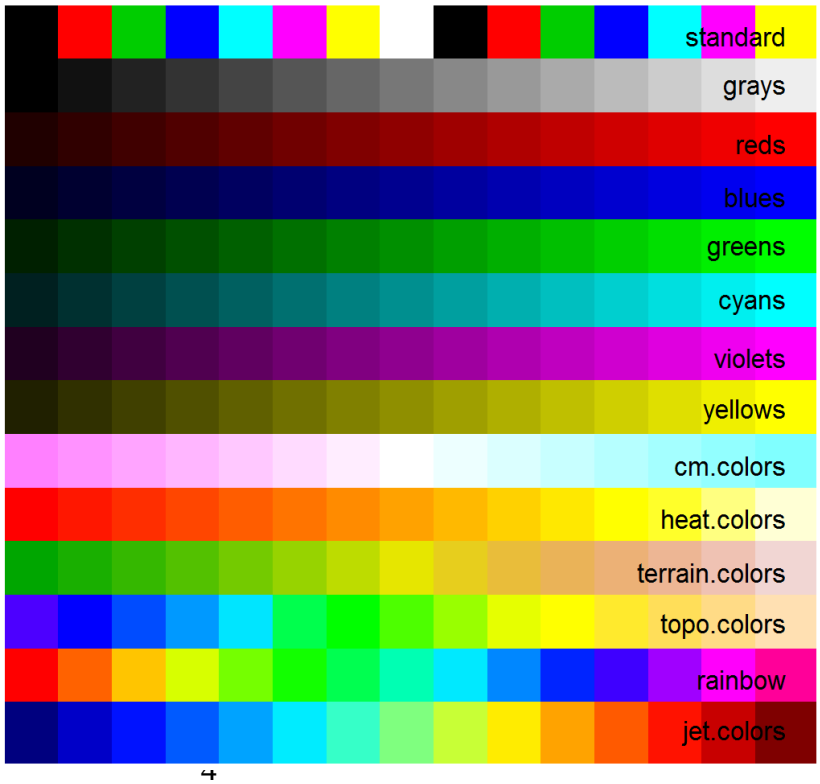
It is recommended that a column named "**Mineral**" contains specification of mineral species (i.e. recognized full name or abbreviation) for each of the analyses. If no such column is present or if it contains some unrecognized names, the desired mineral is to be chosen from a drop-down list of the mineral classes available in the database.



A column named "**Symbol**" (if any) is taken as containing specification of plotting symbols. They can be specified either as single character strings or numeric codes, whose overview is given on the right or can be obtained by invoking the menu item *Data handling|Show available symbols*.

Another column, whose name is "**Colour**" or "**Color**" (if any, capitalization does not matter) may contain codes (1–49) or English names for the plotting colours. There are 657 of the latter, see Help, invoke menu *Data handling|Show available colours* or type *colours()* into the *R Console*.

Alternatively, colours can be coded using the RGB model (8-bit hexadecimal colour code) (e.g. `#FF0000` for red) or RGBA model (the same but with two extra hexadecimal digits appended for the so-called alpha channel, expressing the colour's opacity). Therefore, for instance, `#FF0000BF` is red with 75% opacity (~ 25% transparency).



If specifications of both the plotting symbols and the plotting colours are missing completely, and at least one non-numeric variable is present, the user is prompted whether he does want to have symbols and colours assigned automatically, i.e. according to the individual

values (levels) of the selected label. Otherwise default symbols (empty black circles) are used.

Column named "**Size**" or "**cex**" may contain relative size for each of the plotting symbols (use 1 for standard size, 2 for double and 0.5 for half the size).

Apostrophes and other special characters should be avoided in the sample or variable names. Otherwise, the data files are practically freeform, i.e. no specified oxides/elements are required and no exact order of these is to be adhered to. Analyses can contain as many numeric columns as necessary; the names of oxides and trace elements are self-explanatory (e.g. 'SiO2', 'Fe2O3', 'Rb', 'Nd'). Total iron, if given, should be expressed as ferrous oxide ('FeOt', 'FeOT', 'FeOtot', 'FeOTOT' or 'FeO*') or ferric oxide ('Fe2O3t', 'Fe2O3T', 'Fe2O3tot', 'Fe2O3TOT' or 'Fe2O3*'). Structurally bound water can be named 'H2O.PLUS', 'H2O.P', 'H2O+', 'H2OPLUS' or 'H2O_PLUS'.

