

User Guide for the *HELM Antibody Editor V1.1*

Stefan Klostermann, pREDi

Roche Innovation Center Penzberg

23.03.2015

Table of Contents

1	General.....	2
2	First steps with the <i>HELM Antibody Editor</i>	2
3	Run an example.....	3
3.1	Sequence Loader.....	3
3.2	Domain Detection Window.....	4
3.3	Antibody Editing Window	4
3.4	Details Box.....	5
3.5	Connect domains manually.....	5
3.6	Extend the example and add an oligo to a domain	6
4	Options, parameters and processes	9
4.1	Antibody Editor Settings	9
4.1.1	Domain Detection Settings	9
4.1.2	Sorting of hits	11
4.2	Domain Detection Window.....	11
4.3	Antibody Editing Window	12
4.3.1	Cys-Cys bond creation.....	13
4.3.2	Canvas options	13
5	Libraries: Use, syntax and editing	14
5.1	Domain library.....	14
5.2	Mutation library	15
5.3	Autoconnector rule set	16
6	Contact	18

1 General

The *HELM Antibody Editor* V1.1 is a separate application that employs HELM technology to handle complex antibodies for their analysis and registration. It uses the *HELM Editor* V1.2 to enable chemical modifications of domains e.g. to add payloads.

This document describes features of the *HELM Antibody Editor* V1.1 and provides background details on internal libraries used and the internal processing steps.

2 First steps with the *HELM Antibody Editor*

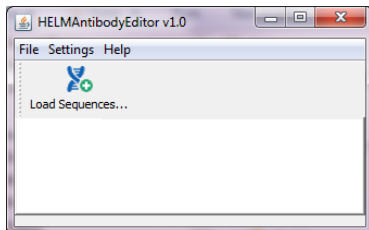
- Build jar file
- Run HELMAntibodyEditor.jar

3 Run an example

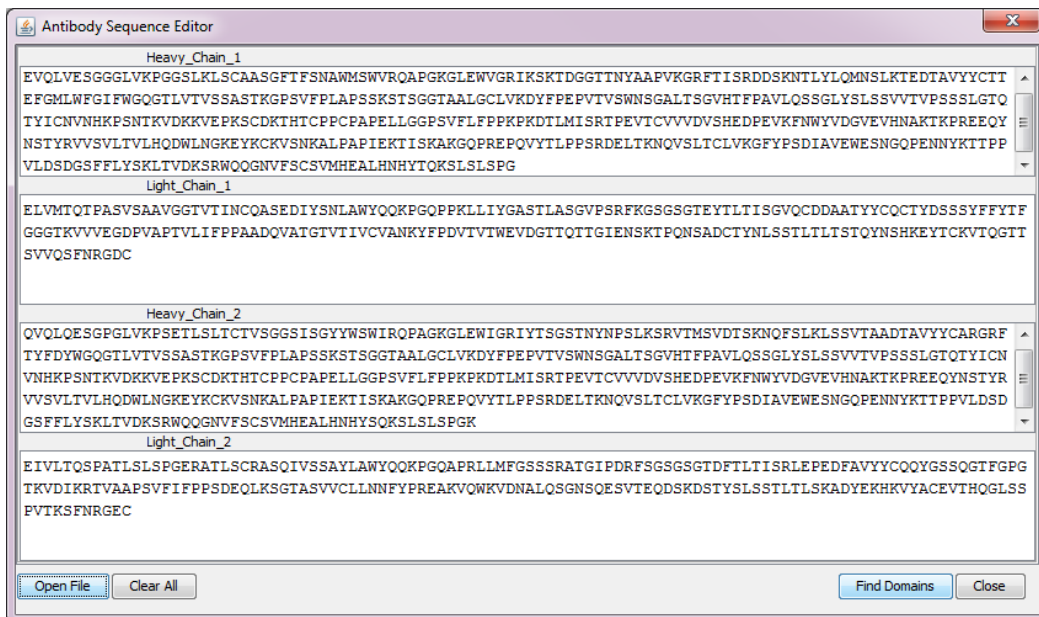
As a practical illustration of the new *HELM Antibody Editor* functionality the following section uses an provided example and showcases the steps and options.

3.1 Sequence Loader

- Start the *HELM Antibody Editor*



- Click “Load Sequences ...” in the ribbon
 - Click “Open File” and open “sample\Test antibody”
- to result in:



3.2 Domain Detection Window

- Click “Find Domains” to result in:

Antibody Domains Found

Heavy_Chain_1

hVh3 (1,0E-54) 72% ID, 100% Cov 1-121 (1-125/125)	hIgG1_CH1 (9,0E-65) 100% ID, 100% Cov 122-219 (1-98/98)	hIgG1_hg (3,0E-08) 100% ID, 100% Cov 220-234 (1-15/15)	hIgG1_CH2 (9,0E-77) 100% ID, 100% Cov 235-344 (1-110/110)	hIgG1_CH3 (5,0E-75) 100% ID, 99% Cov 345-450 (1-106/107)
---	---	--	---	--

Light_Chain_1

rbVka (2,0E-53) 71% ID, 98% Cov 1-111 (1-110/110)	rbCka4 (5,0E-75) 100% ID, 100% Cov 112-215 (1-104/104)
---	--

Heavy_Chain_2

hVh4 (3,0E-63) 83% ID, 100% Cov 1-116 (1-124/124)	hIgG1_CH1 (8,0E-65) 100% ID, 100% Cov 117-214 (1-98/98)	hIgG1_hg (3,0E-08) 100% ID, 100% Cov 215-229 (1-15/15)	hIgG1_CH2 (8,0E-77) 100% ID, 100% Cov 230-339 (1-110/110)	hIgG1_CH3 (3,0E-75) 99% ID, 100% Cov 340-446 (1-107/107)
---	---	--	---	--

