Biochemfusion

Protein Derivatives Notation (DerNot)

Specification

version 1.1

2014-01-19

Table of Contents

Changes since version 1.0

Inline definitions of named modified terminal and residue structures are now supported.	See the "Properties" section.

Introduction

The Biochemfusion protein **Der**ivatives **Not**ation (DerNot) specification defines a format for protein editing instructions - DerNot expressions. DerNot expressions can be used to create protein derivatives/analogs from existing protein entries or express the difference between two existing protein entries.

The notation is loosely inspired by the IUPAC trivial peptide naming[1] but has been adapted so it can be expressed and transmitted in plain text. This facilitates easy exchange via electronic documents, e.g. in e-mail messages or in spreadsheet documents.

Residues, sequences, and terminals are expressed by the same notational means as in the Biochemfusion PLN notation.[2] Throughout this document PLN notation will also be used to show example protein entries.

General structure of DerNot expressions

A DerNot expression consists of 1 to 6 tokens separated by whitespace.

```
<prepends> <insertions> <substitutions> <deletions> <reference entry> <appends>
```

Only the `<reference entry>` token is required. All other tokens are optional.

The `<appends>` token may only occur *after* the `<reference entry>` token. All other tokens must occur *before* the `<reference entry>` token. Each token type may occur only once.

A DerNot expression may be followed by any number of property tokens. A property token defines a key-value pair; more details in the Properties section.

Tokens may not contain whitespace. An exception is the `<reference entry>` token which may contain a name with whitespace if properly quoted. Other circumstances may also require the reference entry name to be quoted - see the following section.

Implementations must discard linefeeds when reading DerNot expressions. Otherwise e-mail or document transmission of DerNot expressions would be likely to corrupt data due to line wrapping.

The following sections will detail each token's format and intended usage.

Reference entry

The `<reference entry>` token identifies a reference protein entry by name. The name may be any sequence of characters with the exception of linefeed characters which implementations must discard when reading the token.

Instead of identifying the reference protein entry by its full name the asterisk character '*' may be used instead.

In a few circumstances the name must be enclosed in double quotes ("). This is when the name:

- Contains whitespace characters.
- Contains double quotes.
  - Each double quote inside a name must be represented by two consecutive double quotes.
- Ends with a closing parenthesis.
  - A closing parenthesis at the end would make the name look like a `<substitutions>` token.
- Starts or ends with a hyphen.
  - Hyphens at either end would make the name look either like an `<appends>` token or a `<prepends>` token without chain locant.
- Starts with 'endo-' or 'des-'.
  - You should not be able to confuse the name with an `<insertions>` token or a `<deletions>` token.

Examples:

Name	Correctly quoted
Iupac_Ubiquitin	<no quoting required>
Human insulin	"Human insulin"
Very "important" protein	"Very ""important"" protein"
GLP(6-36)	"GLP(6-36)"
-minus_protein	"-minus_protein"

How the named reference protein entry is found and how the edit operations defined by the rest of the tokens are applied to the reference entry to yield the derived protein entry is implementation-dependent.

Chain locants

A chain locant identifies a particular chain within the reference protein. The format of a chain locant is a chain identifier within a pair of parentheses.

The chain identifier is a single uppercase alphabetical character where 'A' denotes the first chain of the reference entry, 'B' the second chain and so forth.

Residue locants

A residue locant identifies a specific residue or residue range within a protein chain. All residue locants reference residues of the reference entry *before* any edit operations are applied. Residue locants are always enclosed in a set of parentheses.

A residue locant is a chain identifier followed by an integer residue number.

Residue numbers are always relative to a chain and 1 identifes the first residue within the chain.

A residue locant range is a chain identifier followed by a pair of integers where the first integer must be less than the second integer. The integer pairs are separated by a hyphen '-'.

If the reference entry is a single-chain protein the chain identifier is optional.

Substitutions

The `<substitutions>` token will change existing residues of the reference entry into the ones listed in the token. The token may contain single residues followed by residue locants or residue sequences followed by residue locant ranges.

DerNot expression	Reference entry	Derivative
`A(2)G(4)KHR(6-8) single`	`H-QWEKSDATY-OH name=single`	`H-QAEGSKHRY-OH`

`[Gla](A5)[Sar](B2) dbl`	`H-ASDFE-OH.H-QGE-OH name=dbl`	`H-ASDF[Gla]-OH.`
		`H-Q[Sar]E-OH`

Insertions

`<insertions>` lists residue(s) that should be inserted at a given residue locant position. An `<insertions>` token starts with 'endo-' followed by a residue list.

Insertion will take place between the residue indicated by the residue locant and the residue immediately following the residue locant ("the position between the qth and the (q+1)th residue..." as specified in [1]).

