Design and practical usage of web biological databases for the annotation and classification of proteins

Submitted for the Degree of Doctor of Philosophy in Biotechnology by
Toni Hermoso Pulido

The work was supervised by **Dr. Enric Querol Murillo**
and **Dr. Francesc Xavier Avilés i Puigvert** of the
Institut de Biotecnologia i de Biomedicina - Universitat Autònoma de Barcelona

Toni Hermoso Pulido

Enric Querol Murillo   Francesc Xavier Avilés
i Puigvert

Bellaterra (Cerdanyola del Vallès), May 2015

If you torture the data enough, nature will always confess.

Ronald Coase. "How should economists choose?" Warren Nutter Lecture, 1981.

Reprinted in *Essays on Economics and Economists* (1994) p. 27.

SUMMARY

In the so-called Information society, data represent a key structural parts of knowledge generation. Likewise, present-day Bioinformatics ultimately relies on the proper management and processing of data originated from both traditional and high-throughput biological analyses.

As a primary aim of this thesis, different practical approaches for the handling of raw protein sequence data and their applied resulting Bioinformatics analyses are compiled. Nonetheless, as a preliminary step, pre-existing Bioinformatics tools and algorithms are adapted to the characteristics of current informational paradigm, basically revolving around the World Wide Web. After a proper adaptation of those applications (in this document: ProtLoc, TransMem, TranScout and Bypass), it becomes possible to face the massive processing of protein data originating from the last genome sequencing projects.

The outcomes of these analyses are made available for the world-wide scientific community in the form of user-friendly and fully accessible web-based biological databases. As examples, two cases are presented: TrSDB, a compendium of well-known and putative transcription factors from different model organisms, and ArchDB, a structural web database of protein loops.
As a further step, starting from Bypass program —a fuzzy-logic based tool for the re-annotation and evaluation of protein homology search alignments—, a complete annotation and sequence management framework is deployed. As as strong point, the system is also flexible and modular enough for allowing the input of different data (e. g. Gene Ontology) or cross-communicate with other future and existing applications.

Parallely to this, and also taking advantage of the explosion of raw sequence data and database curation, a Bioinformatics characterization of a new metallacarboxypeptidase family is introduced. By using computational means, it was possible to present a first hypotheses for the enzymatic activity and phylogenetical history of these group of proteins. Notably, their actual identification may represent an enlightening focus for dealing with biological processes such as malaria or neurodegenerative disorders, where these molecules are intimately linked.

RESUM

En l'anomenada societat de la informació, les dades representen unes parts estructurals clau en la generació del coneixement. De la mateixa manera, la Bioinformàtica depèn en darrer terme d'una adequada gestió i processament de les dades que s'originen de les anàlisis tant tradicionals com d'alt rendiment.

Com a objectiu principal d'aquesta tesi, s'han compilat diferents aproximacions pràctiques per a la manipulació de les dades de seqüenciació de proteïnes i les anàlisis resultants que s'hi apliquen. En tot cas, com a pas preliminar, eines i algorismes bioinformàtics preexistents s'han adaptat abans al paradigma de la informació actual, bàsicament centrat en el World Wide Web. Després d'una pertinent adaptació d'aquestes aplicacions (en aquest document: ProtLoc, TransMem, TranScout i Bypass), esdevé possible encarar el processament massiu de dades proteiques que s'han originat dels darrers projectes de seqüenciació de genomes.

Els resultats d'aquestes anàlisis es fan disponibles per a la comunitat científica d'arreu del món mitjançant bases de dades biològiques basades en el web, de ple accés i d'ús amigable. Com exemples, es presenten dos casos: TrSDB, un compendi de factors de transcripció coneguts i putatius de diferents organismes models, i ArchDB, una base de dades web estructural de llaços de proteïnes.
Com a pas posterior, a partir del programa Bypass —una eina basada en lògica difusa per a la reanotació i avaluació d'alineaments proteics obtinguts amb cerca per homologia— s'implementa un entorn complet d'anotació i gestió de seqüències. Com a punt fort, el sistema també és suficientment flexible i modular per a permetre l'entrada de diferents tipus de dades (p. ex., Gene Ontology) o comunicar-se amb altres aplicacions ja existents i potencialment futures.

De forma paral·lela, i aprofitant l'explosió de dades crues de seqüenciació i la curació de les bases de dades, es presenta una caracterització bioinformàtica d'una nova família de metal·locarboxipeptidases. Mitjançant aproximacions computacionals, va ser possible plantejar unes primeres hipòtesis per a l'activitat enzimàtica i la història filogenètica d'aquest grup de proteïnes. Notablement, llur identificació pot representar un enfocament esperançador per a tractar processos biològics com ara la malària o desordres neuronals, on aquestes molècules s'hi troben implicades estretament.

RESUMEN

En la llamada sociedad de la información, los datos representan unas partes estructurales clave en la generación del conocimiento. Del mismo modo, la Bioinformática depende en último término de una adecuada gestión y procesado de los datos que se originan de los análisis tanto tradicionales como de alto rendimiento.

Como objetivo principal de esta tesis, se han compilado diferentes aproximaciones prácticas para la manipulación de los datos de secuenciación de proteínas y los análisis resultantes que se aplican. En todo caso, como paso preliminar, herramientas y algoritmos bioinformáticos preexistentes se han adaptado antes al paradigma de la información actual, básicamente centrado en el World Wide Web. Después de una pertinente adaptación de estas aplicaciones (en este documento: ProtLoc, TransMem, TranScout y Bypass), se hace posible encarar el procesamiento masivo de datos proteicos que se han originado de los últimos proyectos de secuenciación de genomas.

Los resultados de estos análisis se hacen disponibles para la comunidad científica mundial mediante bases de datos biológicas basadas en el web, de pleno acceso y de uso amigable. Como ejemplos, se presentan dos casos: TrSDB, un compendio de factores de transcripción conocidos y putativos de diferentes organismos modelos, y ArchDB, una base de datos web estructural de lazos de proteínas.
Como paso posterior, a partir del programa Bypass —una herramienta basada en lógica difusa para la reanotación y evaluación de alineamientos proteicos obtenidos a partir de búsqueda por homología— se implementa un entorno completo de anotación y gestión de secuencias. Como punto fuerte, el sistema también es suficientemente flexible y modular para permitir la entrada de diferentes tipos de datos (p. ej., Gene Ontology) o comunicarse con otras aplicaciones ya existentes y potencialmente futuras.

De forma paralela, y aprovechando la explosión de datos crudos de secuenciación y la curación de las bases de datos, se presenta una caracterización bioinformática de una nueva familia de metalocarboxipeptidasas. Mediante aproximaciones computacionales, fue posible plantear unas primeras hipótesis para la actividad enzimática y la historia filogenética de este grupo de proteínas. Notablemente, su identificación puede representar un enfoque esperanzador para tratar procesos biológicos como por ejemplo la malaria o desórdenes neuronales, donde estas moléculas se encuentran implicadas estrechamente.

ACKNOWLEDGEMENTS

The whole process of writing a thesis is an excellent moment for getting fully acquainted of how much your final work depended on other people.

As you proceed adding references to the body of the text, you can notice how much you owe to so many individuals about that piece of knowledge or technology that you may have used at some point. This can be just a punctual contribution, but very often it can also determine the successful outcome of your project.

Bioinformatics, as any other science or technology discipline, is about collaboration in the end. And this liquid nucleation of human partnership on the generation of knowledge is something that flows along History. When you keep your own story in your mind as you write a document like a thesis, involuntarily you are also assuming the role of an inexpert historian.

As any of the people that helped to fill up the bibliography of this document, there are many conditionings and personal experiences, sometimes mutually shared, that allowed them to attain what I could benefit and tried to write down here.

This section is a great chance for acknowledging all those people who helped and impacted this author, myself, during the personal trajectory (arguably more or less errand) behind the presented works. Some of these people may not fit into a bibliographic citation, but sometimes I can tell I may owe to them most of what I finally could accomplish.

First of all, I want to thank the whole people that made up *Institut de Biotecnologia i Biomedicina* before my own enrollment there until today. Looking into perspective, and counting all current and former members, I can safely say how important the contribution of this center has been to the development of Bioinformatics in Catalonia.

At a more mundane level, I must say it is the place I've enjoyed the best workplace Christmas parties so far. These celebration parties, named 'Fondues' after their main served course, are a heritage that deserves a place in the future of this institution. I would even dare to say that they should be a cultural example for any valuable workplace Christmas celebration anywhere.

At a closer level, I am really thankful to all colleagues that has formed part of the IBB Bioinformatics, Proteomics and Molecular Biology labs with whom I shared different experiences, sometimes simply academical or professional, but normally also personal at different degrees of depth, since my arrival there many years ago.

You can hardly do anything profitable in your daily chores if you do not have the chance to be surrounded by excellent and friendly folks. Just for listing a few —and I deeply regret if I miss anyone I may notice at a later reading—, I can remember Dani Aguilar, Isaac Amela, Ricard Bonet, Silvia Bromsons, Alicia Broto, Juan Cedano, Jordi Espadaler, Narcís Fernandez-

Fuentes, Mario Ferrer, Antonio Gómez, Luis Gonzalez, Mario Huerta, Maria Lluch, Inés Lopez-Torrejón, Montse Morell, Baldo Oliva, Raquel Planell, Mónica Rodriguez, Sebastian Tanco or Sebastian Trejo.

However, if I had to highlight someone both at the professional and at the personal level, I would not hesitate to mention Oscar Conchillo, *aka* grumpy *Txino*, to whom anyone who passes by the Bioinformatics Lab in IBB owes so much in the end.

Moreover, I cannot forget the hospitality of my hosts during my predoctoral stays abroad, specially when I see myself as such a young and tender guy at that time. Concretely, from my visits to the USA, I want to emphasize my gratitude to Andrej Sali, Marc-Martí Renom and Lloyd Fricker, who welcomed and hosted me at their own groups in a sincere and uninterested manner. And comparably, also to Rafael G. Sevilla and José Bautista, who made me feel in Madrid as if I were at home.

On the other hand, I do not want to lose the chance to thank all my current and former colleagues at the Centre for Genomic Regulation (CRG), specially at the Bioinformatics Unit, where I work at the time of writing. Had I not continued keeping a link with Bioinformatics in my daily life during these last years, it would have been very difficult to arrive at this very moment.

Besides, I must also acknowledge the inheritance of my participation in different communities and associations drinking from the “Free and Open Source principles” (e. g., Mozilla, Wikimedia or Softcatalà). Despite this actual involvement may have eluded me from a full focus on aspects such as this thesis, at the same time, I feel privileged for all what I also managed to learn from that participation. Indeed, I am indebted to these collaborations for what I strove to transfer to my technical and scientific practices and projects.

Arriving at the end of this section, I cannot dismiss from my mind my thesis directors. I cannot avoid to wonder how unusual might I be in their list of mentored PhD students. If there might be any kind of mystical parental connection in this director-candidate relationship, I hope they can realize some kind of pride from what they helped me to achieve. Particularly, I want to thank F. Xavier Avilés for offering me the opportunity to go abroad, get a close approach with fellow leading scientists and get me engaged into exciting projects. Likewise, I want to show my higher gratitude to Enric Querol, for his scientific inspiration and specially for his cheerful and continuous support at these latter stages of writing this thesis; without his encouragement these last years this document may not ever have existed.

And, of course, I am deeply indebted to my parents, Isabel Pulido and Antoni Hermoso. Only when you arrive at a stage like mine right now in front of a document like this, you can fully comprehend those homages that can be found when browsing the first pages of any thesis. Parents devote many efforts and go through many worries and hardships for their children, and mine are not an exception. And, as a humble origin family, I feel strongly grateful for their sacrifices on me.

And, somehow, my family, in a wide nurturing aspect, has not only been my strictly biological one, but also our whole society (bodified or, who knows, maybe even caged, into diverse administrations and institutions). Without this other trusting collective family, I would not have ended up writing any of these lines either.

Anyway, I am deeply concerned that these “kinship” society bonds may be dramatically severed in the upcoming years for other people not so different from myself some years ago. This renders me a profound sense of moral responsibility so in the future worthy bibliography can continue emerging from diverse-enough personal scenarios.

Last but not least, I cannot forget in these acknowledgements my partner, Alina Mierluș, with whom I share not only an intimate companionship, but also the opportunity to be daily engaged in an intellectual mutual motivation. She is the one who may have endured more directly this kind of academical penitence process and the one who is more likely to feel more relieved after me from its formal conclusion.

Table of Contents

CHAPTER 1 - INTRODUCTION

Bioinformatics is a broad usage term, which we might simplistically ascribe to the theoretical and practical disciplines that care about biological data. By definition, this is necessarily a multidisciplinary approach at the intersection of both biological and non-biological sciences and engineerings (e. g. statistics or computer sciences). Being such as wide definition concept, it is no surprise that there is a non-fully ended discussion about what Bioinformatics is compared to other terminological coinages, such as Computational Biology (Altman, 2009; Homolog.us, 2012).

Putting apart any further ontological or sociological discussion, it can be safely said this thesis is centered on Bioinformatics. It encompasses different research and technical implementations, mostly focused in data management aspects, spanning several years since the start of some of its comprising works.

Looking into perspective, many of the central ideas and underlying principles and algorithms used in the presented projects, and within Protein Bioinformatics in general, can still be considered essentially valid at the time of writing. Nonetheless, how these are being put into practice is something that has been changing substantially year after year.

It can be argued that one of the main key changes has been the progressive increase of the amount of data researchers can access and how bioinformaticians can handle it without losing sight of what is really significant in the end. Being able to answer relevant questions or discover not necessarily explicit relationships without being overwhelmed by a flood of apparently meaningless data is one of the challenges of present-day science.

Bioinformatics, as a branch of knowledge, has expanded and matured fostered by the computer science, and especially, the so called Internet Revolution (Okin, 2005). Because of this highly-advancing and accompanying milieu, the field, also as a practice, has evolved from a quasi-artistic manual curation of abstract biological entities (as biological sequences are in the end) — where computers were often little more than elongation of dashboards, wide papers and pencils —, to sophisticated intermediaries, where assumptions and definitely, theories, are ultimately tested directly *in silico*.

As it is the case with most of present-day disciplines which deal with huge amounts of data, extensively marketed as *Big Data* (Greene et al., 2015), Bioinformatics activities are not a matter of simply finding a needle in a haystack, but to make needles crystallize from a seemingly homogeneous haystack.

As commented, this situation does not invalidate many of the earlier conventions and algorithms assumed during the first decades of Bioinformatics, e. g., the classic Smith-Waterman one (Smith & Waterman, 1981). However, because of dealing with huge amounts of data, different approaches can be followed for pre-filtering diverse inputs, always struggling with the dilemma of correction versus speed/performance.

The mentioned flood of information and data imposes that care and time must be taken to differentiate what is relevant and innovative and what is not. Within this context, open source (Stajich et al., 2006), open data (Molloy, 2011) and open access (Suber, 2002) have become widely-accepted paradigms in science as a whole, and particularly in Bioinformatics. This is because they make reproducibility practically possible in a short timeframe (ideally the time involved in downloading datasets and performing any computing by just clicking a button or typing a command-line).

On the other hand, part of the Bioinformatics contribution is also offering colleague scientists effective ways to access, generate and test their own data and analyses; and, at the same time, empowering them to continue using these data in other contexts not necessarily foreseen beforehand.

After the Web has become the *de facto* platform not only for accessing but also handling most of nowadays knowledge, there are many additional considerations to be taken into account beyond strictly biological questions.

As a direct consequence, biological sciences publication has largely increased during these last years. There is no doubt that IT resources, such as PubMed Central (Roberts, 2001), has facilitated the exchange of information; but largely, also a plethora of biological resources, notable biological databases and catalogs, have been indispensable tools for common-day research in Biology and Medicine.

1.1. Biological Databases

1.1.1. Context

As a result of genome sequencing projects and the high development of NGS techniques, an ensuing problem is how to deal with all the resulting accumulated data. Especially, how to gather them in a way to be accessible, filterable and comparable to other data we might generate by other different means.

A common approach is trying to index the input and output results in the different involved steps. By indexing, we allow that an entry, which could be a single sequence that can be analyzed, or its actual analysis output, can be more quickly retrieved just by querying it via a unique identifier. Alternately, we can get a set of entries which match a common property, therefore with a non-unique value, shared by them. Otherwise, we would need to transverse, normally rather costly, a whole set for spotting the entry we are interested.

In practice, when indexing, an accompanying binary files (that is, non-human readable) are derived from a file or a set of files. These store pointers for easing access to reference entries. Obviously, this gain in speed is at the expense of an increase of used storage. Moreover, we should not forget that the effort and associated cost (time spent programming, storage, etc.)

may only makes sense if indexed data has to be accessed several times (either by further analyses or through a web service, for instance).

There are different type of indexes and an important variety of data structures under which they can fit, notable implementations are balanced trees, B-trees or B + trees or hashes (key-value).

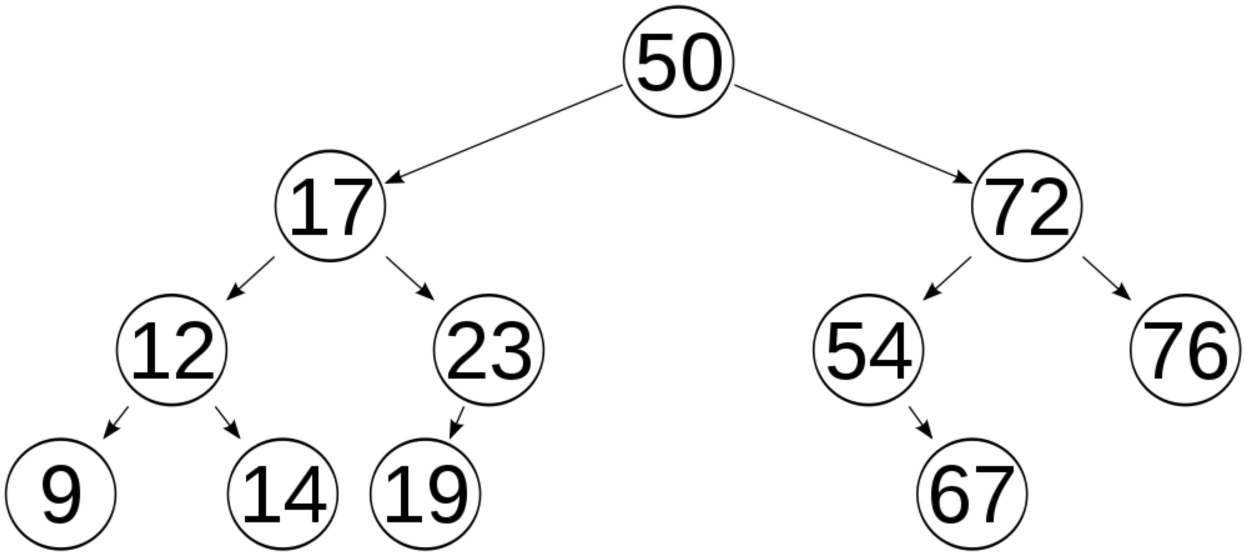


Figure 1. Height fixed (3) balanced-tree. Source: https://commons.wikimedia.org/wiki/File:AVLtreef.svg

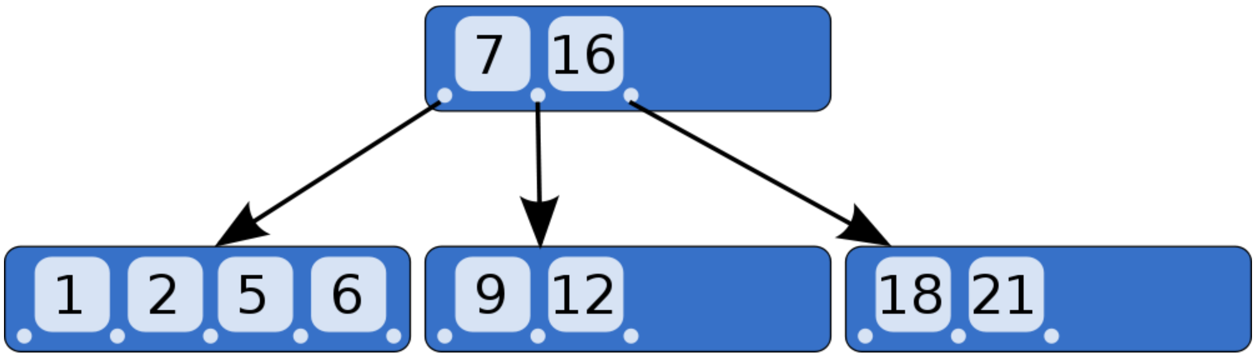


Figure 2. B-tree index. A very popular index structure used in most popular DBMS such as MySQL (Schwartz & Zaitsev, 2012) or CouchDB (Holt, 2011a). Source: https://commons.wikimedia.org/wiki/File:B-tree.svg

The index used has an important impact for improving the speed of retrieval of an entry. A good index for storing and retrieving integers is not necessarily a suitable one for long characters strings such as biological sequences, and the latter is not good for human language text that may be queried in a semantic way.

In real life, most bioinformaticians do not normally design their own index mechanism for their data, but have to make the proper decision about indexing with the available tools they have.

This practically means either deciding either indexed flat file databases or database management systems.

1.1.2. Indexed flat file databases

This is the normal approach of popular application like BLAST. Actually performed by applications such as `formatdb` (now `makeblastdb` in NCBI-BLAST+) (Camacho et al., 2009). By default, NCBI-BLAST uses GI, a unique integer used in NCBI databases as primary key. However, it’s possible to use custom indexes as well.

It’s worth noting that `makeblastdb` also offers the possibility to have a secondary index, specifically intended for looking up sequences associated to a taxonomy ID and restrict BLAST searches to only certain taxa.

Another existing solution, convenient when using pipelines with BioPerl (Stajich et al., 2002) and BioPython (Cock et al., 2009) frameworks is Open Biological Database Access (OBDA) (Osborne, 2011), where different criteria or keys (let’s say in a FASTA header) can be used for building up an index.

Flat file indexes are still object of heavy interest (Agrawal, 2008), specially as more variation sequence information (single nucleotide variants, insertions/deletions, etc.) are being needed as NGS practices are becoming more commonplace (UCSC Genome, 2015).

1.1.3. Database management systems

As data retrieval became more automatized, and specially, as there was an increasing demand to find different source data components, it turned out that scattered files were not just a convenient enough approach. A *DataBase Management System* (DBMS) allows to store under a same common infrastructure different data, and it normally provides a framework to relate them.

DBMSs offer a convenient way for users and external applications to connect, query and interact included data normally via a specific querying language.

As a rule of a thumb, DBMS hosts different databases (DBs), which can be considered a logical set of stored data where different tasks will be performed or mutual relationships drawn within. It must be noted that there are DBMSs that allow relationships to be set between different DBs (with an associated performance cost).

1.1.4. Relational Databases

Relational databases are so far the most popular and widely used database model. Only very recently their dominant position have been challenged by other models, as the ones we will comment in later sections of this document.

A relational database consists of data organized in the form of relations (or tables). A table essentially consists of rows with different fields/columns defined by a fixed schema. A column or a group of columns can normally define a primary key, whose value(s) must identify unequivocally a row (a defined set of data).

Moreover, relationships can be established between a row of one table with the row in another one, which is the main strength of the model. These relationships are created by mapping a column(s) of a table against the column(s) of another, and can be extended subsequently with other tables. The actual fields involved in the mapping of tables with each other are known as foreign keys.

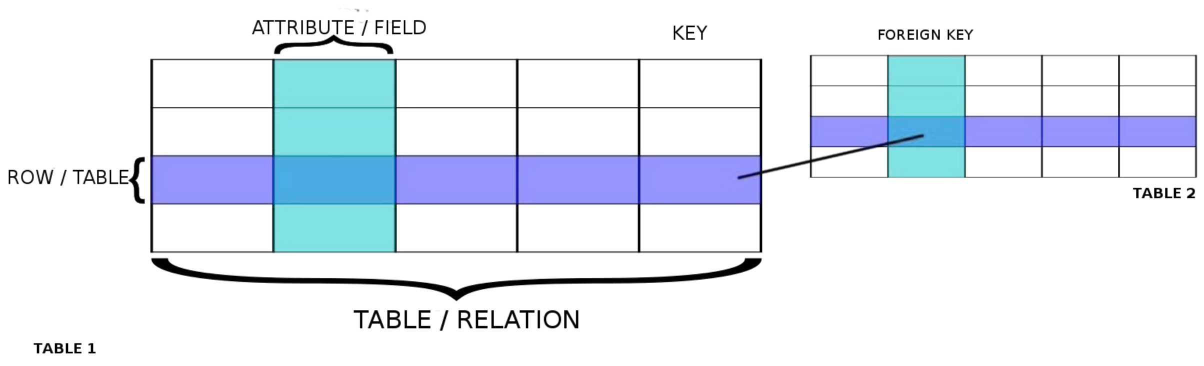


Figure 3. Schematic representation of the relationship between two relational tables. Foreign keys are actually keys in their own tables, foreign to the linked tables. Based on: https://commons.wikimedia.org/wiki/File:Relational_database_terms.svg

Apart from these later keys, generic keys (not necessarily based on univocal values) can be created in different fields so the table can be queried over them. At the computer level, keys are actually translated as file indexes not so different from flat file databases commented above in the end.

How data is stored, linked to each other and the type of involved indexes (which constrains how data can be queried) ultimately depends on the database engines (table types) available in the database management system. As example, at the time of writing, MySQL, one of the most popular relational DBMS available, is providing two main kind of table types: MyISAM (the original one) and InnoDB. If using the former, it’s possible to index human text at the word level so it can be queried (fulltext index), but it’s not feasible to have ensure referential integrity by using foreign keys (that is, simplistically, blocking any data modification if it only affects one table but the table it’s also linked to another where there is a matched mapping). Conversely, InnoDB did not provide any fulltext indexing until very recently (Yang et al., 2011), but it allowed to ensure referential integrity by foreign keys.

Another distinguishing feature of most relational database management systems (RDBMS) is using a Structured Query Language (SQL), which is a specific-purpose programming language which acts as a high-level interface for querying the database. Thanks to this, both the user and any external application must not deal with the computer-specific aspects of the different table types.

One common approach for reducing the redundancy of data is what is called normalization. Data is divided into smaller non-overlapping self-definable sets as tables, which in turn, can be linked between each other via relationships. Great care must be taken for defining proper indexes that link resulting tables in order not affect performance. Indeed, depending on the query, it might even be convenient to allow redundancy (via denormalization) if this implies a significant performance improvement (Sanders et al., 2001). A well-know case in biological applications is BioMart (Zhang et al., 2011), extensively used in projects such as ENSEMBL (Kinsella et al., 2011) and others (Durinck et al., 2005; Zhang et al., 2011).

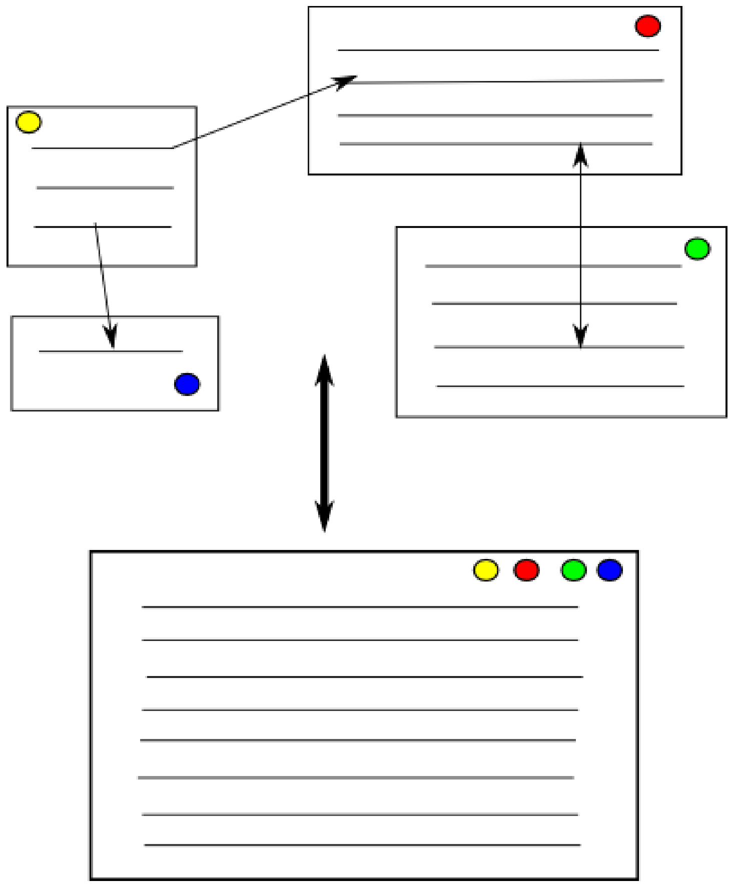


Figure 4. Simplified representation about normalization vs denormalization.

Data in text format is ideal for importing into a RDBMS such as MySQL. If contents are arranged in a comma, semicolon or tab separated value file, they can be normally imported using default functionalities in already pre-defined tables as far as field order is kept the same in both files and relational tables.

If files, even if text format, are already too complex or may be not convenient to process them for just importing (e. g., because they may be reused for other means), they may also be parsed for inserting their contents into a relational table. For this aim there are several database interfaces provided by different programming languages that can be helpful enough. As example, in most of the applications to be discussed below Perl DBI module has been used. This provides a unified and common interface, no matter the underlying RDBMS which is used (e.g. MySQL, Postgres, Oracle, etc., but nowadays also NoSQL database systems).

It's worth noting the existence of solutions such as SQLite, that allow performing typical Relational Database operations within a file context instead of a typical server connection. This will be further explored in `Appendix A`.

In latter pages we will also describe some case examples of biological databases that take advantage the approaches explained above and were actually published in different specific journals such as the yearly *Nucleic Acid Research* reference issue.

1.1.5. NoSQL Databases

In the previous sections some examples were provided of how a database could facilitate we could store different data we had previously gathered and processed in order to set up a resource that could be both useful for consulting researchers and also for planning further analyses.

As we have explained, in order to accommodate that data, we had to define tables that enabled relationships between the different kind of results we were handling. The storage of these data and definition of these tables is done by means of relational database software such as MySQL.

However, at the same time that bigger amounts of data were starting to flow into research, and Internet as a platform progressively became the favorite platform for an increasing number of daily routines, many technologists realized that SQL rigidity might be a concern.

From these circumstances, NoSQL (which means *Not Only SQL*) approaches started to develop. When talking about NoSQL, we actually refer to a plethora of different software implementations that would only be common in the sense that escape from the traditional relational tabular design of relational databases.

3 kinds of NoSQL approximations can be differentiated:

Key-value:
This could be roughly equivalent to porting traditional dictionary/hash variable types or data structures used in traditional programming into the database realm. At first, we would expect that less complex data could fit in this kind of schema (even though it may also expand in many dimensions), and at the same time it’s normally the fastest way to exchange data. Its major use in the real world is for caching. This means that data is not usually saved permanently in this kind of stores but rather kept temporally coming from other sources and delivered to a process queue or to the user that requested the resource. Indeed, that’s the case Redis, a key-value NoSQL software, as deployed in Wikipedia.

Document:
This type of approach can be regarded as an extension of the former. But, instead of simple key-values, we deal with an already more complex kind of data, alike to XML (Extended Markup Language) and, especially JSON (JavaScript Object Notation). Since these 2 kind of formats were already ubiquitously used when exchanging information between applications in Internet, it became convenient that some applications could store with them without the associated drawback of fitting data into a schema. We will describe a specific implementation we used in the next pages.

Graph:
Compared to the previous two types, this kind of database is specifically thought for storing data modelled according to a mathematical graph, which essentially consists of a set of nodes and edges. In this case, relationships are specially relevant and we expect to get answers from perform queries that run along a path. There are many cases of biologically relevant graph approximations (from simple taxonomy relationships to interactomics networks) and we would detail an implementation in the next pages.

As a matter of a fact, some of the available NoSQL software solutions would not be purely of one type or another and, for instance, document-like information can be stored in node and edges of a graph database. There are even products ultimately designed to work as the three types, as it’s the case of ArangoDB (ArangoDB GmbH, n.d.)

Compared to relational databases, NoSQL ones do not force a defined schema (a table) for all the elements of a set. That means that different elements can have different properties/attributes. This would be like having rows of a same table with different types and numbers of columns.This is a huge flexibility but it also translates the responsibility of data integrity at the logical level (at the actual program that deals with the data).

1.1.5.1. Using Document Databases

As commented before, document databases are suited to deal with the exchanging formats of Internet. One of these oldest formats, XML, have also become spread in many other contexts, not necessarily restricted to web usage, so many applications provide output in XML format.

