SitCom

Comparison of Substructure Sites for Macromolecular Phasing

Please Cite:
Dall'Antonia, F. & Schneider, T. R. (2006). *SITCOM: a program for comparing sites in macromolecular substructures*. **J. Appl. Crystallogr. 39**, 618-619.

SitCom has been funded by the European Commission as part of the projects AutoStruct (contract QLRI-CT-2000-00398) and BioXHit (contract LHSG-CT-2003-503420)

SitCom's internal symmetry library was created using CCTBX: Grosse-Kunstleve, R. W. & Adams, P. D. (2003). *Newsletter of the IUCr Commission on Crystallographic Computing*, **1**, 28-38

0.17.3 User Guide / Program Documentation

SitCom is a program for the comparison of heavy atom sites from two or more macromolecule derivative substructures.

The most recent version 0.17.3 has beta-test status.

How to obtain the program

The distribution comes as precompiled binary for Linux and MacOS. It is obtained via download.

- First, go to the download form using the top menu, fill the form and submit.
- after a few minutes you will receive an e-mail with the download page link.
- having been directed to the new page, right-click the actual download link and save the sitcom-0.17.3.tgz to your hard disc.

SitCom set-up and usage are extremely easy. There is no further installation needed after having placed the tarball to the desired place and extracted there. Only if you are planning to use SitCom as a group, some book-keeping might be recommended. Therefore, the following few explanations address both 'administrators' and private single-users. Some of the technicalities mentioned hereare obvious for people with a little Linux experience.

Contents of this guide

Startup 3

Contents of the distribution 3

What to do after the download 3

Input Guide 4

How to run the program 4

Operating SitCom from the command line 4

Card mode and hybrid mode 6

Output 7

Output files 7

Summary file 7

NCS file 7

Visualizing sites in the output PDB 8

Syntax reference 10

Writing input cards 10

About comments 10

Keyword index 11

Keyword pages 12-32

Startup

In this section we will explain the first steps after having obtained the distribution tarball. The startup section was primarily written for the download webserver; if you are reading this HTML document from within the extracted distribution or after having invoked "*sitcom -help*" from your local terminal, you are past the preparation steps. In that case, skip to the input section.

Contents of the SitCom distribution

Upon using tar:

```
tar xzvf sitcom-0.17.3.tgz
```

the distribution will be extracted to a folder 'sitcom' which contains:

- the program binary 'sitcom-0.17.3'
- a very short read.me file (being a copy of this section)
- the user guide sitcom_guide.pdf (this document)

What to do after the download

The initial set-up of SitCom is easy, because the program is precompiled and no proper installation is needed. SitCom (current version: SitCom 0.17.3_debug binary) is a single stand-alone executable file. This means that there are no such dependencies as shared libraries or 3rd party/host/client programs involved. If you are a somewhat advanced UNIX user, feel free to skip the following part and go directly to the input section.

For a personal install:

copy the compressed tar file to some place in your home directory	`cp sitcom-0.17.3.tgz /my/home`
uncompress and extract the archive in one go	`tar xvf sitcom-0.17.3.tgz`
append the new program path to the $PATH variable.	`setenv PATH ${PATH}:/my/home/sitcom`
create a symbolic link, in-place:	`cd /my/home/sitcom` `ln –s sitcom-0.17.3.tgz sitcom`
as a test, start sitcom from your home (then quit with <Ctrl-C>)	`cd /my/home` `sitcom`

A system-wide install would mean to copy sitcom-0.17.3_debug_arch to the /usr/local/bin folder.

Input guide

In this section we will explain the two principal ways to start and operate the program. Every sample command line contains two cells: Left a formalized notation and right a concrete command example.

How to run SitCom

Being a typical terminal-operated program, SitCom is called from the command line. Thus it accepts various command line arguments, but even more parameters can be provided by means of input-card scripts.

If only a pair or a small set of pdb files shall be compared, their names can be supplied directly to the program call. SitCom will then run without using input cards, setting default parameters where appropriate.

For proper computation tasks, it is recommended to direct the terminal output (STDOUT) of SitCom into a log file.