Note that names of variables are **case sensitive** in R. However, any of the fully upper case names of the oxides/elements that appear in the following list are translated automatically to the appropriate capitalization:

SiO2, TiO2, Al2O3, Fe2O3, FeO, MnO, MgO, CaO, Na2O, FeOt, Fe2O3t, Li2O, mg#, Li, Rb, Cs, Be, Sr, Ba, Sc, Y, Ti, Zr, Hf, V, Nb, Ta, Cr, Mo, W, Re, Ru, Os, Co, Rh, Ir, Ni, Pd, Pt, Cu, Ag, Au, Zn, Cd, B, Ga, In, Tl, C, Ge, Sn, Pb, P, As, Sb, Bi, S, Se, Te, F, Cl, Br, I, At, La, Ce, Pr, Nd, Pm, Sm, Eu, Gd, Tb, Dy, Ho, Er, Tm, Yb, Lu.

Also trailing spaces in the end of the column names are disposed of. However, note that **no duplicated column/sample names are allowed!** The correct syntax is checked upon loading.

After loading, all the completely empty columns are removed first. In the text files, any line starting with the hash symbol ('#') is ignored and can be used to introduce comments or to prevent the given analysis from loading temporarily. Any non-numeric items found in a data column with one of the names listed in the above dictionary are replaced by 'NA' automatically. At the next stage all fully numeric data columns are stored in a numeric data matrix *WR*. Decimal commas, if present in text file, are converted to decimal points. For any missing major- and minor-element data (SiO2, TiO2, Al2O3, Fe2O3, FeO, MnO, MgO, CaO, Na2O, K2O, H2O.PLUS, CO2, P2O5, F, S), an empty (NA) column is created automatically.

The remaining, that is all at least partly textual data columns are transferred into a data frame *labels*, in which are also stored codes of plotting symbols, their colours and sizes.

Mineral recalculation procedure

After loading, the rough mineral data (in wt. %, typically coming from electron microprobe) are recalculated automatically to atoms per formula unit (apfu).

The recalculation options for individual mineral species (classes) are stored in the so-called standard database, i.e. the file '`mineral_db.r`', that is read always at the start-up of the *GCDkit.Mineral*. With some care, this file can be edited and even new minerals added (but please do make sure that you keep the original safe in case that something gets wrong…).

If ready, the programme allocates the freshly recalculated apfu to individual crystallographic sites, calculates some extra parameters and/or recasts the analyses to end members.

The complete recalculation procedure (including names of underlying functions) is as follows:

1. determination, for each analysis, of mineral class and setting relevant recalculation options from the standard database: `minAssign`,
2. recalculation to apfu using the specified recalculation scheme: `minFormula`,
3. allocation of atoms to crystallographic sites: `minAllocateAtoms`,

4. calculation of any extra parameters that can be based on rough data in wt.%, apfu or contents of a specific atom(s) in a concrete site(s): `minValues`,
5. calculation of end-member proportions: `minEndMembers`.

NB that this procedure is invoked in its entirety (*via* the front-end function `minMain`) always when new data set is loaded, edited or its subset is created. Each of the functions in *GCDkit.Mineral* has its own help page, that can be invoked from the HTML help system (*GCDkit.Mineral*|*Help*) or directly by typing e.g., `help("minFormula")` or `?minFormula` into the R window.

If everything goes smoothly, all the original and recalculated data are stored, together with recalculation options, into a newly created object (list) `min.data`. This object contains named component(s) for each of the mineral species (classes) present. Each of these components (e.g. `min.data$garnet`) comprises database entries (slots) summarized in the Table 1 below.