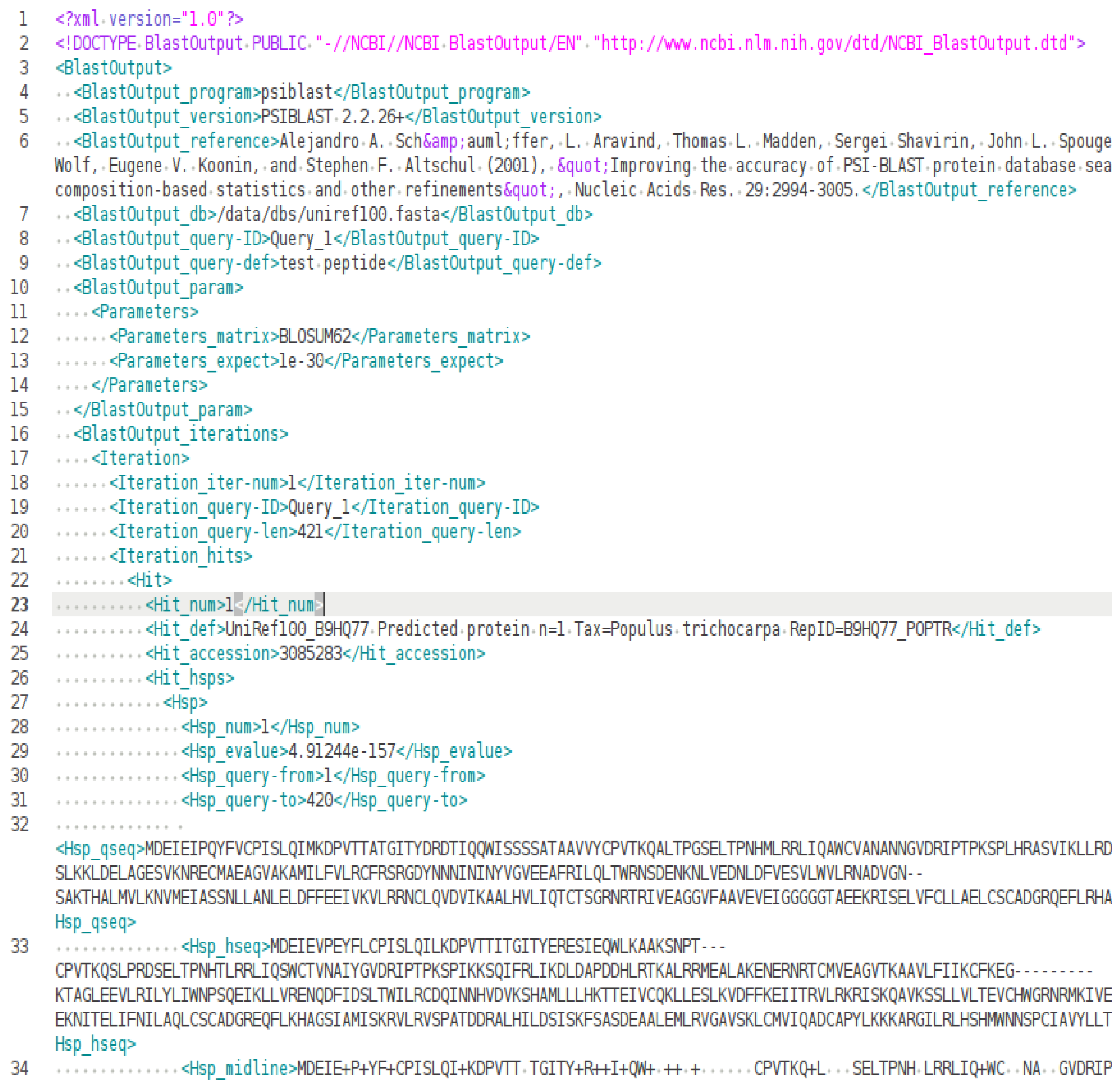
If an XML outcome is provided, normally there exists an additional schema file in order to ensure that results comply (are valid) to what is expected. This schema file can be codified either as DTD (Document Type Definition) or as XSD (XML Schema Definition) and software providers publish them with their program packages or in their websites.

A very-well known case of XML usage is NCBI Blast. Traditionally, BLAST provided results as a simple raw text file. In order to extract information such as hits, bioinformaticians had been elaborating different kind of text-processing tools. These tools might be often very sensitive to slight text particularities (let’s say, an unexpected diacritic) and prone to errors if some changes or new features were introduced in the results outcome with new versions.

Using XML avoids all these issues. Since there are already many existing tools to process XML in quite a generic way, there is no need to write a specific parser for one kind of document. The only inconvenience is that XML output files may be rather huge, as it’s the case, for instance of several iterations PSI-Blast analyses.

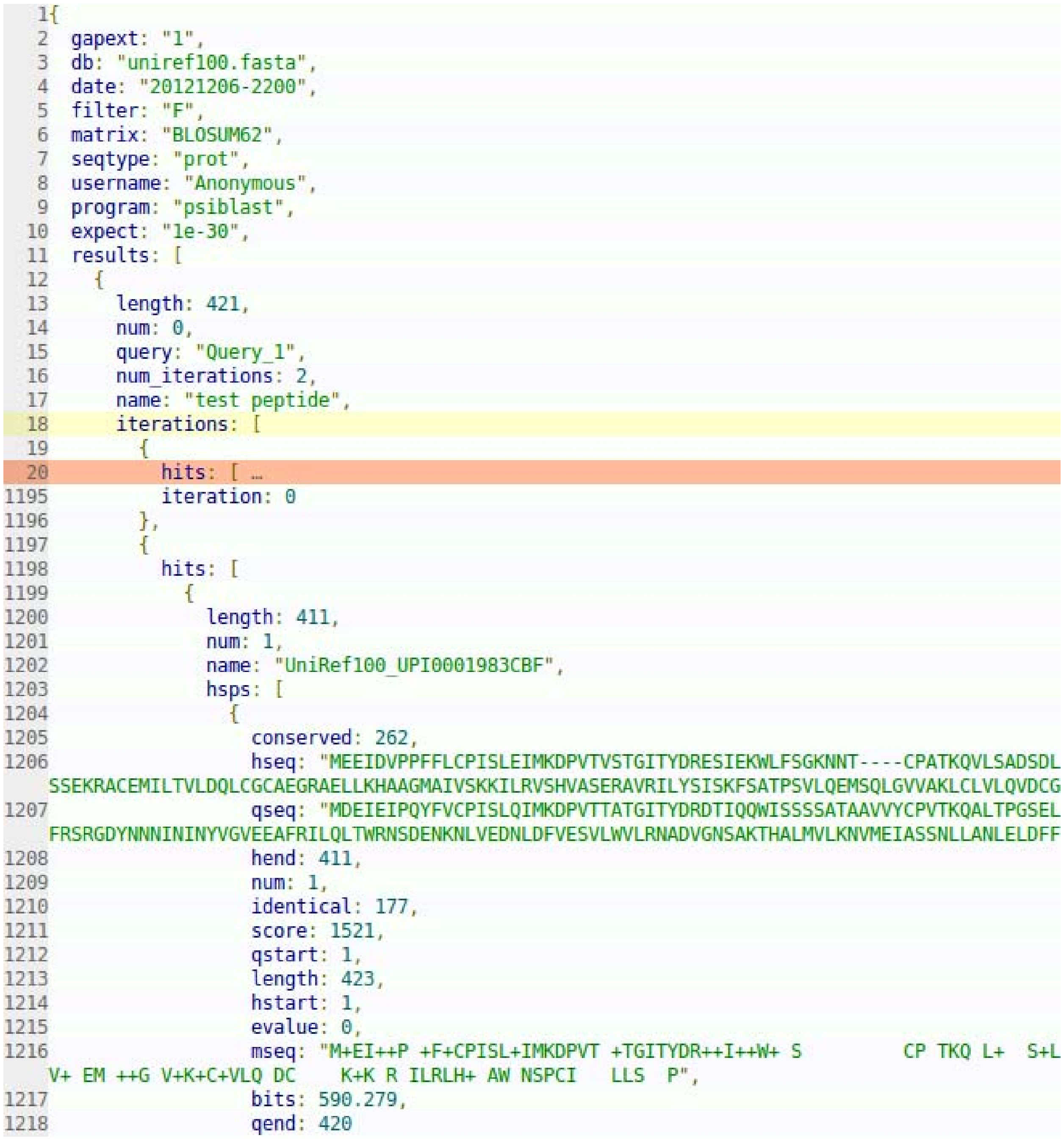
A too big file size is a problem if results are to be offered to the user inside the browser in a reasonable time. So, there is a need to simplify the XML content, for instance by stripping tags and skipping all these analyses details that are not relevant to further analyses or results. For this, a good approach is using XSLT (Extensible Stylesheet Language Transformations), which maps XML structure into any other structure, without any need to actually parse the whole XML file, which would little performant.

Elements of the XML analysis output are selected to be present in the conversion and then a JSON file is generated. For this we use `xsltproc`. JSON is a leaner file format than XML. Tags are substituted by keys but, as the former one, you can keep a hierarchy out the outcome values and the involved fields. Below an example of a XML file and the stripped converted JSON.



```
<?xml version="1.0"?>
<!DOCTYPE BlastOutput PUBLIC "-//NCBI//NCBI BlastOutput/EN" "http://www.ncbi.nlm.nih.gov/dtd/NCBI_BlastOutput.dtd">
<BlastOutput>
  <BlastOutput_program>psiblast</BlastOutput_program>
  <BlastOutput_version>PSIBLAST 2.2.26+</BlastOutput_version>
  <BlastOutput_reference>Alejandro A. Sch&amp;auml;ffer, L. Aravind, Thomas L. Madden, Sergei Shavirin, John L. Spouge
Wolf, Eugene V. Koonin, and Stephen F. Altschul (2001), &quot;Improving the accuracy of PSI-BLAST protein database sea
composition-based statistics and other refinements&quot;, Nucleic Acids Res. 29:2994-3005.</BlastOutput_reference>
  <BlastOutput_db>/data/dbs/uniref100.fasta</BlastOutput_db>
  <BlastOutput_query-ID>Query_1</BlastOutput_query-ID>
  <BlastOutput_query-def>test peptide</BlastOutput_query-def>
  <BlastOutput_param>
    <Parameters>
      <Parameters_matrix>BLOSUM62</Parameters_matrix>
      <Parameters_expect>1e-30</Parameters_expect>
    </Parameters>
  </BlastOutput_param>
  <BlastOutput_iterations>
    <Iteration>
      <Iteration_iter-num>1</Iteration_iter-num>
      <Iteration_query-ID>Query_1</Iteration_query-ID>
      <Iteration_query-len>421</Iteration_query-len>
      <Iteration_hits>
        <Hit>
          <Hit_num>1</Hit_num>
          <Hit_def>UniRef100_B9HQ77 Predicted protein n=1 Tax=Populus trichocarpa RepID=B9HQ77_POPTR</Hit_def>
          <Hit_accession>3085283</Hit_accession>
          <Hit_hsps>
            <Hsp>
              <Hsp_num>1</Hsp_num>
              <Hsp_evalue>4.91244e-157</Hsp_evalue>
              <Hsp_query-from>1</Hsp_query-from>
              <Hsp_query-to>420</Hsp_query-to>
               
<Hsp_qseq>MDEIEIPQYFVCPISLQIMKDPVTTATGITYDRDTIQQWISSSSATAAVVYCPVTKQALTPGSELTPNHMLRRLIQAWCVANANNGVDRIPTPKSPLHRASVIKLLRD
SLKKLDELAGESVKNRECMAEAGVAKAMILFVLRCFRSRGDYNNNININYVGVEEAFRILQLTWRNSDENKNLVEDNLDFVESVLWVLRNADVGN--
SAKTHALMVLKNVMEIASSNLLANLELDFFEEIVKVLRRNCLQVDVIKAALHVLIQTCTSGRNRTRIVEAGGVFAAVEVEIGGGGGTAEEKRISELVFCLLAELCSCADGRQEFLRHA
Hsp_qseq>
              <Hsp_hseq>MDEIEVPEYFLCPISLQILKDPVTTITGITYERESIEQWLKAAKSNPT---
CPVTKQSLPRDSELTPNHTLRRLIQSWCTVNAIYGVDRIPTPKSPIKKSQIFRLIKDLDAPDDHLRTKALRRMEALAKENERNRTCMVEAGVTKAAVLFIIKCFKEG---------
KTAGLEEVLRILYLIWNPSQEIKLLVRENQDFIDSLTWILRCDQINNHVDVKSHAMLLLHKTTEIVCQKLLESLKVDFFKEIITRVLRKRISKQAVKSSLLVLTEVCHWGRNRMKIVE
EKNITELIFNILAQLCSCADGREQFLKHAGSIAMISKRVLRVSPATDDRALHILDSISKFSASDEAALEMLRVGAVSKLCMVIQADCAPYLKKKARGILRLHSHMWNNSPCIAVYLLT
Hsp_hseq>
              <Hsp_midline>MDEIE+P+YF+CPISLQI+KDPVTT TGITY+R++I+QW+ ++ +      CPVTKQ+L   SELTPNH LRRLIQ+WC  NA  GVDRIP
```

Figure 5. Simplified excerpt of an XML blast document.



```
{
  gapext: "1",
  db: "uniref100.fasta",
  date: "20121206-2200",
  filter: "F",
  matrix: "BLOSUM62",
  seqtype: "prot",
  username: "Anonymous",
  program: "psiblast",
  expect: "1e-30",
  results: [
    {
      length: 421,
      num: 0,
      query: "Query_1",
      num_iterations: 2,
      name: "test peptide",
      iterations: [
        {
          hits: [ …
          iteration: 0
        },
        {
          hits: [
            {
              length: 411,
              num: 1,
              name: "UniRef100_UPI0001983CBF",
              hsps: [
                {
                  conserved: 262,
                  hseq: "MEEIDVPPFFLCPISLEIMKDPVTVSTGITYDRESIEKWLFSGKNNT----CPATKQVLSADSDL
SSEKRACEMILTVLDQLCGCAEGRAELLKHAAGMAIVSKKILRVSHVASERAVRILYSISKFSATPSVLQEMSQLGVVAKLCLVLQVDCG
                  qseq: "MDEIEIPQYFVCPISLQIMKDPVTTATGITYDRDTIQQWISSSSATAAVVYCPVTKQALTPGSEL
FRSRGDYNNNININYVGVEEAFRILQLTWRNSDENKNLVEDNLDFVESVLWVLRNADVGNSAKTHALMVLKNVMEIASSNLLANLELDFF
                  hend: 411,
                  num: 1,
                  identical: 177,
                  score: 1521,
                  qstart: 1,
                  length: 423,
                  hstart: 1,
                  evalue: 0,
                  mseq: "M+EI++P +F+CPISL+IMKDPVT +TGITYDR++I++W+ S         CP TKQ L+  S+L
V+ EM ++G V+K+C+VLQ DC    K+K R ILRLH+ AW NSPCI   LLS  P",
                  bits: 590.279,
                  qend: 420
```

Figure 6. Simplified excerpt of a JSON document transformed from XML thanks to an applied XSL transformation sheet: https://gist.github.com/toniher/2b1e96297ade920728b8

This JSON outcome can be accepted quite straightforwardly by Document Database engines. That way information is kept in a more coherent way than just by being stored as separate files and search can also be performed from the added results, as we would explain below.

In our deployments we used CouchDB as Document Database Engine (Agrawal, 2008). As a main peculiarity among other similar solutions, it's only way to interact with is via a REST API. This is an advantage in the sense there is no need to learn a completely different language and, through little configuration, access can be easily granted to different locations. On the other hand, care must be taken to upload a huge amount of data into the database, and for this, instead of submitting only a JSON document in one request, many are submitted at once in a series of batches requests.

In order to search and present input data, it is necessary to index the fields we want to query in advance. This is done in CouchDB, following the same web philosophy as with its API, in Javascript and thanks to map-reduce functions.

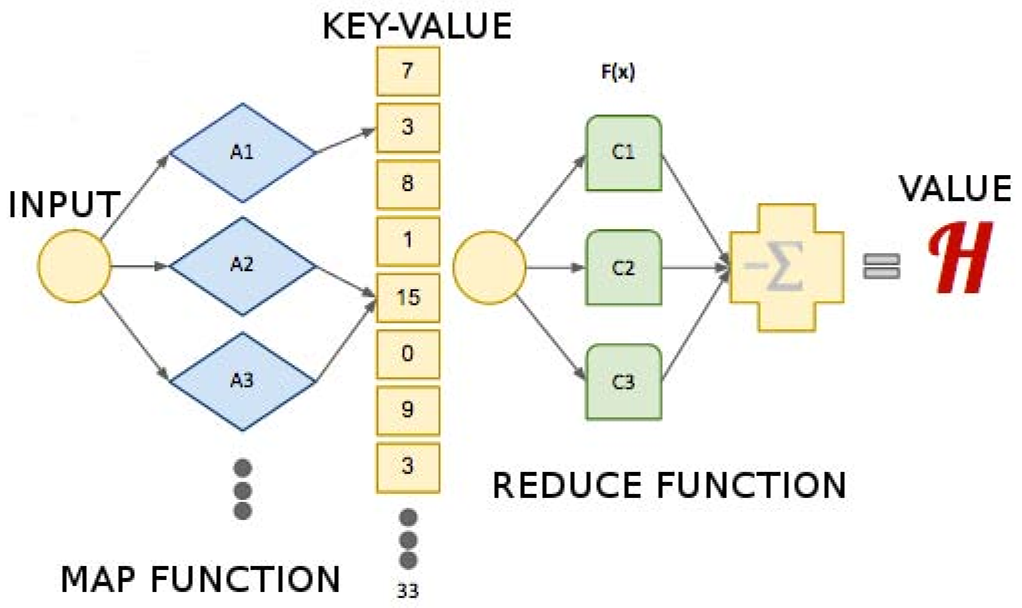


Figure 7. Graphical representation of how a map-reduce processing flow works. Based on: https://commons.wikimedia.org/wiki/File:MapReduceCompleto.png

```
function(doc) {
     if(doc.ref && doc.results && doc.type=='blast') {
               emit(doc.ref, doc);
     }
}
```

Figure 8. Javascript map function for retrieving documents which have *ref* and *results* attributes, and which also have *type* attribute value of 'blast'. *ref* attribute value and the whole document are returned. In a next section this approach is used for linking different kind of documents that are related between each other (e. g., input and output of an analysis).

```
function(keys, values, rereduce) {
  if (rereduce) {
     return sum(values);
  } else {
     return values.length;
  }
}
```

Figure 9. Javascript reduce function coupled to the previous map function. The function above can also be called simply with the built-in function _count. The differential usage depending on rereduce parameter is a constriction posed by B-trees underlying indexing (Holt, 2011b).

1.1.5.2. Graph Databases

As advanced some sections before, graph databases may be convenient for certain biological challenges. Indeed, that’s the case when they can be more easily modeled by graphs.

Two well-known and widely-used graphs modellings in Biosciences are organism taxonomy, such as the one maintained by NCBI (Anderson et al., 2010), and Gene Ontology (GO), curated by Gene Ontology Consortium (Federhen, 2012). Both cases can be mathematically described as directed acyclic graphs (DAGs).

This kind of graphs implies that is not possible to have loops, that is, you cannot go from one vertex (node) and arrive at itself again after transversing along different edges (relationships). Moreover, edges are actually semantic, links between two nodes are meaningful and in the two examples above cannot be reverted.
NCBI Taxonomy DAG is far simpler than Gene Ontology, one node can only have one outcoming relationship and this is only possible type. As example, a species can only have one genus, and the only possible relationship is the intrinsic hierarchy between different taxonomic ranks (species -> genus -> family…)

Gene Ontology is actually 3 DAGs representing 3 ontologies: biological process, cellular component and molecular function. Unless we include gene products in the same graph as intermediaries, there is no way to go through any of the elements of two different ontologies. Moreover, within the same ontology, one node can have more than one relation with another (*is_a*, *part_of*, *regulates*, *positively_regulates*, *negatively_regulates*).

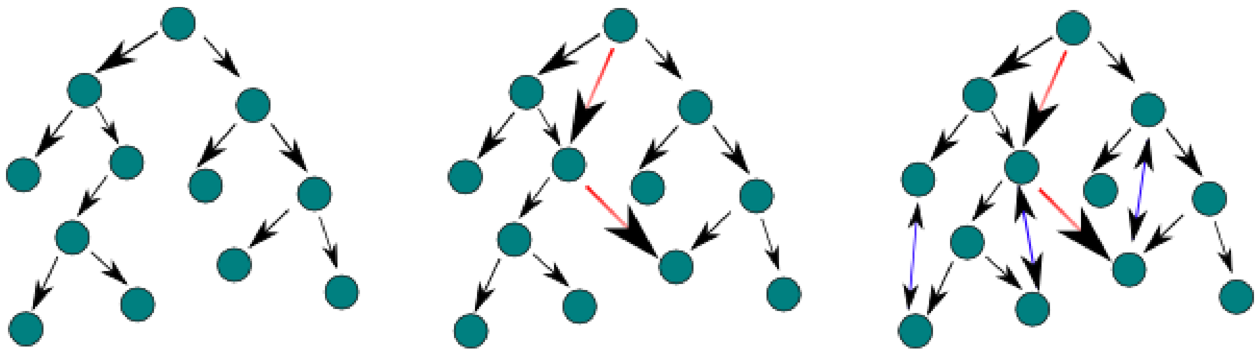
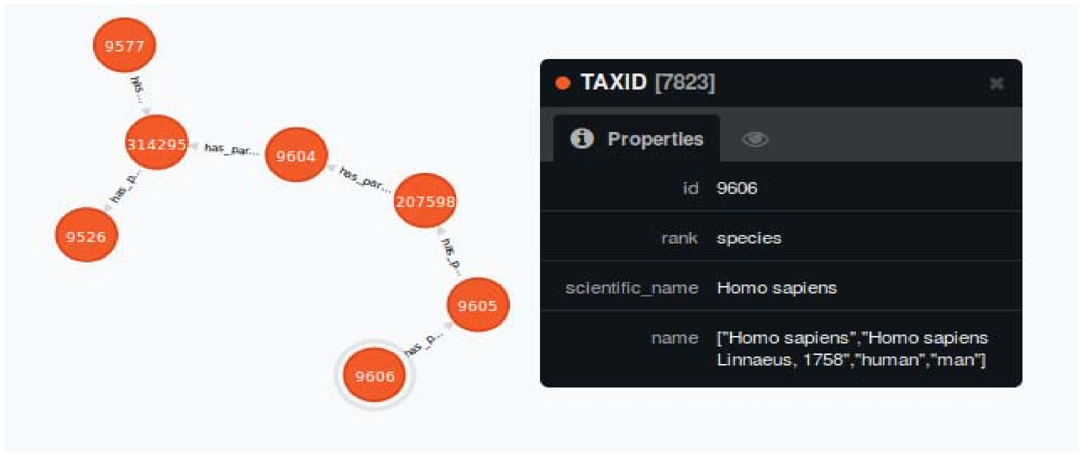


Figure 10. Types of graphs. Left: simple hierarchy. Directed rule, 1 parent (e. g. NCBI taxonomy). Center: Directed Acyclic Graph. Directed Rules, but >= 1 parents (e. g. Gene Ontology). Right: Not directed graph. Nodes can be both parents and children (e. g. metabolic networks) (The Gene Ontology Consortium, 2013).

The fact is that there are already existing and well-known approximations of modeling DAGs suchs a GO or NCBI taxonomy into a MySQL database (Mackey, 2002). However, since data needs to be transformed (by adding extra indices or foreign keys for instance) to fit to the restrictions of the engine requirements, as the number of involved nodes in a query increase, performance heavily degrades.

Graph database engines specifically address this, and are more convenient if distant or many nodes in a path are involved. In our work Neo4j (http://www.neo4j.com) will be used, that is one of the most popular software of this kind at the time of writing. As a relevant feature, both nodes and relations are regarded as property containers. In the end, this means that every node and relationship can actually be a container, that is, like a document in Document database, that can have a particular or predefined set of properties.

As example, a model representation of a NCBI taxonomy node with its stored properties:



TAXID [7823]

Properties

id	9606
rank	species
scientific_name	Homo sapiens
name	["Homo sapiens","Homo sapiens Linnaeus, 1758","human","man"]

Figure 11. Browser graphical representation of Neo4j node attributes and relationships with other nodes. In this case for NCBI taxonomy —highlighting human—.

In order to search for actual values of these properties, care must be taken to set up some indexes. This is normally done at the same time of importing. Behind the curtains, Neo4j uses Lucene (http://lucene.apache.org/), which is a quite widely extended system used in many text-searching environments.

Both NCBI and Gene Ontology Consortium provide the data commented above in tab-separated files. This format is quite straightforward for populating MySQL tables, but it’s also convenient enough for Neo4J. For this we use py2neo (http://py2neo.org) (Python Node4j REST API wrapper) and Pandas (http://pandas.pydata.org/) (Python Data Analysis Library). As it was the case with CouchDB, in order to improve the performance and decrease the import time, submitting many REST requests at once in a batch is recommended.

Neo4j runs as a webservice instance, so it’s possible to have many different hosted graphs accessible as a webservice at the same time. So, one possible approach would be having NCBI Taxonomy graph and GO graph as separate running services. In any case, for sake of simplicity, we kept both them in the same instance, even though they are not linked by any node (they could be linked if we included, let’s say, gene products as well).

In order to query the graphs, Neo4J ships with an expressive SQL-like specific language named Cypher. This is very convenient for testing imported data and as a proof concept but, at the time of writing, it seemed not performant and fast enough for offering queries that are going to interact with the Web. Neo4J also provides a native Java API, which is more convenient in these cases, only at the expense of a longer time devoted to programming.

```
START n=node(nodeid1), m=node(nodeid2) MATCH p=shortestPath(n-[*]-m)
RETURN length(relationships(p)) as distance
```

Figure 12. Example Cypher query calculation the shortest path distance between two nodes.

In the biological context, it's worth noting the platform Bio4j (Pareja-Tobes et al., 2015), that aims to integrate the latter mentioned sets and other more (such as UniProt or Expasy Enzyme DB).

CHAPTER 2 - OBJECTIVES

General objectives

Designing advanced and user-friendly web services from the different programs created by the Bioinformatics Group of the *Institut de Biotecnologia i Biomedicina (UAB)*. As a consequence and thanks to this, creating biological databases as reference resources and empowering other experimental research projects from the gained bioinformatical know-how.

Specific objectives

1. Rewriting different applications formerly developed at IBB-UAB (e. g., *ProtLoc*, *TransMem*, *Transcout* or *Bypass*) so they could be used both as webservices and within a massive data-processing pipelines.

2. Designing biological databases and user-friendly interfaces so resulting analyses from the former tools could be easily accessible by third parties.

3. Implementing and deploying a sequence analysis framework so common-day usage tools such as BLAST and the tools commented above could be used with little technical intervention.

4. Bioinformatics and phylogenetics analysis, function prediction and structural modeling of Nna-like proteins (also known as CCPs) as putative targets for diverse diseases such as malaria or neurodegenerative disorders.

CHAPTER 3 - RESULTS

3.1. ProtLoc

3.1.1. Context

Functionality of protein products make sense in their natural context, that is, in conjunction with other biological products (proteins, RNA, DNA, etc.) within the cell.

On the other hand, the biochemical milieu of the different compartments of the cell are not the necessarily same, having for instance different pH and pKa values. This electrochemical cell diversity determines or influences the presence, absence or quantity of different protein products, which are in the end conditioned by their actual amino acid composition. For instance, certain proteins may not be able to fold, catalyze a reaction, or interact with other proteins in some cellular compartments because of the state of some of their amino acids at a certain pH (being hydrogenated or not).

Based on this assumption, we might think that depending on the chemical properties of containing amino acids of a protein, that might be more likely to be present in one location or another. In some sense, this can be considered in the end a corollary of the central dogma of biology; that is, sequences (at the very gene level) do contain already information about their final functional location.

A subsequent question is elucidating how this final location information may be actually encoded. It may be ‘sequentially’ encoded, that is, the order and presence within a sequence of certain amino acids (sometimes in certain regions of the sequence, such as the extremes) may determine the final location. This is the case of the detection of signal peptides (von Heijne, 1990), present in the N-terminus of the translated peptides, which lead them to be sent to the secretory pathways. Another instance are transmembrane regions or experimentally well-known DNA-binding regions, that necessarily imply a functional location. We will talk about these two latter in upcoming sections. Moreover, the determination in a polypeptide of certain regions or domains which are already known to imply an interaction with other protein products placed in a definite location, transitively implies that the former may share location with the latter (guilty by association) (Schauer & Stingl, 2009).

All these previous approaches are not necessarily incompatible with what introduced above, the fact that certain abundance of some types of amino acids (e. g., these ones that may allow to have transmembrane regions in some sequence stretches) may indicate that a product may be in a cellular location. And this is actually the hypothesis behind Protloc (referring to Protein Location) (Cedano et al., 1997).

As a starting point, 5 different basic non-overlapping sets of proteins were created representing rough cellular locations: <u>Intracellular</u>, <u>Extracellular</u>, <u>Nuclear</u>, <u>Transmembrane</u> and <u>Anchored</u>. Each set was populated thanks to manually-curated annotation from Swiss-Prot database.

In order to assign a putative cellular location for an amino acid sequence, Malahanobis distance statistic was used.

Query protein is converted in a vector of length 20 (one position for every amino acid), with the frequency of every amino acid in the sequence. Vector values necessarily sum up 1.

$$x = (x_1, x_2, x_3, \ldots, x_N)^T$$

amino acid frequency of a sequence

We calculate the distance of this vector against the different 5 sets, using Mahalanobis distance. Each set consists of an average vector and a covariance matrix respect to the different amino acids. If all sequences of a certain set happened to have the same amino acid frequency average, covariance matrix but be an identity matrix, and instance of Mahalanobis distance we would be calculating a simpler Euclidian distance, that is the case as well when calculating the distance between each other.

$$\mu = (\mu_1, \mu_2, \mu_3, \ldots, \mu_N)^T$$

average amino acid frequency of a set

$$D_M(x) = \sqrt{(x-\mu)^T S^{-1} (x-\mu)}.$$

Mahalanobis distance,
where S is the covariance matrix.

As explained in the published paper, the approach turned to be sensitive enough and predicted protein location quite faithfully (either as first, or at worst, second position).

As example, for a protein such as TMM43_HUMAN (Transmembrane protein 43), we get the following result. The lower the distance, the more likely location assigned.

```
Membrane => 8.04007866311015
Anchored => 8.70351093081404
Nuclear => 8.92771314050655
Intracellular => 9.33241032405688
Extracellular => 11.0166179413074
```

3.1.2. Technical deployment

The program was originally implemented and published as an Excel spreadsheet macro (Cedano et al., 1997).

By the time it was originally published, it was not unusual to handle just a few ORFs that could be perfectly analyzed case by case basis. However, the advent of big genome sequencing projects rendered this technical approach as impractical, and the application was reprogrammed as a modern Perl script, so it could be integrated in high-throughput analyses pipelines, as we would comment in a later section. Afterwards, it was also reprogrammed as a CGI (Common Gateway Interface) website.

For a long time CGI was the most used, if not the only, way bioinformaticians could make their tools available to non-tech savvy biologists. CGI consists in a rather language-neutral standard interface method for generating dynamic web content from any kind of underlying program in the web server. Therefore, in principle, converting a command-line program to a CGI web service is a matter of changing the input, from command-line parameters to REST-like requests normally submitted via web forms, and the output, from commonly raw text data to HTML output which can be rendered in a browser window. That is the case of ProtLoc, which could be used as a CGI by any interested biologist who simply pasted their interest sequence in the available form.

CGI triggers an application process per request, so if a heavy computation is performed or many requests are submitted at once, the web server may get overwhelmed and even become unresponsive. As a workaround, developers normally submitted processes in the background and results were offered to the user by being queried after a succession of page refreshes or even by email. In case of long computations, the latter is the most common and realistic approach even if CGI is not actually used.

Later on, as alternatives to CGI, other methods have emerged such as:

- FastCGI, and actual improvement of the former but allowing to fork more easily processes, so blocking of the server is less likely.
- Apache modules, providing a specific language-compiled interface to the web server (Apache in this case), being the most popular case for PHP (mod_php).
- Language-specific interfaces (e. g. PSGI for Perl, or WSGI for Python).

At the time of writing, ProtLoc is not only provided as a website but also as a webservice. This last possibility allow external web-enabled applications to use it straightforward. For this, Mojolicious Perl framework is used, which allows different web interface methods (CGI, FastCGI or PSGI) to be relied on with minimal change depending on the configuration of the hosting server.

3.1.3. Next advances and considerations

After this cellular location prediction approach was first introduced, other richer and more detailed annotation sources have emerged compared to the rather free SwissProt keywords. Notably, that’s the case of Gene Ontology (GO) (The Gene Ontology Consortium, 2013), which for this case would fit into its cellular component domain. How Malahanobis distance approach could be effectively extended to further subsets based on GO layers is something we’re currently investigating.

Another open question for the application of this approach would be whether could be useful extending it for non-canonical amino acids, such as selenocysteine (Böck et al., 1991), or pyrrolysine (Gaston et al., 2011) or even for amino acids that may have posttranslational modifications such as methylations or acetylations.

Moreover, it cannot be ignored the need of comparing this simpler methodology against more sophisticated computer approaches such as SVM (Wan et al., 2012, Kumar et al., 2014)

Code can be found at:
https://github.com/toniher/ProtLoc/

3.2. TransMem

As in the case of ProtLoc trying to elucidate cellular location of ORF products, another program developed in IBB-UAB, TransMem (Aloy et al., 1997), aims to detect hydrophobic transmembrane (HTM) stretches. The prediction of these protein structural elements are samely very relevant for assigning a function.

Compared to the previous approach used in ProtLoc, in TransMem Artificial Neural Networks (ANN) is the preferred method. Briefly, ANNs are a type of computational models initially inspired on living neural systems where a system of interconnected nodes (neurons) compute values which are 'feeded' to them. This feeding enables the system to learn about the input data and, therefore, become capable of recognising certain patterns when new data is presented. For TransMem, the network was based on Multilayer Perceptron mode (Rojas, 1996), which was trained with 5 transmembrane proteins with 39 helices and tested against a set of 55 sequences with transmembrane stretches of a SwissProt database set used by a formerly reference algorism in transmembrane prediction (Aloy et al., 1997).