Operating SitCom from the command line

The command line is the most intuitive input mode, in particular for pairwise substructure comparison. Here we present the essential command line constructs.

sitcom	`sitcom`

Type only the program name.
SitCom will produce the inital text output including self-identification of the program (version & build information), author acknowledgements, and, most importantly, a list of allowed command line arguments. This includes the command to create SitCom documentation.

sitcom -help	`sitcom -help`

Use the **-help switch** as sole argument.
SitCom will write some concise text to STDOUT, and an instance of the program guide html-file, identical to this document, is produced in place (useful to quickly create local file copies of the guide).

sitcom { pdblist }	`sitcom transh_fa.pdb transh_solve.pdb`

type a list of pdb files without extra arguments.
SitCom will do a standard site comparison (most basic usage)
- taking input sites from >= 2 pdb files. { pdblist } = "name1.pdb name2.pdb name3.pdb ..."
- requiring that **the 1st of them contains unit-cell parameters and a space group symbol** (CRYST1 line).
- considering every ATOM/HETATM line, i.e. suited if each of the pdb files contains an experimental substructure (only).

sitcom { pdblist } -sg N	`sitcom jia_fa.pdb jia_solve.pdb \` `jia_snb.pdb -sg 20`

supply the -sg switch followed by an integer number, in addition to the list of pdb files.
SitCom requires it if the pdb files lack the space group symbol. **Caution:** If the 1st pdb file contains the space group, number N will override it!
- for the space-group numbers see sitcom's spacegroup tables

sitcom structA.pdb_EL structB.pdb_EL	`sitcom 1FJ2.pdb_BR haptbr_fa.pdb_S`

use extended file names with _EL appended.
SitCom will use the EL tag (chemical element symbol) to filter input sites from a larger set of pdb atoms. - this is useful in case of either mixed substructures or otherwise composite models, for example if one of the compared pdb files contains a refined protein model (native or derivative).

sitcom single.pdb -ncs N D	`sitcom 1vkd_sites.pdb -ncs 30 20.0`

provide the name of a pdb file together with the -ncs switch and two numerical parameters.
SitCom will perform NCS analysis, screening 2-fold to 12-fold symmetry, using
- N = number of sites to take for triangulation (integer)
- D = search radius for triangulation (real number in Angstrom units).
The NCS module needs only one pdb file. If you give more, the consensus model will be analyzed.

There is quite a number of switches, and each one has an equivalent input card. Therefore, switches are systematically explained in the Reference section. Additionally, every allowed switch is listed at the beginning of the regular program output.

Operating SitCom in card mode and hybrid mode

Input cards are recommended for more complex tasks involving specific parameters. Moreover, card 'script' files are practical for re-usable batch input. Strictly speaking, such plain-text files are not (program) scripts but rather batch files for input card streaming via < STDIN. Thus they are provided directly with the command line call.

sitcom < cardfile	`sitcom < jia.inp`

provide input cards through < STDIN. This can be done using the keyboard, but if a file name is provided (as recommended), SitCom will read the file stream instead.
It expects a plain-text file containing **one interpretable input card per line**. Any file name is accepted, though we recommend the .inp extension.
- the input card syntax is explained in the Reference section.

Hybrid operation mode means that scripted cards can be combined with command line switches. There is one essential restriction, though: If site input is provided through command-line pdb file names, then input cards are completely ignored

sitcom -h -sg N < cardfile > logfile	`sitcom -h -sg 20 < jia.inp > jia.html`

Use one or more switches in deliberate order, then < / > directives.
Common case of hybrid usage: input sites are provided through cards from a file, additional parameters through the command line directly.
- HTML format (-h) is the most likely option you will use a switch for; direct the STDOUT log text to a .html file for readability.
- both switch effects could alternatively be achieved using input cards.

Output

In this section we will explain what output SitCom produces and which information the output provides.

Output files

A core SitCom run produces three explicit output files: A summary file, a pdb file and a res file. Their common base name ('name') is the one given by the str_name card. If this card is missing, or if SitCom has been run without a script, the output files will be called 'sitcom' - be careful not to overwrite them by later jobs. If NCS-analysis has been chosen there will be another pdb file. The summary file will have the extension '_summary.html' if html-format output was generated (-h switch or html_out card), otherwise the extension will be '_summary.txt'.

- **name_summary.html** (name_summary.txt) contains comparison result tables and the scored consensus model.
- **name_consensus.pdb** contains the sites of the consensus model, sorted by their score, in pdb format.
- **name_consensus.res** contains the same model, only in Shelxd res format.
- **name_ncs.pdb** contains the subset of (single-input or consensus) sites that are NCS-related.