Light_Chain_2

hVka3 (5,0E-60) 81% ID, 100% Cov 1-108 (1-112/112)	hCka (5,0E-75) 100% ID, 100% Cov 109-215 (1-107/107)
--	--

Domain Library: jdbc:sqlite:config/config.db3 ('domain_library')

Mutation Library: jdbc:sqlite:config/config.db3 ('mutation_library')

Autoconnector: jdbc:sqlite:config/config.db3 ('autoconnector_config')

Return to Input Re-Annotate changed domains Accept

- Click on domains to see further options (other BLAST hits, assignment of the domain to neighboring ones, restricting domain boundaries to neighboring ones if BLAST hits overlap, edit boundaries manually, delete domain annotation = “None” → s. details in chapter 4.2 “Domain Detection Window”)

e.g.:

rbCka4 (5,0E-75)
100% ID, 100% Cov
112-215 (1-104/104)

rbCka4 (5,0E-75)
100% ID, 100% Cov
112-215 (1-104/104)

rbCka5 (2,0E-53)
77% ID, 98% Cov
115-215 (3-103/103)

rbCka6 (6,0E-51)
75% ID, 98% Cov
115-215 (3-104/104)

rbCka8 (2,0E-52)
71% ID, 100% Cov
112-215 (1-106/106)

rbCka9 (2,0E-46)
67% ID, 98% Cov
114-215 (3-104/104)

rcCka8 (7,0E-34)
50% ID, 97% Cov
116-215 (4-106/106)

Assign to the left

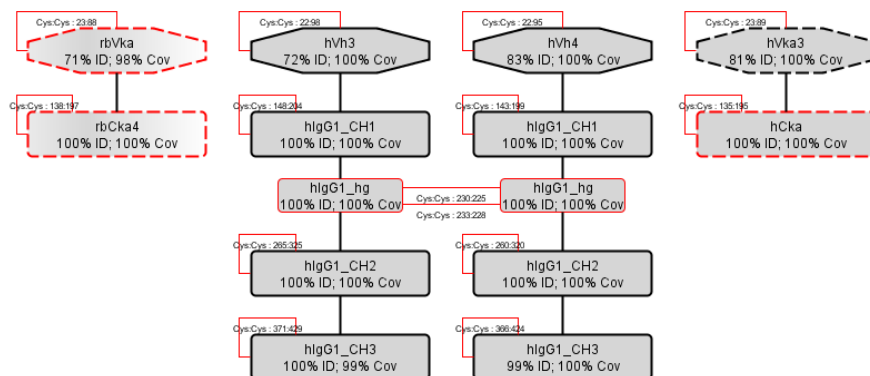
Restrict left border to the neighboring one

Edit boundaries

None

3.3 Antibody Editing Window

- Click “Accept” to result in



- This is a fake antibody with a rabbit light chain to the left. Note that for demonstration purposes in addition to the intradomain Cys-Cys bridge both “rbVka” and “rbCka4” domains contain an additional Cys that form an additional bridge and “rbVka” contains even another (4th) Cys located (located in the CDR3).
- Legend
 - **Shape:** octagons = variable domains, rounded squares = constant domains, small rounded squares = hinge regions, ovals = other domains (e.g. linkers, tags, sites, peptides incl. signal peptides – not present in this example)
 - **Border:** dashed = light chain, solid = heavy chain, red = free Cys residues
 - **Background color:** gray = human, color gradient = can be humanized (e.g. rabbit domains)
 - **Red lines:** indicate the Cys-Cys bridges → the application tries to automatically establish as many of these bridges as possible. Remaining ones have to be done manually.

3.4 Details Box

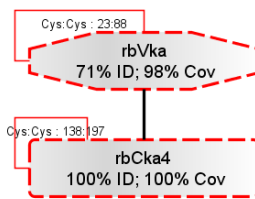
- Click on the upper left domain to see its details in a box to the right:

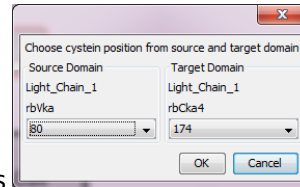
Property	Description
> CHAIN	
NAME	Light_Chain_1
SEQUENCE LENGTH	215
SEQUENCE	ELVMTQTPASVSAAGGTVTINCQASEDIYSNL...
ORIGINAL SEQUENCE	
> DOMAIN	
USER LABEL	rbVka
USER COMMENT	71% ID; 98% Cov
START POS ON CHAIN	1
END POS ON CHAIN	111
LENGTH	111
START POS ON TEMPLATE	1
END POS ON TEMPLATE	110
IDENTITY TO LIB. DOM.	71%
COVERAGE OF LIB. DOM.	98%
LEADING SEQUENCE	
SEQUENCE (DE FACTO)	ELVMTQTPASVSAAGGTVTINCQASEDIYSNL...
TRAILING SEQUENCE	
> ANNOTATIONS	
> MUTATIONS	
> UNKNOWN MUTATIONS	
> CYS BRIDGES	
CYS POS	[23, 80, 88, 90]
CYS BRIDGES	[rbVka:23 <-> rbVka:88]
FREE CYS	[80, 90]
> OTHER BRIDGES	
GENERAL	
> RECOGNIZED DOMAIN FROM LIBRARY	
NAME IN LIB	VKAPPA-CONS_RABBIT
SHORT NAME	rbVka
SPECIES	Rabbit
HUMANNESS	Humanizable
CHAIN	Kappa
DOMAIN TYPE	Variable
SEQUENCE LENGTH IN LIB	110
CYS COUNT	4
PATTERN	[1-3]
COMMENT	
SEQUENCE (TEMPLATE)	DPVLTPSPVSAAGGTVTISCQASQSVYNN...