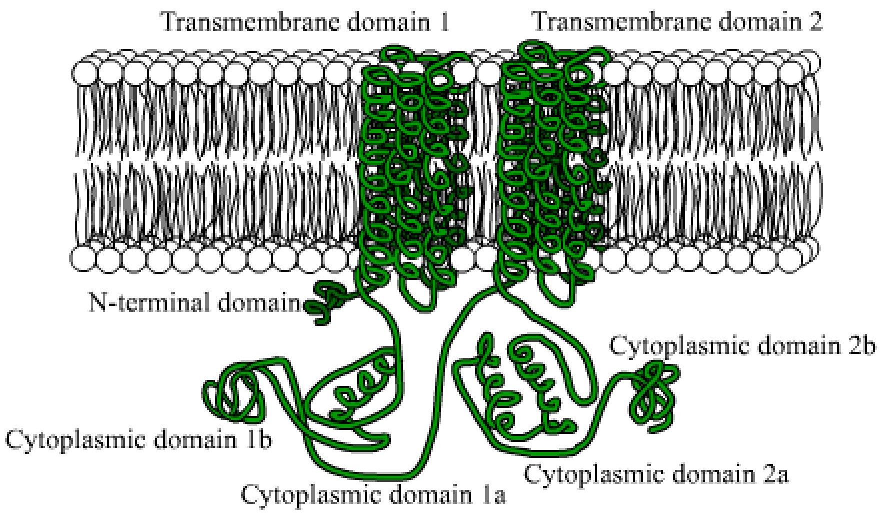


Figure 13. Example transmembrane protein Adenylate Cyclase. Source: https://commons.wikimedia.org/wiki/File:Adenylate_cyclase.png

Initially deployed as an Excel spreadsheet, the program was ported to C/C++. For designing the trained ANN, SNNS (Stuttgart Neural Network Simulator) simulator framework (Eberhard Karls University, n.d.) was used, and exported as a library.

This porting, as it was done with ProtLoc, allowed the application to be integrated in massive pipeline analyses.

Later on, this program was also turned into a CGI (wrapped with Perl using Inline library (http://www.slideshare.net/daoswald/getting-started-with-perl-xs-and-inlinec), and recently as a webservice as well following the same approximation as described with ProtLoc.

An outcome commandline result can be seen below, where a window size of n is used:

```
$ ./transmem -w 10 Q9BTV4.fasta
sp|Q9BTV4|TMM43_HUMAN    GMFVGLMAFLLSFYL    34    49
sp|Q9BTV4|TMM43_HUMAN    VESFMATAPF    174    184
sp|Q9BTV4|TMM43_HUMAN    IGRFFLSSGL    186    196
sp|Q9BTV4|TMM43_HUMAN    AAGWMAMFMGLNLM    312    326
sp|Q9BTV4|TMM43_HUMAN    LVNIGLKAFAFCVATSLTLLTVAAGWLF    342    370
sp|Q9BTV4|TMM43_HUMAN    LWALLIAGLALVPIL    373    388
```

By the time of writing, there are many other applications for predicting transmembrane helices, many of them based on Hidden Markov Model (HMM) approaches (Liakopoulos et al., 2001; Sonnhammer et al., 1998).

A benchmark and comparative analyses of the available tools was performed in 2005 (Cuthbertson et al., 2005), but TransMem was not among them, surely because of not being publicly accessible by that time.

Nevertheless, it’s worth noting that, respect to other competing software, TransMem is open-source and so other researchers can easily adapt it to their needs.

As a recent use case, TransMem has been utilized massively in plant genomes for detecting in a qualitative way ‘Receptor Like Proteins’ (RLP) (Sanseverino et al., 2012), which consist of leucine-rich receptor-like repeats and transmembrane regions of ~25 aa (Contaldi, unpublished).

Code can be found at:
https://github.com/toniher/TransMem

3.3. TranScout

Even though some rough clues of gene products functionality can be drawn from more or less generalistic methods such as the ones presented above, that is: non sequential (as ProtLoc) and partially sequential (as TransMem); the fact, is that in most other cases, sequential information is essential for elucidating any specific function. That is the case of transcription factors, which actually have regions that specifically bind to genomic DNA regions.

Transcription factors have a major role in the complex regulation of gene expression. Their identification, characterization and the exploration of their diversity is an actual important step towards understanding determinant biological processes such as cell development, tissue differentiation and apoptosis. Knowledge of the triggers and effectors that link these latter processes with their expression basis have evident biological and biomedical implications.

Classifying transcription factors in related groups among different organisms (some used as model organisms) may help us to highlight evolutionarily conserved or dissimilar strategies and propose shared solutions to different problems where related transcription factors occur.
In recent years, a large amount of information has been accumulated about proteins with transcriptional regulatory activity.

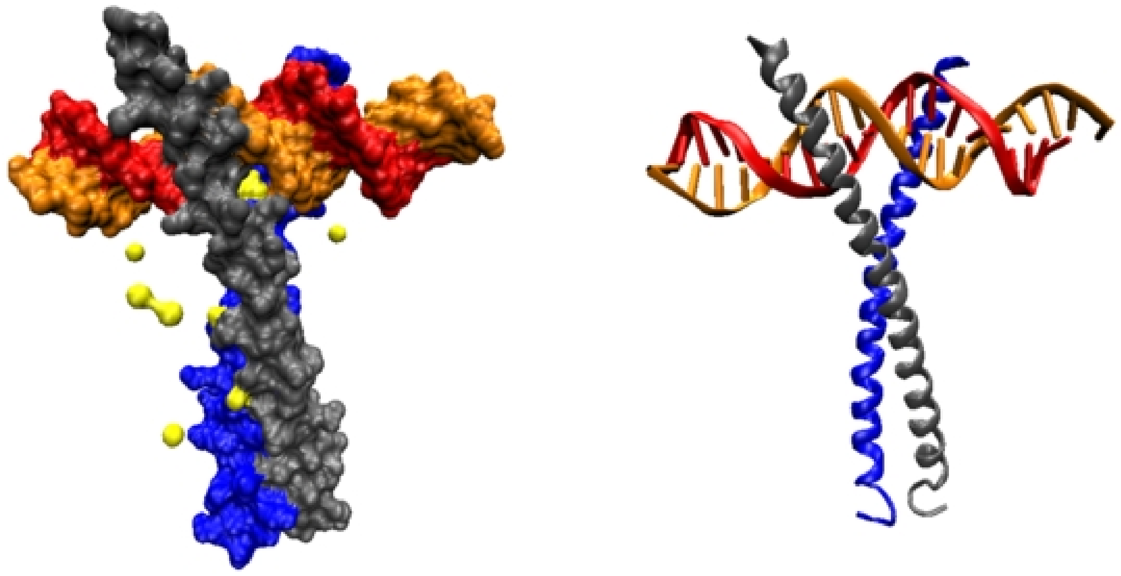


Figure 14. Leucine zipper type transcription factor in complex with DNA. Created from MMDB entry 48655 using visualization package VMD. Source: https://commons.wikimedia.org/wiki/File:Leucine_zipper2.png

One of the frequent and most used methodologies is by using Position-Specific Scoring Matrices (PSSM). By this approach, a set of well-defined aligned sequences can be encoded into a profile matrix of occurrence frequencies which can be used to search against a query sequence or database of sequences. Resulting probabilities matrices are normally weighed against existing substitution matrices, such as PAM (Cuthbertson et al., 2005) or BLOSUM ones (Jones et al., 1992), which normally represent the rate of amino acid substitution as deduced from experimental datasets. At the time of writing, BLOSUM62 is the most commonly used one and the default option of popular programs such as BLAST (Mount, 2008).