The pdb sites can be used for many programs, for example for phasing with SHARP. The res sites are for input to the phasing program SHELXE. The NCS sites are assigned to PDB chains according to the determined monomers and can serve as input to DM (see later).

- Most likely, there will be another output file, namely the log file into which the direct SitCom terminal messages were directed.
- If the **make_pdbfit** card has been used, there will also be a file **name_fit.pdb**.

The log file is of interest if you want to learn more about the crystallographic relationship between the solutions. The fit-file does not contain the unique site positions (consensus model), but ALL input sites with chain numbers according to the solution they belong to. The site positions have been transformed to common (fitting) equivalents.

The summary file

SitCom's major results are contained in this file. It is designed in a self-explanatory way, so that no extra help topic is neccessary. Here, it shall only be mentioned that the summary file consists of two sections, the first being two tables of solution cross-comparison results, the second being a verbose table of the consensus model made of unique site positions and their (SFOM) scores.

The NCS file

This PDB-format file can be conveniently used as an interface to DM, as it contains the NCS-related sites (if any), assigned to monomeric groups, as well as verbose remarks on the NCS operators. Copy the ATOM lines of a monomer to a separate file and run NCSMASK on it to create a mask for DM. The lines with the operators (matrix, polar or euler angles) can be directly copied to a DM script or to the corresponding CCP4i interface.

Visualizing sites in the pdb files

Both the standard output file name_consensus.pdb and the optional file name_fit.pdb can be used to visualize the consensus substructure. Use a program like Rasmol and chose a space-filling mode to display the sites as spheres.

- The consensus model file (name_consensus.pdb) contains B-values derived from the SFOMs of the unique sites: The more reliable a consensus site, the lower the B-value. Therefore, use a temperature color-scheme in your graphics program and look for 'cold' sites.
- The all-sites-fit file (name_fit.pdb) contains **every** input site, and a different chain number for every input solution. Use a color-by-chain scheme in your graphics program. The solutions are not containing their original site positions, but have been transformed to the hand and origin of the consensus model (which was derived from the first solution).Moreover, for individual sites, if agreeing to unique positions, the fitting symmetry equivalent is used. Therefore, this 'solution-superposition' structure looks similar to the consensus model. At the high-reliability positions (low B-values in name.pdb), here you should find very close-by sites of different solutions that account for the same unique site.

The NCS file is also composed by PDB-chains, each containing the site group that corresponds to a protein monomer. Therefore the chain visualization in a graphics program quickly reveals the NCS arrangement of the substructure. Note that in some cases sites seem to be "mixed up" between monomers. This is not a wrong result, as it is consistent to the NCS operators, but it means that the monomers may interfuse. For best DM results, consider to swap the "outlyers" manually.

Tutorial

The tutorial is still under development, sorry.

Syntax reference

This reference treats all available input cards, together with the corresponding command line switches, if existing.

Dual approach

SitCom understands all of its instructions on what to compare and how to compare by means of input cards. Exploiting the card syntax, every adjustable program parameter can be accessed. The user will experience that using a script with input cards is the most powerful way to control SitCom.

On the other hand the command-line mode is very convenient and user-friedly, suited to address most of the typical tasks in a more straight-forward fashion than possible with cards. You will find that in particular the one-keyword cards (fix_asunit, html_out, etc.) can be substituted by switches (-fix, -h) for more flexible and convenient usage.

Writing input cards

Most of the input cards are not essential, but useful (see Tutorial section). Every input card is defined by a name (= keyword). Most input cards have additional arguments, mainly numerical values, that are interpreted as parameters.

the general syntax is: `KEYWORD [ARG-1 [ARG-2 [...] ] ]`

- each card is a separate line of an input script file.
- the line must start with the keyword, preceding blank-spaces are not accepted.
- the space between the arguments may be any number of blank-spaces (free format).

Is the order of input cards important?

The order of the cards is in general not important, but there are two exceptions:

- Input substructures are read in the order of their source cards (read_sol or read_set). It is recommended to specify the most-trusted solution first, because its site peaks will usually define most of the consensus model coordinates and occupancies.
- If you want to check alternative indexing of a certain data/substructure source, the alt_index card should directly follow the respective source card.