This box contains all data on the full chain (“>CHAIN”), the domain found as entered (“>DOMAIN”), recognized formats (“> ANNOTATIONS”), known and unknown mutations (“>MUTATIONS”, “>UNKNOWN MUTATIONS”), all, connected, and free Cysteins (“>CYS BRIDGES”), all other bonds e.g. to chemical linkers (“> OTHER BRIDGES”) and the domain from the domain library that has been used to annotate the given domain (“>RECOGNIZED DOMAIN FROM LIBRARY”).

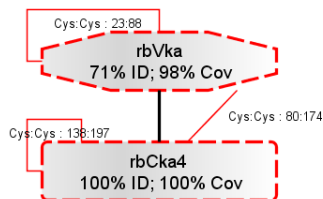
3.5 Connect domains manually

- Drawing a connection is done by clicking on a domain, keeping the (left) mouse-button pressed and dragging to the domain you want to connect to.

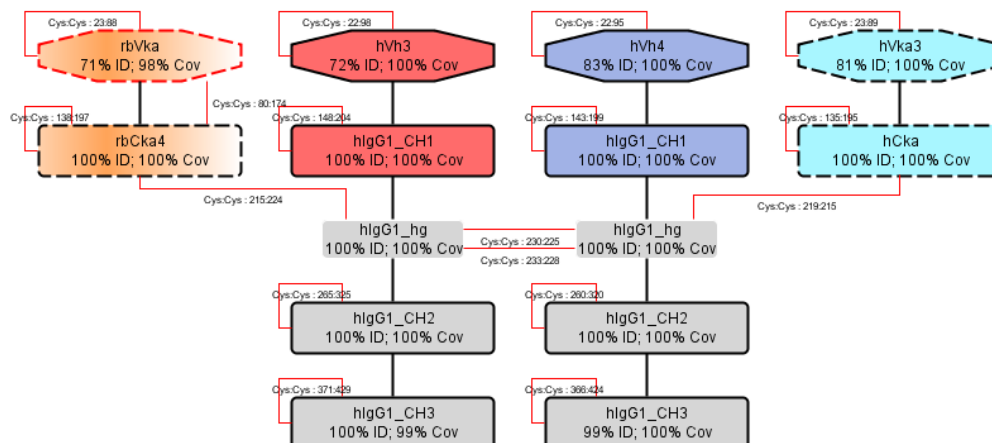
- Thus draw a line between  to connect the additional (rabbit specific)



Cys-Cys bond of these 2 domains. This opens → Click “OK” to see



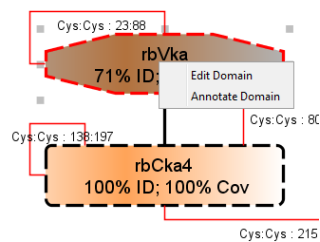
- In the same manner connect “rbCka4” and the left hinge “hlgG1_H” and the right hinge “hlgG1_H” with “hCka” to result in :

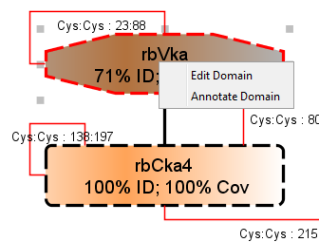


Note that the paired Fv are marked by a common background color and light chains are marked by a more light color shade.

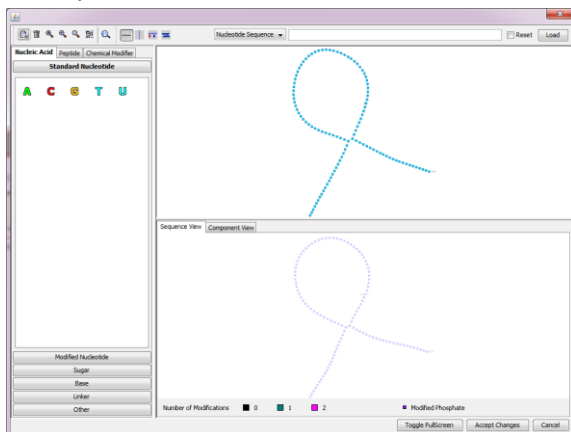
- This antibody is now ready for registration.

3.6 Extend the example and add an oligo to a domain



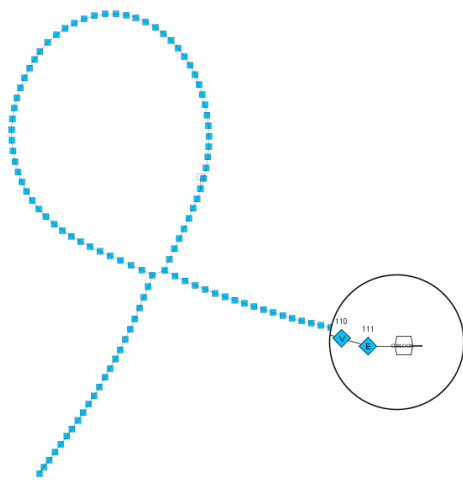
- Right-click the upper left domain  and choose “Edit Domain”

- This opens the *HELM Editor* :

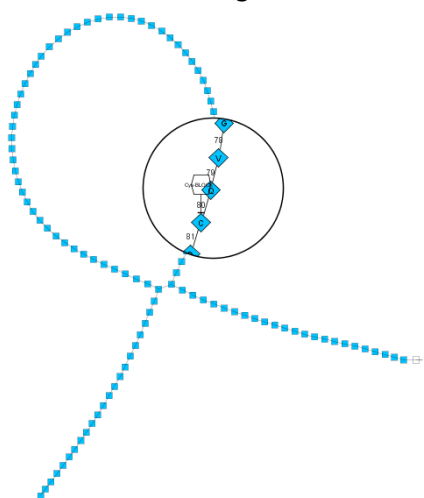


Note that calling the *HELM Editor* the 1st time installs a folder for the monomer library in C:\Users\<user>\.helm if not already done.