In the specific case of TranScout (Aguilar et al., 2002), different well-known protein domains associated to transcription factors (not necessarily only DNA-binding ones) were originally derived from TransFac classification (Matys et al., 2003) and later extended providing up to 73 possible profiles.

```
DEF    Basic domain (bZIP)
SIGN    bZIP transcription factors
RNG    Class
FUNC    DNA-binding
ALIN    leuzip.basicdom.bench.alin.swiss
MAT    leuzip.basicdom.bench.matvor
MAX    8.22
MIN    4.65
THRES    19.00
GAPP    2.00
GAP    1.00
CAU    0

14
0.00 0.06 0.08 0.00 0.00 0.00 0.04 0.70 0.74 0.10 0.06 0.05 0.00 0.05 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.04 0.13 0.25 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.05 0.00 0.00 0.00 0.00 0.00 0.00 0.13 0.00
0.00 0.00 0.29 0.00 0.00 0.00 0.28 0.00 0.00 0.00 0.10 0.00 0.00 0.05 0.00
0.04 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.08 0.00 0.00 0.00
0.00 0.00 0.04 0.00 0.00 0.13 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.12 0.00 0.00 0.00 0.24 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.13 0.14 0.13 0.29 0.00 0.23 0.00 0.00 0.00 0.13 0.24 0.00 0.00 0.23 0.39
0.00 0.35 0.11 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.03 0.00 0.00 0.06 0.00
0.00 0.16 0.13 0.00 0.00 0.00 0.06 0.00 0.00 0.00 0.00 0.00 0.00 0.03 0.00
0.00 0.00 0.00 0.13 0.87 0.13 0.00 0.00 0.00 0.00 0.04 0.00 0.00 0.07 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.07 0.15 0.00 0.00 0.00 0.00 0.19 0.16 0.09 0.00 0.00 0.22 0.05
0.83 0.30 0.05 0.33 0.13 0.52 0.00 0.00 0.00 0.24 0.32 0.13 0.87 0.15 0.56
0.00 0.00 0.00 0.11 0.00 0.00 0.00 0.30 0.00 0.26 0.00 0.42 0.00 0.02 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.12 0.00 0.00 0.07 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.08 0.00 0.08 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.13 0.00 0.00 0.00 0.00 0.07 0.13 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

Figure 15. Example definition and associated matrix of one of the classes scanned by TranScout.

A web page describing the involved transcription classes and a demonstration of the application can be found at: http://bioinf.uab.es/transcout/

For sake of reference, another well-known program that uses PSSM is the popular PROSITE (Falquet et al., 2002), which has also be used for classifying putative transcription factors (Sigrist et al., 2009).

At the time of writing, there has not been any update of this method with newer classification updates and against other popular application, as the mentioned before.

3.4. TrSDB: a proteome database of transcription factors

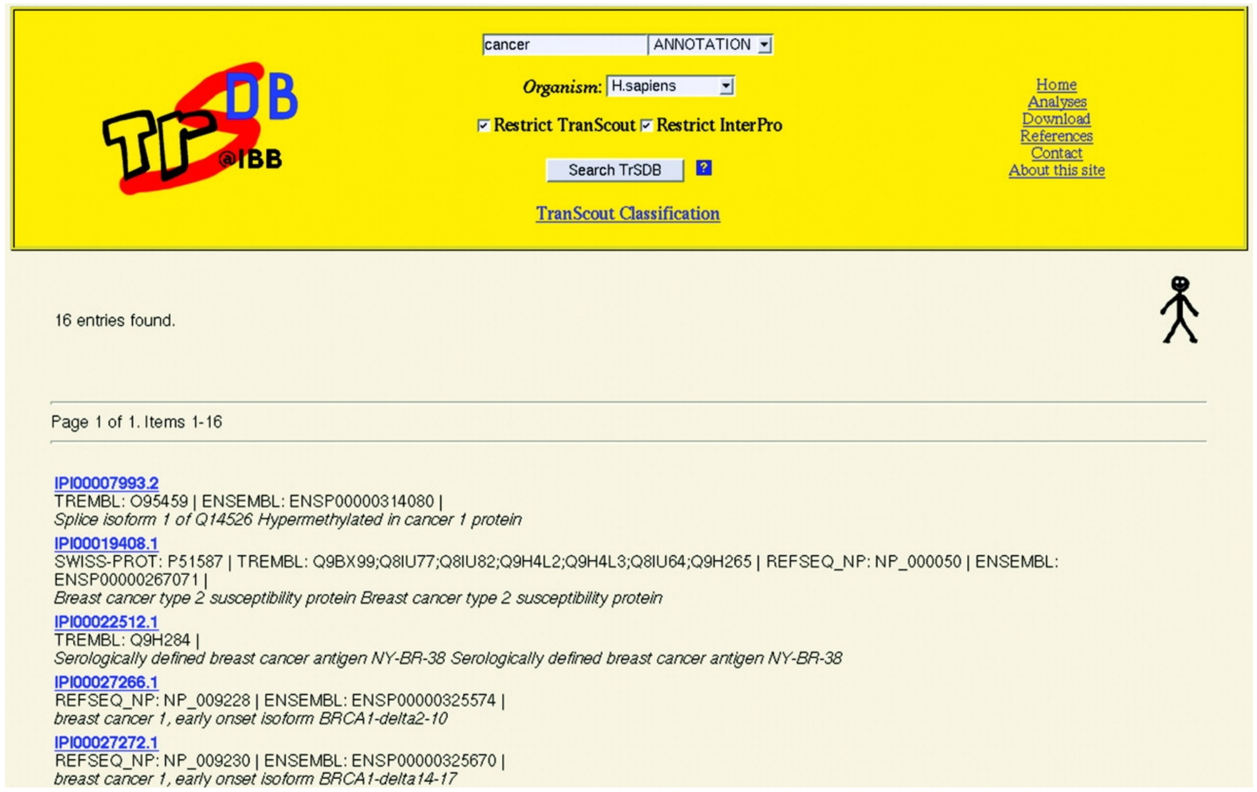
3.4.1. Introduction

The increasing knowledge on transcriptions factors has been progressively integrated in some comprehensively annotated biological databases such as TRANSFAC (Matys et al., 2003). In our approach, TranScout classification and algorithm (Aguilar et al., 2002) was used for analyzing nine eukaryotic non-redundant proteomes (*Homo sapiens*, *Mus musculus*, *Rattus norvegicus*, *Drosophila melanogaster*, *Caenorhabditis elegans*, *Arabidopsis thaliana*, *Guillardia theta*, *Saccharomyces cerevisiae* and *Schizo saccharomyces pombe*). Predictive analyses with ProtLoc (cellular localization assignment tool) (Cedano et al., 1997), TransMem (transmembrane domains prediction program) (Aloy et al., 1997), InterPro (Mulder et al., 2003) and Gene Ontology Annotation (GOA) (Camon et al., 2003) are also integrated in our database, providing additional validation to our results.

The result is TrSDB (Hermoso et al., 2004) —TranScout Database— (http://ibb.uab.es/trsdb), a proteome database of eukaryotic transcription factors based upon predicted motifs by TranScout and data sources such as InterPro and Gene Ontology Annotation

3.4.2. Web interface and automated analyses

TrSDB can be freely accessed at the website. Users may query TrSDB after selecting an organism by entering keywords to be searched for in the sequence annotation (e.g cancer), by sequence accession code (e.g. Q9Y5A6) or by accession codes that match InterPro (e.g. IPR001356), GOA (e.g. GO:0003700) or TranScout analyses or predictions (e.g. TRS0023). If preferred, queries may be restricted to two subsets: those entries showing positive TranScout matches (which are more likely to be transcription factors) or those that are annotated by InterPro. After submitting a query, a list of matching entries with a brief description (alternative sequence accession codes and annotation) is displayed.



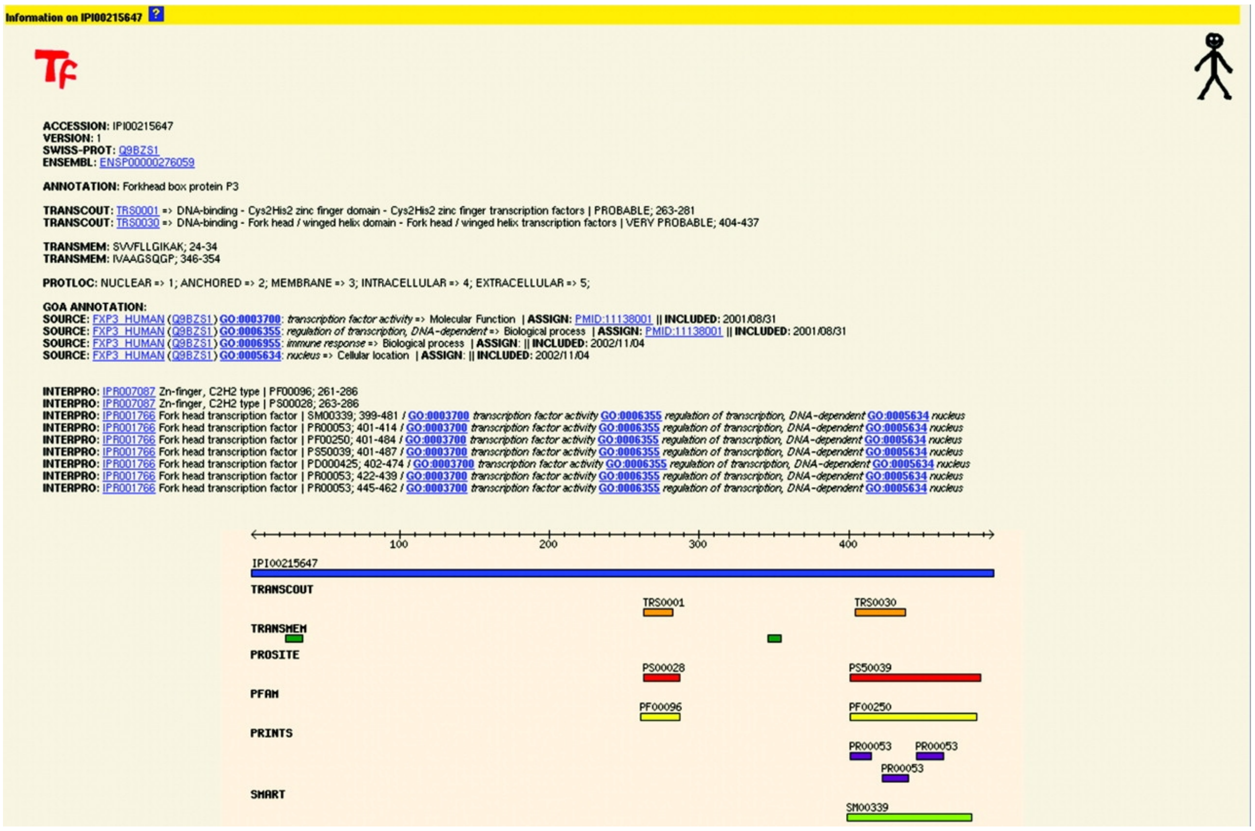
16 entries found.

Page 1 of 1. Items 1-16

IPI00007993.2
TREMBL: O95459 | ENSEMBL: ENSP00000314080 |
Splice isoform 1 of Q14526 Hypermethylated in cancer 1 protein
IPI00019408.1
SWISS-PROT: P51587 | TREMBL: Q9BX99;Q8IU77;Q8IU82;Q9H4L2;Q9H4L3;Q8IU64;Q9H265 | REFSEQ_NP: NP_000050 | ENSEMBL: ENSP00000267071 |
Breast cancer type 2 susceptibility protein Breast cancer type 2 susceptibility protein
IPI00022512.1
TREMBL: Q9H284 |
Serologically defined breast cancer antigen NY-BR-38 Serologically defined breast cancer antigen NY-BR-38
IPI00027266.1
REFSEQ_NP: NP_009228 | ENSEMBL: ENSP00000325574 |
breast cancer 1, early onset isoform BRCA1-delta2-10
IPI00027272.1
REFSEQ_NP: NP_009230 | ENSEMBL: ENSP00000325670 |
breast cancer 1, early onset isoform BRCA1-delta14-17

Figure 16. Detail of a query search output of the organism-based TrSDB browser. Source: (Hermoso et al., 2004)

When clicking one entry, all TrSDB available information about this sequence is shown, ranging from relevant external database crosslinks to matches of used predictive programs. A graphic is available with all the assumed protein signatures located along the sequence. Each matching signature is displayed in a different color together with the associated signature code of the prediction.


Information on IPI00215647

ACCESSION: IPI00215647
VERSION: 1
SWISS-PROT: Q9BZS1
ENSEMBL: ENSP00000276059

ANNOTATION: Forkhead box protein P3

TRANSCOUT: TRS0001 => DNA-binding - Cys2His2 zinc finger domain - Cys2His2 zinc finger transcription factors | PROBABLE; 263-281
TRANSCOUT: TRS0030 => DNA-binding - Fork head / winged helix domain - Fork head / winged helix transcription factors | VERY PROBABLE; 404-437

TRANSMEM: SVVFLLGIKAK; 24-34
TRANSMEM: IVAAGSQGP; 346-354

PROTLOC: NUCLEAR => 1; ANCHORED => 2; MEMBRANE => 3; INTRACELLULAR => 4; EXTRACELLULAR => 5;

GOA ANNOTATION:
SOURCE: FXP3_HUMAN (Q9BZS1) GO:0003700: *transcription factor activity* => Molecular Function | ASSIGN: PMID:11138001 || INCLUDED: 2001/08/31
SOURCE: FXP3_HUMAN (Q9BZS1) GO:0006355: *regulation of transcription, DNA-dependent* => Biological process | ASSIGN: PMID:11138001 || INCLUDED: 2001/08/31
SOURCE: FXP3_HUMAN (Q9BZS1) GO:0006955: *immune response* => Biological process | ASSIGN: || INCLUDED: 2002/11/04
SOURCE: FXP3_HUMAN (Q9BZS1) GO:0005634: *nucleus* => Cellular location | ASSIGN: || INCLUDED: 2002/11/04

INTERPRO: IPR007087 Zn-finger, C2H2 type | PF00096; 261-286
INTERPRO: IPR007087 Zn-finger, C2H2 type | PS00028; 263-286
INTERPRO: IPR001766 Fork head transcription factor | SM00339; 399-481 / GO:0003700 *transcription factor activity* GO:0006355 *regulation of transcription, DNA-dependent* GO:0005634 *nucleus*
INTERPRO: IPR001766 Fork head transcription factor | PR00053; 401-414 / GO:0003700 *transcription factor activity* GO:0006355 *regulation of transcription, DNA-dependent* GO:0005634 *nucleus*
INTERPRO: IPR001766 Fork head transcription factor | PF00250; 401-484 / GO:0003700 *transcription factor activity* GO:0006355 *regulation of transcription, DNA-dependent* GO:0005634 *nucleus*
INTERPRO: IPR001766 Fork head transcription factor | PS50039; 401-487 / GO:0003700 *transcription factor activity* GO:0006355 *regulation of transcription, DNA-dependent* GO:0005634 *nucleus*
INTERPRO: IPR001766 Fork head transcription factor | PD000425; 402-474 / GO:0003700 *transcription factor activity* GO:0006355 *regulation of transcription, DNA-dependent* GO:0005634 *nucleus*
INTERPRO: IPR001766 Fork head transcription factor | PR00053; 422-439 / GO:0003700 *transcription factor activity* GO:0006355 *regulation of transcription, DNA-dependent* GO:0005634 *nucleus*
INTERPRO: IPR001766 Fork head transcription factor | PR00053; 445-462 / GO:0003700 *transcription factor activity* GO:0006355 *regulation of transcription, DNA-dependent* GO:0005634 *nucleus*

Figure 17. Partial entry output showing several crosslinks and all the protein signatures found for this sequence. Source: (Hermoso et al., 2004)

At the bottom of the page, NCBI BLAST premade analyses against the proteomes in the current database (using default parameters and a E-value threshold of 10^{-20}) are available in order to allow users to browse similar entries within the same organism or in others.

Besides proteome browser facilities, TrSDB offers a set of compiled analyses and statistics of TranScout predictions with significant annotation from other resources.

Derived sets of analyses may help researchers to test previously annotated or reannotated proteins that may have a novel and/or conflicting function. A selected list of GO entries are used as a criterion for considering which entries may have been previously annotated as transcription factors by cross-checking against the InterPro (through InterPro2GO) and/or GOA databases (Camon et al., 2003).

Previously poorly annotated entries are highlighted since they are hot candidates for which our predictions could be particularly useful. Furthermore, a list of proteome entries detected as positive by TranScout, which have high similarity (BLAST with default parameters and a E-value threshold of 10^{-20}) to other TranScout positive protein entries in TrSDB are offered for each organism in the database.

Interested users may also download MySQL dumps of core TrSDB data obtained from TranScout, ProtLoc and TransMem runs against considered proteome sets of the current TrSDB version, as well as TranScout classification definitions.

TrSDB relied upon proteome sets, derived InterPro analyses and GOA assignments maintained by EBI facilities (Mulder et al., 2003). Most data are stored and accessed through a MySQL relational database management system. Analyses and interchanging files are generated and stored in a suitably parseable way in specially formatted text or XML format. Many scripts for carrying out analyses, linking components and handling the web interface use BioPerl tools and modules

3.4.3. Possible future approaches

As commented before in the document, at the time of writing, TranScout has not been recently updated to include any newer categorised transcription factor family. Moreover, being TransFac a licensed resource, it may be argued that there may a need to provide an updated Transcription Factor database based on already existing open-access data and with the new experimental and theoretical data from long-range chromosome interactions (Bartkuhn et al., 2008). A recent example resource going into that direction would be HOCOMOCO (Kulakovskiy et al., 2013), which would be based on TransFac (as TrSDB) and specialy JASPAR (Mathelier et al., 2014), that seems to be fully open-data based.

3.5. ArchDB: Functional Annotations and Structural Classifications of Protein Loops

3.5.1. Introduction

Loops represent an important part of protein structures. The study of loop is critical for two main reasons:

1. Loops are often involved in protein function, stability and folding.
2. Despite improvements in experimental and computational structure prediction methods, modeling the conformation of loops remains problematic.

In the following pages, a structural classification of loops is presented, ArchDB, a compendium of information with application in both mentioned fields: loop structure prediction and function prediction.

ArchDB is a database of classified protein loop motifs. At the time it was first designed, the database provided four different classification sets tailored for different purposes. ArchDB-40, a loop classification derived from SCOP40, well suited for modeling common loop motifs. Since features relevant to loop structure or function can be more easily determined on well-populated clusters, another set, ArchDB-95, was developed, which is a loop classification derived from SCOP95.

Other loops classifications are presented, ArchDB-EC, a classification of loop motifs from enzymes, and ArchDB-KI, a manually annotated classification of loop motifs from kinases. Information about ligand contacts and PDB sites has been included in all classification sets.

The lengths of classified loops range between 0 and 36 residues long. ArchDB offers an exhaustive sampling of loop structures. Functional information about loops and links with related biological databases are also provided. All this information and the possibility to browse/query the database through a web-server outline a useful tool with application in the comparative study of loops, the analysis of loops involved in protein function and to obtain templates for loop modeling.

3.5.2. Biological context

In a protein structure, loops are the regions of non-repetitive conformation connecting regular secondary structures, namely α-helices and β-strands. Loops are involved in protein function, stability and folding (Fetrow, 1995), and so represent an important part of protein structures. They can play a wide repertoire of roles related to protein function: (i) recognition sites Complementary Determining Regions (CDRs) (Kim et al., 1999), (ii) protein-protein interactions: signaling cascades (Bernstein et al., 2004), dimerization (Fritz-Wolf et al., 1996), PDZ-motifs (Feng et al., 2003), (iii) ligand binding (p loop (Saraste et al., 1990) EF-hands (Kawasaki et al., 1995), Nicotinamide adenine dinucletotide phosphate (NAD(P)) binding loops (Wierenga et al., 1986), glycin-rich-loop (Schenk et al., 1999)), (iv) DNA-binding (helix-turn-helix motifs (Tainer et al., 1995), M13 phage (Coleman et al., 1986)); (v) forming enzyme active sites (e.g. Ser-Thr kinases (Johnson et al., 1998)) or serine proteases (Wlodawer et al., 1989)). Moreover, loops play a vital role in correctly positioning catalytically important residues (Gunasekaran et al., 2003).

Experimental and theoretical evidences suggest that local structural determinants are frequently encoded in short segments of protein sequence. Local sequence-sequence-structure relationships derived from local structure/sequence analyses could significantly enhance the capacities of protein structure prediction methods (Yang et al., 2003). The reports of Shindyalov and Bourne (Shindyalov et al., 2000) and (Tendulkar et al., 2004) suggest that folds are mainly made up of a number of simple local units of super-secondary structures, formed by few secondary structures connected by loops.

There is a large difference between known protein sequences (Bairoch et al., 2005) and protein structures (Berman et al., 2000). In the absence of an experimentally determined structure, *ab initio* and threading methods or comparative modeling methods can sometimes provide a useful 3D structure of a protein (Baker et al., 2001). Nevertheless, the recent improvements on the performance of fold prediction and homology modeling methods in successive CASP (Critical Assessment of Structure Prediction) experiments (Venclovas et al., 2005)have not proved to be as successful as in loop model building. In general, these methods tend to correctly predict the protein core but not the loop regions. Errors in loops are the dominant problem in comparative modeling and often are the most difficult parts to model (Fiser et al., 2000).
Thus, a database of structurally classified protein loops will have widespread applications (i.e. in model building or to complete locally undefined regions from an X-ray diffraction map).
The impact of loop modeling is significant.

According to (Pieper et al., 2004), approximately 60% of all protein sequences can have at least one domain modeled on a related, known protein structure. At least two thirds of the comparative modeling cases are based on less than 40% sequence identity between the target and the templates, and thus generally require loop modeling (Sánchez et al., 1997).

Structural genomics initiatives attempt to infer details of protein function via 3D structure determination (Shapiro et al., 2000). If a new protein structure adopts a previously observed fold, functional details might be inferred by considering the function of other proteins adopting the same fold (Dietmann et al., 2002).

If fold similarities are ambiguous or if a protein adopts a new fold, it is still possible to infer function by comparing key active site residues (Russell et al., 1998). Common structural motifs contain particularly useful information on the conservation of specific residues across species, being occasionally involved in the protein function (i.e. the activation loop of some kinases) or in the folding nucleus (Mirny et al., 2001).
Several works in loop classification had been published before (Burke et al., 2000; Donate et al., 1996; Oliva et al., 1997, 1998; Wintjens et al., 1996; Wojcik et al., 1999). However, these

classifications were not web accessible or updated regularly. ArchDB (Espadaler et al., 2004) has been updated since its creation, and the new version presented here includes three new classifications: ArchDB-95, ArchDB-EC and ArchDB-KI, plus the added value of functional annotations. The web-server, accessible at http://sbi.imim.es/archdb.

3.5.3. Material and Methods

3.5.3.1. PDB sets and loops extractions

The considered version of ArchDB contains 4 different types of loop structure classification of loops, namely: ArchDB-40,ArchDB-95,ArchDB-EC and ArchDB-KI each of them extracted from a different set of structures. ArchDB-40 is based on a list of protein domains of SCOP 1.67 (Lo Conte et al., 2002) with less that 40% sequence identity. ArchDB-95 is based in SCOP 1.67 using sequences with identity smaller than 95%. The two lists of protein domains were downloaded from ASTRAL compendium (Chandonia et al., 2004). ArchDB-EC is derived from a set of structures with known Enzyme Commission (EC) number (Kotyk, 1999) downloaded from http://www.bioinf.org.uk/pdbsprotec/ (Martin, 2004). The program cd-hit (Li et al., 2002) was used to obtain a set of chain with less than 95% sequence identity. Finally, ArchDB-KI is derived from a set of structures with EC number 2.7.X.X (transferring phosphorus-containing groups).

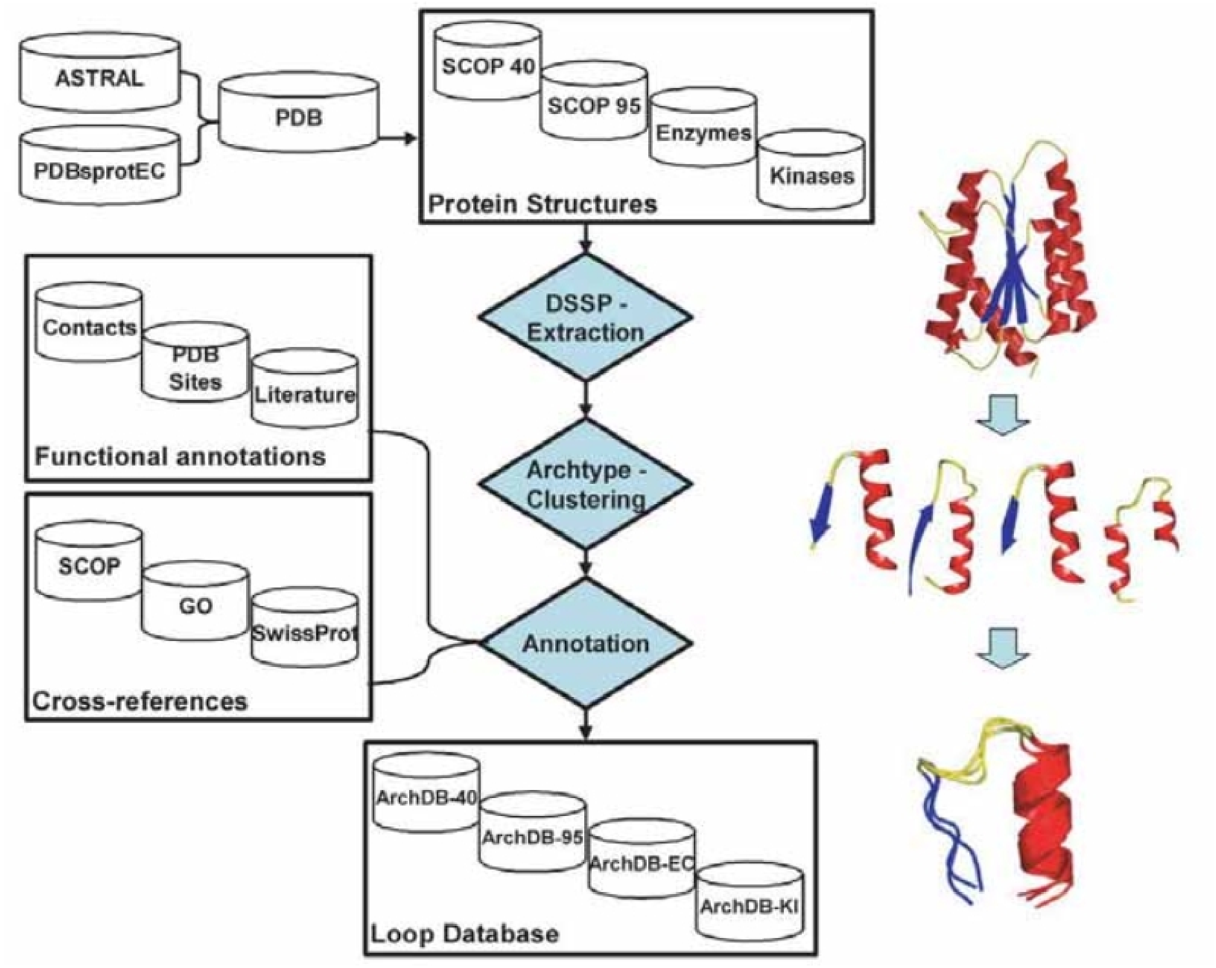


Figure 18. Overview of construction and annotation process of ArchDB. Four different PDB datasets were constructed to derive the four different classifications of loops. The process of the building of the database includes the extraction of loops, their clustering, etc. Source: (Hermoso et al., 2007)

The process of construction of the loop classifications is similar for the four sets included in ArchDB. First, structures not obtained by X-ray crystallography or with resolution larger than 3.0 Å are removed from the initial sets. The DSSP (Dictionary of Protein Secondary Structures) program (Kabsch et al., 1983) is used to locate loop segments, defining loops as fragments between any two regular secondary structures. The initial dataset of loops is further filtered by a quality rule: no loops were considered with missing residues or missing main chain atoms (including C_β, except for Glycine).

3.5.3.2. Clustering process

Loops extracted in the previous step are clustered according to structural similarity. The structural clustering of loops is obtained with an improved version (Espadaler et al., 2004) of the Arch-Type program (Oliva et al., 1997). In short, the clustering algorithm is based on a geometry comparison of the flanking secondary structures and on a density search on the [φ, ψ] space of the loop conformation. Geometry is defined by four internal coordinates of flanked secondary structures, a distance, D, between ending points and three angles: hoist, packing and meridian. Two loop motifs share the same geometry if Δ (D, hoist, packing, meridian) belongs to the four-dimensional semi-open interval I = ((0, 0, 0, 0), (2, 45, 45, 45)] (Fernandez-Fuentes et al., 2006). The possible conformations of the loop fragment were defined by assigning the most accessible regions in [φ, ψ] space. The regions are α, $α_λ$, γ, β, $β_p$ and ε (encoded by ArchType as "a", "l", "g", "b", "p" and "e"). Two special regions denoted "l/g" and "b/p" are defined as transition regions between the l and g conformations and between the b and p conformations, respectively. For a pair of loops, a conformational similarity score is obtained as the percentage of the total number of residues that can be equivalent with identical conformational codes.

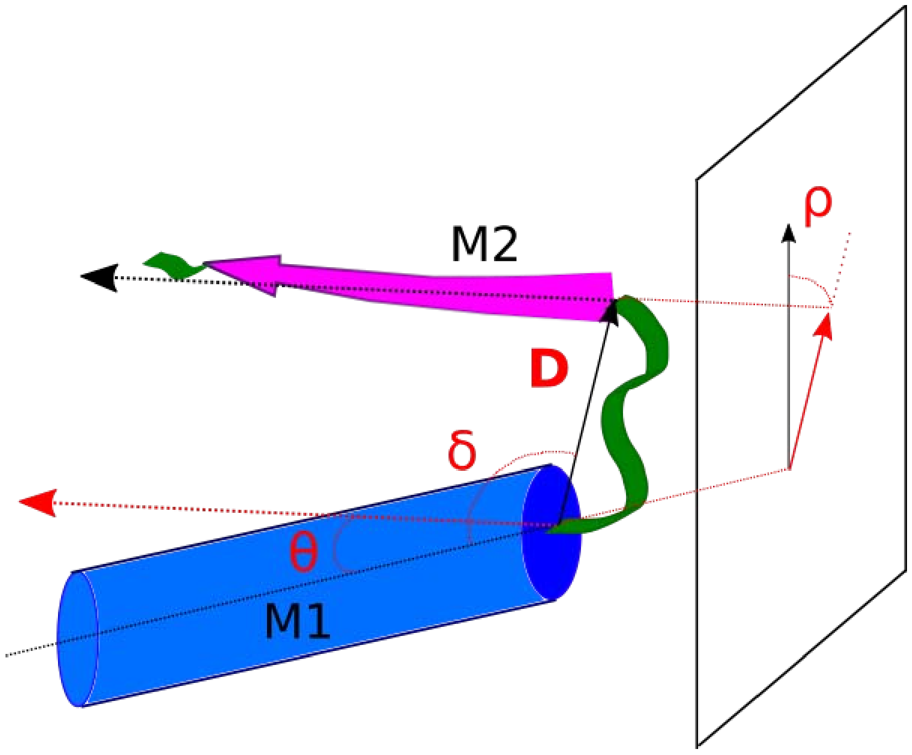


Figure 19. Graphical representation of relevant loop geometry variables. D: loop distance, θ: packing angle, δ: hoist angle, ρ: meridian angle.

Owing to the ±1 residue extension in loop length definition allowed because of the difficulty in defining the termini of the secondary structures and to the wide definition around [φ, ψ] regions in "l/g" and in "b/p" conformations, loops can cluster into more than one group. A re-clustering protocol has been devised to deal with the overlap between clusters. Overlapping clusters are merged depending on the percentage of shared loops. A cluster-membership p-value is calculated for each loop motif (see below Statistic significance of clusters). Overlapping clusters are merged if they have more than 80% of loops or if there is a common loop with membership p-value < 0.002 to both clusters. Averaged coordinates are recalculated and the process is repeated until convergence of the classification. The result is an optimized partition of the conformational space of loops that joints clusters (as obtained in Arch-Type (Oliva et al., 1997)) that contain structurally similar loops and a minimum overlap between subclasses.

3.5.4. Outcomes

3.5.4.1. Organization and annotation

ArchDB was structured into four levels of hierarchy: (i) at the *classification level*, there are links to the four loop classifications included in ArchDB: ArchDB-40, ArchDB-95, ArchDB-EC and ArchDB-KI; (ii) at the second level of the classification, loops were identified according to the bracing secondary structure type: α–α loops α–β loops, β α loops and β–β loops that are further split into β βhairpins (which are those loops between two β strands with at least one hydrogen bond between both strands) and β–βlinks, the complementary set in β–β loops; (iii) at *class* level, loops are grouped according to the loop length and [φ–ψ] loop conformation; and (iv) at *subclass* level the classes are subdivided according to the orientation of secondary structures or motif geometry. Each subclass is identified in ArchDB by a three-number code as defined in the original paper (Oliva et al., 1997). For instance, a subclass with a classification code αβ4.1.1 means that: it belongs to type α–β, it is the most populated class αβ4.1 with loops of length 4 ± 1 and it is the most populated subclass αβ4.1.1.

Subclasses were classified as *putative structure/function-related subclasses* (PSFRS) or *functional subclasses* according to the degree of conservation of the annotations (DCA). The considered annotations have been obtained from:

(i) SCOP identifiers;
(ii) GO terms;
(iii) EC codes among the original PDB chains;
(iv) ligand contacts, i.e. residues found within a cut-off distance of 6Å from an hetero-atom, ligand, inhibitor, cofactor or complex partner molecule (protein or DNA) with the exception of D_2O or crystallization buffer molecules;
(v) PDB site information (residues identified by functional information from ACTSITE and SITE records in the PDB file header); and
(vi) residues identified by the functional annotation collected from the literature and assigned to specific motifs (only for ArchDB-KI).

The functional annotation process is as follow. Each loop is annotated by its SCOP, EC and GO number. The conservation of these annotations is explored among the loops included in the same subclass.

Three groups of DCA were defined: <50% conservation, between 50 and 75% conservation and >75% conservation. We define a subclass with more than 75% conservation of a given annotation as PSFRS. In case of ArchDB-KI, subclasses are considered functional subclasses when there is a meaningful conservation of functional residues in the loops of the cluster and more than 50% of its loops belong to proteins of the same SCOP superfamily. Besides the quantitative conservation of the SCOP, EC and GO numbers, a qualitative measure of potential function is also given if any loops included in the subclass have any annotation extracted from the PDB header (annotated as ACTSITE and SITE) and/or contacts with ligands.

3.5.4.2. Content and web interface

Users can browse through ArchDB data-sets or perform queries searching for loops motifs satisfying particular features:

i. Belonging to a PDB structure by specifying the PDB identifier or SWISS-PROT accession code;
ii. Browsing through ArchDB levels: i.e. classes and subclasses;
iii. Loop with particular bracing secondary structures type and geometry, loop size or loop [φ, ψ] conformation;
iv. Loops with a specific SCOP family, super-family and fold, SWISS-PROT keywords or GO accession codes;
v. Loops from subclasses with residues in contact with ligands and/or with PDB SITE annotations (and with bibliographical annotations for ArchDB-KI);
vi. PSFRS with DCA > 50%, between 50% to 75% or DCA > 75%;
vii. Sequence search. The search is performed on the selected classification using BLOSUM 62 as mutation table to calculate the sequence score;
viii. Classes with the same conformation and subclasses with the same geometry and/or conformation of the loops of an uploaded protein structure. Structural classes and subclasses are assigned comparing the loop geometries and conformations of all the loops of an uploaded protein structure with the loops from the database. Secondary structure and loops of the uploaded coordinates of the query protein are defined with DSSP (Kabsch et al., 1983).

Points (iii) and (vii) will allow the user to obtain potential templates for loop modeling, as well as retrieving functional information about similar loops to check whether our loop could play a functional role or not. Analogously, for non-clustered motifs (single member subclasses), information described in points (iii), (v) and (vi) can also be retrieved. However, not all the structures classified in PDB databan) are represented in ArchDB. If a structure is not present in our classification, the PDB code(s) of the closest protein(s) in homology (i.e. the smallest e-value and the largest percentage of identity as aligned by PSI-BLAST) are shown.

Other type searches can be the list of motifs found in a given PDB structure, the list of subclasses satisfying specific features or the content of a given subclass. Structural and

functional information for each PDB structure is accessible, including resolution, R-factor, PDB source, GO annotation, Enzyme annotation, and the SCOP domain classification.
For each subclass, a table describing consensus features (sequence, geometry, percentage of sequence identity, averaged RMSD and its standard deviation) can be obtained. Additional information includes a PROSITE-like pattern (Falquet et al., 2002) with calculated position-specific entropy (Pei et al., 2001) and a BLOSUM-like PSSM (Position-Specific Scoring Matrix) profile obtained with the multiple sequence alignment.

3D Images of superimposed motifs and averaged coordinates can be viewed using any molecular visualization program, such as Jmol (The Jmol Team, 2007), that can handle atomic coordinates in PDB format.

Users can download coordinates for superimposed motifs or the average structure, which may be useful for loop reconstruction. Multiple alignments of sequences, secondary structures and [φ/ψ] conformations of the loops are provided. Information about residues in contact with ligands and residue with PDB site annotations (and with bibliographic annotations for ArchDB-KI) are also given, if any.

Finally, ArchDB is cross-linked to other important databases such as Protein Data Bank (PDB) (Berman et al., 2000), GO (Ashburner et al., 2000), SWISS-PROT (Bairoch et al., 2000) and SCOP (Lo Conte et al., 2002).

3.5.5. Discussion and conclusions

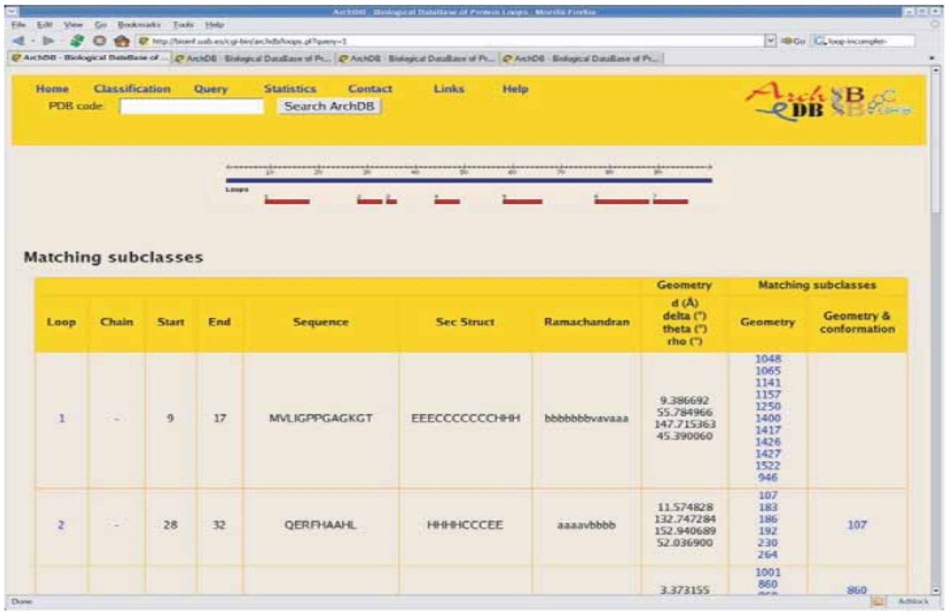
The two major motivations for this study are: (i) to help to predict loop conformation in comparative modeling and, (ii) the availability of a functional annotated loop classification for the study of loops.

We provide a classification of the conformation of loops with their associated sequence patterns and a PSSM profile for each structural alignment; together with the ability to search ArchDB database, provides a powerful tool to analyze loops in protein sequences. We have proved the usefulness of sequence profiles in loop structure prediction (Oliva et al., 1998)

Instead of searching for potential templates, users can be interested on functional annotated loops that are related with its query sequence by browsing among the functional annotations of the subclasses delivered with the sequence search.

Functional annotated subclasses may help in the central problem of protein annotation. When sequence or structure comparisons fail to suggest a function, insights can come from discovery of functionally important local structural patterns. A subclass is a set of conserved local structural patterns. Conserved short stretches of amino acid sequences or motifs contain useful information on the conservation of specific residues involved in the protein function (catalysis or binding) or in the folding nucleus. The analysis performed on ArchDB-40 showed that up to 35% of active site residues are located in loops.

Functional annotated subclasses can be used to search for matches of loops in a newly determined structure and thereby suggest putative function or bindings. It can be of special interest given the pace of structures production on structural genomic initiatives worldwide, where functional insights can come from discovery of functionally important local structural patterns. For that reason, we created ArchDB-*EC*, a subset of ArchDB restricted to structures from proteins with known enzymatic function. ArchDB-EC is aimed at users focusing on loops involved in active sites. We expect this subset to be of interest when searching for loops with catalytic roles in protein structures. Figure below shows an example of a search using the loops extracted from a structure (noted as feature (viii) at the Browsing and Querying section). After uploading a protein structure, ArchDB extracts all loops and structurally compare with the classes and subclasses (and single loops if selected) classified. Users can easily explore and browser the results and assess the significance of the results to their specific queries. In addition, this type of search yields all possible loop conformations that bridge two secondary structures. Users could be interested on comparing its own loop conformation with alternative ones (i.e. structural models, alternative loop conformations in catalytic/mobile loops, etc.).



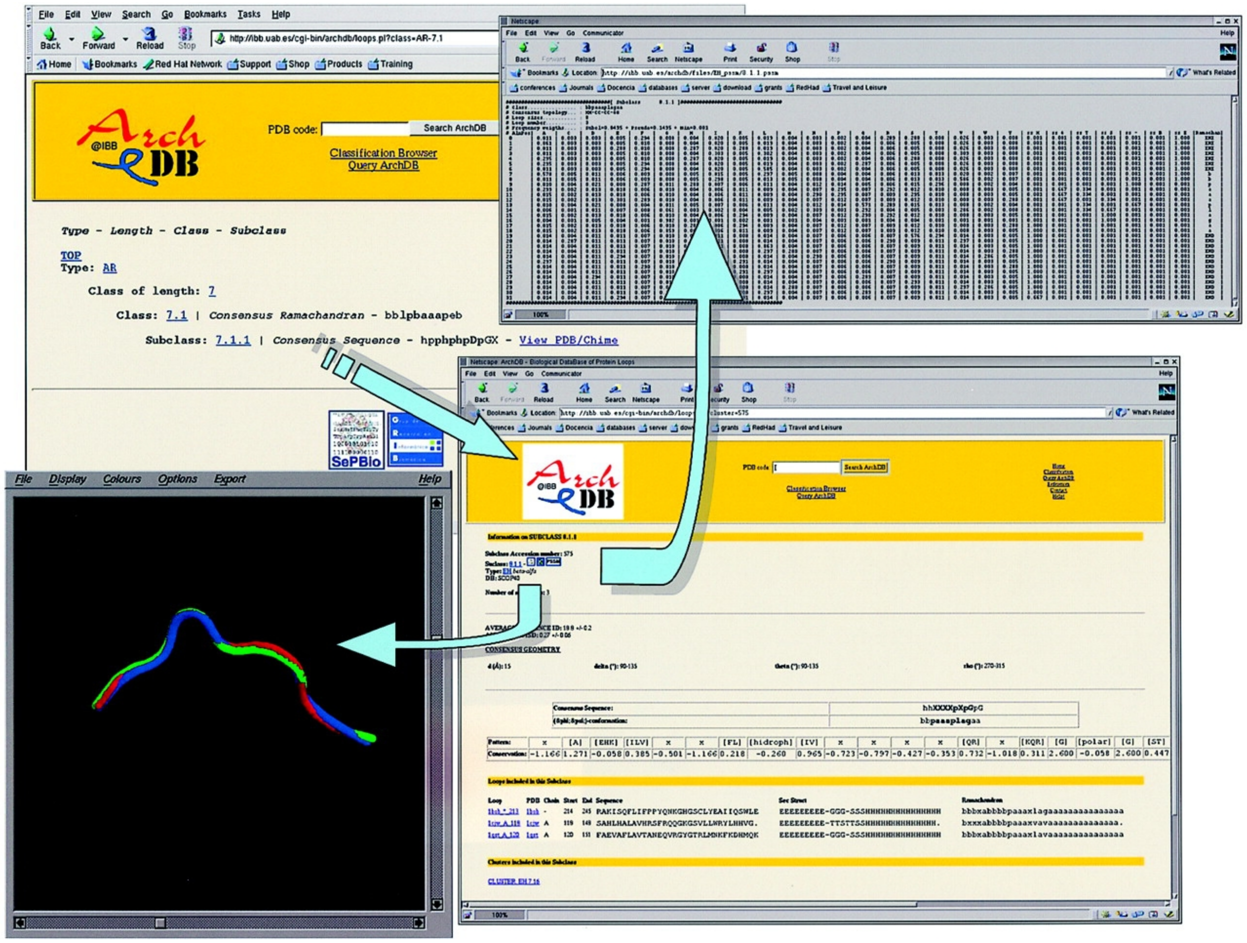
Matching subclasses

							Geometry	Matching subclasses	
Loop	Chain	Start	End	Sequence	Sec Struct	Ramachandran	d (Å) delta (°) theta (°) rho (°)	Geometry	Geometry & conformation
1	-	9	17	MVLIGPPGAGKGT	EEECCCCCCCHHH	bbbbbbbvavaaa	9.386692 55.784966 147.715363 45.390060	1048 1065 1141 1157 1250 1400 1417 1426 1427 1522 946	
2	-	28	32	QERFHAAHL	HHHHCCCEE	aaaavbbbb	11.574828 132.747284 152.940689 52.036900	107 183 186 192 230 264	107
							3.373155	1001 860	860

Figure 20. A snapshot of ArchDB website showing an example of a search using atomic coordinates. Loops are assigned using DSSP (Fernandez-Fuentes et al., 2005) and its location in the sequence is shown. Matching subclasses by loop geometry and matching subclasses by geometry. Source: (Hermoso et al., 2007)

On the other hand, the search using protein structures can be also used for loop modeling. All subclasses that fit the geometry of the adjacent secondary structures of a motif can be retrieved from ArchDB. Consequently, for a missing or wrongly modeled loop region, users can download the atomic coordinates of the subclasses and superimpose them to the known framework.

Broken or missing loops are shown as '*-loop incomplete-*' at the result table if the loop region was missing while a list of compatible subclasses according to motifs-geometry is provided. This feature is also applicable in case of structural models, namely structures predicted by computational means. Users might be interested on searching for loops that can span a fixed core (i.e. secondary structure elements) obtained by comparative modeling, threading, or an *ab initio* prediction.



http://ibb.uab.es/cgi-bin/archdb/loops.pl?class=AR-7.1
ArchDB
PDB code:
Search ArchDB
Classification Browser
Query ArchDB
Type - Length - Class - Subclass
TOP
Type: AR
Class of length: 7
Class: 7.1 | Consensus Ramachandran - bblpbaaapeb
Subclass: 7.1.1 | Consensus Sequence - hpphphpDpGX - View PDB/Chime
SePBio
File Display Colours Options Export Help
Information on SUBCLASS
CONSENSUS GEOMETRY
Consensus Sequence:
hhXXXXpXpGpG
Loops included in this Subclass
Clusters included in this Subclass

Figure 21. Overview of the different screens of ArchDB. Source: (Espadaler et al., 2004)

The database, consisting on an exhaustive classification of loop structures, was composed of four different classifications customized for specific requirements and includes functional annotations. Moreover, a flexible search engine was designed that allowed the querying/browsing of the database in a number of ways, either using sequence, structure, and feature-based information. All this classified data and the wide range of possibilities of the search engine shapes a powerful tool with applications in different areas of biological sciences and bioinformatics.

3.5.6. Current status

At the time of writing this document, a major update of ArchDB was published (Bonet et al., 2013) with a 5-fold increase in the number of loops considered. Compared to what is explained above, precisely because of computational restrictions posed by the included loops, clustering is also performed thanks to Markov Clustering (MCL) algorithm (Van Dongen, 2008), but density search classification is still kept for back-compatibility.

3.6. Design and setup of a sequence analysis management system

3.6.1. Introduction

In the previous pages we detailed a few projects where different analyses with a considerable amount of data were involved. In some of the cases, it leaded to the creation and publication of the results in the form of an online public database that could be accessed and queried by the whole scientific community. In contrast, in the next pages, we will elaborate a slightly different approach. An initial web framework will be set-up that might be flexible enough to accommodate different kind of analyses and, at the same, both linking initial data sources and final results which, in turn, can be accessed as any common web resource.

The core idea is making possible a coherent way of managing biological data, processes and results within the same location, so they can be easily shared (via a URL).

The considered biological data are biological sequences (mostly protein ones), and the processes, that is, analyses, are essentially PSI-BLAST and Bypass ones. The later program, designed and built in the IBB (Gómez et al., 2008), is discussed in more extensive detail beforehand.

3.6.2. Bypass: A fuzzy logic approach for predicting protein function

The advent of Genomics yielded thousands of reading frames without a function Typically, from the initial analysis of sequenced genomes 20%-60% of the ORFs cannot be functionally annotated. For example, the human genome contains about 40,000 genes, roughly 50% of which can be assigned a putative biological function on the basis of sequence similarity to proteins with a known function (Venter et al., 2001).

The exponential growth of sequence databases and the difficult task of the functional annotation of the new protein sequences (Lander et al., 2001) represent a major drawback not only for genomic science but also for its biotechnological and biomedical applications such as DNA arrays, drug target identification, filing a patent of a protein sequence (because, of the three requirements for patentability, the most elusive, “Utility”, depends on the biological function of the protein).

Therefore, *in silico* prediction of the function of a protein has become a key objective for most of the “omics” science and derived biotechnology areas (Gilks, et al., 2005). For example, there are other algorithms and programs which help to predict function from protein sequence, such as GeneQuiz (Ouzounis et al., 2002), ProtFun (Hoersch et al., 2000).

One of the best procedures to computationally predict the function of a protein from its sequence is by finding a related protein with high sequence similarity for which biological information is available. This is performed by means of alignment algorithms, such as, BLAST (Jensen et al., 2002), FASTA (Altschul et al., 1990), PSI-BLAST (Pearson et al., 1988) or HMMER (Finn et al., 2011). The main problem comes out when analyzing and interpreting sequence results in order to infer functional similarity from homology, even more so when the proteins have limited sequence similarity (remote homologues) (Altschul et al., 1997). Analysis and interpretation of the results remains a major challenge and typically requires a case-by-case analysis of the output (Devos et al., 2000). A typical search of similar sequences by alignment algorithms displays a sometimes-long list of protein sequence stretches presenting significant similarity to different protein targets, ordered from the top by their respective E-values. However, statistical and biological significance are not equivalent and, sometimes, important biological matches can have relatively low scores.

As Tian and Skolnick stated, the E-value is not a good measure for transferring function from sequence, specially when multiple iterations of PSI-BLAST are executed. It is been frequently observed that function starts to diverge quickly and homologous proteins may evolve to different functions, especially when sequence identity falls below 40%. For example, using a permissive threshold for inclusion in a profile may cause the inclusion of unrelated sequences in the profile and lead to diversion from the original sequence (Tian et al., 2003). Nevertheless, although difficult to identify these true positives, it is a very important fact that upon running a PSI-BLAST search, usually a true positive for the function of a query protein in some position of the output can be found

A way to facilitate the functional annotation of a query protein sequence would be the rearrangement of the PSI-BLAST program output according to characteristics other than non directly sequence derived parameters, trying to find the true functional targets.
When no experimental evidence is available, methods based only on information derived from sequence are required (Yona et al., 2002). This is actually the approach of Bypass (Gómez et al., 2008).

Since long ago, it is known that protein sequence hides characteristics related to its structure and function. In the present work several protein sequence characteristics have been checked in relation to their ability for helping to rearrange a PSI-BLAST output, raising lower E-value matches to the top positions, to their biological functions. Four characteristics were found to contribute to the accuracy of the functional prediction of proteins:

- hydropathic profile
- flexibility profiles
- amino acid content
- length of the matched sequence hit

A multiple parameters scoring function with these characteristics is used. The parameters may involve very different representations from numerical and their relative importance.

3.6.2.1. Sequence Characteristics

Amino Acid Composition

In order to use the amino acid composition of a protein in a numerical value, we used the Euclidean distance as a classification tool. Euclidean distance is a dissimilarity index. It evaluates the difference in terms of amino acid composition between the query protein and each subject protein from the database.

$$d_{xy} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

where ***x*** is the amino acid frequency of one amino acid ***i*** and ***j*** in the subject protein and query protein.

Segment Length

For each homologue to the query protein, its length in terms of number of amino acids is reported.

Hydropathic Profile

For each homologue to the query protein, a hydropathic profile is calculated according to Kyte and Doolittle algorithm (Kyte & Doolittle, 1982). The hydropathic profile of a protein is calculated by assigning each amino acid a numerical value (‘‘hydropathy index’’) and then repetitively averaging these values along the peptide chain. The values assigned to each amino acid are the Kyte–Doolittle values. The window length over which the hydropathy indices are averagedmust also be set.

Flexibility Profile

For each homologue to the query protein, the flexibility profile is calculated according Karplus and Schulz (Karplus & Schulz, 1985; Vihinen et al., 1994) as follows: In a sliding window along the sequence, the local flexibility if 7-amino acid stretches and averages them. The individual measure for each amino acid is based on crystallographic temperature factors. The numerical values for these properties are introduced as inputs in the fuzzy logic algorithm.

3.6.2.2. The Fuzzy Logic algorithm

The fuzzy logic algorithm takes the decision of combining these four parameters: Amino acid composition, Segment length, Hydropathic profile and Flexibility profile, and giving a fuzzy score for each protein. These data are converted to fuzzy data in a process called “fuzzyfication” converting numerical parameters into linguistic values (“low”, “medium” and “high”), which can be viewed as fuzzy sets or membership functions.

From a protein sequence, fuzzy logic integrates several features from sequence data into single rules in order to include as much as possible protein information from a remote homology search. The fuzzy scores for the four above different parameters from a protein sequence are then combined to give a single score using a fuzzy inference algorithm which reflects the relative importance of each criterion in a similar way than a human judgment does. A decision matrix is constructed with fuzzy rules in order to compare the data included in these rules. This is process is called the “decision-making logic”, which simulates human decision-making and which infer fuzzy control actions employing fuzzy implication and the rules of inference in fuzzy logic. The inference algorithm combines scores sequentially by pairs: first the Amino acid composition and the Segment length (Numerical parameters) into a common score and then, the Hydropathic profile and Flexibility profile (Profile parameters) into a second composite score. Finally, these two composite scores are combined into a single score for all the parameters. No training set are needed because the fuzzy inference engine uses the rules of maximum value and of minimum value to make the combinations of them and automatically give the weights to each one.

Finally, in a process called “defuzzification”, the qualitative fuzzy data is transformed to a numerical value (in a scale from 0 to 100) that is used for the rearrangement of the initial PSI-BLAST profile. Thus, sequences from lower scores in the initial PSI-BLAST profile can “Bypass” the others and climb to top positions, suggesting another function for the query protein.

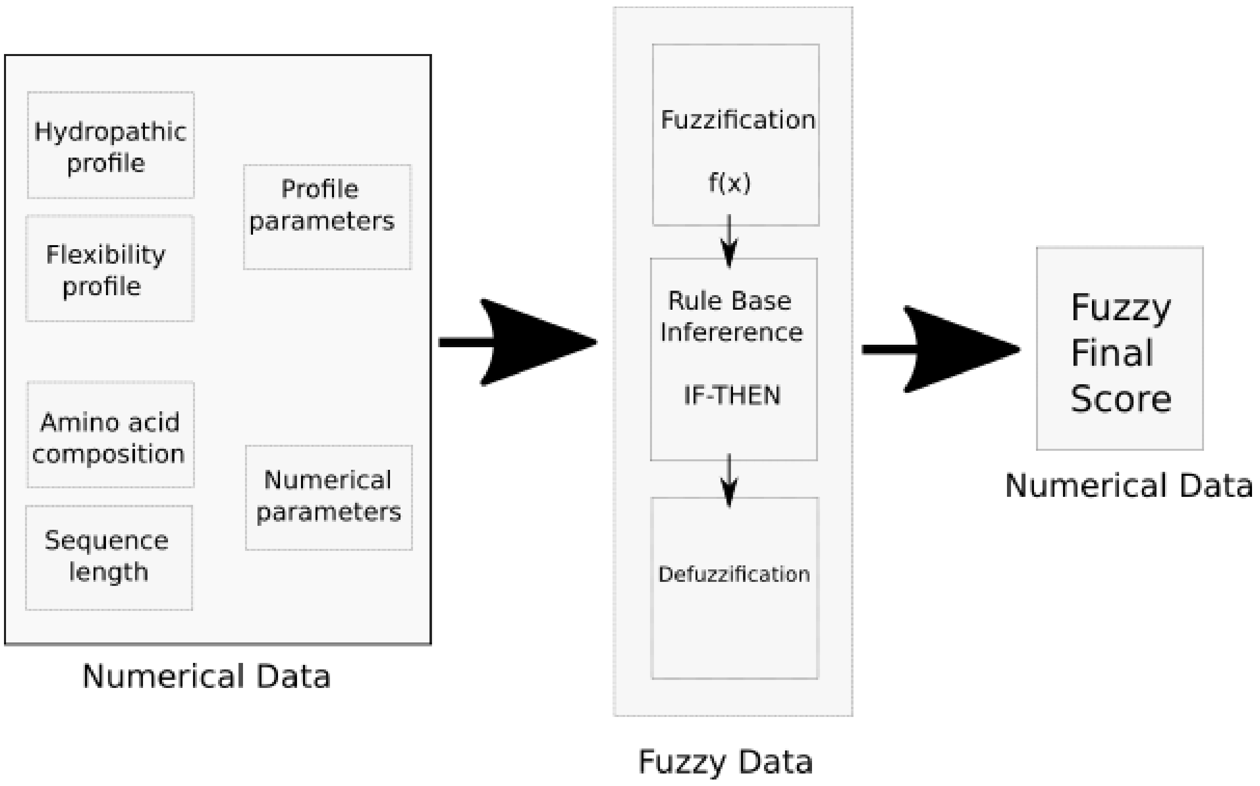


Figure 23. Flowchart describing the Fuzzy Algorithm. First, the numerical data obtained from calculations (i.e. hydropathic profile, amino acid content, etc) are divided into numerical and profile parameters. Once they are introduced in the algorithm, they are fuzzified in order to obtain fuzzy data that can be applied to the controller. The output fuzzy set is defuzzified obtaining a crisp numerical data that can rearrange the numerical PSI-BLAST profile. Based on: (Gómez et al., 2008)

3.6.2.3. Evaluating the accuracy of the prediction

In order to evaluate the Bypass sensitivity in finding true positives in a non redundant database, we used a database of sequences with less than 40% identity derived from ASTRAL SCOP 1.67 (King et al., 2004), that is ASTRAL 40% PDB subset (http://astral.berkeley.edu.). SCOP entries were excluded with sequences shorter than 25 residues length, errors in their header annotation or if they included the terms “hypothetical protein”.

The involved procedure was:

1. compare every sequence to the nr protein database (www.ncbi.nlm.nih.gov) and save the matrix computed after the fifth PSI-BLAST iteration, or when the program stops because no further similarities are found. (E-value threshold for inclusion in the next iteration was 0.005, BLOSUM 62 matrix);
2. each saved PSSM is used to search against the Astral SCOP1.67 database using the restart option of PSI-BLAST;

3. amino acid sequences of the hits above the selected threshold in the PSI-BLAST profile generated are extracted from every pairwise alignment and analyzed according to their properties: Amino acid composition, Segment length, Hydropathic profile (Brenner et al., 2000), and Flexibility profile (Kyte & Doolittle, 1982);
4. the numerical values for these properties are introduced as inputs in the Fuzzy Logic algorithm.

The procedure involved measuring the number of true positives found at 100% specificity (that is, before the first false positive appears) and at 99% specificity. ASTRAL 40% PDB is used at two levels, Fold level and Family level; so, we consider true positives if two proteins belong to the same family or share the same fold. Table I shows that Bypass gives a higher number of true positives before the first false positive appears than PSI-BLAST. These results show the improvement obtained by Bypass improves PSI-BLAST in removing false positives that will corrupt the PSSM profile in subsequent iterations.

The principal improvement of Bypass versus PSI-BLAST is to remove the false positives in the first positions, so the usual ROC curve is not the best way to show the accuracy of the method, because it plots the True positive rate versus the False positive rate. In our case, we want to show only the removal of false positives from high score positions. So to assess the accuracy of Bypass and PSI-BLAST on the data sets, Specificity vs. Score curves are used. It is also known as True positive rate before the first false connection.

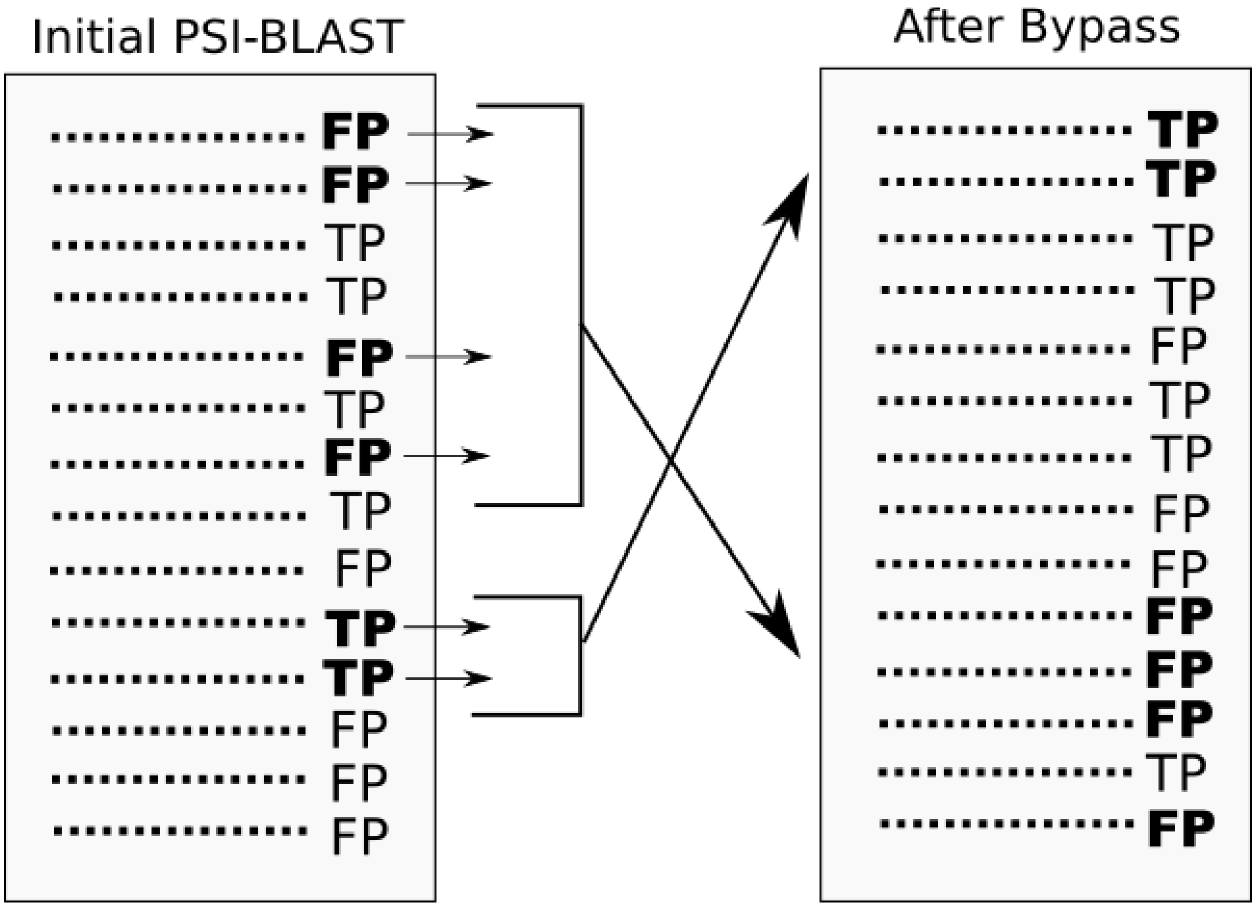


Figure 24. Effect of Bypass removing false positives in high positions of PSI-BLAST output and moving up of true positive hits located in lower positions. Higher positions of the output correspond to higher levels of specificity, lower levels correspond to specificity low values. Based on: (Gómez et al., 2008)

Several examples will show different results and situations selected from a set of proteins related to human diseases for their biomedical relevance whose function had been previously experimentally established was used. Two larger tables can be found in the web server with more examples of proteins whose functions have been properly predicted by Bypass. Showing some examples of previously hypothetical or incorrectly bioinformatically predicted functions (prior to the experimental determination of the function). When PSI-BLAST has applied to these sequences no experimental data of the protein function were available and the obtained prediction using the top hits of the PSI-BLAST was full of hypothetical, ambiguous or poorly annotated hits. But in the same output there were others well-annotated hits with high E-values.

3.6.2.4. Examples

Running Bypass, the user generally finds one of different situations (Gómez et al., 2008):

a) PSI-BLAST output has detected the protein function but in low positions

The McKusick-Kaufman syndrome (MKKS) and the Bardet-Biedl syndrome correspond to a rare autosomal recessive genetic disease with classic hexadactyly and hydrocolpos in females, and sometimes cardiac abnormality. Both are inherited in an autosomal recessive pattern caused by mutations in the MKKS gene**.** The protein has similarity to a *Thermoplasma acidophilum* chaperonin. The PSI-BLAST analysis shows in the first hit an (AK025741) unnamed protein product, in the next positions a group of MKK proteins of non determined functions from different species was found. The Bypass rearrangement moves a chaperonin up, from the 88th position in the original PSI-BLAST output, to the top position (AF226725).

b) PSI-BLAST output depicts a huge number of hypothetical/putative proteins and Bypass rearrange the set, moving the right one up to top positions and suggests a function for the whole set

The Smith-Lemli-Opitz syndrome (SLOS), or RSH syndrome, is a multiple congenital mental retardation condition due to the deficient activity of the enzyme 7-dehydrocholesterol (7-DHC) reductase. PSI-BLAST shows several functionally unknown proteins at the top matches. Bypass reorders the output and a group of proteins moves up to the top positions. The top hit is sterol delta 14 reductase, gi545906, (from PSI-BLAST position 29). In this case, we observe that Bypass reorders PSI-BLAST output according to the proper function of the protein (sterol reductase), but it also moves up several putative proteins: gi21645453, gi1370272 and gi21293863, CG17952-PC and D7SR gi2935281, another sterol reductase. A CLUSTALW multi-alignment (as shown below) not only suggests that they are not false positives, but also that this set of previously functionally unannotated proteins could belong to the reductase sterol metabolism functional class**.**

3.6.3. Data management and analysis framework

3.6.3.1. Using CouchDB and MediaWiki

As main storage solution, CouchDB was chosen because its document database model was highly convenient for mapping the hierarchical diversity of PSI-BLAST and Bypass analyses (as explained in previous NoSQL section). It's actually not the first case this relatively young NoSQL DBMS is used in biological context (Manyam et al., 2012).

Concatenation of analyses results are kept in the database just by having key values that match the identifiers of other documents. This allow us to browse the results. The relationship link can be described as: Protein sequence submission -> PSI Blast outcome -> Bypass result.

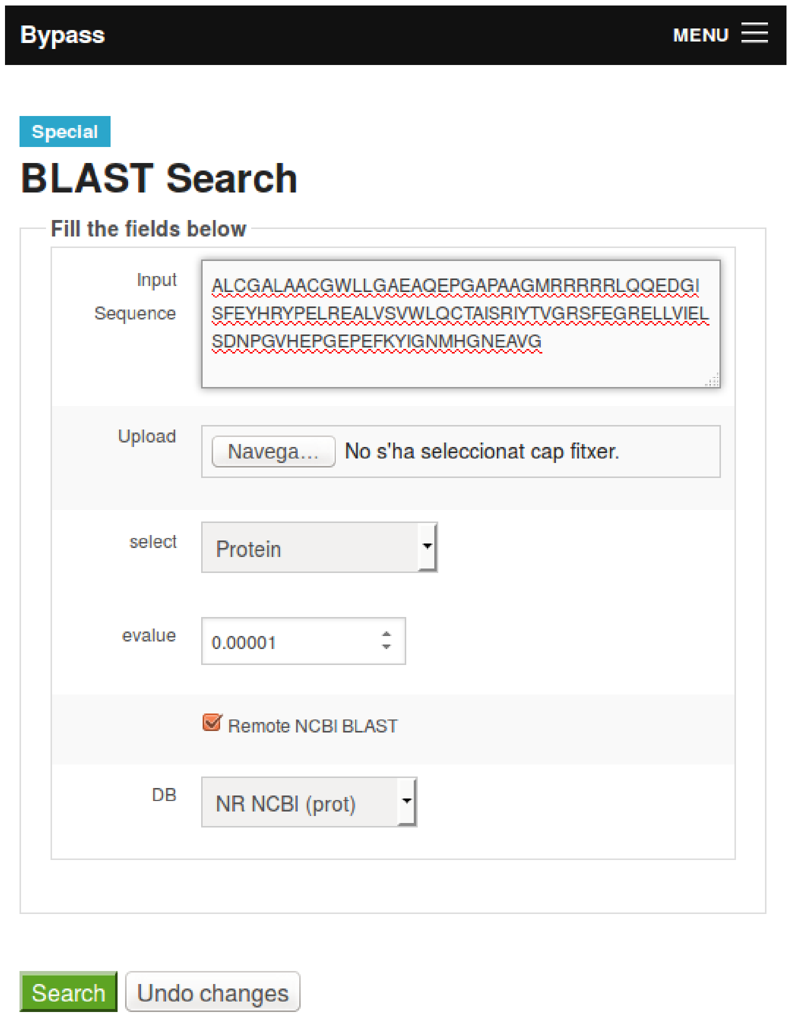
In order to make possible that Bypass program can be more easily integrated into a web framework, this was rewritten so it could accept and return JSON content. Moreover, by using cURL library (http://curl.haxx.se/), the program could interact directly with HTTP-friendly CouchDB, so saving any additional file write/read process in the disk (still inevitable for other used components, such as PSI-BLAST).

MediaWiki, the software behind Wikipedia, was chosen as a CMS (Content Management System), that is, a web platform for hosting everything that was being developed. This wiki system provides a quick and straightforward content input mechanism and because of other potential benefits (e. g., integrated API or the possibility data structuration of user-input data) that could be explored if needed in the future. Accompanying Appendix B presents in further detail the possibilities of this platform.

At the time of writing, the deployment can be accessed from: http://bypass.uab.cat, replacing the previous CGI interface to PSI-BLAST and Bypass program.

3.6.3.2. User interaction and data-input

Users can paste their sequences and analyse them against a set of predefined BLAST analyses (e. g. PSI-BLAST or BLASTN) and databases. In case the hosting server does not have enough memory, in order to reduce the delay using certain databases (such as NCBI non-redundant), NCBI-BLAST network option is used. This means that analyses are sent to NCBI servers and retrieved back from there.



BLAST Search

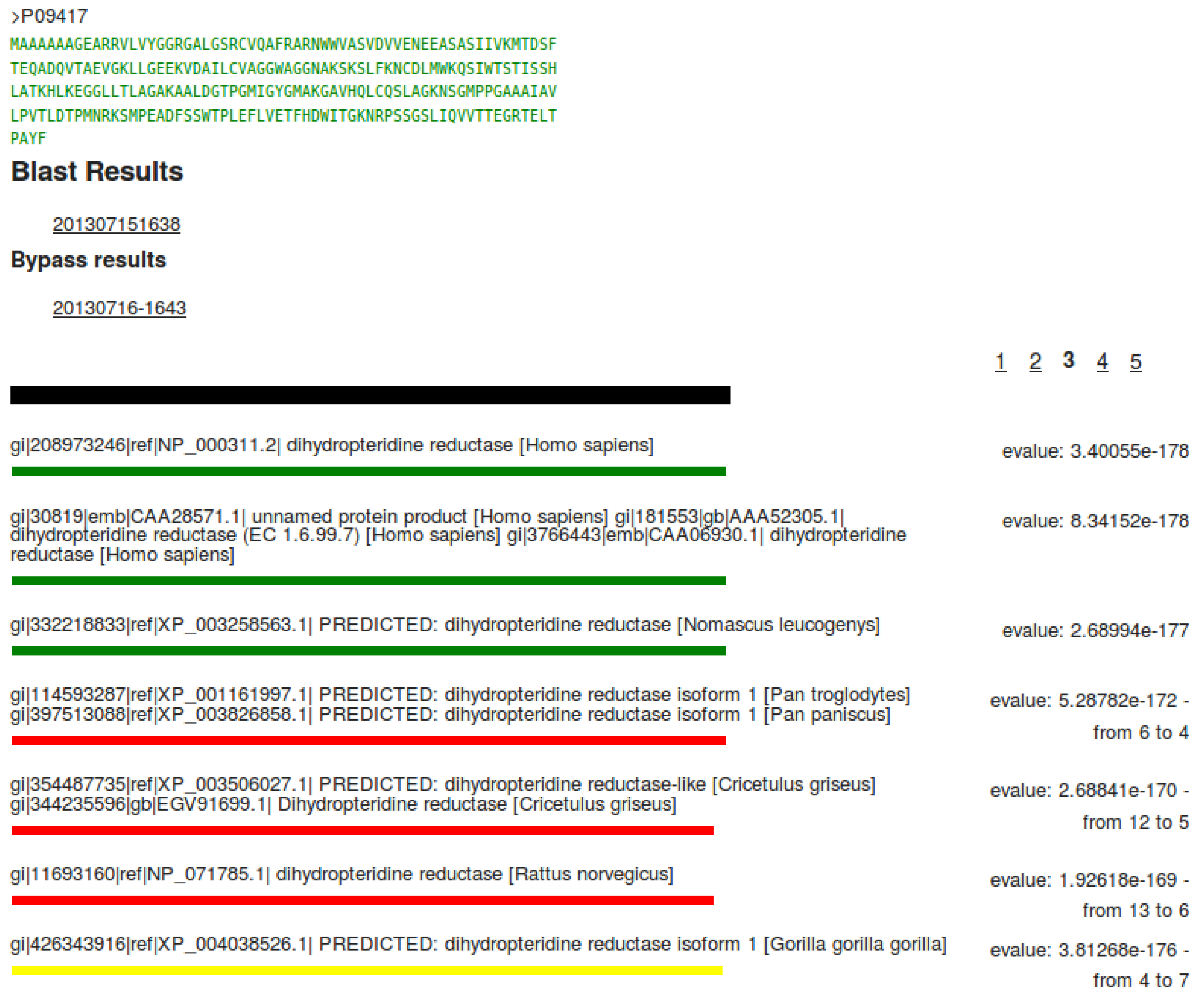
Fill the fields below

Input Sequence	ALCGALAACGWLLGAEAQEPGAPAAGMRRRRRLQQEDGI SFEYHRYPELREALVSVWLQCTAISRIYTVGRSFEGRELLVIEL SDNPGVHEPGEPEFKYIGNMHGNEAVG
Upload	Navega… No s'ha seleccionat cap fitxer.
select	Protein
evalue	0.00001
	✓ Remote NCBI BLAST
DB	NR NCBI (prot)

Figure 25. Look of main search interface.

Once submitted, sequences are stored and accessible from a single page. From that location, launched BLAST analyses can be retrieved at any time, and it's also possible to launch additional runs with different parameters.

BLAST results can be browsed in a graphical way, switching between iterations if available, and Bypass can be run (with default or custom weight parameters) for the selected results set.