About comments

Comments can be placed anywhere in the input card file. They are initiated with the hash character (#) You can write separate comment lines, starting with a hash, or put a comment in a card line, after the last argument. For example:

```
    # comment1: the following line contains the unit cell    unit_cell
73.46  55.35  110.7  90.00  103.25  90.00  # comment2: this line contains
the cell
```

Keyword index

Xtal-related	| unit_cell | space_group | alt_index | fix_asunit |
Site-related	| read_sol | read_set | deriv_atyp | nsites_exp |
Comparison-related	| max_dist | max_proj | set_weights | restrain_comp | tric_pedantic |
NCS-related	| ncs_triang | ncs_match | ncs_loop |
Output-related	| str_name | make_pdbfit | merge_coord | merge_occup | html_out |

unit_cell

This card provides the unit cell parameters, mainly used to derive the matrices for coordinate transformations (fractional <-> cartesian). If no script is used, the cell is taken from a pdb file.

Prototype	`unit_cell EDGE-A EDGE-B EDGE-C ALPHA BETA GAMMA`
Example:	`unit_cell 73.46 55.35 110.7 90.00 103.3 90.00`

Parameter	Default	Explanation
EDGE-A, ...	*none*	Decimal Ångstrom values of the three unit cell edges
ALPHA, ...	*none*	Decimal degree values of the three unit cell angles

Equivalent switch: *none, but can be read from pdb*

space_group

This card provides the spacegroup, used to derive the symmetry operators and allowed origin shifts to be applied on the sites.

Prototype	`space_group NUMBER`	
Example:	`space_group 19`	
Parameter	Default	Explanation
NUMBER	*none*	the spacegroup number as defined by the International Tables, see sitcom's spacegroup-tables.
Equivalent switch: -sg NUMBER		

alt_index

This card applies to the solution(s) defined by the latest previous read_sol or read_set card, i.e. it should be placed directly in the following line. It causes the transformation of the substructure(s) to a different setting, corresponding to the indices given. This is relevant for some space groups, but only if (a) the substructures are due to different data sets and (b) a lack of consistence is observed upon comparison.

Prototype	`alt_index SCHEME`	
Example:	`alt_index -h,-k,l`	
Parameter	Default	Explanation
SCHEME	*none*	The string for the new indices h',k',l'. An alternative indexing regime of reflection data must be considered for 3 crystal classes: [3]: h',k',l' = -h,-k,l || k,h,-l || -k,-h,-l [32]: h',k',l' = -h,-k,l [23]: h',k',l' = k,h,-l If you want to test all indexing schemes in one run, use multiple solution cards refering to the same source, each followed by the appropriate alt_index card.
Equivalent switch: *none*		

fix_asunit

keyword-only card to keep the original input site positions for the consensus model, respectively change coordinates only with respect to consistent enantiomorph/origin. By default, the program will initially transform all site positions to their equivalents closest to the origin, in order to limit the comparison range. However, this may result in a weaker NCS-search performance in rare cases.

Prototype	`fix_asunit`
Example:	`fix_asunit`

Equivalent switch: -fix

read_sol

This card provides access to a single-solution file containing sites. For inter-solution comparison, at least two of these cards should be used. Format must be PDB.

Prototype	`read_sol TAG WEIGHT FILENAME N(SITES) [ SYMBOL ]`	
Example:	`read_sol SOLVE 1.0 jia_solve.pdb 8`	
Parameter	Default	Explanation
TAG	'SOL-*N* '	a meaningful label to specify the source of the solution, e.g. a program name
WEIGHT	1.0	a decimal value, 0.0 <= WEIGHT <= 1.0, to indicate the (subjective) 'importance' of the solution relative to others
FILENAME	*none*	the complete name (including path if neccessary) of the file from which the sites shall be read
N(SITES)	250	how many sites to read from the pdb file, if less than the complete set shall be read.
SYMBOL	*none*	Type tag (EL symbol) to select a subset of atoms. Optional parameter (use for refined protein models).