Table 1 Slots used to store rough and recalculated data for each of the minerals

rough	original analyses (wt. %) as imported from the data file, but there may be some additions calculated by the system based on mineral stoichiometry, such as H_2O*, FeO* or Fe_2O_3*
labels	all at least partly textual information on individual analyses from the file plus plotting attributes (plotting symbols, colours, sizes)
recalc	analyses recalculated to apfu
formula	apfu allocated to individual sites
values	calculated extra parameters
end.members	calculated end-member proportions

Contents of these slots can be displayed from menu *Calculations.* Alternatively, they can be queried directly from the Console window, typing e.g., `min.data$garnet@recalc`.

1. minAssign

This function redistributes relevant parts of the input data matrix (`WR`) into a list `min.data` that contains named component(s) for each of the mineral classes present in the file.

For each analysis, mineral species is determined based on the column '**Mineral**' in the original file. This column may contain any of the full/abbreviated names listed in the standard database. The abbreviations include the standard ones from Kretz (1983), Whitney & Evans (2010) and Warr (2021), as well as those used by other software packages, e.g. MinPet (Richard, 1995), ThermoCalc (Holland & Powell, 1998) or PET (Dachs, 1998; 2004).

Entries in the standard database serve to determine the recalculation options for each of the minerals (Table 2).

2. minFormula

This function is a front-end to several more specialized functions that serve for recalculation of electron-microprobe analyses to mineral formulae on the basis of fixed number of atoms, oxygen equivalents or charges.

The function `formulaFixedOxygens` recalculates chemical analyses to a given number of oxygen equivalents as specified, for each mineral class, in the slot `oxygens`. If O is not the only anion present, the functions can handle halogens, F and Cl, as well (Deer *et al.*, 1992).

Table 2 Overview of the recalculation options stored in the standard database

Slot	Explanation
oxygens	number of O equivalents the formula should be recalculated to
charges	number of charges the formula should be recalculated to; optionally, Fe^{II}/Fe^{III} estimation can be carried out in attempt to balance the formula precisely
atoms.sum	number of atoms in the formula unit (recalculations to a fixed number of atoms – i.e., neither *oxygens* nor *charges* are given)
atoms.recalc.list	atoms that are to be summed (for formula recalculation to a fixed sum of specific atoms)
cations	number of cations when Fe^{II}/Fe^{III} is to be estimated by the methods *Droop* (*'cations'* refer to the total sum of cations) or *FixedCats* (*'cations'* give sum of the site specified by *'cations.site'*)
cations.site	name of the site that should be summed by iterative iron estimation (*iron = "FixedCats"*)
iron	Fe^{II}/Fe^{III} estimation method, implemented for minerals are *'Droop'* and *'FixedCats'*, *'allFeII'* and *'allFeIII'*. For amphiboles can be used also *'8Si','16CAT','15eNK', '15eK','13eCNK'","8SiAl','10sumFeIII'* and *'avg'*, for pyroxenes *'PxPapike'*
atom.names	names of all possible atom names to be returned by formula recalculation (not all need to be present in the current data file)
sites	list, whose each component contains names of atoms that should be allocated to the given crystallographic site
site.sums	sums of individual sites, or *NA* when unconstrained
values. formulae	formulae for calculation of additional parameters
values.names	their names
end.member.formulae	formulae for calculation of end-members
end.member.names	their names

Please see `help("mineral-class")` for complete and up-to-date list of recalculation options.

The function `formulaFixedAtoms` recalculates chemical analyses to a given number of atoms. The sum is specified, for each mineral class, in the slot `atoms.sum`. The calculation is carried out for all atoms, or just those listed in the slot `atoms.recalc.list` (if specified).

In addition, the Fe^{II}/Fe^{III} ratio can be estimated by various methods as specified by the slot `iron` (see Table 2). Alternatively, when both FeO and Fe_2O_3 have been determined, no Fe^{II}/Fe^{III} estimation is carried out and both oxides are recalculated as given.

The function `formulaFixedCharges` recalculates the chemical analyses using a charge-balance method. The number of desired charges is specified, for each mineral class, in the slot `charges`. Optionally, when slots `atoms.sum` and `atoms.recalc.list` are specified, Fe^{II}/Fe^{III} estimation can be carried out in attempt to balance the formula precisely. For analyses in which this cannot be achieved, a warning message about their non-stoichiometry is displayed.