DerNot expression	Reference entry	Derivative
`endo-K(3) single`	`H-QWETS-OH name=single`	`H-QWEKTS-OH`
`endo-A(2)GKH(3) single`	`H-QWETS-OH name=single`	`H-QWAEGKHTS-OH`

Insertion residue locants referencing the last residue of a chain are not allowed. If a peptide is a 5-residue single chain you can insert residues at (A4) but not at (A5). The latter corresponds to a C-terminal chain extension and must instead be specified as part of an `<appends>` token.

Deletions

`<deletions>` lists residues by locant that should be deleted from the reference entry. A `<deletions>` token starts with 'des-' followed by residues (optional) and their residue locant(s).

The following expression will delete the Ile residue in position A2 and the Lys-Thr sequence starting in position B29 from a human insulin entry.

```
        des-I(A2)KT(B29-30) Insulin
```

If listed residues do not perfectly match the residues present in the reference entry the implementation must raise an error.

Residues may be omitted for brevity, e.g. the following deletion token is identical to the above.

```
        des-(A2)(B29-30) Insulin
```

If residue locants within a `<substitutions>` token overlap residue locants within a `<deletions>` token the implementation must raise an error.

Residue locants in an `<insertions>` token do not necessarily conflict with residue locants within a `<deletions>` token. If the `<insertions>` token is processed before the `<deletions>` token conflicts will not arise, e.g. applying "`endo-AC(A2) des-W(A2) *`" to "`H-QWE-OH`" would simply produce "`H-QACE-OH`". Implementations may choose to allow this kind of overlap or treat it as an error.

Prepends

N-terminal chain extensions or terminal modifications are defined in the `<prepends>` token.

A chain extension starts with an N-terminal specification followed by a hyphen and then by a sequence of residues followed by a hyphen. Both the N-terminal specification and the sequence are optional, but at least one of them must be specified.

If the reference entry contains multiple chains each chain extension must be followed by a chain locant. For a single-chain reference entry the chain locant is optional. Multiple chain extensions must be delimited by commas. See examples below.

DerNot expression	Reference entry	Derivative/Analog
-AG[Gla]T- single	H-QWE-OH name=single	H-AG[Gla]TQWE-OH
[biotin]- single	H-QWE-OH name=single	[biotin]-QWE-OH
[biotin]-AGKT- single	H-QWE-OH name=single	[biotin]-AGKTQWE-OH
H- single	[biotin]-QWE-OH name=single	H-QWE-OH
-AGKT-(A),-HR-(B) dbl	H-QWE-OH.H-ASD-OH name=dbl	H-AGKTQWE-OH.H-HRASD-OH
- single	H-QWE-OH name=single	<not allowed>

Appends

C-terminal chain extensions or terminal modifications are defined in the <appends> token.

Chain extensions are structured exactly like the corresponding <prepends> chain extensions only with hyphens, sequence, and terminal in reverse order.

When the reference entry contains multiple chains a chain locant is required after each chain extension and extensions must be separated by commas. For single chain reference entries the chain locant is optional.

DerNot expression	Reference entry	Derivative/Analog
single -GK-	H-QWE-OH name=single	H-QWEGK-OH
single -[NH2]	H-QWE-OH name=single	H-QWE-[NH2]
single -AS-[NH2]	H-QWE-OH name=single	H-QWEAS-[NH2]
single -OH	H-QWE-[NH2] name=single	H-QWE-OH
dbl -[NH2](A),-HR-(B)	H-QWE-OH.H-ASD-OH name=dbl	H-QWE-[NH2].H-ASDHR-OH
single -	H-QWE-OH name=single	<not allowed>

Properties

A property is a key-value pair delimited by an equals sign '='. Currently the only properties allowed are inline modification definitions.

Inline modification definitions are necessary to make sure that all DerNot expressions can be applied successfully. If a DerNot expression is generated by comparing two proteins that contain inline modifications, the resulting DerNot expression may reference those inline modifications by name and in that case the inline modifications need to be preserved in the DerNot expression.

The inline modifications in DerNot expressions are written in the same format as inline-mod properties in PLN. [2]

Residues, terminals, and modification names

Residue codes and names must adhere to the PLN format rules. In general:

- Unmodified residues are defined via their one-letter codes.
- Modified residue and terminal names must be enclosed in square brackets.
- D-forms are indicated by a '{d}' prefix.

Consult the PLN specification sections "Residues", "Non-standard residues", "D-form residues", and "Modification names" for the exact details.[2] ***Note*** that DerNot expressions do not yet allow the use of three-letter residue codes.

References

(1) IUPAC - "Names and symbols for derivatives of named peptides"
http://www.chem.qmul.ac.uk/iupac/AminoAcid/AA22.html#AA22

(2) Biochemfusion PLN 1.3 specification
http://www.biochemfusion.com/doc/Biochemfusion_PLN_1.3_spec.pdf