Equivalent switch: filename.pdb[_EL] (*command line argument without switch status*)

read_set

This card provides access to a multi-solution file in SHELXD-format (.lst). All sets of sites ('solutions') or a selection thereof will be read. **The total number of solutions (from all read_sol and read_set cards combined) must not exceed 120**

Prototype	`read_set TAG WEIGHT FILENAME N(SOLUTIONS) N(SITES)`	
Example:	`read_set REMOTE 0.7 jia_remote.lst 5 11`	
Parameter	Default	Explanation
TAG	'SET-*N* '	a meaningful label to specify the source of the solution: the program name, a dataset-wavelength etc.
WEIGHT	1.0	a decimal value, 0.0 <= WEIGHT <= 1.0, to indicate the relative 'importance' of the solutions from this source, compared to others.
FILENAME	*none*	the complete name (including path if neccessary) of the file from which the sites shall be read
N(SOLUTIONS)	1	how many solutions to select. After sorting all solutions by FOM values, SitCom will store the top N.
N(SITES)	250	how many sites-per-solution to read from the lst file (for each selected solution alike)
Equivalent switch: *none, multi-solution files can only be used through input cards*		

deriv_atyp

This card provides a chemical element symbol, which is used as atom type label in some output files.

Prototype	`deriv_atyp SYMBOL`	
Example:	`deriv_atyp BR`	
Parameter	Default	Explanation
SYMBOL	'SE'	can be any label; chemical element symbols (EL) are most meaningful for the site atom type.
Equivalent switch: -at SYMBOL		

nsites_exp

Specifies the number of expected sites, if there is such prior knowlewdge.

Prototype	`nsites_exp NUMBER`	
Example:	`nsites_exp 40`	
Parameter	Default	Explanation
NUMBER	*none*	the NUMBER top sites of the consensus model will be highlighted. Otherwise there will be no effect.
Equivalent switch: -nx NUMBER		

max_dist

This card determines the distance tolerance/limit for the identification of matching sites.

Prototype	`max_dist THRESHOLD`	
Example:	`max_dist 3.0`	
Parameter	Default	Explanation
THRESHOLD	1.5	a decimal value for the allowed inter-position distance (in Ångstrom) between two sites of different solutions. If their distance is below THRESHOLD, they are considered as two instances of the same unique site. If it is greater than THRESHOLD, they will become two separate unique sites.
Equivalent switch: -d THRESHOLD		

max_proj

This card determines the distance tolerance for sites in the non-polar plane of polar unit cells. The parameter is ignored for non-polar spacegroups.

Prototype	`max_proj THRESHOLD`	
Example:	`max_proj 0.7`	
Parameter	Default	Explanation
THRESHOLD	1.0	a decimal distance value in Ångstrom, as for max_dist. If used it should be 50-100 ÷ of the max_dist value.
Equivalent switch: -p THRESHOLD		

set_weights

With this card, the scoring of output sites is controlled by weighting three contributions to the SFOM.

Prototype	`set_weights W-FREQ W-ACCUR W-OCCUP`	
Example:	`set_weights 1.0 0.3 0.0`	
Parameter	Default	Explanation
W-FREQ	1.0	weight for the frequency of a site (also called consensus-rate). This is, how many positional instances of a unique site are found in different solutions. 0.0 < W-FREQ < 1.0
W-ACCUR	0.5	weight for the positional accuracy of a site. This is the mean distance between positional instances of a unique site, if more than one are found in different solutions. 0.0 < W-ACCUR < 1.0
W-OCCUP	0.0	weight for the occupancy of a site. The occupancy (corresponding to the peak height) of a unique site in the consensus model is taken from the first instance of that site found. 0.0 < W-OCCUP < 1.0
Equivalent switch: -w W-FREQ W-ACCUR W-OCCUP		

restrain_comp

keyword-only card to refer all subsequent comparisons to the first ('reference') site list. This restrained comparison will prevent the initial seed of sites (from the 1st input) from being extended, and the subsequent site lists can be independently probed for their agreement to the reference.

Prototype	`restrain_comp`
Example:	`restrain_comp`

Equivalent switch: -res

tric_pedantic

keyword-only card that may be used for spacegroup P1. Two solutions in P1 are displaced by a certain 3-dimensional translation vector, and the algorithm assumes that this shift is the most freqent vector found between combinatorically checked site pairs. In case of very small substructures, this approach may not lead to a clear and/or correct result, therefore the pedantic mode may be activated to consider every vector found more than once.

Prototype	`tric_pedantic`
Example:	`tric_pedantic`

Equivalent switch: -tp

ncs_triang

This is one of three cards employed for NCS analysis, and of these it is the only mandatory one. If the ncs_triang card is encountered, the NCS analysis module is activated. If the other two NSC cards are not given, their defaults are automatically set.