3. minAllocateAtoms

This function allocates the recalculated atoms per formula unit to crystallographic sites of the given mineral(s).

The sites are filled by individual atoms in the order specified, from left to right, in the slot `sites`. If some of the atoms may be present in two sites, the slot '`site.sums`' must give the required sum for the first of them. When the site is filled (the sum in '`site.sums`' reached or exceeded), the excess of the given atom is passed to the next position available.

In the following hypothetical example:

```
sites=list(Z=c("Si","Al"),X=c("Mg","Fe"),Y=c("Ti","Al")),
site.sums=c(6,NA,NA),
```

the site 'Z' is filled by all Si and part of Al (6 – Si). Any excess Al will be transferred to the site 'Y'. In other words: Al_Z = 6 – Si_Z; Al_Y = Al – Al_Z.

In this case, no sums have to be defined for sites 'X' and 'Y', hence the value 'NA' standing for 'not available'. If desired, a special symbol (atom name) for vacancy (`Vc`) can be used if the site is to be filled up by vacancies to the sum specified.

4. minValues

This function calculates extra values specified by formulae stored in the database or in an external R script. The additional parameters can be defined in the database (in the slots `values.formulae` and `values.names`), similar to the following example:

```
values.formulae=c("Ca/(Ca+Mg+FeII+Mn)", "FeII/Mg", "Al_Z/Al_Y"),
values.names=c("XCa","Fe/Mg","AlIV/AlVI")
```

This means that the formulae can refer to valid atom names (as are the first two above) and/or the names of the atoms allocated to the crystallographic sites by the function `minAllocateAtoms` (the third parameter above).

An example of recalculation script `garnet.r`:

```
# Garnet extra values
if((3-Si)<0){
    AlIV<-0
}else{
    if((3-Si)>Al) AlIV<-Al else AlIV<-3-Si
}
c(Mg/(Mg+FeII),Mg/(Mg+FeII+FeIII),FeII/Mg,AlIV,Al-AlIV)
```

And the corresponding entries in the standard database:

```
values.formulae="garnet.r",
values.names=c("XMg","XMg(Fetot)","FeII/Mg","AlIV","AlVI"),
```

5. minEndMembers

This function recasts the analyses into end-member components specified by formulae or an external R script.

The end-member formulae and definitions can be given in slots `end.member.names` and `end.member.formulae` stored in the database, similarly to the case of feldspar:

```
end.member.formulae=c("Na/(Na+Ca+K)","Ca/(Na+Ca+K)","K/(Na+Ca+K)"),
end.member.names=c("Ab","An","Or") # Simplified
```

Note that the formulae can refer to valid atom names (see `minFormula`) and/or to names of the atoms allocated to the individual crystallographic sites (see `minAllocateAtoms`). In addition can be used any special parameters calculated by `minValues`, e.g.:

```
values.formulae=c("FeII/(Ca+Mg+FeII+Mn)", "Mg/(Ca+Mg+FeII+Mn)",
   "Mn/(Ca+Mg+FeII+Mn)"),
values.names=c("XFe", "XMg", "XMn"),
end.member.formulae=c("XMg", "XFe", "XMn", "FeIII/2"),
end.member.names=c("Prp","Alm","Sps","Adr") # Simplified
```

Examples of the possible recalculation types

1. **Recalculation to a fixed total number of atoms (5), no Fe^{II}/Fe^{III} estimation**

   ```
   atoms.sum=5, # no oxygens, charges, iron specified
   ```

2. **Recalculation to a fixed number (5) of selected atoms (here Si + P), no Fe^{II}/Fe^{III} estimation**

   ```
   atoms.sum=5, # no oxygens, charges, iron specified
   atoms.recalc.list=c("Si","P")
   ```

3. **Recalculation to a fixed number of oxygens (4), no Fe^{II}/Fe^{III} estimation**

   ```
   oxygens=4, # no cations, iron, charges specified
   ```

4. **Recalculation to a fixed number of oxygens (12), estimation of Fe^{II}/Fe^{III} assuming certain number of cations (8) in the whole formula (Droop, 1987)**