- Note the 2 Cys residues that connect to other domains are blocked:
 - The domain C-terminus connected to the N-terminus of the “rbCka4” domain

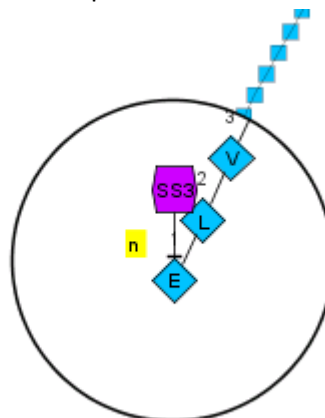


- and the additional bridge into the “rbCka4” domain:



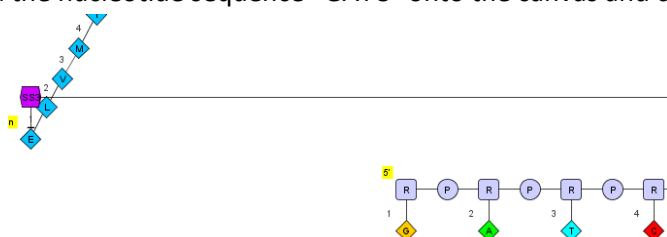
- Add “SS3” from the monomers: “Chemical Modifier” > “Bi-Functional” > Select it and drag it onto the canvas. Connect it to the free N-terminus and chose the appropriate connection

points (or just click “OK” in the dialogue windows for specification of the R-groups involved




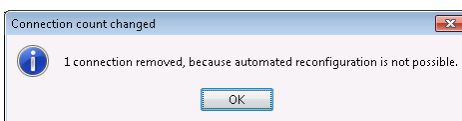
in this connection). This will lead to :

- Load the nucleotide sequence “GATC” onto the canvas and connect it to the “SS3” to see:

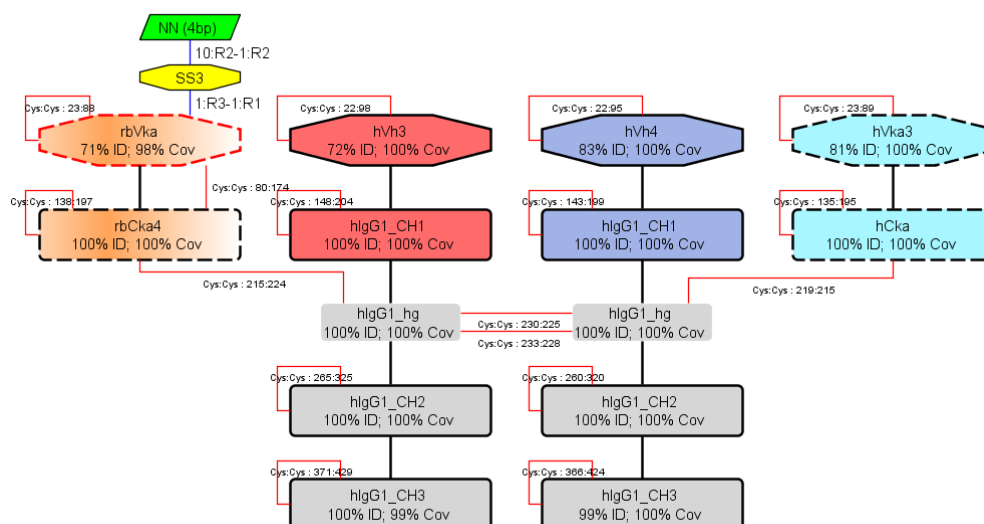


Note that you might also add peptides as well as CHEM payloads.

- Click  to return this modified domain to the *HELM Antibody*



Editor. Click “OK” in the window that indicates that the Cys-Cys bridge between “rbVka” and “rbCka4” had to be removed for technical reasons and thus would have to be re-drawn. Once done this is the final molecule ready for registration:



4 Options, parameters and processes

The following *HELM Antibody Editor* functionality is available under the menus “File”, “Settings” and “Help”:

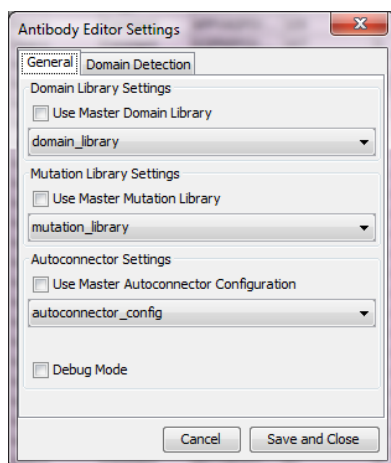
- **File**
 - **Load Sequences** is shown in the example [above](#). Files can be loaded by “Open File” or by drag & drop. File formats can be (multiple) FASTA or Vector NTI (Protein).
 - **Load Antibody From XML ...** loads antibodies from XML files.
 - **Save Antibody To XML ...** saves antibodies to XML files.
- **Settings**
 - **View Domain Library** displays the antibody domain library (s. chapter [5.1](#)). The editing functionality (“Add”) will be delivered in a future release.
 - **Antibody Editor Settings** are detailed below.

4.1 Antibody Editor Settings

In the *HELM Antibody Editor* settings (Main menu > Settings > Antibody Editor Settings) there are 2 tabs:

- The “General” tab provides the selection of the libraries to be used for domain detection, mutation annotation and assembly of modules.
- The “Domain Detection” tab summarizes parameters that influence the detection of domains within an entered peptide chain.