Prototype	`ncs_triang N(SITES) RANGE`	
Example:	`ncs_triang 20 15.0`	
Parameter	Default	Explanation
N(SITES)	*all stored*	The number of sites to be used for initial triangle generation (integer), referring to a subset of either a single-solution or a site comparison consensus. By default all sites of the respective set are taken. Reduction of this value limits the search set to a probably more accurate site basis.
RANGE	25.0	This is a decimal Angstrom value directly influencing the number of triangles built. The larger the search range around a given site, the more neighboring sites are found and more combinations of triangle vertices are possible. Increase this parameter in case of large asymmetric units.
Equivalent switch: -ncs N(SITES) RANGE		

ncs_match

This card addresses the parameters for triangle matching, once a list of triangles has been generated. Given a sufficient number of N-fold matches (with N being the NCS-order), up to 100 attempts per NCS run can be made to derive NCS operators from the respective match groups.

Prototype	`ncs_match SIMILARITY #FIRSTTRY N(ATTEMPTS)`	
Example:	`ncs_match 2.0 0 10`	
Parameter	Default	Explanation
SIMILARITY	3.0	a decimal Angstrom value defining the allowed deviation of triangle edges to satisfy the matching condition. The selection of matching triangle pairs (respectively groups) is the basis for NCS operator determination. With the default value, the number of matches is usually less than 100.
#FIRST_TRY	0	starting number for a batch of matches to proceed with. Given a large triangle list in combination with a wide SIMILARITY tolerance, thousands of matches may be found. Since a maximum of 100 matches can be analysed for closed-loop NCS at a time, it might be necessary to run several batches separately.
N(ATTEMPTS)	3	The number of attempts to use from the initial match list, starting with the number given by the previous parameter. `ncs_match 5.0 100 10` would use 10 matches, N(i) = 100 to 109. N(ATTEMPTS) is always truncated to 100. If the running number N(i) exceeds actual maximum match index, it is stopped at N(max).
Equivalent switch: *none*		

ncs_loop

This card specifies the closed-loop NCS order to look for, and thereby the search mode. Secondly, it controls the strictness of NCS-matching for single sites tested against the NCS operators.

Prototype	`ncs_loop ORDER ACCURACY`	
Example:	`ncs_loop 4 3.0`	
Parameter	Default	Explanation
ORDER	1	integer number guiding the search for N-fold NCS, if there is an a-priori expectation. 2 <= N <= 12 : the NCS module searches groups of N matching triangles and determines N NCS-operators in parallel. N = 1 : sequential search for orders 2 - 12, yielding a score-based prediction of the most likely order. N = 0 : the closed-loop search is abandoned in favor of a more basic analysis of unrelated operators.
ACCURACY	3.0	a decimal Ångstrom value defining the limiting distance between putative NCS mates. Upon application of a determined NCS operator, the transformed site will be marked as NCS-consistent only if its distance to a mate is <= ACCURACY.
Equivalent switch: *none*		

str_name

This card provides a structure name, which is used for the output file names.

Prototype	`str_name NAME`	
Example:	`str_name jia`	
Parameter	Default	Explanation
NAME	'sitcom'	anything that you can associate with your (sub)structure.
Equivalent switch: -name NAME		

make_pdbfit

keyword-only card that makes SitCom write a file name_fit.pdb of all input solutions with different chain numbers. For the site atom coordinates, the fitting equivalents are written, so that clusters of atoms are formed at unique sites with high consensus rate.

Prototype	`make_pdbfit`
Example:	`make_pdbfit`

Equivalent switch: *none*

merge_coord

keyword-only card that will cause the averaging of coordinates from all instances of sites at a unique position, instead of keeping the coordinates of the first site found at that location.

Prototype	`merge_coord`
Example:	`merge_coord`

Equivalent switch: -mc

merge_occup

keyword-only card that will cause the averaging of occupancies from all instances of sites at a unique position, instead of keeping the occupancy of the first site found at that location.

Prototype	`merge_occup`
Example:	`merge_occup`

Equivalent switch: -mo

html_out

keyword-only card that will switch on the html-formatting of all STDOUT text after the input parsing.

Prototype	`html_out`
Example:	`html_out`
Equivalent switch: -h	

Please remember that the most recent keyword reference is included in the self-generated HTML guide.