   ```
   oxygens=12,
   cations=8,
   iron="Droop",
   ```

5. **Recalculation to fixed number of oxygens (12), estimation of Fe^{II}/Fe^{III} assuming a fixed sum of cations (2) in a given site ("Y")**
   The formula is calculated iteratively; in the following example the site is Y occupied by (part of) Al, Ti, Cr, Y, P and Fe^{III}

   ```
   oxygens=12,
   cations=2,
   cations.site="Y",
   iron="FixedCats",
   sites=list(Z=c("Si","Al"), X=c("Mg","FeII","Ca","Mn"),
      Y=c("Al","Ti","Cr","Y","P","FeIII")),
   ```

6. **Recalculation to a certain number of charges (22) in the whole formula, no Fe^{II}/Fe^{III} estimation**

   ```
   charges=22, # no atoms.sum, oxygens, iron specified
   ```

7. **Recalculation to a certain number of charges (22) in the whole formula, Fe^{II}/Fe^{III} estimation carried out in an attempt to balance the charge of the whole formula precisely**

   ```
   charges=22, # no atoms.sum, oxygens, iron specified

   cations=7,
   atoms.recalc.list=c("Si","Al","FeIII","Mg","Mn","Cr","FeII"),
   ```

User-defined mineral subclasses

As a means for saving several recalculation approaches for a single mineral species serve the so-called subclasses. In a such a subclass. only a handful of slots is defined; the rest is inherited from the “master” class. In the following example, a subclass is created on the basis of the `amphibole` master class to recalculate the amphibole formulae on the basis of the 23O 15eNK method. Make sure that abbreviations are cleaned and new full name is unambiguous:

Table 3 Some examples of available recalculation schemes

	Recalculation to apfu				Fe^{II}/Fe^{III} allocation		
	oxygens	charges	atoms.sum	atoms.recalc. list	cations	cations.site	iron
No Fe^{II}/Fe^{III} estimation							
Fixed number of atoms in entire formula			5				
Fixed number of **selected atoms** (here Si + P)			5	c("Si","P")			
Fixed number of **oxygen equivalents**	4						
Fixed number of **charges**		22					
With Fe^{II}/Fe^{III} estimation							
Fixed number of **oxygen equivalents**, Fe^{II}/Fe^{III} estimation assuming certain number of cations in the whole formula **(Droop 1987)**	12				8		"Droop"
Fixed number of **oxygen equivalents**, Fe^{II}/Fe^{III} estimated assuming a certain **number of cations in the given site** ("Y")	12				2	"Y"	"FixedCats"
Fixed number of **charges**, Fe^{II}/Fe^{III} estimation carried out to balance the charges in formula precisely		22		c("Si","Al", "FeIII","Mg", "Mn","Cr", "FeII")	7		

```
setClass("amphibole_23O_15eNK",representation(),contains="amphibole",
  prototype(
      abbreviated=character(0),
      full="amphibole_23O_15eNK",
      oxygens=23,
      iron="15eNK",
      site.sums=c(8,5,2,1,2)
  )
)
```

For advanced users: using *GCDkit.Mineral* in direct or batch mode

Sample files

GCDkit.Mineral contains several built-in datasets, derived from the third edition of the monograph on rock-forming minerals by Deer et al. (2013). These are invoked by the function `sampleDataset(mineral)`, like in the following example:

```
sampleDataset("garnet")
```

Available at the moment are the following datasets: `alumosilicates, amphibole, apatite, feldspars, garnet, micas, olivine, pyroxene`.

Recalculation on demand, in direct or batch mode

Apart from the GUI, the *GCDkit.Mineral* code can be invoked in direct or batch mode. In both cases, the front-end function `minMain` can be invoked directly, specifying the mineral name and an (optional) list of recalculation options as arguments. Note that all obsolete options are cleared first, before the newly desired ones are set. Examples of custom recalculations:

```
sampleDataset("garnet")
# default options (taken from the standard database)
minMain("garnet")
# Examples of alternative calculations
minMain("garnet",list(oxygens=12))
minMain("garnet",list(oxygens=12,cations=8,iron="Droop"))
minMain("garnet",list(oxygens=12,cations=2,cations.site="Y",
   iron="FixedCats"))
minMain("garnet",list(atoms.sum=8))
minMain("garnet",list(atoms.sum=8,values.formulae="Al_Z/Al_Y",
   values.names="Al ratio"))

sampleDataset("amphibole")
minMain("amphibole",list(oxygens=23,iron="13eCNK"))

sampleDataset("pyroxene")
minMain("clinopyroxene",list(oxygens=6,iron="PxPapike"))
```

