

OM 5 Useful functions

Standard recalculation, classification and plotting

Function	Purpose
<code>minMain(mineral,min.slots)</code>	Master function for complete recalculation of analyses of the given <code>mineral</code> , using the list of options in <code>min.slots</code>
<code>minClassify(mineral)</code>	Classifies all the analyses belonging to the given <code>mineral</code> class
<code>minClassifyPlot(mineral)</code>	Plots one or more classification diagrams applicable to the given <code>mineral</code>
<code>HTMLFormula(mineral)</code> <code>HTMLFormula.all(minerals)</code>	Produces a HTML version of formatted structural formula for the given <code>mineral</code> (or several of them)
<code>recalcOptions(mineral)</code>	Returns recalculation options for the given <code>mineral</code> class
<code>selectByMineral(slot,mineral)</code>	Returns data stored in the given <code>slot</code> for selected <code>mineral(s)</code>

Auxiliary functions

Function	Purpose
<code>molecularWeight(formula)</code>	Returns a molecular weight of an oxide specified by the <code>formula</code> . The atomic weights come from the official CIAAW web site, https://www.ciaaw.org .
<code>oxide2oxide(formula1, formula2)</code>	Returns a recalculation factor for recalculation between two different oxides of the same element; e.g., <code>oxide2oxide("Fe2O3", "FeO")</code>
<code>oxide2ppm(formula, where="WR")</code> <code>ppm2oxide(formula, where="WR")</code>	Recast concentrations of an oxide (in wt. %) to that of appropriate cation (in ppm) and vice versa.
<code>minComp(n.at, oxides=major)</code>	Calculates chemical composition (in wt. % of oxides) based on a named vector with numbers of individual atoms..
<code>plComp(An=0)</code>	Function calculating chemical composition of an ideal plagioclase with the given molar proportion of anorthite (An) component (0 to 1).
<code>olComp(Fo=1)</code>	Function calculating chemical composition of an ideal olivine with the given molar proportion of forsterite (Fo) component (0 to 1).
<code>idealMineralCompositions(my.mins)</code>	Calculates chemical compositions (in wt. % of oxides) of the selected rock-forming minerals. Their ideal mineral formulae are based on Le Maitre (1982). For instance, <code>idealMineralCompositions(c("Fluorite", "Albite", "Magnetite"))</code>

Useful for programming, for instance of plugin modules

Function	Purpose
<code>.which.min(multiple = TRUE)</code>	Allows to choose a name of single mineral (or several of them, as controlled by the parameter <code>multiple</code>) with any analyses in the current dataset
<code>.formulaCount(mineral)</code>	Returns formula, number of cations and anions for any mineral class in the standard database; e.g., <code>.formulaCount("feldspar")</code>
<code>.getClassPrototype(mineral, slot)</code>	Returns a default value for the specified <code>slot</code> of the given mineral in the standard database; e.g. <code>.getClassPrototype("feldspar", "full")</code>
<code>.atoms.from.formula(oxides, valency=FALSE, Fe.only=FALSE)</code>	Converts names of oxides to atom names, optionally with indication of a valency state; e.g., <code>.atoms.from.formula(c("MgO", "Al2O3"))</code>
<code>.callGCDkit(fun, mineral = NULL, what = NULL)</code>	Call the function "fun" of the standard <i>GCDkit</i> for analyses of mineral stored in the slot <code>what</code> . For instance, <code>.callGCDkit("binary", "clinopyroxene", "recalc")</code>

Le Maitre, R.W. (1982) Numerical Petrology. Developments in Petrology 8, Elsevier, Amsterdam, 281 pp.