```
>P09417
MAAAAAAGEARRVLVYGGRGALGSRCVQAFRARNWWVASVDVVENEEASASIIVKMTDSF
TEQADQVTAEVGKLLGEEKVDAILCVAGGWAGGNAKSKSLFKNCDLMWKQSIWTSTISSH
LATKHLKEGGLLTLAGAKAALDGTPGMIGYGMAKGAVHQLCQSLAGKNSGMPPGAAAIAV
LPVTLDTPMNRKSMPEADFSSWTPLEFLVETFHDWITGKNRPSSGSLIQVVTTEGRTELT
PAYF
```

Blast Results

201307151638

Bypass results

20130716-1643

1 2 **3** 4 5

gi|208973246|ref|NP_000311.2| dihydropteridine reductase [Homo sapiens]	evalue: 3.40055e-178
gi|30819|emb|CAA28571.1| unnamed protein product [Homo sapiens] gi|181553|gb|AAA52305.1| dihydropteridine reductase (EC 1.6.99.7) [Homo sapiens] gi|3766443|emb|CAA06930.1| dihydropteridine reductase [Homo sapiens]	evalue: 8.34152e-178
gi|332218833|ref|XP_003258563.1| PREDICTED: dihydropteridine reductase [Nomascus leucogenys]	evalue: 2.68994e-177
gi|114593287|ref|XP_001161997.1| PREDICTED: dihydropteridine reductase isoform 1 [Pan troglodytes] gi|397513088|ref|XP_003826858.1| PREDICTED: dihydropteridine reductase isoform 1 [Pan paniscus]	evalue: 5.28782e-172 - from 6 to 4
gi|354487735|ref|XP_003506027.1| PREDICTED: dihydropteridine reductase-like [Cricetulus griseus] gi|344235596|gb|EGV91699.1| Dihydropteridine reductase [Cricetulus griseus]	evalue: 2.68841e-170 - from 12 to 5
gi|11693160|ref|NP_071785.1| dihydropteridine reductase [Rattus norvegicus]	evalue: 1.92618e-169 - from 13 to 6
gi|426343916|ref|XP_004038526.1| PREDICTED: dihydropteridine reductase isoform 1 [Gorilla gorilla gorilla]	evalue: 3.81268e-176 - from 4 to 7

Figure 26. Graphical browser outcome linking a PSI-BLAST result an its associated Bypass processing. Red lines show alignments that are given higher importance after processing, yellow ones are the ones that are put behind and green ones are the ones that remain in the same position order.

As extra feature for helping in interactive manual annotation, integration with Gene Ontology is also provided, as it is explained below.

3.6.3.3. Gene Ontology cross-integration

Within our framework application we provide researchers the possibility of comparing a candidate set of matching sequences against their subject one in order to foresee new or unexpected functionalities. Even though they can simply compare them by reading the annotation, it might help things if we can stress what matching sequences have already in common.

There already existing applications, as it is the case of BLAST2GO (Conesa et al., 2008), that assign GO annotation to sequence queries from a set of matches (e.g. a BLAST result). However, in our case, compared to the former application, we provide both an interactive and selective way to generate this annotation.

From candidate sequences, which user may selectively choose, GO annotations are retrieved via a MySQL database populated by a UniProt-GOA. We include assigned GO accession codes by UniProt ID and also a mapping of identifiers among different DBs (RefSeq, GenBank, etc.) As reference, in this case we prefer a traditional SQL storage approach, because original data were huge tabular files that can be crossed along certain indices (UniProt ID) a reduced number times. In cases like this, SQL is still simpler and outperforms other methods when looking up by ID.

From retrieved GO accession codes of every hit, a subset of those more abundant codes is offered (weighted by the number of involved sequences). Furthermore, in order to try to extract additional annotation, codes that are not necessarily shared in different hits are analyzed for common parent accession codes that might be shared though. Finding a common parent in a tree or a graph, also known as ‘Least Common Ancestor’, is a common problem in graph manipulation and analysis. In the case of GO DAG, this is actually a bit more complex, since every node can have different kind of relations respect to other nodes (is_a, part_of, regulates, positively_regulates, negatively_regulates). In our case, only ‘is_a’ and ‘part_of’ relations are considered to set up a parent relationship, since the other ‘regulates’ ones can extend in faraway locations of the tree.

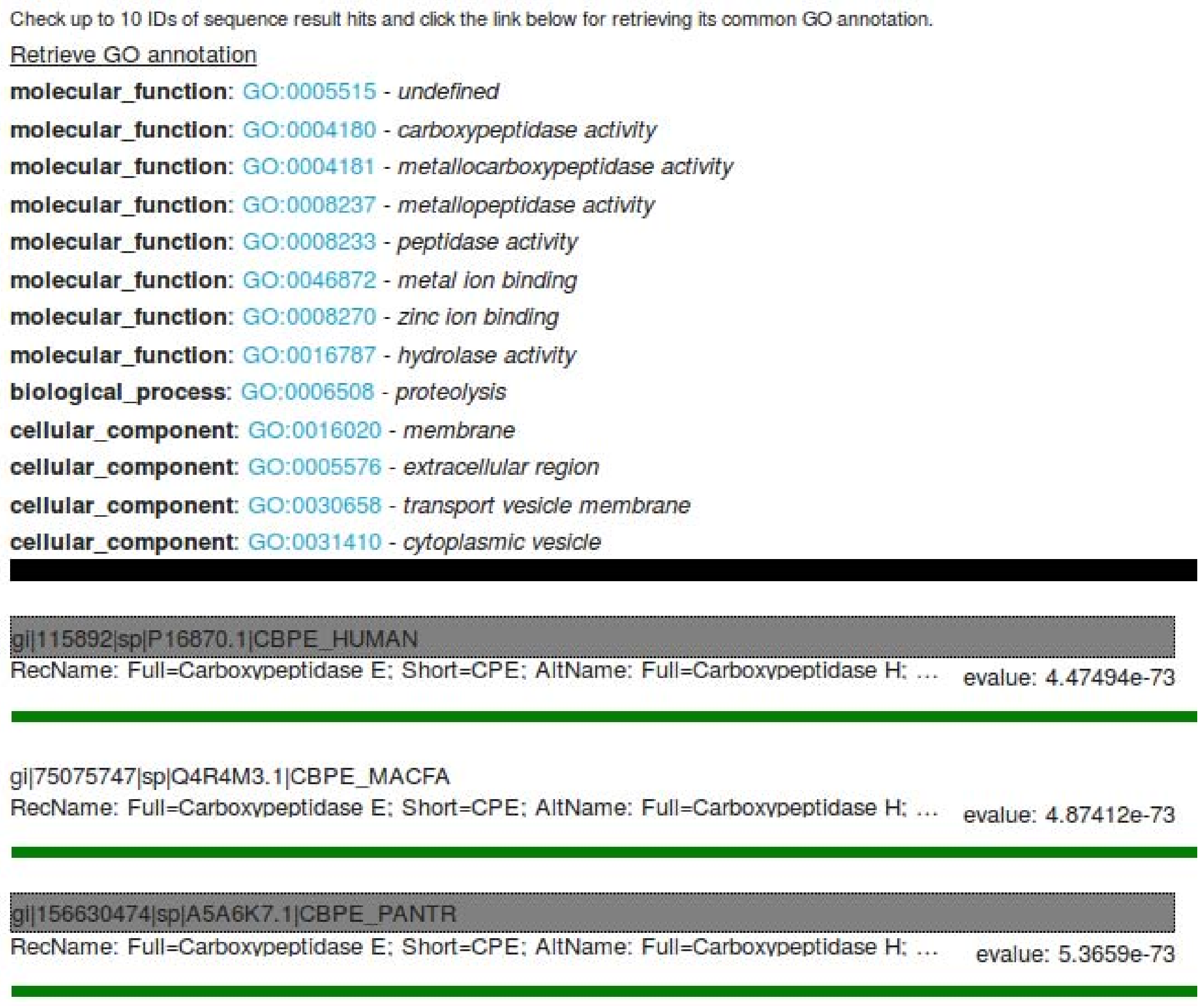
A brute-force approach is stopping at every node and decide the next node to follow in a path by checking (for every possible available relation) which is the next node that has a smaller distance to the root node.

Root nodes are already defined for GO (GO:0008150: *biological_process*, GO:0005575: *cellular_component*, GO:0003674: *molecular_function*).

Implementing this algorithmically in top of Neo4j (e.g. with a Py2Neo script) is time prohibitive. Instead of the former approach, we recur to already built-in Java native Neo4j path discovery methods and we collect all the paths that go from the candidate GO accession codes to their root node and we try to find the least common ancestor (Pareja-Tobes, 2012). If the least common ancestor is actually one root node, we consider it as meaningless, otherwise we save it.

Since performing it on the whole set of GO access codes may easily render a root node result, ignoring the fact that a gene product may have different assigned annotations, the analyses are performed in all different possible subsets with a size lower than the whole set (e. g. `(n/2)+1` vs `n`). That way we are able to retrieve any additional possible annotation that might be more or less spread among the hit matches. Of course, any singular annotation (either because of a specific functionality of a gene product, or because of a lack of annotation coverage) is not covered by this method.

As explained above, both MySQL and Neo4j are used for querying different biological datasets (semantically linked, though). Since it's not recommended that final users may be somehow aware of this underlying infrastructure diversity, a wrapper tool is set up for providing transparent queries. For this, Express (StrongLoop, n.d.), a Nodejs framework, is used for creating a REST API. The script tool performs HTTP requests and MySQL connections and always returns a JSON outcome that can be easily parsed. This provides a read-only way to query and integrate what has been explained in the preceding sections. Service is offered to be used in Bypass interface, but we offer it as well as a service by itself for 3rd party applications.



Check up to 10 IDs of sequence result hits and click the link below for retrieving its common GO annotation.

Retrieve GO annotation

molecular_function: GO:0005515 - *undefined*
molecular_function: GO:0004180 - *carboxypeptidase activity*
molecular_function: GO:0004181 - *metallocarboxypeptidase activity*
molecular_function: GO:0008237 - *metallopeptidase activity*
molecular_function: GO:0008233 - *peptidase activity*
molecular_function: GO:0046872 - *metal ion binding*
molecular_function: GO:0008270 - *zinc ion binding*
molecular_function: GO:0016787 - *hydrolase activity*
biological_process: GO:0006508 - *proteolysis*
cellular_component: GO:0016020 - *membrane*
cellular_component: GO:0005576 - *extracellular region*
cellular_component: GO:0030658 - *transport vesicle membrane*
cellular_component: GO:0031410 - *cytoplasmic vesicle*

gi|115892|sp|P16870.1|CBPE_HUMAN
RecName: Full=Carboxypeptidase E; Short=CPE; AltName: Full=Carboxypeptidase H; ... evalue: 4.47494e-73

gi|75075747|sp|Q4R4M3.1|CBPE_MACFA
RecName: Full=Carboxypeptidase E; Short=CPE; AltName: Full=Carboxypeptidase H; ... evalue: 4.87412e-73

gi|156630474|sp|A5A6K7.1|CBPE_PANTR
RecName: Full=Carboxypeptidase E; Short=CPE; AltName: Full=Carboxypeptidase H; ... evalue: 5.3659e-73

Figure 27. Screenshot of the interface for crosslinking GO annotations of different results.

Code of data import can be found at:
https://github.com/toniher/biomirror
Code of Express REST API can be found at:
https://github.com/toniher/gogoapi

3.6.3.4. Batch upload of analyses

At the time of writing, Bypass web platform does not allow to perform analyses of a huge amount of sequences at once (e.g. a whole candidate proteome). However, it’s actually possible to import into the platform a whole set of different analyses performed apart, browse the results, get some statistics and perform further analysis on every entity if desired.

As example of import mechanism, we analyzed different strains of *Mycobacterium bovis* with Blast and PSI-BLAST (5 iterations), InterPro (Hunter et al., 2009) and Blast2GO using the former Blast analysis. This represented an average of 1 GB per strain in different text format outputs (as tab separated and XML files, the latter preferred for sake of information).

As a first step, we create a CouchDB database. Therefore, we include the different views that will be used afterwards for querying the data we are going to import. The actual views (map functions) are codified as documents themselves and imported in the newly created database. As a next step, we generate a project, that will group a batch of sequences (for instance from a single strain).