Note that the crystallographic sites allocations, calculations of extra values and end-members, may not be compatible with the chosen user-defined recalculation method. In the following example, the `site.sums` of `c(2,2,NA)` are defined in the standard database for recalculation to 8 O equivalents. Therefore, we need to double these sums if recalculating to 16 O:

```
minMain("feldspar",list(oxygens=16,site.sums=c(4,4,NA)))
```

Choosing numeric variable(s), calculation core routine

Many functions (e.g. binary plots, some statistics) require a single numeric variable to be chosen first. The easiest way is to type in the name of the numerical column (e.g., 'SiO2') or its sequence number (2 for the second column). However, it is not necessary to enter the name in its entirety. Only a substring that appears somewhere in the column name or other forms of regular expressions can be specified.

If the result is ambiguous, the correct variable has to be picked manually from a drop-down list of the multiple matches. Ultimately, empty response invokes list of all available variables.

As an useful alternative in many instances, the system allows to enter a formula that can involve any combination of names of existing numerical columns with constants, brackets, arithmetic operators +-*/^ and R functions. The most useful of these are '*sqrt*' (square root), '*log*' (natural logarithm), '*log10*' (common logarithm) and '*exp*' (exponential function). Potentially useful can be also '*min*', '*max*' (extremes), '*length*' (number of elements/cases), '*sum*', '*mean*', and '*prod*' (product of the elements). In any case, no one prevents the advanced users from writing their own functions that can be invoked here.

When multiple columns are to be selected (e.g. for multiple or correlation plots), the easiest way is to type in directly their names separated by commas. Alternatively, a comma-delimited list of sequence numbers can be used that may also contain ranges expressed by colons. Also user-defined or built-in lists can be employed, such as 'REE', or their combinations with the column names.

Calculations

All the calculation algorithms produce output temporarily stored in the memory, in the variable (a vector, matrix or list) *results*. Before they are rewritten by the next calculation procedure, they can be saved or copied to clipboard. If the recalculation of the complete current dataset is needed, type *minMain()* into the R command prompt.

Plotting

The plotting symbols and colours can be allocated, if need be independently of each other, according to current grouping or any of the labels. If two distinct criteria for symbols and colours are chosen, two legends are built (see *http://blog.gcdkit.org/2017/04/legends.html*).

The *GCDkit.Mineral* produces publication quality plots in PDF or PostScript formats that can be easily imported into other graphical or DTP package for further editing (simply right-click the graphical window and select the desired format). All the plots can be saved simultaneously to PDF (*Save all graphics to PDF*) or PostScript (*Save all graphics to PS*).

The strength of *GCDkit.Mineral* consists in the fact that most of the plots (the stand alone, i.e. not multiple, diagrams) are defined as templates compatible with *Figaro*, a set of graphical utilities for R.

Figaro provides means to create figure objects, which contain both the data and methods to make changes to the figure (via the menu *Plot editing*). So, for example, the title can be changed or the histogram fill colour altered and any changes are automatically made visible on interactive devices. In addition, one can zoom in and out of the data. *Figaro* objects currently permit the editing of the text, font, size and colour of the main title, sub title and axis labels; colour, size, and symbol of points; colour, line type, and width of lines. Thus *Figaro* provides a degree of interactive editing before committing to hardcopy.

GCDkit.Mineral also employs the concept of the so-called plates, i.e. collections of two or more Figaro plots in a single plotting window. The examples are Harker plots, or diagrams using the same Figaro template plotted for individual data groups (menu item "*Multiple plots by groups*" – function *figMulti*), based e.g. on the locality they come from. Alternatively, plates can be fully user-defined and contain any mixture of Figaro-compatible plots, such as binary or ternary plots.