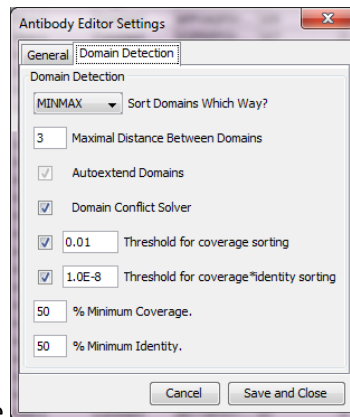
The tab “General” allows to select the libraries. In a company installation the 3 check boxes “Use Master ...” select for a remote database system. As a starter the public version on GitHub contains a SQLite library file in config\config.db3 containing different instances of the 3 libraries. See details in chapter [5](#). In the GitHub version there is only one instance for each library that is set as default:

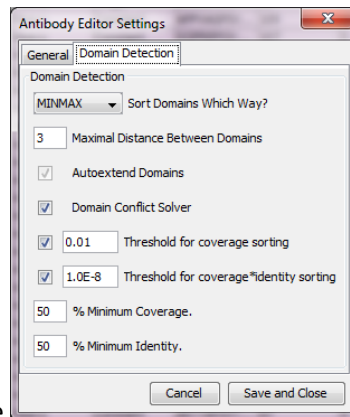


4.1.1 Domain Detection Settings

To understand the settings in this chapter first the domain detection procedure has to be briefly described: The entered peptide chains are chopped into domains using BLAST versus the domain

library. Domain hits are filtered for their %identity and %coverage (s. parameters below), sorted and finally assembled. In the Domain Detection Window the entered chains are listed below each other and the domain hits represented next to each other in rows. These domain hits are sorted depending on different criteria that again depend on the BLAST e-value (given in brackets after the domain name) according to options in the Domain Detection settings (s. below).



The default (and recommended) window looks like  and contains the following options:

- **Domain Detection – How to sort domains?:** The MINMAX algorithm combines the strength of the alternative algorithms MIN (finds domains as small / granular as possible) and MAX (finds domains as large as possible): It maps the filtered and sorted BLAST hits to the entered peptide sequence. Any domain with a lower sorting rank is either also mapped onto the entered peptide sequence as space allows (max. 1 residue overlap accepted to already mapped domains) or otherwise is placed below an already mapped domain and thus assigned to its given position. Thus the best (top sorted) hits determines the position and size of the detected domains.
- **Maximal Distance between Domains:** If BLAST can't annotate stretches between domains these "domains" are labelled "NN (...AAs)". In case the number of such residues is equal or smaller than this parameter this inter-domain stretch is not marked as a domain.
- **Auto-extend Domains:** In case a given (entered) domain has terminal differences to the best matching domain from the domain library, BLAST will fail to return the full alignment. If this option is checked the given domain is extended beyond the BLAST hit until either the length of the library domain is reached or other detected domains would cause an overlap conflict. This is useful to extend especially variable domains to their full size as the given antibodies might well diverge from the consensus sequences in the domain library. (This option is e.g. applied for the domain "rbVka" in the provided example.)
- **Domain Conflict Solver:** In case two domains share one common residue at their ends BLAST might return overlapping domains that are marked by a red border of the conflicting domains in the Domain Detection Window. If the Domain Conflict Solver option is checked the software assigns this one common residue to that domain with the higher priority (i.e. lower domain_priority number, s. chapter 4.2). This option is quite useful to support batch registration of biologicals as it avoids further user interaction in case of conflicting domains.
- **Threshold for coverage sorting:**
 - BLAST hits with an e-value lower than this parameter are significant and sorted for their e-value.

- BLAST hits with an e-value higher than this parameter are often linkers (e.g. 2G3S) and just sorted for their coverage (1st) and e-value (2nd) to select the best hit.
- **Threshold for coverage*identity sorting:**
 - BLAST hits with an e-value lower than this parameter are highly significant (e.g. full domains) and sorted for their coverage*identity. This helps to select the better match from close consensus sequences independent of their length – an attribute that influences the BLAST e-value in a non-controllable manner.
- **% Minimum Coverage:** Filters for BLAST hits equal or above % coverage specified by this parameter. “%Coverage” relates to the extent to which a domain from the domain library is used to annotate a sequence stretch of the input sequence. This parameter shall remove small stretches of domains from the library to be matched to the input sequence.
- **% Minimum Identity:** Filters for BLAST hits equal or above % identity specified by this parameter.

4.1.2 Sorting of hits

As detailed [above](#) 2 parameters influence the sorting of the hits. The following list illustrates the scale from low (significant) to high (not significant) e-values:

Coverage*Identity sorting (for large domains)

...

Threshold for coverage*identity sorting → to be applied only on large domains to focus on them.

Alternatively this value can be set equal or close to “Threshold for coverage sorting” with the effect that large and small domains are treated the same way and thus the entered peptide chain might be chopped into a set of smaller domains instead of one modified larger one. This has to be evaluated for each user group / company independently.

...

e-Value sorting (for small domains)

...