```
curl -X PUT http://admin:passwd@127.0.0.1:5984/mbovis
```

Once a project is generated, a unique ID and a timestamp are returned, which will be used for linking the following import steps between each other.

The first actual data import is from original sequences, normally stored in FASTA format. Either in advance or just before the import process, it is a good practice to codify the whole sequence string as a unique MD5sum string which we will use as base identifier for the sequence document and also for the documents related to analyses performed to that sequence (e.g. Blast or InterProscan). For avoiding the eventuality of the same sequence happening in different batches (very likely to be the case for batches of different organism strains), we generate a new MD5 hash string with batch ID generated before. For analyses such as Blast, MD5sum of query sequence is also used, but adding a batch ID, an analysis distinctive descriptor (which can be a string such as ‘blast’ or simply ‘b’) and the timestamp retrieved before. If more analyses are performed lated using the same program against the same (set of) sequences, it’s simply a matter of using another timestamp.

Batch Document

```
{
   "_id": "2b4e2",
   "_rev": "3-72762f1d426e405fcb85489d3587a2fa",
   "username": "EQuerol",
   "date": "20140616-2305",
   "type": "batch",
   "name": "Mycoplasma bovis PG45"
}
```

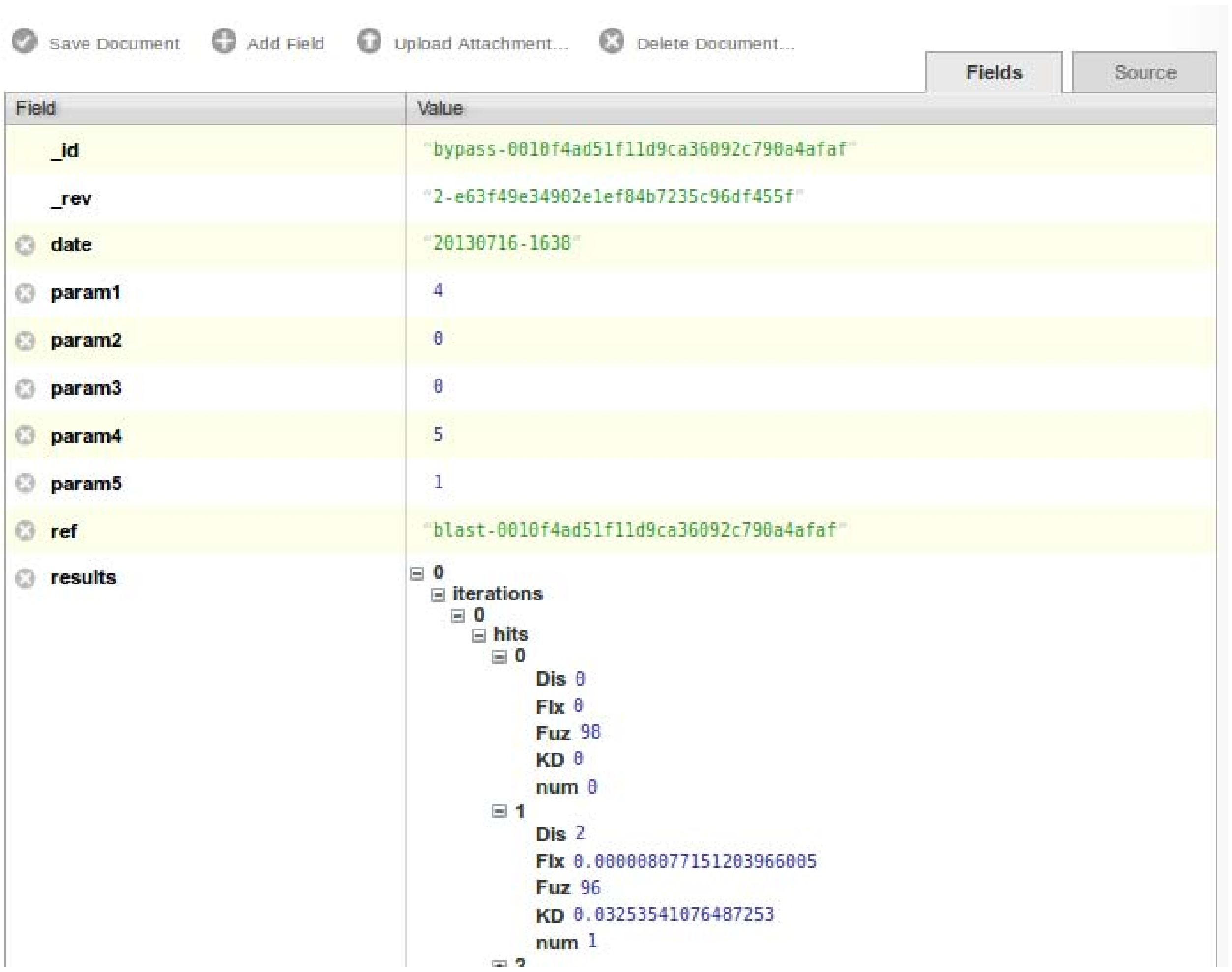
Sequence Document
ID -> MD5hash(Batch ID + MD5 SeqID)

```
{
   "_id": "615de85a28f2f13aeb9166a3de59d100",
   "_rev": "1-9672bd50f23d82b8561ace60f3967855",
   "username": "Anonymous",
   "date": "20140616-2306",
   "seqs": [
       {
           "name": "sp|O84898|UVRC_MYCBG",
           "desc": "sp|O84898|UVRC_MYCBG UvrABC system protein C OS=Mycoplasma bovis
(strain ATCC 25523 / PG45) GN=uvrC PE=3 SV=1",
           "id": "sp|O84898|UVRC_MYCBG",
           "seq":
"MDKNEIILMLKNVSTSPGVYLWKDAKQNVLYVGKAKNLRKRMLQYFDGAINSYKTNKLVSLIYDFDVYICKTNKEALLLEKAMI
DRYNPEFNILLLDDRKYPYLKVQLLKDSLLINLSRKVNAKDSKNTFYYGPFPSGYGAKPILKLLQHETLYENGLLIKNKDYNFWI
NQFNKIKEILSFKNNNYINELTNKMHQAANNMQFELALFLRDGLTYLKKLKESQIIELSQYKNIDVFAYKTDEKLIFATVLFYRY
GILINKVNLTIPLGLSVDESLRVFFEQFYEDKILPDNLIVQEELLNFDLNLSSEYKFISPKIGTNKKVLDLAILNLNDYYEKEHL
VIKNQLDKASNMLDSLNKYLNLPKLKNIVVFDNSNINNINPVGVAIVYTNGIKNKSLYRKFNLEALNERSADVEYIKQSISKFFS
SNKNTKDYDLVIADGGIQQVNEAKKTLKTLNINIPVIGLVKNEFHKTKALIDLDMNEIHINDLELYNYLVQIQVEVDRFAKSHFR
NRQKISSLEGKLRNIKGLGPNMEQNLLNHFKSYAKIYDASVEELSKIVPLNIAKSIKNKDYE",
           "md5": "a8231e9b2b9670f39522eba15e668683"
       }
   ],
   "type": "submit",
   "seqtype": "prot",
   "ref": "2b4e2"
}
```

Blast Document

ID -> MD5hash(Batch ID + “Blast” + MD5 SeqID + Timestamp)

```
{
   "_id": "10c8022d2184d1f0d1f0e1fbc0a82a75",
   "_rev": "1-18b1b5cd788da43d1aefa5fe93896dcc",
   "username": "Anonymous",
   "ref": "801030615d506565633d660c5d003d90",
   "maxiters": "5",
   "db": "blastdb",
   "results": [
       {
           "iters": [
               {
                   "hits": [
                       {
                           "hsps": [
                               {
                                   "qend": 334,
                                   "score": 1730,
                                   "positive": 334,
                                   "evalue": 0,
                                   "hend": 334,
                                   "hit_frame": 0,
                                   "query_frame": 0,
                                   "length": 334,
                                   "num": 1,
                                   "hstart": 1,
                                   "hseq": "MY..NE",
                                   "qseq": "MY..NE",
                                   "bit_score": 671.003,
                                   "qstart": 1,
                                   "identity": 334,
                                   "gaps": 0
                               }
                           ],
                           "name": "fatty acid/phospholipid synthesis protein PlsX",
                           "length": 334,
                           "num": 1,
                           "id": "gi|313678610|ref|YP_004056350.1|",
                           "gi": 313678610
                       },...
          ],
           "md5": "0009b8af1d6c83e23da345447700348c"
       }
   ],
   "program": "psiblast",
   "params": {
       "filter": "F",
       "gap_extend": 1,
       "expect": 10,
       "gap_open": 11
   },
   "date": "20140616-2304",
   "type": "blast"
}
```



Field	Value
_id	"bypass-0010f4ad51f11d9ca36092c790a4afaf"
_rev	"2-e63f49e34902e1ef84b7235c96df455f"
date	"20130716-1638"
param1	4
param2	0
param3	0
param4	5
param5	1
ref	"blast-0010f4ad51f11d9ca36092c790a4afaf"
results	

```
0
  iterations
    0
      hits
        0
          Dis 0
          Flx 0
          Fuz 98
          KD 0
          num 0
        1
          Dis 2
          Flx 0.000008077151203966005
          Fuz 96
          KD 0.03253541076487253
          num 1
```

Figure 28. Futton (CouchDB web interface) preview of a Bypass document. ID used here is the same as blast linked analysis. This is not an issue as far as no further Bypass runs are expected.

MD5 hash is 16 bytes long, which can be converted to ASCII hexadecimal resulting in 32 characters string length. This is the same length as default random ID generated by CouchDB. This should be convenient enough, however, it must be taken into account that querying indexes size will be dependent on ID size of the involved documents. One way to reduce ID length is using Base64 instead of hexadecimal, getting 24 character string length, or even other approaches such as Base91. However, as far as we could test, the index size difference was not significant enough, and at the same time, using hash approaches that resulted in a reduced index ID length actually imply having to cope with non-URL friendly characters such as '?'.

In order to avoid unwanted errors during imports (which are critical in second step analyses, such as Bypass —which require previous Blast analyses—), file checks are included between different steps in the workflow. For Blast or InterPro, since we opt for XML as output format (since this can converted to JSON and then imported document databases), `xmlwf` command-line tool is used. This checks well-formedness of the document (XML validity is skipped). As example, PSI-BLAST under many iterations may fail and this is a way to check any failed query. For other analyses which directly return JSON, such as Bypass, NodeJS `jsonlint` is used.

Even though disk space may become cheaper as consequence of Kryder's law (Walter, 2005), though disputed (Mellor, n.d.), handling and indexing text fields in a database continues to be an issue at the performance level. So, if the whole bulk of BLAST (or other) analyses are stored in CouchDB document, involved hit or query sequences (if chosen to be kept) continue being an important burden because of their relatively wide length, likely redundancy among entries and sequence informational complexity (higher in proteins than in nucleic acids). With this in mind, sequences are removed from documents and appended in a FASTA file, being replaced in the document by a MD5sum hash string generated from the actual sequences and used as sequence IDs in the FASTA file.

```
>MySeq (sent on 20150101)
METGPHYNYYKNRELSIVLAPFSGGQGKLGVEKGPKYMLKHGLQTSIEDLGWSTELEPS
MDEAQFVGKLKMEKDSTTGGSSVMIDGVKAKRADLVGEATKLVYNSVSKVVQANRFPLT
LGGDHSIAIGTVSAVLDKYPDAGLLWIDAHADINTIESTPSGNLHGCPVSFLMGLNKDV
PHCPESLKWVPGNLSPKKIAYIGLRDVDAGEKKILKDLGIAAFSMYHVDKYGINAVIEM
AMKAVHPETNGEGPIMCSYDVDGVDPLYIPATGTPVRGGLTLREGLFLVERLAESGNLI
ALDVVECNPDLAIHDIHVSNTISAGCAIARCALGETLL
```

Seq Hash function (MD5) --> `c89c50faff250aba83f2b5691b4efa08`

```
>c89c50faff250aba83f2b5691b4efa08
METGPHYNYYKNRELSIVLAPFSGGQGKLGVEKGPKYMLKHGLQTSIEDLGWSTELEPS
MDEAQFVGKLKMEKDSTTGGSSVMIDGVKAKRADLVGEATKLVYNSVSKVVQANRFPLT
LGGDHSIAIGTVSAVLDKYPDAGLLWIDAHADINTIESTPSGNLHGCPVSFLMGLNKDV
PHCPESLKWVPGNLSPKKIAYIGLRDVDAGEKKILKDLGIAAFSMYHVDKYGINAVIEM
AMKAVHPETNGEGPIMCSYDVDGVDPLYIPATGTPVRGGLTLREGLFLVERLAESGNLI
ALDVVECNPDLAIHDIHVSNTISAGCAIARCALGETLL
```

Once all the biological sequences from the different documents in the database are pushed into a FASTA file, this is processed so no redundant entries are left. Specifically, BioPython (Cock et al., 2009) is used for parsing the target FASTA file and we rely on IDs as unique identifiers (since IDs are generated from sequences).

If sequences are formatted with NCBI BLAST tools (`makeblastdb`), they can be later retrieved using `blastdbcmd` and, at the same time, being used as well as database to be searched with BLAST applications.

Unfortunately, this approach cannot be followed if among original sequences there are strings that contain dashes ‘-’, e.g. from alignments, since `makeblastdb` do not accept them at the time of writing. Luckily, there are other indexers that can handle this, as it is the case of `samtools faidx` (Li et al., 2009). Other possibilities could be using relational databases or key-value stores such as Redis, or even an apart document database only for the sequences and their IDs. None of these latter options provide the versatility of running BLAST (or other programs if FASTA is suitably indexed) on the stored sequences though.

```
>5114b61fbe9e12123335a8a192deb321
PPKCTSCGSQSPQ----HAEMCLHTA-GPTFPEEMGETQSEYSD------SSCENGAFF
CNECDCRFSEEASLKRHTLQTHS-DKPYKCDRCQASFRYKGNLASHKTVHTGEKPYRCN
ICGAQFNRPANLKTHTRIHSGEKPYKCETCGARFVQVAHLRAHVLIH
```

3.6.3.5. Enabling querying of sequences as a webservice

In the case of dealing with sequences with common biological alphabetical characters (e. g., not from alignments), we can choose to keep them in a FASTA formatted file instead of storing them in another DBMS (such as a MySQL or a CouchDB) but, in any case, keeping the same ID references in both places. Thus, sequences can be retrieved by using their ID thanks to `blastdbcmd` and, at the same time, programs such as BLAST can be run against the resulting binary files database. By following this approach, we allow each software to address what it was more proficient for.

Even though this same functionality can be directly integrated into any software (e. g., via a mere system call and some IO magic), since this is likely to be a service that could be used in many other instances, a dedicate REST webservice API was specifically created.
Based on Express NodeJS framework, it was possible to expose both sequence retrieval and BLAST searches, hiding the system calls on the background. JSON outcome, but also bulk FASTA files, could be retrieved by using HTTP calls. So, the service could be used in MediaWiki-based websites, using client jQuery Javascript or, even plugged to a CouchDB, proxying requests with Externals API (http://docs.couchdb.org/en/latest/externals.html)

More Databases can be simply included in a configuration JSON file.

```
{
    "express":{
         "db": {
                 "def": "prot",
                 "list": {
                         "prot": {
                                 "drosoph_aa": {
                                         "path": "/home/toniher/seqservice/db/drosoph.aa"
                                 },
                                 "moonlight": {
                                         "path": "/home/toniher/seqservice/db/moonlight.fasta"
                                 }
                         },
                         "nucl": {
                                 "drosoph_nt": {
                                         "path": "/home/toniher/seqservice/db/drosoph.nt"
                                 }
                         }
                 }
         },
         "exec": {
                 "path": "/home/toniher/seqservice/soft/ncbi-blast-2.2.30+/bin",
                 "method": {
                         "blastdbcmd":"/home/toniher/seqservice/soft/ncbi-blast-2.2.30+/bin/blastdbcmd",
                         "samtools": "/home/toniher/seqservice/soft/samtools-1.1/samtools"
                 }
         },
         "xsl": {
                 "blastn": "/home/toniher/seqservice/xsl/blastn.xsl",
                 "blastp": "/home/toniher/seqservice/xsl/blastp.xsl"
         },
         "port": "10030",
         "jsonp": false,
         "basepath": ""
    }
}
```

Figure 29. Example configuration file for defining sequence service.

And below an example REST FASTA query:

```
curl -X GET http://127.0.0.1:10030/db/drosoph/entry/7290020/fasta/1

{
	"def":">gi|7290020|gb|AAF45487.1|:1-1624 (AE003417) EG:BACR37P7.2 gene product
[Drosophila melanogaster]",
	"seq":"MTDFVELMNSMSSTFNSDCATSTAEGGTLLNLNLAEDKTLKWRNLANNQFASKEKKHKDKEEEERKEARNQ
EEIEDIKAL\nLADVVDAAAVKLEEEEAQNAEKVEPHTKCEIEEEGRKEMEYDQDVAKQDSEMEKKQNGKATSITVKMESNERAE
KHATEI\nATTSTERWENESFKTEQQNKKAAEKEEEPILAATQKLEANAEPLTTTRIEVAVASPLVVSSASVKLAADATNQMRAA
TSA\nGAATLADKNVQVSPGGTRRSRRTPRPIDTPTSVTDEHVQVENKKFGKSEQYTDCSSHLERFTLDDNTAIVRLQLKSEPDK
\nPSLTALSPEENSAPAPKRGRGRARKIRPDAEVETSEVILPCEDSLGEKKPGRKRKLPDEPIDQQQLSDLVVVKTEQEELG\nD
APLGDVKRMRRSVRLGNRLHADGSPWEEVKTEALHPQPSAELSFAEVTSEILPLAVLDEKTPPKKRGRKAKTPCVKLES\nETSC
GLPFANGNKKTNSSGGCELQLPKRSKRRIKPTPKILENDELRCEFETKHIERMTQWESAAAVDGDFETPTTGGNGS\nNSSTSRQ
KSDKSDGSNFEGGPGHPAGTSAIKKRLFSKSQRDIENYGAAMLAKSKLPPCPDVEQFLNDIKASRINANRSPE\nERKLNKKQQR
KLAKQKEKHLKHLGLQKNHRDEPSDNDSSNTDNEFFPTTRVQVGKPSVTLRVRNSVTKELPTTATLKSRR\nNPVVQAAKLTRRI
GARAAGEVTEAARASVPISTPDAEQLHSLDTSIQADVTPIRDLDMRPSTSRVSKFICLCQKPSQYYA\nRNAPDSSYCCAIDHID
DQKIGCCNELSSEVHNLLRPSQRVSYMILCDEHKKRLQSHNCCAGCGIFCTQGKFVLCKQQHFF\nHPDCAQRFILSTSYEKELG
DEEDQGVKFSSPVLVLKCPHCGLDTPERTSTVTMKCQSLPVFLRTQKYKIKPARLTTSSHL\nTQFGTVENANTPGATARNKGGL
STAVTLSAASSPASKTNGAQRGRAGTSNSNSRHALNSINFAQLIPESVMNVVLRGHVV\nSASGRVTAEFTPRDMYYAVQNDDLE
RVAEILAADFNVLTPIREYLNGTCLHLVAHSGTLQMAYLLLCKGASSPDFVNIVD\nYELRTALMCAVMNEKCDMLNLFLQCGAD
VAIKLGNLEATQLIVDSYRTSRNITSFLSFIDAQDEGGWTAMVWAAELGHTD\nIVSFLLNQDADPNICDNDNNTVLHWSTLHND
GLDTITVLLQSGADCNVQNVEGDTPLHIACRHSVTRMCIALIANGADLM\nIKNKAEQLPFDCIPNEESECGRTVGFNMQMRSFR
PLGLRTFVVCADASNGREARPIQVVRNELAMSENEDEADSLMWPDF\nRYVTQCIIQQNSVQIDRRVSQMRICSCLDSCSSDRCQ
CNGASSQNWYTAESRLNADFNYEDPAVIFECNDVCGCNQLSCK\nNRVVQNGTRTPLQIVECEDQAKGWGVRALANVPKGTFVGS
YTGEILTAMEADRRTDDSYYFDLDNGHCIDANYYGNVTRF\nFNHSCEPNVLPVRVFYEHQDYRFPKIAFFSCRDIDAGEEICFD
YGEKFWRVEHRSCVGCRCLTTTCKYASQSSSTNASPT\nNATTAPENETGTLSSTNTEKIGHA"
}
```

Letting simple sequence retrieval apart, performing analyses such as BLAST (as for example PSI-BLAST with different iterations) is a completely different story. These cases take longer than what we expect from the ‘immediateness’ of browsing the web. With these constraints, different workarounds have been devised for handling this situation.

As the analysis process is computed in the background in order not to block any other ongoing web browsing, one way or another must be envisaged to inform users that their analyses have actually finished and inform them about their associated results. Especially for long-time analyses, the most traditional approach is just email the users about their progress (normally at the end of this, but also at the submission time).

Parallelly, a common practice is reloading the submission page within the browser to show users the final results. Once the submission state is detected as finished, analyses results are fetched and they will be printed, otherwise there will be a periodical delay until this is retried and page is reloaded again. However, a more modern approach is available nowadays: Websocket (Fette & Melnikov, 2011), an Internet protocol which allows a more intimate communication between server and client. Compared to the traditional approach, page reloading (via HTTP), where there is a client request to the server, with Websocket server does actually send direct output to the client once a two-way communication is established. This protocol is normally used in chat or gaming services, but it might be also convenient in moderate time analyses, where user may expect some kind of interaction and check for instance how a workflow is progressing without the hassle of continual page reloads (or non-necessary data storage).

A real-time communication with Socket.IO library (Socket.IO Contributors, n.d.) (defaulting to WebSocket in modern web browsers) was implemented in the same Express NodeJS webservice used for sequence retrieval. However, this approach might only be practical if the actual database is not too large enough (e. g. NCBI non-redundant might be too large). Nonetheless, a practical application could be for monitoring the status of an ongoing long analyses, and even depending of its output, checking live the resulting data.

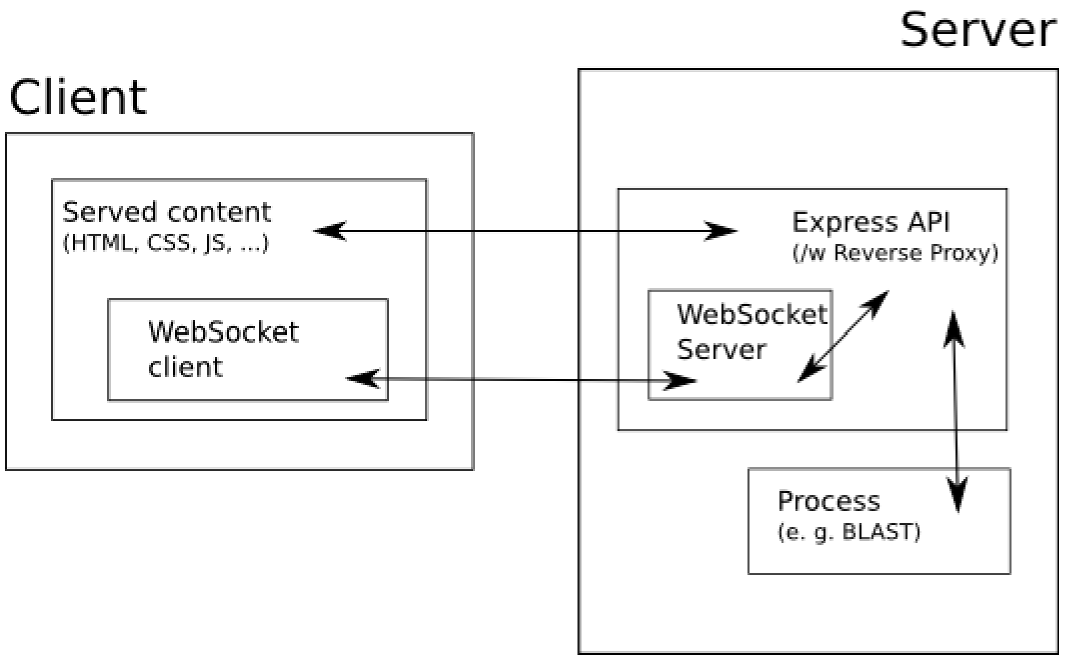


Figure 30. Schematic representation of the WebSocket communication when performing a process such as a BLAST.

Code of the sequence retrieval service can be found at:
https://github.com/toniher/seqservice

3.6.4. Conclusion

As stated as main objective in the introduction of this section, we managed to develop a biological sequence analysis management platform centered on the 3rd party NCBI BLAST and specially Bypass program. Moreover, it is a system flexible enough to include other applications such as Blast2GO, ProtLoc or TransMem, or any other program that can generate an output that can be converted into a key-based document object.

By using this environment, this platform has already been used with Bypass in other more recent studies for assessing other moonlighting proteins (Hernández et al., 2014).
As a side note, just as a reference of the interest in product annotation, it's worth to mention that there are other recent approaches that try to process PSI-BLAST output, in this case, using text-mining and machine learning approach (Biodados, n.d.).

Focusing back on the web framework, even though MediaWiki is being used as a base CMS for containing the sequence functional analyses platform described above, all its powerfulness as platform is not fully taken into advantage. For that reason, in the Appendix B of this document, some technology approaches already used by the author of this thesis in other projects are presented as possible ways to allow and enhance integration with other data.

In addition, within the Appendix C, a more throughout and detailed study of pros and cons regarding storage and retrieval of biological sequences is presented.

Moreover, at the time of writing, all the different involved software applications involved in this workflow platform; that is, scripts, web applications or even the underlying supporting DBMS are being ported into application containers using Docker technology (https://www.docker.com/). Despite the additional complexity of including another layer, the idea is to improve the maintenance and posterior deployment of the different software and associated data components. As a bonus, precisely this kind of approach is also a way to warrant a better reproducibility by 3rd parties. Indeed, this is a trend that is already being followed in some HPC Bioinformatics practices (Chatzou, 2014) (e. g., nucleotide assemblers benchmarks (Barton, 2014)) and already widely used in non-scientific web services environments.

As a final comment, the idea is that some of the other analyses approaches that have been introduced in previous sections of this document could be integrated in this platform and benefit from these last explained approximations as well. Furthermore, it would not be such a far-fetched plan to include more complex workflows (ideally compatible with Galaxy (Goecks et al., 2010) or Apache Taverna (Oinn et al., 2004))or, alternately their outcome results, in this presented framework.

3.7. Bioinformatics elucidation of Nna1-like proteins as active metallocarboxypeptidases

3.7.1. An introduction to metallocarboxypeptidases

Metallocarboxypeptidases (MCPs) are exopeptidases that catalyze the hydrolysis of C-terminal amino acids from their substrates. They are found in genomes from phyla of all five biological kingdoms and belong to Clan MC in MEROPS database (Rawlings et al., 2006) , which contains only one peptidase family, M14. The active site of M14 peptidases contains an essential catalytic zinc atom per molecule, which is penta-coordinated in a slightly distorted tetrahedral manner to two His, one Glu, and a water molecule. One of the histidines and the glutamate occur in the motif His-Xaa-Xaa-Glu-Xbb; the third zinc ligand is 103–143 residues C-terminal to this motif. Based on sequence conservation within the motifs containing the zinc ligands and around the catalytic residues, family M14 is grouped into three subfamilies: M14A, M14B, and M14C. Such divergence is confirmed by phylogenetic analyses.

In subfamily A, the zinc ligands occur within the motifs His-Xaa-Arg-Glu-Xbb, in which Xaa is Ser or Ala, and Xbb is Trp or His; and Xaa-His-Xbb-Tyr-Ser-Xcc, in which Xaa is a hydrophobic residue, Xbb is Ser or Thr, and Xcc is Gln or Glu. In subfamily B, the motifs are His-Gly-Xaa-Glu-Xbb, in which Xaa is Asp or Asn and Xbb is uncharged; and Xaa-His-Gly-Gly-Xaa-Xbb, in which Xaa is any small amino acid, and Xbb is hydrophobic or Arg.
Members of the M14A subfamily contain a catalytic domain ~300 residue in length preceded by a prosegment of 90–100 residues at the N terminus. The precursor of the enzymes is either completely inactive or has greatly reduced enzyme activity relative to the form with the prosegment removed.

The A/B subfamily (M14A) includes the pancreatic digestive enzymes CPA1, CPA2, and CPB, the mast cell CPA, a plasma CPB that functions in clot lysis, and four additional genes that are expressed in a limited number of tissues (Huang et al., 1999; Wei et al., 2002)

Members of M14B subfamily are not produced as inactive proenzymes; instead of the prosegment, these proteins contain a transthyretin-like subdomain at the C terminus of catalytic domain. Members of this subfamily contain other domains and even repeats of the carboxypeptidase domain.

The N/E subfamily (M14B) includes CPN, a serum enzyme, CPE, a neuroendocrine enzyme that activates peptides, CPD, a broadly expressed enzyme that functions in the processing of proteins within the trans Golgi network, CPM, a broadly distributed enzyme that functions on the cell surface, and CPZ, an enzyme present in the extracellular matrix (ECM) that regulates the activity of specific forms of Wnt. In addition to these active members of the N/E subfamily, there are three members of the N/E family (CPX1, CPX2, and ACLP/AEBP1) that do not appear to encode active CP enzymes based on the absence of critical active site residues as well as on direct biochemical analyses (He et al., 1995; Layne et al., 1998; Lei et al., 1999; Xin et al., 1998).

The M14C subfamily comprises the bacterial orthologs of d-glutamyl-(l)-meso-diaminopimelate peptidase I (Hourdou et al., 1993).

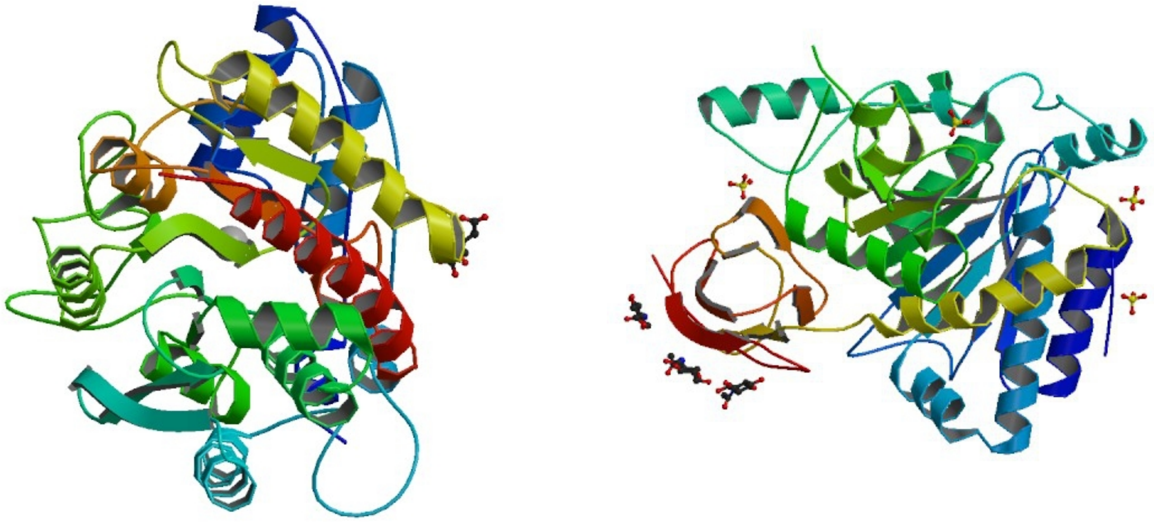


Figure 31. Ribbon structural representation of typical carboxypeptidase family members. Left: Human procarboxypeptidase B (1KWM). Right: Human carboxypeptidase N (Kininase I) catalytic domain (2NSM). Source: Protein Data Bank.

3.7.2. Linking Nna1-like proteins to metallocarboxypeptidases

3.7.2.1. Introduction

It was presumed that there was cytosolic metallocarboxypeptidase which acted in the removal of Tyr from the C terminus of α-tubulin (tubCP). However, despite much effort, there was no sequence information of any tubCP, and the gene (or genes) corresponding to this activity had not previously been reported . Even though two members of the M14B subfamily may be expressed in the cytosol and nucleus: the adipocyte enhancer binding protein-1 and CPD-N, but neither are functional as the tubCP because both were reported to be specific for C-terminal basic residues (He et al., 1995; Too et al., 2001).

As it will be presented later in this document, a gene transcript highly homologous to MCPs was identified as being up-regulated in spinal cord of mice subjected to sciatic nerve transection or crush injury. After the initial rise following nerve crush, transcript levels decline in affected motor neurons, with a time course coincident with target reinnervation. This transcript was named Nna1 (nervous system nuclear protein induced by axotomy) (Harris et al., 2000) and was identified as the gene mutated in the classical recessive mouse mutant Purkinje cell degeneration (pcd) (Fernandez-Gonzalez et al., 2002) . Nna1 is also known as ATP/GTP binding protein (AGTPBP-1), and human related genes have been named ATP/GTP binding-like proteins: AGTPBP1, AGBL2, AGBL3, AGBL4, and AGBL5 [Human Genome Organization (HUGO) Gene Nomenclature Committee: http://www.gene.ucl.ac.uk/nomenclature/; human, mouse and rat degradomes: http://www.uniovi.es/degradome/.

The large number of putative unclassified MCPs sequences is what prompted us to perform a detailed scanning of genomic, cDNA, and protein-based databases to give rise to a structurally driven classification analysis for sequences carrying the M14 signature. This was accompanied with an extensive phylogenic analysis to demonstrate that Nna1-like proteins represent a new M14 subfamily (named M14D).

Different conformational modeling also suggested representative Nna1-like proteins from a variety of species indicated an unusually open active site, a property that might facilitate its action on a wide variety of peptide and protein substrates. To test this, a recombinant form of one of the Nna1-like peptidases from *Caenorhabditis elegans* was expressed. This leaded us to demonstrate that this protein is a fully functional metallocarboxypeptidase that cleaves a range of C-terminal amino acids from synthetic peptides. The enzymatic activity is activated by ATP/ADP and salt-inactivated, and is preferentially inhibited by Z-Glu-Tyr dipeptide, which is without precedent in metallocarboxypeptidases and candidates as tubulin carboxypeptidase, being so a fundamental protein for cytoskeleton and an important pharmaceutical target.

Overall, all these works provided the necessary groundwork for other studies aimed at understanding the multiple physiological and pathological processes catalyzed by Nna1-like proteins, and stresses the importance of this new subfamily of metallocarboxypeptidases. A new name, cytosolic carboxypeptidases (CCPs), was then proposed for members of this M14D subfamily.

3.7.2.2. Database searches

The Protein Information Resource (PIR) nonredundant reference database (Wu et al., 2003) was scanned using the HHMER program (Durbin et al., 1998) to detect metallocarboxypeptidase sequences using the peptidase_M14 PFAM model (Bateman et al., 2004). Those sequences that matched the Hidden Markov Model profiles with an e-value of $<1 \times 10^{-10}$ were grouped as a candidate dataset. All sequences shorter than 230 aa were discarded. All candidate sequences in the dataset were aligned to one another using basic local alignment search tool (BLAST) 2.2.x (Altschul et al., 1990) `blastp` option of `blastall` program with default parameters. BLAST scores for each sequence analysis were used to generate a lower triangular matrix of all members in the dataset. This matrix was used to generate a clustering tree with the UPGMA algorithm of the PHYLIP neighbor program (Tuimala, 2006).

The novel putative metallocarboxypeptidase subfamily had already been defined as a clustered separate block of sequences at considerable BLAST-derived distance from well-known classic carboxypeptidases.

JOY (Mizuguchi et al., 1998) structural alignment information used in the modeling procedure (explained below) was taken into account during this iterative process and was stopped when no more promising sequences could be added to the working set.

To expand the analysis, in those organisms where information from current genome projects could be extracted, working sequences were mapped to chromosomal locations using available annotation and/or by searching the all of the currently sequenced genomes with BLAST and SSAHA (Ning et al., 2001) applications in ENSEMBL (Birney et al., 2006). Sequences with an incomplete carboxypeptidase domain or ones that appeared to represent alternative splicing variants were sorted out in order to reduce and refine the working set of proteins.

3.7.2.3. Comparative conformational modeling

Caenorhabditis elegans gene EEED8.6 encodes the *C. elegans* AGBL4 homologue (ceAGBL4). The amino acid sequence of ceAGBL4 was taken from the *C. elegans* ORFeome cloning project database (http://worfdb.dfci.harvard.edu/). The amino acid sequences and 3-dimensional (3-D) structures of pancreatic CPA from *Bos taurus* (2ctc) and *Sus scrofa* (1pca), pancreatic carboxypeptidase B (CPB) from *Sus scrofa* (1nsa), and CPT from *Thermoactinomyces vulgaris* (1obr) were obtained from the Protein Data Bank (PDB) (www.rcsb.org/pdb/) and used as templates in comparative modeling.

Secondary structure predictions of ceAGBL4 were obtained using PSIPRED (McGuffin et al., 2000) and its carboxypeptidase domain (CP domain) limits were defined. The ceAGBL4 CP domain sequence was aligned to templates using BLAST and FUGUE (Shi et al., 2001) . Key residues involved in substrate binding and catalysis were annotated with a secondary structure. Models were generated using MODELLER 8v1(Fiser & Sali, 2003a; Sali & Blundell, 1993) and later evaluated for correctness of stereochemistry, energy distribution, and fold assessment quality with PROCHECK (Laskowski et al., 1996), VERIFY3D (Eisenberg et al., 1997) and JOY. The process of modeling and manual realignment using GeneDoc (Nicholas et al., 1997) was repeated until models with good geometry and conformation were obtained. Root mean square (r.m.s.) deviation calculations of the modeled structures with respect to the crystallographic ones were obtained using COMPARER (Sali & Blundell, 1990) . PyMOL (Rother, 2005) was used for representation and visual inspection of models. All residues are numbered according to mature bovine carboxypeptidase A1.

3.7.2.4. Phylogenetic analyses

A total of 151 protein sequences belonging to M14 family from 46 different species representing the major lineages from Eubacteria and a wide range of eukaryotes were aligned with ClustalX (Thompson et al., 1997) in order to infer the evolutionary history of these proteins. Manual arrangement was carried out to correct disruptive inserts in the alignment. These corrections respected all conserved motifs and functional domains of M14 family. The definitive maximum length alignment for the complete phylogenetic analysis ranged from 370 sites in the case of the described metallocarboxypeptidase subfamilies M14A, M14B, and M14C to 565 sites for the novel putative carboxypeptidase unclassified sequences. In this alignment, 531 sites were "parsimony informative". A manual correction of the alignment was required especially for the sequences from Trichomonads, Kinetoplastids (Trypanosomatidae), Apicomplexans, *Rhodopirellula baltica*, and *Idiomarina loihiensis.*

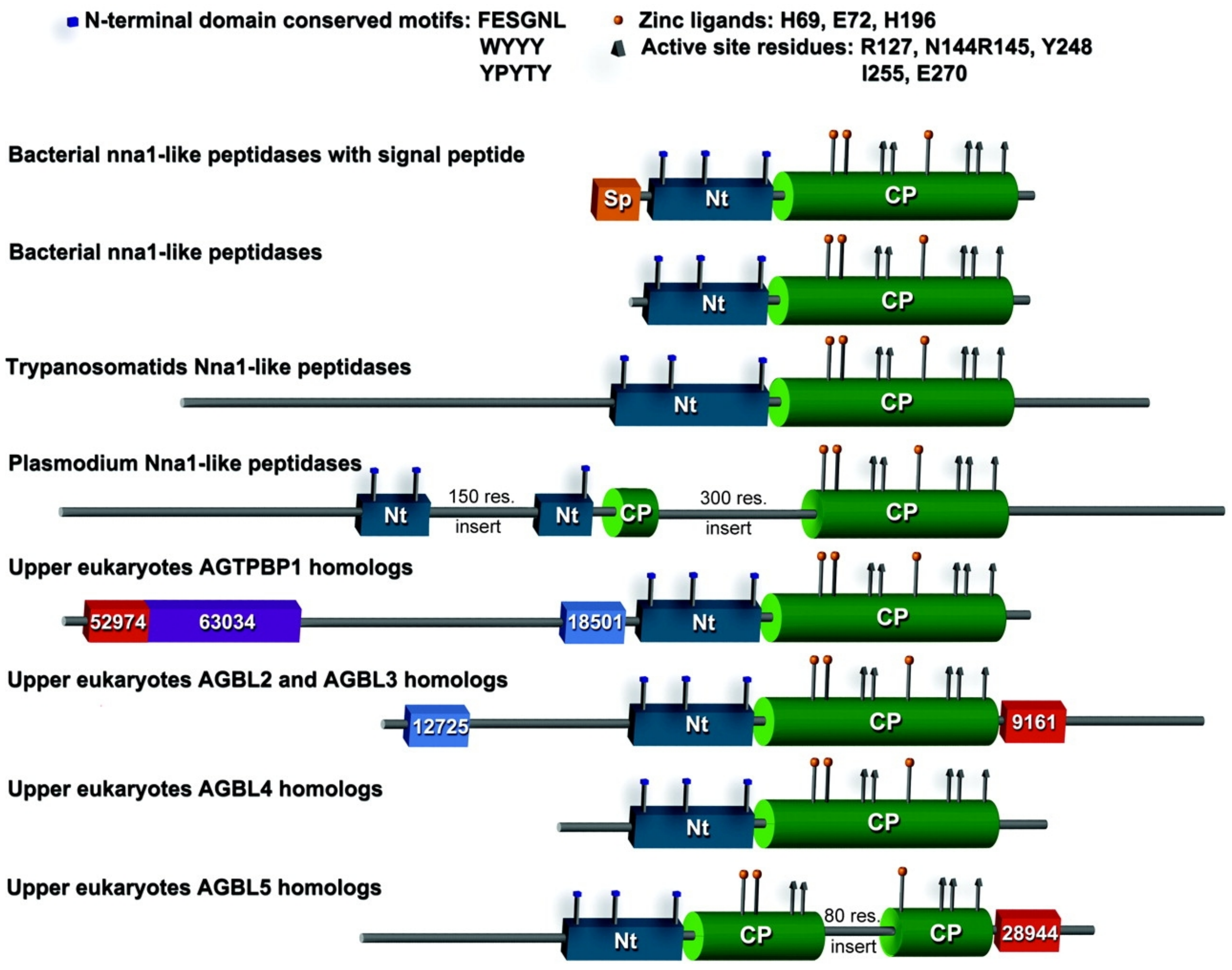

N-terminal domain conserved motifs: FESGNL
WYYY
YPYTY
Zinc ligands: H69, E72, H196
Active site residues: R127, N144R145, Y248
I255, E270
Bacterial nna1-like peptidases with signal peptide
Sp
Nt
CP
Bacterial nna1-like peptidases
Trypanosomatids Nna1-like peptidases
Plasmodium Nna1-like peptidases
150 res. insert
300 res. insert
Upper eukaryotes AGTPBP1 homologs
52974
63034
18501
Upper eukaryotes AGBL2 and AGBL3 homologs
12725
9161
Upper eukaryotes AGBL4 homologs
Upper eukaryotes AGBL5 homologs
80 res. insert
28944

Figure 32. Diverse domain organization of Nna1-like proteins. Conserved N-terminal domain (Nt) and carboxypeptidase domain (CP) are represented as blue boxes and green tubes, respectively. Other conserved motifs among subgroups are represented as colored boxes, with numbers indicating the PfamB family. Bacterial signal peptide is represented as an orange box labeled Sp. Conserved motifs located at the N-terminal domain and active site residues of the peptidase unit are indicated. Source: (Rodriguez de la Vega et al., 2007)

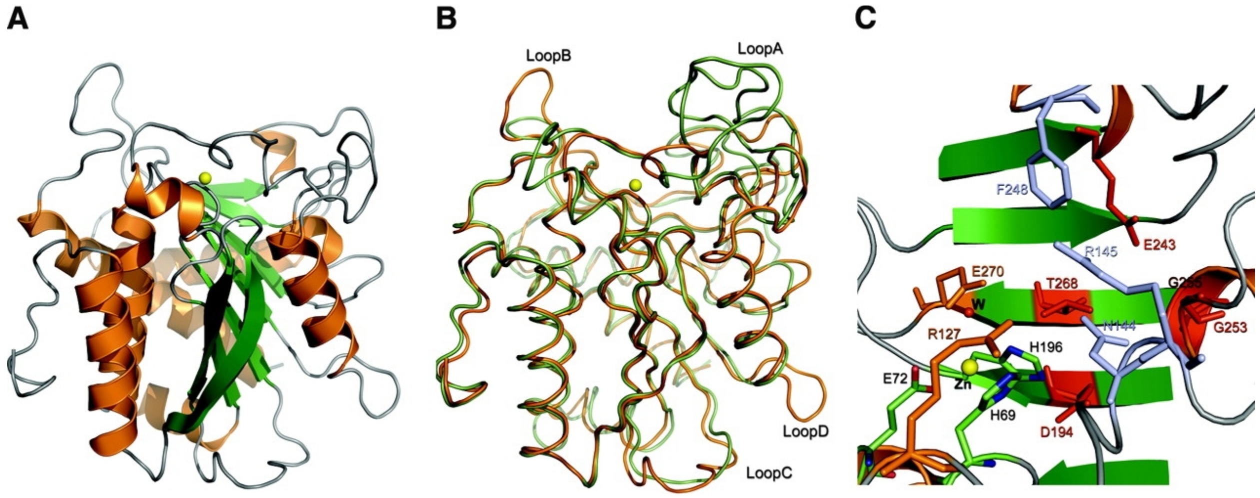


Figure 33. *A)* Structural model of *C. elegans* AGBL4 carboxypeptidase domain obtained by comparative modeling exhibits the characteristic topology of well-known zinc carboxypeptidases. Zn atom is represented as a yellow sphere. *B)* Superimposition of ceAGBL4 CP domain model (orange) and bovine pancreatic CPA (green) (PDB code: 2ctc). Overall topology is coincident (r.m.s.d value: 0.571). Main differences have been found in LoopA (α4-α5), LoopB (β8-β9), LoopC (β2-β3), and LoopD (α5-β5). *C)* Representation of the ceAGBL4 model active site. Zn ligands, catalytic residues, and substrate binding residues are represented as sticks. Zn ligands are shown in green, catalytic residues in orange, S1′ site in blue, and specificity pocket residues in red. Zinc atom (Zn) and water molecule (W) are represented as yellow and red spheres, respectively. Bovine CPA numbering has been used. Source: (Rodriguez de la Vega et al., 2007)

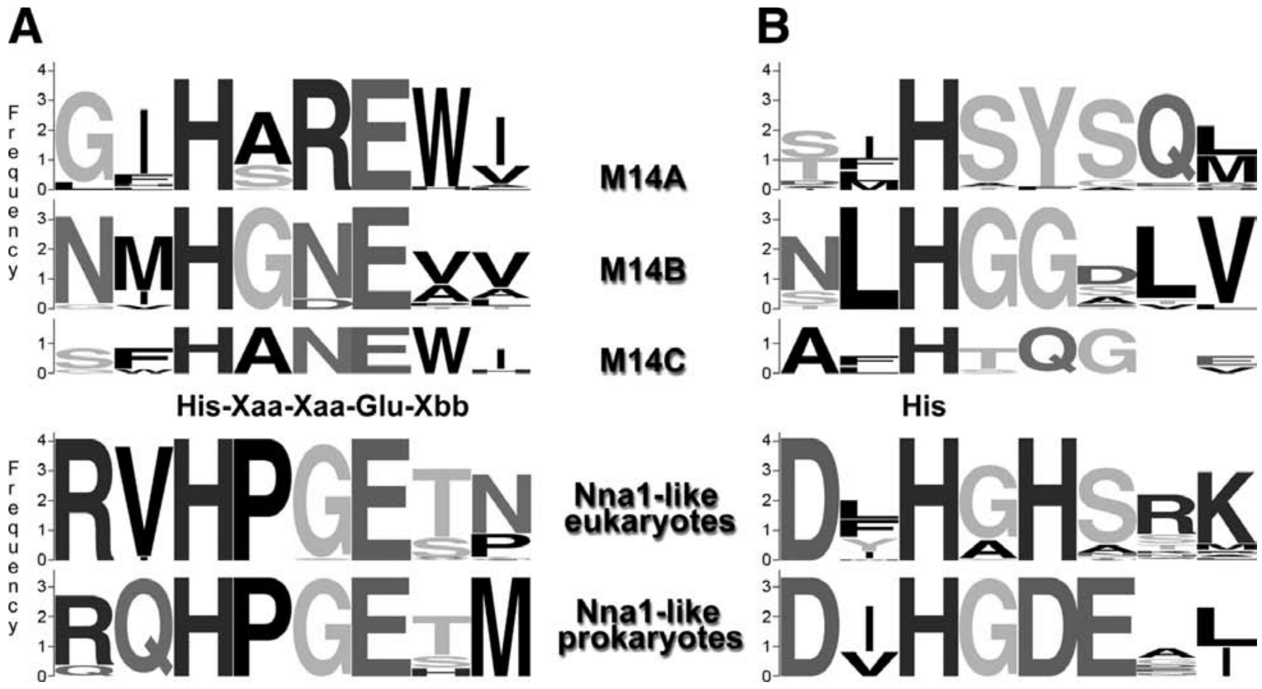


Figure 34. Sequence Logos for first (*A*) and second (*B*) zinc binding motifs in M14 peptidase subfamilies. The representation was generated using Weblogo (Crooks et al., 2004). The height of each amino acid indicates the relative frequency of that amino acid at that position. Source: (Rodriguez de la Vega et al., 2007)

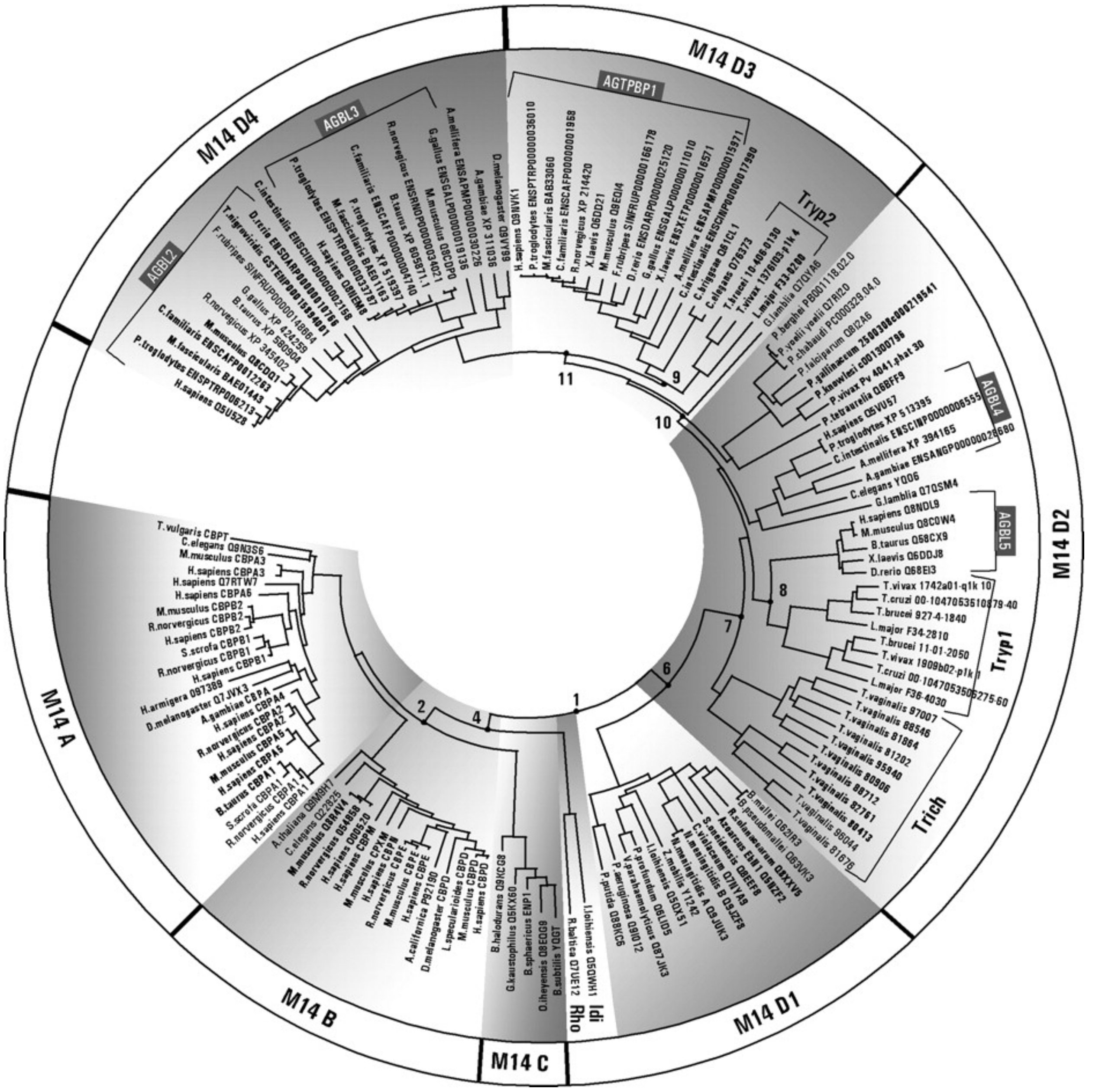


Figure 35. Phylogeny of M14 family. Reconciled tree of metallocarboxypeptidases obtained from the definitive alignment by the Minimum Evolution method under the Poisson Correction model. Numbered nodes were specifically analyzed for consistency by Bootstrap and Interior Branch Test; AGTPBP1, AGBL2, AGBL3, AGBL4, and AGBL5 group those homologues to the human Nna1-related genes named according to the HUGO Gene Nomenclature Committee: http://www.gene.ucl.ac.uk/nomenclature/; and the human, mouse, and rat degradomes: http://www.uniovi.es/degradome/; Tryp1 and Tryp2 group Clade 1 and Clade 2, respectively, from the family Trypanosomatidae; Trich groups *T. vaginalis* paralogous; Rho, *Rhodopirellula baltica*; Idi, *Idiomarina loihiensis*. Source: (Rodriguez de la Vega et al., 2007)

3.7.3. Outcomes and discussion

Available genomes, transcriptomes, and protein sequences databases were searched against in order to profile the Nna1 gene family in various taxa. More than 100 Nna1 homologues were collected in bacteria, Protista, and Animalia, but not in Archaea, Fungi, or Plantae. Exhaustive sequence studies with all the putative Nna1-like sequences revealed distinctive architecture in domains despite sharing a characteristic zinc carboxypeptidase signature confirmed by both Pfam (Punta et al., 2012) and MEROPS (Rawlings et al., 2006) databases. Nna1-like genes that did not appear to conserve the peptidase unit were excluded throughout the study; for example, mouse and human Nna1-like gene products that had sequence similarity only to the N-terminal portion of Nna1 were not further considered in this study, and so only five mouse and human Nna1-like gene products were detected by this analysis. In a more thorough analysis of the mouse genome, this partial sequence in the mouse database was extended using PCR and found to contain further Nna1-like sequence similarity over the entire peptidase domain; thus there are a total of six Nna1-related genes in the mouse genome (Kalinina et al., 2007), and presumably in the human genome as well.

3.7.3.1. Domain architecture and sequence analysis

Nna1-like proteins range from 400 to 2000 residues in length and comprise a common metallocarboxypeptidase domain of □300 residues. From the full-length sequence alignments, an N-terminal conserved domain was also identified that is □150 residues in length and contains three highly conserved motifs: F[E,D]SGNL at the N terminus, W[F,Y][Y,H,N]Y 60 residues downstream, and [F,Y]P[F,Y][S,T]Y at the C terminus, right before the peptidase domain.

Nna1-like genes from the different phyla are highly diverse. Some bacterial Nna1-like proteins display a signal peptide at their N terminus, which suggests they may be secreted to their environment. In contrast, eukaryotic Nna1-like proteins lack this signal peptide, which correlates with the experimentally demonstrated intracellular localization of Nna1 in cultured murine neurons.

Three different organizations in protist eukaryotes were found. Species from the genera Plasmodium have only one large Nna1-like gene (>1000 residues), with long N- and C-terminal extensions and widespread insertions of up to 300 residues long that were not conserved, probably caused by the particular nucleotide composition bias shown in this genera (Aravind et al., 2003). *Trichomona vaginalis* shows 11 paralogous genes encoding Nna1-like peptidases from 500 to 600 residues, all with N- and C-terminal extensions <100 residues in length. On the other side, Trypanosomatids encoded three paralogous genes of 800-1200 residues, depending on N-terminal extensions size.

Nna1-like genes of multicellular eukaryotes also display several organization patterns. The AGTPBP1 homologues are the largest proteins, reaching up to 1200 residues in length. They present the CP domain located at the C terminus and wide N-terminal extensions with highly conserved motifs, among them the PfamB families (conserved protein motifs with unknown function) PB052974, PB063034, and PB018501 (Pfam 20.0, PfamB clusters information are given as supplemental data). AGBL2 and AGBL3 homologues are similar in sequence, the main differences found in Cys distribution. Their size varies from 400 to 1000 residues due to the presence or absence of C-terminal extensions. The presence of two conserved motifs, PB012725 and PB009161, at the N and C terminus, respectively, also defines these closely related AGBL genes. AGBL4 homologues are relatively small due to the lack of N- or C-terminal extensions and the absence of other un-Nna1-like conserved domains. These AGBL4 genes are unique in their active site, with two conserved Arg residues close to position 255 instead of the single Arg found in the rest of the Nna1-like peptidases. AGBL5 homologues extend between 700 and 800 residues, with a large CP domain centered in the sequence and an additional highly conserved domain, the proline-rich PB028944 in the C-terminal side of the CP domain. Two other features distinguish AGBL5 homologues: first, substitution of the motif FESGNL located at the beginning of the N-terminal domain, by FDSGNL; second, the presence of two main insertions in the CP domain just before the zinc-anchoring motifs: the shortest is 20 residues in length (PB017213 and PB036922) and is located in β2-β3 loop; the largest consists of a 80 residue-long insertion (PB019879) located in α5-β5 loop.

M14A carboxypeptidases have a conserved 100 residue-long proregion located at the N terminus of the peptidase domain. This proregion folds in a globular-independent unit called the activation domain, which is necessary for the folding of the adjacent peptidase domain (Ventura et al., 1999).There is no amino acid sequence similarity between M14A prodomain and the conserved N-terminal domain of Nna1-like peptidases and, based on secondary structure predictions and fold recognition analyses using FUGUE and GeneTHREADER (Jones, 1999; McGuffin et al., 2003), we found no clear structural relations between them. N-terminal domain sequences were submitted to PSI-BLAST (Altschul & Koonin, 1998) analysis, but no significant sequence homologues were found other than Nna1-like species. Searches using Pfam HMMs (hidden Markov models) showed clear links to PB003298, PB030815, PB029118, PB017470, PB011640 and PB003904 domains. The PfamB families identified above only comprise Nna1-like proteins, confirming earlier PSI-BLAST results

.

Although it was not possible to assign a function to the N-terminal domain of Nna1-like peptidases by sequence homology or by fold recognition analysis, we suggest that this domain might act as a folding domain because it is conserved among Nna1-like peptidases, it is specific for these proteins and does not appear to exist in proteins that do not contain an adjacent CP domain. Furthermore, all known M14 metallocarboxypeptidases have a domain adjacent to the catalytic one that seems to help in folding M14A peptidases (Ventura et al., 1999) at the N terminus and M14B peptidases at the C terminus. The N-terminal domain might also act as a regulatory domain (discussed later) or as a binding domain, as is the case of the N-terminal extensions found in M14C peptidases (Bateman & Bycroft, 2000).

3.7.3.2. ceAGBL4 CP domain model

To examine whether Nna1-like peptidases conserve the typical secondary structures and the spatial conformation of the active site of known MCPs, we built a 3-D model of ceAGBL4 CP domain. Based on secondary structure predictions, the ceAGBL4 CP domain was defined from residue 131 to 415 and the conserved N-terminal domain motif YPYTY was located right before the first α helix of the peptidase unit. An initial alignment of templates and ceAGBL4 CP domain sequence was achieved using fold recognition software FUGUE. After >10 rounds of modeling and manual realignment, the Ramachandran plot for the final model showed 87.4% residues in most favored regions, 12.6% in allowed regions, and 0.0% in disallowed regions. The PROCHECK overall G-factor was –0.19 and the VERIFY3D report indicated that no poor areas were present in the model.

The ceAGBL4 CP domain model shows a fold containing an α/β/α sandwich structure with an antiparallel β-sheet of eight strands. The main chain atoms of the ceAGBL4 CP domain model and templates can be superimposed fairly well using COMPARER, displaying closest topological similarity to pancreatic MCPs. The overall r.m.s. deviation values calculated between the enzyme moieties of the model and the templates denote that all regular secondary structures of the model lie in regions topologically equivalent to porcine CPA (0.679), bovine CPA (0.571), and porcine CPB (0.567). A mayor divergence was observed between the model and the CPT from *T. vulgaris*, with an r.m.s. deviation value of 1.574.

The main differences between the model and templates were found within the lengths of the loops. At the active site entrance, the largest loop (α4-α5) of pancreatic carboxypeptidases has been reduced to half the number of residues in Nna1 homologues (LoopA), but the β8-α9 loop is a few residues larger in ceAGBL4 (LoopB) and other Nna1-like peptidases. LoopB might interact with natural substrates of these enzymes or confer resistance against natural protein inhibitors, as is the case of the *Helicoverpa zea* CPB (Bayés et al., 2005). However, the ceAGBL4 CP domain model clearly has a more accessible active site when compared with pancreatic carboxypeptidases, suggesting that the Nna1-like enzymes are capable of hydrolyzing bulky substrates like compact proteins, which do not have an extended C-terminal tail. At the opposite side of the peptidase domain, the β2-β3 loop (LoopC) and the α5-β5 loop (LoopD) are both larger in ceAGBL4 than in M14A peptidases. Most of the insertions found in Nna1-like sequences are located in those loops, suggesting they could play a role in the biological function of the native protein.

The predicted active site of ceAGBL4, the residues involved in the coordination of the Zn atom and the series of conserved key residues that form the different active center subsites have essentially the same conformation described in other well-characterized metallocarboxypeptidases.

3.7.3.2. Active site analysis

Nna1-like peptidases conserve the Zn ligands and the catalytic residues of metallocarboxypeptidases, but the motifs where they occur differ from the ones defined for M14A, M14B, or M14C subfamilies. In M14 family, the Zn atom is held in place by penta-coordination with two His residues (His69 and His196), one Glu residue (Glu72), and a water molecule (the numbering system corresponds to bovine CPA and will be used throughout). Zinc ligands are located in two zinc-anchoring motifs. The first Zn binding motif of Nna1-like peptidases contains a highly conserved proline adjacent to His69, in contrast to known MCPs, which contain an alanine or glycine in this position. A new consensus pattern for the first Zn-anchoring motif in Nna1-like proteins can be defined as His-Pro-Gly-Glu-[Ser,Thr]. The second Zn-anchoring motif of Nna1-like peptidases also differs from those found in M14 subfamilies. It can be defined as [aliphatic,aromatic]-His-[Gly, Ala,Ser]-His-[Ser, Ala] for eukaryotic Nna1-like peptidases and [aliphatic]-His-Gly-Asp-Glu for prokaryotic ones.

In addition to zinc ligands, other important MCPs residues involved in catalysis or in substrate binding are conserved in Nna1-like peptidases; these include Arg127, which helps to stabilize the oxyanion hole in the S1 site; Glu270, which is the general base for catalysis (Kim & Lipscomb, 1990); and Asn144 and Arg145 at the S1′ site, which bind to the C-terminal carboxylate group of the substrate. The Arg71 residue at the S2 binding site is substituted by an Asn residue in M14B or M14C subfamilies. In contrast, Nna1-like peptidases lack basic or amide polar residues at this position.

The role of Tyr248 at the S1′ site of MCPs has been the subject of much debate over the years. From the X-ray crystal structures of CPA in complex with Gly-Tyr it has been proposed that Tyr248 plays a role as a proton donor (Rees & Lipscomb, 1981). Subsequent high-resolution X-ray crystallographic study of the complex failed to confirm the proposition and demonstrated that the Tyr phenolic hydroxyl instead forms a hydrogen bond with the terminal carboxylate of the substrate (Christianson & Lipscomb, 1986) ; a kinetic study performed with the Y248F mutant rat CPA showed that Tyr248 was not required for the catalytic process (Gardell et al., 1985). In contrast, when this study was repeated with bovine CPA and its Y248F and Y248A mutants, it was shown that Tyr248 is essential for the catalytic activity of bovine CPA and that its aromatic ring plays a significant role in it (Cho et al., 2001). Nna1-like peptidases have either a Tyr or Phe at position 248, preserving the aromatic ring. Based on this substitution, T. Wang *et al.* proposed that the substrate(s) for Nna1 may have structural characteristics that set them apart from those of other members of M14 family (Wang et al., 2006) ; controversial results regarding the role of Tyr248 in the enzymatic reaction indicate that this is an important issue to be investigated.
In M14A members, residues Ser-194, Ile243, Ser-253, Ile255, and Thr268 define the substrate specificity pocket. Predictions of individual specificities are based largely on the conformational and space-filling effects of the amino acids in these positions (García-Sáez et al., 1997). In Nna1-like peptidases, CPA1 Ser-194 has been replaced by a highly conserved Asp residue (this position is adjacent to the second zinc binding motif;) and CPA1 Ile243 has been replaced by a Glu residue in ceAGBL4. According to the ceAGBL4 model, the acidic side chain of Asp 194 and Glu243 point into the active site cleft, suggesting a possible catalytic role (discussed bellow).

The major contributor to the substrate specificity of M14A peptidases is residue 255, which interacts with the side chain of the substrate's C terminus (Wei et al., 2002). In M14B peptidases, this role is carried out by a highly conserved Asp in position 207 (Gomis-Rüth et al., 1999). M14A, M14C, and Nna1-like peptidases have a conserved Gly residue at position 207; therefore, we can be relatively confident that residue 255 is responsible for the specificity of Nna1-like peptidases. Given that there is no sequence similarity with the M14A peptidases in this region, it is not straightforward to accurately predict this position in Nna1-like peptidases. Based on secondary structure predictions and fold recognition results, there are three optional residue types that could occupy position 255: an aliphatic/hydrophobic residue, a conserved Arg residue, or an Ala, Ser, or Gly residue. The first possibility restricts the substrate specificity of Nna1-like peptidases to CPA-like; the second option restricts it to CPO-like activity, and the third confers less restrictive substrate specificity. The presence of a small uncharged residue at position 255 is thought to confer broad specificity to *H. armigera* CPA1 (haCPA1), which has Ser at this position and hydrolyzes CPA-like and CPB-like substrates (Estébanez-Perpiñá et al., 2001). haCPA1 also presents Ser-194 substituted by an Asp residue.