On the Windows GUI, a single plot can be selected by graphical cursor from the plate already plotted (just right click the plot window and invoke the menu option *Select slot – Select another slot*) and subsequently edited in the manner previously available exclusively to stand-alone Figaro-compatible plots (submenu of the same menu called *Plot editing*). Moreover, properties of the whole plate or all its diagrams can be changed simultaneously (submenu *Plate editing*). For instance, the whole plate can be set to black and white, the font size of axis labels can be altered, scaling of the common axis on Harker plots set, or the minima of all y axes of binary plots set to 0.

NOTE Use menu item *Plate redraw* (attached to the function *plateRedraw*), or dedicated *Export to PS (Colour)* and *Export to PS (BW)* menu items from *Plate editing* submenu to produce "clean" version of your plate before exporting, if prior further editing in a specialized graphical program (CorelDraw, Adobe Illustrator…).

Limitations of *GCDkit.Mineral* 1.00

The current version of *GCDkit.Mineral* is intended especially for processing of electron-microprobe data, possibly with supplementary FeO and Fe_2O_3 analyses by an alternative method. Its known limitations are:

- Apart from Fe, it cannot handle elements with multiple valencies (such as Mn)
- No H_2O determinations are taken into account.
- Missing is import and treatment of supplementary trace-element data, acquired by laser-ablation inductively-coupled mass-spectrometry (LA ICP-MS) or ion probe. Also spiderplots are not implemented yet.
- Missing is switching and overplotting of multiple datasets.

- Limited palette of classification schemes and pre-defined graph templates for individual minerals
- the data can be currently imported (exported) solely from (to) text files or clipboard; also some other functionality (like command history or interactive of editing of plates of several graphs) cannot be implemented.

Known main differences of implementation on Mac and Linux from the (native) Windows version

- Supported are only reading/writing of text files (TXT/CSV), clipboard, and DBF files,
- Missing import/export from/to XLS, XLSX and MDB,
- No interactive editing of plates,
- Language versions for classification plots not implemented,
- Command history does not work correctly.

Where to go next?

Please note that *GCDkit.Mineral* shares many functions, feel and look of its older cousin designed for recalculation and geochemical presentation of whole-rock geochemical data, called *GCDkit* (Janoušek *et al.*, 2003; 2006; 2011). Arguably a very useful introduction to using R with, or without, *GCDkit* in interpretation and modelling of whole-rock geochemical data provides our monograph (Janoušek *et al.*, 2016). Courtesy of the publisher, the Appendices explaining the R syntax and the workings of the *GCDkit* system, are freely available online at: *http://blog.gcdkit.org/2015/10/*. But if you want to support our efforts, and learn more, buy a paper or electronic copy!

Enjoy! On behalf of the authors

Vojtech Janousek
vojtech.janousek@geology.cz

23 May 2023

http://www.gcdkit.org

References

Dachs, E. (1998). PET: petrological elementary tools for Mathematica. *Computers & Geosciences* **24,** 219–235.

Dachs, E. (2004). PET: Petrological Elementary Tools for Mathematica®: an update. *Computers & Geosciences* **30,** 173–182.

Deer, W. A., Howie, R. A. & Zussman, J. (1992). *An Introduction to the Rock-Forming Minerals*. Harlow: Prentice Hall.

Deer, W. A., Howie, R. A. & Zussman, J. (2013). *An Introduction to the Rock-Forming Minerals*. London: Mineralogical Society.

Droop, G. T. R. (1987). A general equation for estimating Fe^{3+} concentrations in ferromagnesian silicates using stoichiometric criteria. *Mineralogical Magazine* **51,** 431–435.

Holland, T. J. B. & Powell, R. (1998). An internally consistent thermodynamic data set for phases of petrological interest. *Journal of Metamorphic Geology* **16,** 309–343.

Janoušek, V., Farrow, C. M. & Erban, V. (2003). GCDkit: new PC software for interpretation of whole-rock geochemical data from igneous rocks. *Geochimica et Cosmochimica Acta* **67,** 186–186.