Threshold for coverage sorting

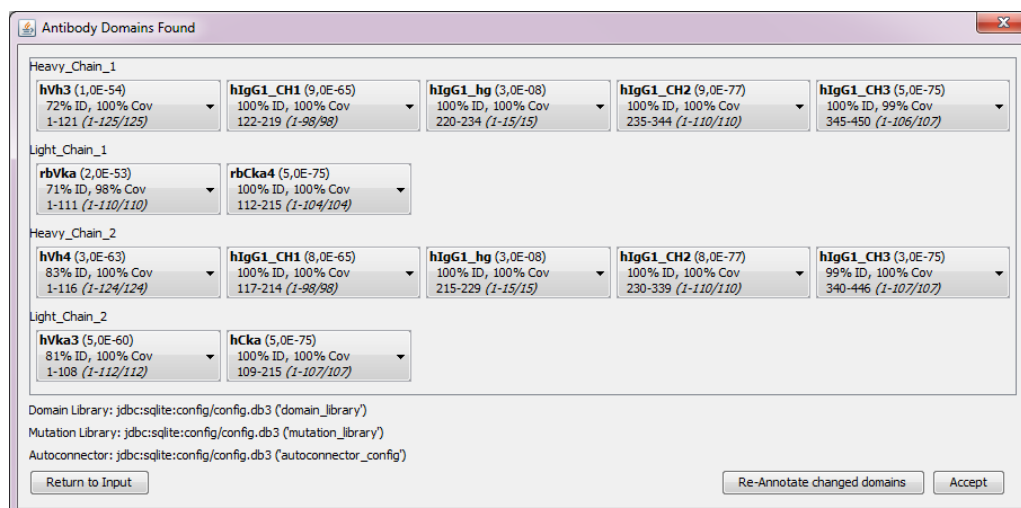
...

Coverage / e-value sorting (for e.g. xGyS linkers)

4.2 Domain Detection Window

Note: s. chapter [4.1.1](#) for details on the domain detection by BLAST.

In the Domain Detection Window (chapter [3.2](#)) the entered chains are listed below each other and the domains are represented next to each other in rows.



In case two domains share one common residue at their ends BLAST might return overlapping domains that are marked by a red border of the conflicting domains in the Domain Detection Window. If the Domain Conflict Solver option is checked the software assigns this one common residue to that domain with the higher priority (i.e. lower domain_priority number, s. chapters 4.1.1 and 5.1).

When clicking on a domain further BLAST hits from the same regions are listed below according to their sorting order with their colors indicating their sorting blocks (s. sorting details in chapter 4.1.2). Any of those domains can be selected to manually override the presented top hit.

Further options are:

- **Assign to the left/right:** assign a complete domain to the neighboring left/right one. Used if a (smaller) domain is obsolete and should be part of another neighboring one.
- **Restrict left/right border to the neighboring one:** adjust left/right domain boundary to the border of the neighboring one such that the overlap is solved. Used in case of overlapping BLAST hits.
- **Edit boundaries:** manually edit the boundaries of a domain. Used to have full control over the domain boundaries.
- **None:** delete domain assignment in case the annotation is wrong. This domain can later be manually annotated in the Antibody Editing Window.

Any “modified” domains are labeled as such to indicate this manual intervention. By clicking “Re-annotate modified domains” these domains are re-BLASTed and re-annotated.

The button “Return to Input” returns to the Sequence Loader and “Accept” transfers the domains to the Antibody Editing Window.

4.3 Antibody Editing Window

Note: The legend (s. chapter 3.3), the Details Box (s. chapter 3.4), manual connection of domains (s. chapter 3.5) and adding oligos, peptides and/or chemicals to domains (s. chapter 3.6) are explained in the example described in chapter 3.

From the Domain Detection Window the domains are analyzed for mutations and annotated accordingly (s. details chapter [5.2](#)).

4.3.1 Cys-Cys bond creation

Next the antibody is assembled with as many Cys-Cys bonds automatically created as possible according to the following steps:

- Create intra-domain Cys-Cys bonds as defined by the Domain library
- Create hinge to hinge Cys-Cys bonds (hard coded)
 - Only if same subtype, same number of Cys, sequences might differ
- Create hinge to constant light chains Cys-Cys bonds (hard coded)
 - Only if HC or LC or both are fully identical (assuming 2xHC + 2xLC)
- Create mutation defined inter-chain Cys-Cys bonds as defined by the Mutation library
 - E.g. knob into hole
- Create intra-chain / inter-chain Cys-Cys bonds as defined by the Autoconnector rule set
 - E.g. scFv, scFab

4.3.2 Canvas options

All bonds (thin red lines) can be selected, the position of their anchor points be adjusted and even deleted.

Domains can be right-clicked providing the following options:

- **Edit Domain:** s. chapter [3.6](#) “Extend the example and add an oligo to a domain”.
- **Annotate Domain:** re-annotate domain (by BLAST of just this domain)

A right-click on the canvas opens a menu with the following options:

- **Back to Domain Recognition:** opens the Domain Detection Window. This can be used to restore the automatically generated antibody assembly if needed.
- **Reset Layout:** resets layout to yFiles defaults
- **Reset Cystein Bridges (+Layout):** resets layout and removes all intra-domain Cys-Cys bonds as defined by the domain library
- **Show HELM:** shows the always up-to-date HELM code for the complete molecule on the canvas

5 Libraries: Use, syntax and editing

The three libraries described below are provided as SQLite file in config\config.db3

We can recommend <http://sourceforge.net/projects/sqliteman/> as easy and standalone viewer / editor for SQLite database files that does not has to be installed under Windows. This tool allows you to extend the library for e.g. linkers, novel Fc mutations , and rules that you use to detect special components or constructs.

5.1 Domain library

The domain library is used to identify and annotate the domains as building blocks of peptide chains. All additional domains, linkers, peptides, conjugate proteins that are used should be entered in this library.