Janoušek, V., Farrow, C. M. & Erban, V. (2006). Interpretation of whole-rock geochemical data in igneous geochemistry: introducing Geochemical Data Toolkit (GCDkit). *Journal of Petrology* **47,** 1255–1259.

Janoušek, V., Farrow, C. M., Erban, V. & Trubač, J. (2011). Brand new Geochemical Data Toolkit (GCDkit 3.0) - is it worth upgrading and browsing documentation? (Yes!). *Geologické výzkumy na Moravě a ve Slezsku* **18,** 26–30.

Janoušek, V., Moyen, J. F., Martin, H., Erban, V. & Farrow, C. (2016). *Geochemical Modelling of Igneous Processes – Principles and Recipes in R Language. Bringing the Power of R to a Geochemical Community*. Berlin: Springer.

Kretz, R. (1983). Symbols for rock-forming minerals. *American Mineralogist* **68,** 277–279.

Richard L. R. (1995). MinPet: Mineralogical and Petrological Data Processing System, Version 2.02. MinPet Geological Software, Québec, Canada.

Warr, L. N. (2021). IMA-CNMNC approved mineral symbols. *Mineralogical Magazine* **85,** 291–320.

Whitney, D. L. & Evans, B. W. (2010). Abbreviations for names of rock-forming minerals. *American Mineralogist* **95,** 185–187.

Appendix: Error messages produced by the function testing the database integrity

1. **"Unclear recalculation scheme"** Neither *oxygens*, *charges* or *atoms.sum* set

2. **"For fixed oxygens recalculation, neither number of atoms or charges can be set"** Recalculation to a fixed number of oxygens in the formula required, but *atoms.sum*, or *charges* are set that are not needed

3. **"For fixed charge recalculation, neither number of oxygens or iron recalculation options can be set"** Recalculation to a fixed number of charges in the formula required, but *oxygens*, or *iron* are set that are not needed

4. **"For fixed atoms pfu recalculation, neither number of oxygens, charges or iron recalculation options can be set"** Recalculation to a fixed number of atoms per formula unit is required, but *charges*, or *iron* are set that are not needed

5. **"Number of oxygens not set"** Oxygens not set and Fe recalculation is required

6. **“Invalid atoms in atom names…”** Invalid (non-existent in the nature) item(s) specified in the slot *atom.names*
7. **“Duplicated atoms in atom names…”** Duplicated items in *atom.names*
8. “**Invalid atoms in sites allocations…”** An attempt to allocate non-existent atoms (i. e. some atoms specified in sites allocation *(sites)* were not listed in *atom.names*)
9. **“Missing atoms in allocations…”** Mismatch between *atoms* and site allocation *(sites)*
10. **“Missing allocated atoms in atom.names…”** Mismatch site allocation *(sites), atoms*
11. **“Mismatch in crystallographic sites definition”** Mismatch in length of sites definition (unequal number of items in *sites* and *site.sums*)
12. **“Invalid Fe recalculation option”** Invalid Fe^{II}/Fe^{III} option (neither of "Droop", "FixedCats", "allFeII", "allFeIII", "PxPapike","8Si", "16CAT", "15eNK", "15eK", "13eCNK", "8SiAl", "10sumFeIII", "avg")
13. **“FeII/FeIII required and number of cations not set”** Fe^{II}/Fe^{III} estimation required by fixed number of cations *(iron=”FixedCats”)* and number of *cations* is not set
14. **“Site required for Fe recalculation not found”** For Fe^{II}/Fe^{III} estimation by *FixedCats,* the *cations.site* must be a valid site (i.e. listed in *sites*)
15. **“Site required for Fe recalculation ….. cannot have a set sum in atoms allocation”** For Fe^{II}/Fe^{III} estimation by *FixedCats*, the *cations.site* cannot have a fixed sum in the sites allocation (site allocation algorithm is used as a first estimate for the iterations)
16. **“Mismatch in extra parameters definition”** Mismatch in length of extra parameters definition (unequal length of *values.formulae* and *values.names*)
17. **“Mismatch in end-members definition”** Mismatch in length of end-members definition (unequal length of *end.member.formulae* and *end.member.names*)