The domain library (only 1st 4 lines) looks like

	domain_pk	domain_name	domain_short_name	species	humanness	chain_type	domain_type	domain_sequence	domain_cys_pattern	_co	ain	domain_usage	is_active
1	1	IGHG1_CH1_HUMAN	hIgG1_CH1	human	human	heavy	constant	ASTKGPSVFPLAPSSKSTSGGTA...	1-2			GENERAL	1
2	2	IGHG1_HINGE_HUMAN	hIgG1_hg	human	human	heavy	hinge	EPKSCDKTHTCPPCP	1-LC,2-H,3-H			GENERAL	1
3	3	IGHG1_CH2_HUMAN	hIgG1_CH2	human	human	heavy	constant	APELLGGPSVFLFPPKPKDTLMISR...	1-2			GENERAL	1
4	4	IGHG1_CH3_HUMAN	hIgG1_CH3	human	human	heavy	constant	GQPREPQVYTLPPSRDELTKNQV...	1-2			GENERAL	1

...

and thus contains the following columns:

- **domain_pk**: used as internal primary key.
- **domain_name**: full description of the domain. Name that is referred to in the mutation library.
- **domain_short_name**: name used in BLAST and in the GUI.
- **species**: human, mouse, rat, rabbit or empty / “-” for e.g. linkers
- **humanness**: human, humanizable if not human yet, non-human if not humanizable as e.g. linkers
- **chain_type**: heavy, kappa, lambda, empty / “-”
- **domain_type**: variable, hinge, constant, empty / “-”
- **domain_sequence**: amino acid sequence
- **domain_cys_pattern**: the pattern of Cys residues as ordinal numbers that pair within this domain, heavy chain domains might pair with light chains “LC” and light domains might pair with hinge “H”. Multiple bonds are separated by a comma.
- **domain_comment**: an arbitrary comment for your convenience.
- **domain_sawi**: not used
- **domain_usage**: only “GENERAL” entries will be used by BLAST.
- **is_active**: only entries with is_active=1 will be used by BLAST. 0 means ‘inactive’.
- **domain_priority**: In case of conflicting domains (s. chapter [4.2](#)) the residue shared by 2 domains is assigned to the domain with the higher priority (i.e. lower domain_priority number).

The domain library contains the variable and constant antibody domains for the species human, mouse, rat, and rabbit. The **variable domains** (heavy, kappa, lambda only for human) are provided as consensus sequences that have been generated from according IMGT entries with CDR3s

represented by X. For human heavy and kappa chains the general consensus has been additionally split into gene family specific consensus sequences to allow a better characterization of such domains and to avoid a mistyping of more rare human domains as non-human in case e.g. the mouse consensus would be closer to a rare human variable region than the human main consensus itself.

The **constant regions** are based on Swissprot (that by itself has received the Ig constant regions in 2008 from IMGT) and IMGT and are chopped into domains like CH1, hg (hinge), CH2, and CH3.

- Lefranc, M-P.
Immunoglobulins: 25 years of Immunoinformatics and IMGT-ONTOLOGY.
Biomolecules. 2014, 4, 1102-1139; doi:10.3390/biom4041102 Open access, <http://www.mdpi.com/2218-273X/4/4/1102>
- Lefranc M-P, Giudicelli V, Duroux P, Jabado-Michaloud J, Folch G, Aouinti S, Carillon E, Duvergey H, Houles A, Paysan-Lafosse T, Hadi-Saljoqi S, Sasorith S, Lefranc G, Kossida S. IMGT®, the international ImMunoGeneTics information system® 25 years on. Nucleic Acids Res. 2014 Nov 5. pii: gku1056. [Epub ahead of print] [PMID: 25378316](https://pubmed.ncbi.nlm.nih.gov/25378316/)
Free Article: <http://nar.oxfordjournals.org/content/early/2014/11/05/nar.gku1056.abstract>
- The UniProt Consortium
Activities at the Universal Protein Resource (UniProt)
[Nucleic Acids Res. 42: D191-D198 \(2014\).](https://doi.org/10.1093/nar/gku1056)

5.2 Mutation library

The mutation library contains all deviations from the domains available in the domain library. Enter variations as e.g. mutations and allotypes of such domains to be identified. Mutations are grouped into sets (identical “mutation_name”) that will only be recognized as such if all the listed mutations of a Mutation group are found - otherwise the mutations are listed as unknown mutations.

The mutation library looks like

mutation_pk	domain_name	mutation_name	mutation_short_name	mutation_num_var	mutation_position_in_domain	mutation_kabat_index_eu	mutation_original_aa	mutation_new_aa	mutation_visible	mutation_connection_group
1	IGHG1_CH3_HUMAN	G1m17, nG1m1	G	3	16	356	D	E	0	
2	IGHG1_CH1_HUMAN	G1m17, nG1m1	G	3	97	214	K	K	0	
3	IGHG1_CH3_HUMAN	G1m17, nG1m1	G	3	18	358	L	M	0	
4	IGHG1_CH1_HUMAN	G1m3, nG1m1	A	3	97	214	K	R	0	
5	IGHG1_CH3_HUMAN	G1m3, nG1m1	A	3	16	356	D	E	0	
6	IGHG1_CH3_HUMAN	G1m3, nG1m1	A	3	18	358	L	M	0	
7	IGHG1_CH3_HUMAN	Hole	H	3	28	368	L	A	1	
8	IGHG1_CH3_HUMAN	Hole	H	3	67	407	Y	V	1	
9	IGHG1_CH3_HUMAN	Hole	H	3	26	366	T	S	1	
10	IGHG1_CH3_HUMAN	HoleC	H	4	67	407	Y	V	1	
11	IGHG1_CH3_HUMAN	HoleC	H	4	28	368	L	A	1	
12	IGHG1_CH3_HUMAN	HoleC	H	4	26	366	T	S	1	
13	IGHG1_CH3_HUMAN	HoleC	H	4	9	349	Y	C	1	KH
14	IGHG1_CH3_HUMAN	Knob	K	1	26	366	T	W	1	
15	IGHG1_CH3_HUMAN	KnobC	K	2	14	354	S	C	1	KH
16	IGHG1_CH3_HUMAN	KnobC	K	2	26	366	T	W	1	
17	IGHG1_CH2_HUMAN	LALA	LA	2	5	235	L	A	1	
18	IGHG1_CH2_HUMAN	LALA	LA	2	4	234	L	A	1	
19	IGHG4_HINGE_HUMAN	SPL	S	2	10	228	S	P	0	
20	IGHG4_HINGE_HUMAN	SPL	S	2	5	235	L	E	0	
21	IGHG1_CH2_HUMAN	3A	3A	3	23	253	I	A	0	
22	IGHG1_CH2_HUMAN	3A	3A	3	80	310	H	A	0	
23	IGHG1_CH3_HUMAN	3A	3A	3	95	435	H	A	0	

and thus contains the following columns:

- **mutation_pk**: used as internal primary key.

- **domain_name**: name of the domain as listed in the domain library under “domain_name”
- **mutation_name**: full name of the mutation (group)
- **mutation_short_name / mutation_visible**: name that is shown in brackets within the domain as shown on the canvas if “mutation_visible” is set to 1 else the mutation is not shown there but just in the details box to the right
- **mutation_num_var**: number of mutations (‘variation’) belonging to one mutation group
- **mutation_position_in_domain**: absolute position of the mutation within the domain
- **mutation_kabat_index_eu**: position of the mutation according to Kabat EU Index (only added for documentation; not used in the HELM Antibody Editor yet)
- **mutation_original_aa / mutation_new_aa**: original and mutated amino acid
- **mutation_connection_group**: identical strings (not essentially numbers) indicate Cys residues to be connected if both Cys are found, and only if all mutations of the corresponding mutation group are found.

5.3 Autoconnector rule set

Rules recognize modules like e.g. scFv and scFab and define the Cys bond pattern between domains of such modules or even beyond those modules. They are provided as text in a separate table (with line numbers to define their sequence) that look like:

```
#standard pairing
*-VH-CH<' %CH1%'>-H-CH<' %CH2%'>-CH<' %CH3%'>-* , *-VL-CL-*=>3&7<1>

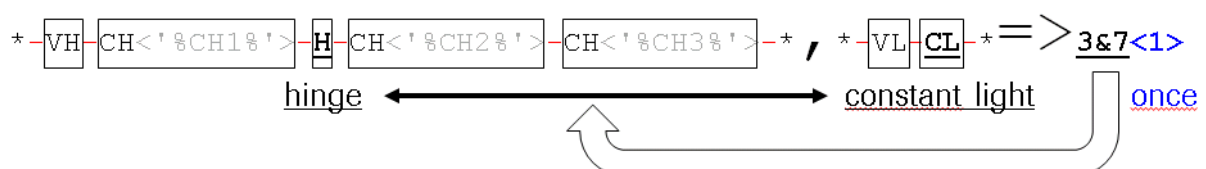
#scFv
VH-X<' ?G?S'>-VL-*=>1&3<n>
*-VH-X<' ?G?S'>-VL=>1&3<n>
VL-X<' ?G?S'>-VH-*=>1&3<n>
*-VL-X<' ?G?S'>-VH=>1&3<n>

#scFab connect V
VH-CH<' %CH1%'>-X-VL-CL-X<' ?G?S'>-*=>1&4<n>
*-X<' ?G?S'>-VH-CH<' %CH1%'>-X-VL-CL=>2&5<n>
VL-CL-X<' ?G?S'>-VH-CH<' %CH1%'>-X-*=>1&4<n>
*-VL-CL-X<' ?G?S'>-VH-CH<' %CH1%'>-X=>1&4<n>

#scFab connect C
VH-CH<' %CH1%'>-X-VL-CL-X<' ?G?S'>-*=>3&5<n>
*-X<' ?G?S'>-VH-CH<' %CH1%'>-X-VL-CL=>4&6<n>
VL-CL-X<' ?G?S'>-VH-CH<' %CH1%'>-X-*=>2&6<n>
*-VL-CL-X<' ?G?S'>-VH-CH<' %CH1%'>-X=>2&6<n>
```

The connector rule for the standard pairing exemplifies the syntax as follows:

#standard pairing



- Rules are processed bottom down.
- **#** precedes comment lines that are not processed.
- The rule notation separates domains by **|**.
- The domain acronyms refer to the domain library as follows:

Acronym	Chain	Domain
VH	heavy	variable
H	heavy	hinge
CH	heavy	constant
VL	kappa or lambda	variable
CL	kappa or lambda	constant
X	any	any

- Domains can be further specified by a text string to occur in the short name, set in **<>** brackets after the domain acronym, framed by **|**.
 - **%** is used as wildcard for 0-n characters.
 - **?** is used as wildcard for a single character.
- **X** is used as wildcard for any domain but usually applies e.g. to linkers.
- N- or C-terminal ends might be extended by ***** to indicate that the chain is allowed to be extended by e.g. modules like scFv but these ***** domains do not count when numbering the domains.
- Different chains are separated by **|**. Domains are continuously numbered across chains – just from left to right.
- The connection rule is given after the **=>** symbol: The ordinal numbers of the domains to be connected by a Cys-Cys bridge are concatenated by **&**.
 - Per default it is assumed that such domains have only 1 free Cys left. Otherwise the ordinal number of the free Cys within a domain has to be specified after the domain number and separated by a **#**:
E.g. 3#2 specifies the 2nd free Cys of the 3rd domain.
- At the end of the syntax **<1>** indicates that the rule applies only if the given pattern is found once, otherwise the rule is not applied. A **<n>** specifies the pattern to be processed as often as it occurs.

The sequence of the rules define which rule will be used if 2 or more rules are matching. In this case, the first rule (lower sequence number) wins.

6 Contact

- Please contact Stefan.Klostermann@Roche.com in case of general and conceptional questions and feedback. Please contact Bernhard Schirm schirm@quattro-research.com for technical questions.