The role of Asp194 should be further investigated to confirm whether the presence of such an acidic residue at this position could be implicated in CPB-like specificity of haCPA1 and Nna1-like peptidases (see below for kinetic results). The Glu243 substitution in ceAGBL4 could also contribute to the CPB-like activity of this enzyme. As shown below, *C. elegans* AGBL4 displays CPA-like and CPB-like activities; those results were also obtained for human AGBL3 peptidase (not shown). Thus, based on our substrate specificity results and the sequence similarities with haCPA1 on specificity pocket residues, it can be suggested that Nna1-like peptidases have Ala, Ser, or Gly at position 255. Nevertheless, until the 3-D structures of these new peptidases are determined, the exact residues that contribute to substrate specificity remain unknown.

In summary, Nna1-like peptidases conserve the zinc ligands as well as the key catalytic and substrate binding residues of the M14 family, but the motifs where they occur are distinct from the other M14 subfamilies. Some Nna1-like peptidases are classified into the M14B subfamily in the present MEROPS classification; however, on the basis of well-defined and different sequence conservations at the active site motifs, we propose that Nna1-like peptidases constitute a new M14 subfamily, which we have tentatively named M14D. The divergence of M14D peptidases has been demonstrated by phylogenetic studies.

3.7.3.3. Distinct and early separation of carboxypeptidases and Nna1-like proteins

The molecular phylogeny of M14 carboxypeptidase family and their homologous Nna1-like proteins was reconstructed based on their amino acid sequences by using the Minimum Evolution method with the Poisson Correction model. Phylogenetic inference with Neighbor Joining and Maximum Parsimony produced similar and congruent trees with respect to the four main groups in M14A, M14B, M14C, and M14D. All analysis and methods yielded highly concordant topologies, with only slight differences in Bootstrap support for a few basal nodes within the new M14D subfamily members corresponding to sequences from *R. baltica*, *I. loihiensis*, *T. vaginalis* (11 sequences), and those from the family Trypanosomatidae, which distributed into two different clades (8+4 sequences). We can draw several interesting evolutionary trends from a comparison of Bootstrap and Interior Branch Test supports on those nodes (Supplemental Table B) and from the main topologies. First, from a common ancestor there is a distinct and early separation of Nna1-like proteins from the classical carboxypeptidases belonging to the M14A, B, and C subfamilies, with 100% Bootstrap support in all trees. This split is a clear example of how the addition of new functional signatures on a gene causes functional divergence to acquire a new protein function in a group of organisms. This partition supports classifying Nna1-like sequences into a new subfamily, which we propose to name M14D.

Second, gene duplication has accompanied the specialized evolution of Nna1-like peptidases. Thus, the earliest duplication found is in *I. loihiensis*, which has two paralogous genes clustered separately (one is the most ancestral to all Nna1-like genes and the second is within the bacterial grouping M14D1), also contains two recent copies from *N. meningitidis*. The derivation of three main subgroups of eukaryotic Nna1-like genes (M14D2, M14D3, and M14D4) reflects several duplication events that are not restricted to vertebrates but also extend to protozoan species.Trypanosomatids Nna1-like genes cluster at two main groups, M12D2 and M14D3, revealing two duplication events with great divergences among them. Remarkably, *Trichomonas vaginalis* shows the most abundant gene duplication, with 11 copies in its genome from two main duplication events clustering together as sister groups within the M12D2 subfamily. The most striking exception of a whole group where the evolution of Nna1-like genes do not proceed through gene duplication is the genus *Plasmodium*, where a single gene copy was present in the genome of each of the seven species analyzed. According to phylogenetic clustering, gene duplications took place before the divergence of vertebrates (Meyer & Schartl, 1999) , since genes from Arthropoda also distributed along the three Nna1-like eukaryotic groups (M14D2, M14D3, and M14D4). This observation agrees with the fact that many early chordate gene families were formed or expanded by large-scale DNA duplications (McLysaght et al., 2002). According to the AGBL nomenclature regarding its ability to bind ATP/GTP (AGTPBP1, AGBL2, AGBL3, AGBL4, and AGBL5), numbering of these M14D peptidase paralogous genes in humans and mice does not match the phylogenetic distribution. Hence, the ancient M14D2 cluster includes AGBL5 and AGBL4, while the most recent M14D4 group encompasses AGBL2 and AGBL3. Although this alternative nomenclature can be applied to most multicellular eukaryota, some exceptions are found (rat has only AGTPBP1, AGBL2, and AGBL3 and mouse has the entire set). Moreover, since not all organisms might contain the full set of paralogous AGBL genes.

Third, Nna1-like proteins might have evolved from a common ancestor shared in two bacteria *R. baltica* and *I. loihiensis*. The sequences from these two species show sister-group relationship with both, classical carboxypeptidases and Nna1-like peptidases . It should be mentioned that only a few bacterial genomes contains either previously classified metallocarboxypeptidases or Nna1-like proteins. Bacterial carboxypeptidases from the subfamily M14C are restricted to two families from Firmicutes and one from Actinobacteria.[2] Nna1-like genes are spread along nine families from Proteobacteria and Plantomycetes. All of them are Gram-negative and flagellated bacterial. The mosaic pattern of bacterial phylogenetic distribution is most readily explained by vertical inheritance and selective loss in many bacterial lineages, with specific retention in metazoans and some protozoans.

Fourth, the simplest explanation for the absence of Nna1-like proteins in plants and fungi is that a common mutual ancestor already lacked a Nna1-like gene, and thus the function executed by the Nna1-like proteins is either not required in these organisms or is covered by another gene family. The protozoan and metazoan lineages of Nna1-like genes were assembled from unique components (N-terminal domain) not found in any other gene family, and thus alternative Nna1-like function in plants and fungi could be evolved by assembling components from other origins, like the different lineages of assembling the hedgehog genes (Ventura et al., 1999) , which in turn did not reveal their identify through a data mining search scheme.

Bacterial, protozoan, and metazoan Nna1-like homologues are virtually identical in domain organization and belong to the same M14D subfamily, suggesting that the N-terminal domain is an integral part of the protein that cannot be separated from the carboxypeptidase domain during gene duplication; this association precedes the eubacterial/parazoan split, the earliest divergence among existing organisms containing Nna1-like proteins. This observation is consistent with the proposal that the conserved N-terminal domain functions in the folding of the carboxypeptidase domain, as described above.

At the time of the study, only 14 bacteria genomes contained Nna1-like genes (<1.5%). Since genomes from Archaea (29 complete and 28 in progress), Fungi (9 and 56), and Plantae (2 and 34) are notably less profuse in databases than those from bacteria, we cannot discard the possibility that Nna-1-like genes may later be found in those other kingdoms. Thus, the low occurrence of Nna-1-like genes in bacterial genomes could also be explained by the relatively small horizontal transfer rate described from eukaryotes (Snell et al., 2006), which has been associated with pathogenicity (Koonin et al., 2001).

3.7.3.4. Link between tubulinyl-Tyr carboxypeptidase (EC 3.4.17.17) and Nna1-like proteins

As shown above, the computer-based modeling of ceAGBL4 indicates that the active site entrance is much wider and open than in the previously studied carboxypeptidases. Because this AGBL4 of *C. elegans* is representative of the entire M14D subfamily with respect to the gaps and inserts required for alignment with the other subfamilies, it is likely that the other Nna1-like proteins will have generally similar structures. This, together with the absence of an N-terminal signal peptide in the eukaryotic members of this new protein subfamily, implies that they function in the cleavage of cytosolic proteins.

The only cytosolic protein known to undergo C-terminal processing is the α subunit of tubulin. This protein is initially produced with a C-terminal Tyr, which is removed by the tubulinyl-Tyr carboxypeptidase (tubCP) and reattached by a specific tubulin ligase. Although the ligase has been identified (Ochman et al., 2000), tubCP has not, despite a great deal of effort since it was first described in 1977 (Erck et al., 2005).

It has been reported that tubCP activity is increased with ADP, inhibited by intermediate to high concentrations of NaCl, and shows strong specificity for Z-dipeptides, which, in the carboxyl end, contain Tyr or Phe linked to glutamic acid (Hallak et al., 1977). Such an enzyme is thought to prefer polymerized over dimeric tubulin as substrate and performs tubulin detyrosination better on microtubules (Argaraña et al., 1978). In proliferating cells, tubCP activity is not detectable, but when cells undergo differentiation, the enzyme is activated. In the case of nervous cells, tubCP activity seems to be restricted to differentiating neurons whose processes are rich in detyrosinated tubulin, indicating that tubCP is involved in growing of processes (Bosc et al., 1996; Gundersen et al., 1987; Wehland & Weber, 1987).

Several observations from the present study are consistent with the possibility that Nna1 and/or related enzymes function as tubCP. First, cytosolic localization of Nna1 and Nna1-related enzymes fits with this proposed role. Second, the modeling suggests the ability to remove C-terminal Tyr residues from proteins based on both a consideration of the substrate binding pocket (which can accommodate a Tyr) and the loops surrounding the active site pocket (discussed above). Third, the direct demonstration that one of the Nna1-like proteins has enzymatic activity toward small peptides that contain C-terminal hydrophobic groups implies that this enzyme is capable of cleaving C-terminal Tyr. Fourth, the nucleotide dependence (ADP/ATP) of the activity of that recombinant enzyme is congruent with its potential involvement in a pathway regulated energetically, as is the transformation of tubulin in cytoskeleton remodeling, and with the reported activation of tubulin carboxypeptidase by ADP (Contín & Arce, 2000). Fifth, the higher inhibitory capability of the dipeptide Z-Glu-Tyr *vs.* Z-Gly-Tyr, and the negative effects of moderate or high salt concentrations on the activity of such enzyme, match the properties of the previously reported tubulin Tyr carboxypeptidase and contrast with other well-known metallocarboxypeptidases from different subfamilies.

The biological significance of tubulin would easily justify the occurrence of various alternative processing forms, although some organisms might survive with a single one or might develop substitutive mechanisms. Perhaps this would be the case for fungi and plants, which apparently lack Nna1-like enzymes. It is important to mention that the tyrosination cycle is highly conserved among eukaryotes and has been found in most cells where it has been searched for, with the exception of the fission yeast *S. pombe* (Argaraña et al., 1978). It was demonstrated that in *Saccharomyces cerevisiae* there is no turnover of the α-tubulin C-terminal and that the tubulin ligase activity is absent despite the presence of a gene with a significant homology to tubulin-tyrosine ligase in other organisms (Alfa & Hyams, 1991). The lack of Nna1-like peptidases in fungi would be consistent with the lack of the detyrosinylation cycle of α-tubulin in those organisms.

3.7.4. Study of Nna1 subfamily of mouse cytosolic carboxypeptidases

3.7.4.1. Mouse Nna1-like proteins

A screen for mouse mRNAs up-regulated by axonal regeneration identified a gene product with homology to the carboxypeptidase (CP) family; this gene product was named Nna1 (Bloom, 2004). However, the initial alignments of Nna1 with the enzymatically active CPs did not support a role for Nna1 as an active enzyme due to the absence of critical active site residues.

Because Nna1 was found to be both cytosolic and nuclear when expressed as a fusion protein with green fluorescent protein (GFP), Nna1 was proposed to function in the processing of nuclear factors needed for neuronal regeneration and survival. Support for a role in neuronal survival was provided by the discovery that *Purkinje cell degeneration* (*pcd*) mice have a defect in the gene encoding Nna1 (Harris et al., 2000) . These mice, which resulted from a spontaneous mutation, have an altered gait due to the degeneration of Purkinje cells (Fernandez-Gonzalez et al., 2002). The *pcd* mice also show degeneration of olfactory bulb mitral cells (Landis & Mullen, 1978) , retinal photoreceptor cells (Greer & Shepherd, 1982) , and other defects such as sterility (LaVail et al., 1982) and abnormal sperm shape (Landis & Mullen, 1978) . Altogether, five variants of the *pcd* mutation are known.

In order to address these questions, the mouse genome was searched for additional Nna1-like sequences. From these searches, it was clear that five additional Nna1-like genes existed in mouse and human genomes. Alignment of these various sequences revealed several highly conserved residues, including critical active site residues that are necessary for the catalytic activity of A/B and N/E subfamily CPs. Structural modeling based on these alignments resulted in structures that share the basic CP catalytic domain structure of both A/B and N/E CPs.

Parallely, via accompanying experiment studies, the distribution of the mRNAs encoding the various Nna1 homologs in mouse tissues suggested a broad function with some redundancy, thus explaining why only a few cell types die in the *pcd* mice.

The absence of a signal peptide from all forms of the five novel Nna1 homologs suggested a cytosolic distribution; this was confirmed by expression in cell lines and immunofluorescence analysis. Because the name Nna1 was based on the previous study of nuclear localization in neurons and from the present data, it is clear that neither Nna1 nor the other family members are restricted to neurons or the nucleus, and this subfamily was so named cytosolic carboxypeptidase (CCP). The new members are numbered from CCP2 through CCP6, and we refer to Nna1 as Nna1/CCP1. One function of these family members appears to be tubulin processing, based on the absence of C-terminally-truncated alpha tubulin forms in the mitral cells of *pcd* mice.

3.7.4.2. Bioinformatics and cDNA sequence analysis

To identify genes and gene products with homology to Nna1/CCP1, the mouse genome was searched with mouse Nna1/CCP1 protein sequence using `tblastn`. The non-redundant (NR) and expressed sequence tag (EST) GenBank databases were searched using mouse Nna1/CCP1, and the fragments were identified from the genome search. Full-length cDNA sequences of the various splice forms were compiled from the NR and EST databases. Several cDNA clones with incomplete sequences in GenBank were purchased (Invitrogen, Carlsbad, CA) and sequenced in both directions; these included clone 3994516 (CCP2), 6813901 (CCP5), and 6446357, 5024144, and 6742069 (CCP6). For CCP4, the longest cDNA sequences in the NR and EST databases appeared to encode only a portion of the CP domain, whereas the genome showed the presence of additional potential exons with homology to the CP domain of Nna1/CCP1. To confirm this, reverse-transcription polymerase chain reaction (RT-PCR) was performed and the resulting PCR product was actually sequenced.

3.7.4.3. Alignment of CCPs and modeling

Amino acid alignments of the six mouse CCPs were performed using DNA-Star MegAlign version 5.05 using gap penalty and gap wt penalty values of 9. For this analysis, a representative amino acid sequence of each mammalian metalloCP was included. Structural modeling was performed on mouse CCP1 using a CPA/B structural alignment extracted from HOMSTRAD database (Stebbings & Mizuguchi, 2004), which includes pancreatic CPA from *Bos taurus* (2ctc) and *Sus scrofa* (1pca) and the pancreatic CPB from *Sus scrofa* (1nsa) and CPT from *Thermoactinomyces vulgaris* (1obr). These structures were checked as templates against mouse CCP1 and used for modeling the latter protein sequence with MODELLER 8v2 modeling suite environment (Fiser & Sali, 2003b; Sali et al., 1995).Batches of at least 500 submits were performed until models with good geometry, and suitable fold requirements were retrieved and verified with PROCHECK (Laskowski et al., 1996), VERIFY3D (Eisenberg et al., 1997), and JOY (Mizuguchi et al., 1998).

The CCP1 derived alignment was transferred to the other candidate CCPs, and they were subsequently modeled following the same procedures and crosschecked against each other and their templates to ensure their quality. Swiss-PDB Viewer (Guex & Peitsch, 1997), Jmol (The Jmol Team, 2007), and PyMOL (Rother, 2005) were used for visually checking the quality of the final models and to determine the spatial orientation of certain key residues (discussed below). PyMOL was also used for generating surface representations of the models.

3.7.4.4. Analysis of Nna/CCP genes and cDNAs

A bioinformatics approach was used to screen for homologues of Nna1/CCP1 in the mouse and human genomes and in GenBank NR and EST databases. Altogether, five additional family members were found in both human and mouse databases. These five additional family members have been designated CCP2–6, based on the order by which they were detected. Each of these CCP genes is present on a distinct chromosome. The length of the CCP2–6 genes varies from 18,000 nucleotides for mouse CCP5 to over 1,200,000 nucleotides for mouse CCP6. Interestingly, the human gene for CCP6 is also over 1,200,000 nucleotides, and the introns are generally similar in length between human and mouse genes. The length of the cDNA for each of the CCPs ranged from 1881 bp for CCP6 to 4369 bp for CCP2. Representative sequences of the splice forms of each mouse CCP have been deposited in GenBank (accession numbers DQ867026 through DQ867040).

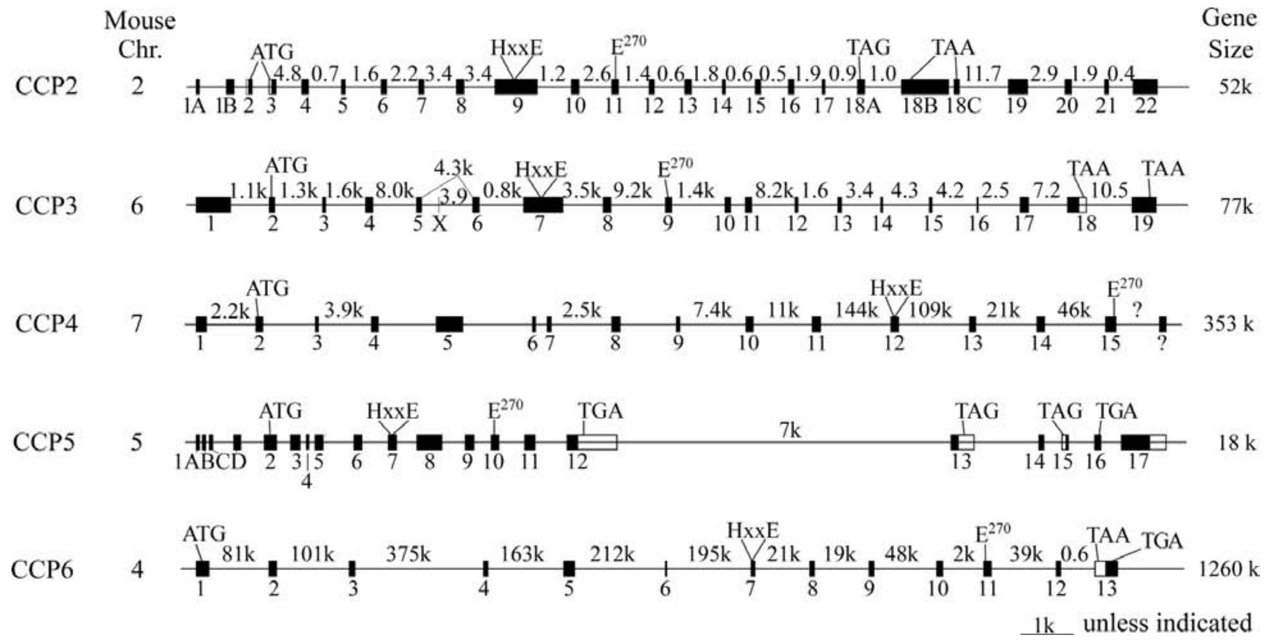


Figure 36. Gene structure of Nna1/CCP1 homologs. Filled boxes represent exons, and open boxes indicate alternatively spliced regions of exons. Size of exons and introns correspond to scale bar shown (lower right), unless indicated. Initiation ATG(s) and stop codon(s) are indicated. Active site amino acids that are near N-terminal region of CP domain (the zinc-binding HxxE) and C-terminal region of CP domain (E^{270}) are indicated. Chromosome is indicated on left and size of gene is indicated on right (kb). Source: (Kalinina et al., 2007)

3.7.4.5. Alignments and modeling of the CCPs

The amino acid sequences of the CCPs have similarities to the 300-residue CP catalytic domain of A/B and N/E family CPs. In addition, all six mouse CCPs have a 120–150 residue N-terminal domain that is moderately conserved among CCPs. This N-terminal domain begins with the FEGSN sequence and ends with the YPYTYS sequence. The length and location of the 120–150 residue N-terminal domain in the CCPs are similar to the pro domain within A/B family CPs, although there is no substantial amino acid sequence similarity. Nna1/CCP1 has a long N-terminal extension of over 500 amino acids, while CCP5 has only 10–15 amino acids preceding the N-terminal domain. There is also considerable variability in the length of the C-terminal extension following the catalytic CP domain. The various CCPs show no sequence similarity among their N-terminal or C-terminal extensions.

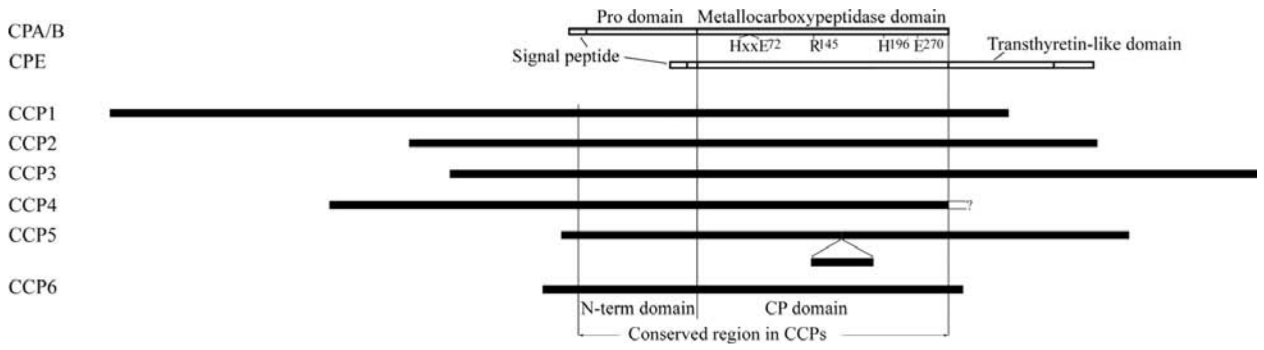


Figure 37. Alignment of CCPs. Relative positions of CP domains are indicated for CPA/B subfamily enzymes and also for CPE, a prototype of N/E subfamily. Key residues in CPA/B subfamily CPs are indicated such as the HxxE sequence, R145, H196, and E270; same residues are present in comparable positions in N/E CPs and in each of the CCPs. In addition to the CP domain, CPs in the A/B subfamily all contain an N-terminal pro domain of 90–100 amino acids, and all members of the N/E subfamily contain a conserved region on C-terminal side of the CP domain, which has structural homology to transthyretin. All A/B and N/E subfamily CPs also contain a signal peptide domain, whereas none of mouse CCPs have this domain. Each CCP contains an N-terminal domain that has amino acid sequence similarity to other CCPs but not to other subfamily CPs or to proteins in the various databases. In alignments small gaps or insertions required to align the various sequences are not shown; only large insert in CCP5 is indicated. Source: (Kalinina et al., 2007)

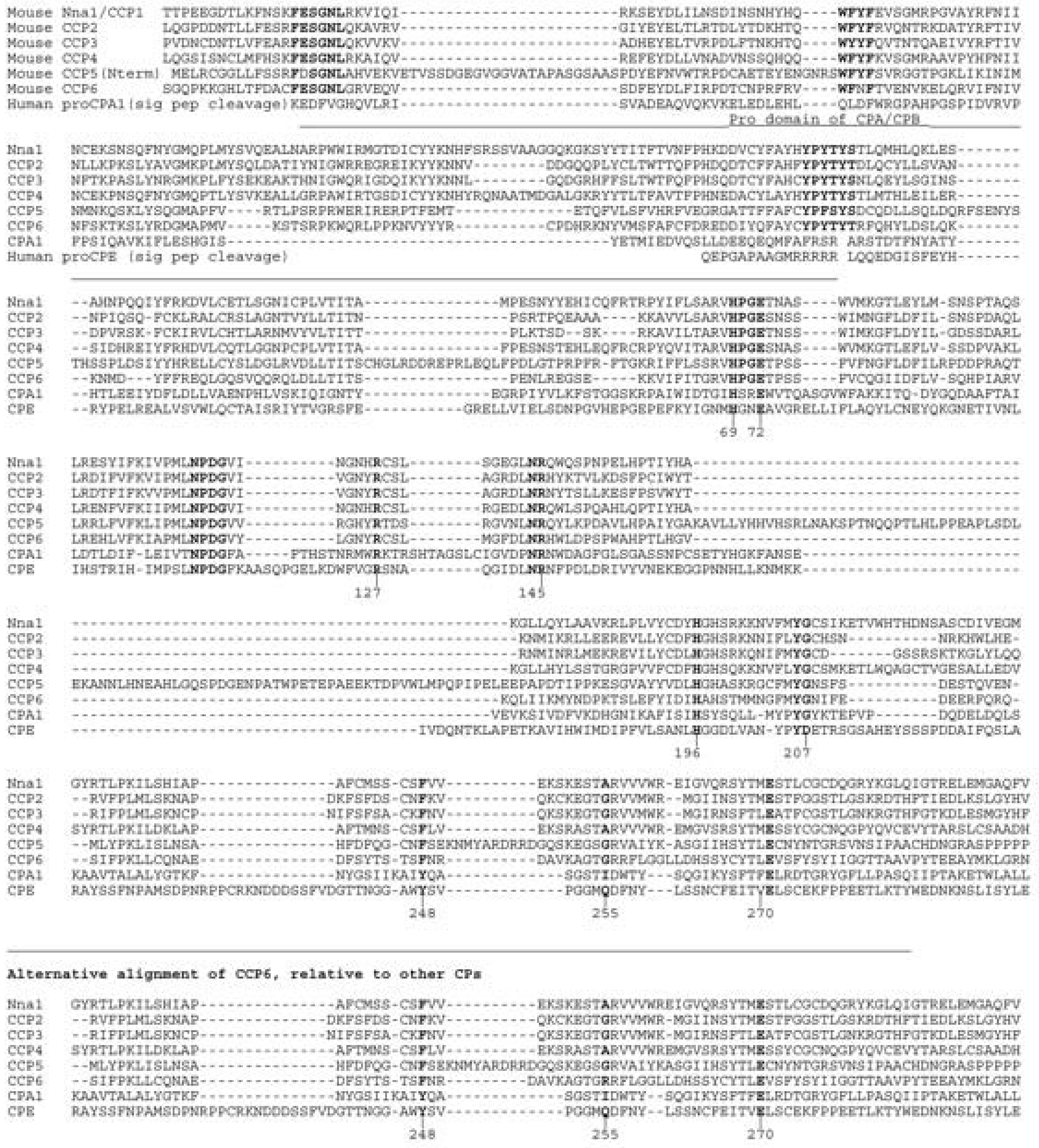


Figure 38. Amino acid sequence alignments of N-terminal and CP domains of each of CCPs and representative members of A/B and N/E subfamilies. Indicated sequences of human proCPA1 and CPE start immediately following the signal peptidase cleavage sites; ~90 residue pro domain of CPA and 15-residue pro domain of CPE are shown. CCP sequences start 10–14 residues upstream of conserved N-terminal domain. Key active site residues and/or other conserved motifs are indicated in bold. Numbering below alignments represents position within bovine CPA, numbered relative to first amino acid after removal of pro domain. Source: (Kalinina et al., 2007)

Within the N-terminal and the CP domains, most of the CCPs require similar short gaps or inserts (<20 residues) to align both the sequences and the predicted secondary structure elements with other CPs. The exception is CCP5, which contains several large inserts relative to the other CPs. Structural modeling suggests that mouse CCP catalytic domains display the typical fold containing an alpha/beta/alpha sandwich structure with an antiparallel beta-sheet of eight strands. The residues involved in the coordination of the active site zinc share the same conformation described in other metalloCPs of the A/B and N/E subfamilies. The Zn atom is held in place by penta-coordination with two His residues, one Glu residue, and a water molecule (data not shown). A comparison of the structure of mouse CCP1 and bovine CPA gave a root mean square of 1.602 and a DRMS of 1.191, indicating that the two structures are quite similar.

There are several motifs found in all CCPs, some of which correspond to active site or important structural residues of A/B and N/E family CPs. Others, such as the FESGN motif, the WFYF motif, and the YPYTYS motif found in the N-terminal domain of the CCPs, have no counterpart in other family CPs. Of these regions, the FESGN corresponds to the beginning of the conserved N-terminal domain. Interestingly, the YPYTYS sequence corresponds to the junction between the N-terminal domain and the CP domain, which is located □70 residues upstream of the HxxE sequence.

Motifs common to the CCPs and other CP families include the zinc binding residues His69, Glu72, and His196 (by convention, the numbering system of bovine CPA and CPB is used, which assigns residue 1 as the N terminus after pro peptide removal). Other conserved residues include Arg127, which in the A/B and N/E family CPs binds to the carbonyl bond at the cleavage site, and Glu270, which transfers a proton from the incoming water molecule to the leaving amine. Another conserved motif is the NPDG sequence located upstream of Arg127; this NPDG sequence is highly conserved in all CPs and presumably plays a role in the structure or interactions of the catalytic domain. Based on the presence of the active site residues that are critical for catalytic activity and the modeling results that suggest these residues are properly located in the 3-dimensional structure to perform a catalytic role, it is likely that the CCPs encode functional peptidases. This proposal is supported by experimental results in the accompanying paper, in which a *C. elegans* CCP homologue was shown to be enzymatically active.

Analysis of the residues in the putative substrate binding site allows for predictions of the cleavage specificity of the CCPs. The presence of an Asn and Arg in the positions equivalent to 144–145 of CPA/B implies that the CCPs will cleave C-terminal residues (in CPA and CPB, Arg145 binds to the carboxylate of the C terminus of the substrate). Another critical substrate-binding site in CPs is residue 255. It is not possible to accurately predict the residue in the position equivalent to 255 for any of the CCPs due to limited sequence similarity with A/B or N/E CPs in this region. For Nna1/CCP1 and CCP4, the residue in position 255 is likely to be an Ala, and for CCP2, CCP3, CCP5, and possibly CCP6, a Gly. If so, then these enzymes would presumably cleave C-terminal hydrophobic and/or basic amino acids, based on the broad substrate specificity of an insect CP with a Ser in position 255. For CCP6, it is also possible that position 255 is occupied by an Arg. This would imply that CCP6 cleaves acidic C-terminal residues. In the modeling studies, the difference between models was small since the particular region involved does not offer a high local stability within the model.

3.7.4.6. Discussion

Nna1/CCP1 and the related CCP genes encode active enzymes that function in the processing of cytosolic proteins such as tubulin. Modeling and analysis of the residues in the putative substrate binding pocket suggests that Nna1/CCP1, CCP2, CCP3, CCP4, and CCP5 have similar specificities toward bulky C-terminal residues, such as hydrophobic and/or basic amino acids. CCP6 either has a similar specificity as the other CCPs or else it cleaves acidic residues; modeling could not resolve these two possibilities.

All of the mouse CCPs considered lacked an N-terminal signal peptide sequence, in contrast to the previously characterized CPs of the A/B and N/E subfamilies, which function either in the secretory pathway or following secretion. By experimental evidence, the cytosolic distribution of CCP2, 5, and 6 observed seem to be consistent with the absence of a signal peptide. Previously, human Nna1/CCP1 was reported to show a cytosolic and nuclear distribution when expressed in primary cortical neurons as a fusion protein with GFP (Estébanez-Perpiñá et al., 2001).

By different experiments in several cell lines, Nna1/CCP1 was shown to be predominantly localized to the cytoplasm. However, CCP5 showed a nuclear-like distribution in each cell line tested, with some lines showing a more pronounced nuclear staining than others. The deduced amino acid sequence of CCP5 contains several stretches of three or more basic residues in the C-terminal region; these may function as nuclear targeting signals.

In addition to the CP domain that is conserved in all mouse CCPs and all other members of the M14 CP family, there is also an N-terminal domain immediately adjacent to the CP domain that is highly conserved among mouse CCPs but not with other M14 subfamily CPs. The other subfamilies of M14 CPs each contain a conserved domain of about the same length (100 amino acids) that functions in the folding of the CP domain, although there is no sequence similarity between subfamilies. In the case of the A/B subfamily of M14 CPs, the N-terminal pro domain functions both in folding of the CP domain and in maintaining the enzyme in an inactive state until cleavage by an endopeptidase. N/E subfamily members do not have this pro domain and instead have a domain immediately to the C-terminus of the CP domain; this C-terminal domain has structural but not amino acid sequence homology to transthyretin. Unlike A/B CPs, the transthyretin-like domain of N/E CPs does not need to be cleaved before the enzyme is fully active. Although the CCPs appear to be more similar to the A/B subfamily by the presence of an N-terminal conserved domain and the absence of a transthyretin-like C-terminal domain, the amino acid sequence similarity is too low between the pro region of CPA or CPB and the N-terminal domain of CCPs to permit modeling. Also, it is not clear if the N-terminal domain of the CCPs is removed like the pro domain of A/B CPs or if it remains attached like N/E CPs.

Mouse Nna1/CCP1 was proposed to contain an ATP-GTP binding motif (Harris et al., 2000). Although this sequence is not conserved in other members of the mouse CCP family or in a *C. elegans* homologue commented in the previous sections, the cleavage of small peptide substrates by the *C. elegans* enzyme seems to be stimulated by ATP or ADP (Harris et al., 2000). This stimulation might reflect the movement of an inhibitory domain away from the active site.

The role of the tyrosinylation and detyrosinylation of tubulin is likely to contribute to microtubule stability and/or function. In various human carcinomas, the extent of tubulin processing correlates with tumor progression (Lafanechère et al., 1998; Kato et al., 2004). Thus, inhibitors of the these studied proteins may have value as therapeutics in cancer treatment.

3.7.5. Current status

After the resulting publications of the work mentioned in the pages above (Rodriguez de la Vega et al., 2007; Kalinina et al., 2007), different other articles have reinforced the relevance of this new carboxypeptidase subfamily in existing key biological processes.

Notably, a following paper confirmed the involvement of mouse CCP1 (cpd) in α- and β-tubulin Glu processing (Berezniuk et al., 2012). Because of its wide presence among Eukarya, some of the members of this family have been highlighted as one of the components in tubulin posttranslational modifications in what is being referred as tubulin code (Verhey & Gaertig, 2007).

CCP functions, which may be directly involved in axonal regeneration, axogenesis, dendritogenesis, and ciliary axoneme seem to be related to cilia and their basal bodies. It is speculated that during eukaryotic evolution there may have been a transfer of these functions from these latter locations to inside the cell in axon microtubules (Wehenkel & Janke, 2014; De La Vega Otazo et al., 2013).

In the meantime, the first CCP structure has also been resolved. That structure, from *Pseudomonas aeruginosa* bacterium, named PaCCP (Otero et al., 2012), displays a novel β-sandwich N-terminal domain followed by the classical carboxypeptidase α/β-hydrolase domain. The actual crystal structure appeared to be an inactive version and might only be active against intracellular or specific substrates. In any case, it opens a window for more accurate homology modelling against eukaryotic M14D candidates than the ones described above.

4 - DISCUSSION AND FUTURE PROSPECTS

10 years before the conclusion of this thesis, most of biological scientific community was still not aware of the importance of data management for generating new knowledge. Only after genome sequence projects contributed huge amounts of sequence data to the public databases, and different laboratories purchased their first high-throughput machinery, it started to become clear that data was not a mere nuisance anymore. Handling data at their different stages is an unavoidable necessity for performing any meaningful research nowadays. And, this is not only for practical and technical research, but also for basic research itself, when more nations are joining in this innovation effort (Qiu, 2014).

However, at the time of writing, despite the situation is being finally acknowledged, most research facilities endure an important shortage of human capabilities for facing this already present scenario. There are still very few researchers or technicians with the suitable skills for these computational challenges (Rubinstein & Chor, 2014), and worst of all, the academic and institutional milieu does not always provide a proper professional path for these experts, who may decide to divert their careers to other sectors (Chang, 2015).

In the pages of this document, we have described with many peer-reviewed examples some of the different necessary steps for developing a full long approach to data in Bioinformatics, or widely speaking, even in present-day biological sciences. Hopefully, all these experiences can be a useful guide for other research laboratories that may be struggling to join this “data revolution” in a good shape.

First of all, as a preliminary step to any later stage, it is important to review and adapt any existing tools that may be relevant for our research so they can work in this massive data paradigm. This is what we initially performed with many of the tools described above: ProtLoc, TranScout, TransMem or Bypass. Achieving this requires a good knowledge of the actual tools, both at the technical and scientific level, and also being openly exposed to different programming practices (Dudley & Butte, 2009), so it may be feasible to perform any wanted analyses within the constraints of our computer infrastructure.

Concurrently to this need, there is an ongoing emphasis on the possible benefits of using “the cloud” in Bioinformatics (Krampis et al., 2012; Schatz et al., 2010). However, we should not forget that the simplification in the “infrastructure burden” (computing and storage hardware), specially if externalized —because in-house 'clouds' are also an existing possibility—, normally entails a major complexity in routine DevOps (Edwards, 2010) and, in the case of Bioinformatics, particularly in storage management.

As already commented during the text, as hardly-manageable data became to accumulate, the WWW got the merits to become the natural platform where all these data could be not only exchanged but also consumed by the final users.

The Web leverages the inconveniences of having to deal with different operative systems at the client-side. Likewise, the emergence of HTML 5, the progressive displacement of 3rd party plugins (e. g., Flash or Java applets) (Gille et al., 2014) and the improvements of client programming languages such as JavaScript (Corpas et al., 2014) or technologies such as WebGL (Rego & Koes, 2015; Rose & Hildebrand, 2015), are minimizing the hindrances that actually existed for having to adapt your work for the experience of visitors with different browsers.

Moreover, even though final users were initially fellow researchers, more and more often, other user profiles, such as medical doctors, patients and, although arguably, also citizens (Franzoni & Sauermann, 2014) should be considered.

Both the above mentioned tools and the databases that were created afterwards, such as TrSDB or ArchDB, could not be understood without being hosted on a web server. On one side, the former research tools and their processed predictions could be easily accessible to non tech-savvy scientists so they could test their own experimental hypothesis (e. g., checking putative transcription factors). And, on the other side, specialty-focused bioinformaticians could take advantage from the resources we provided, for instance for refining loops of protein structural models.

However, as it is the case with literature mining and reading (Lok, 2010; Round & Campbell, 2013), directly accessing and browsing the presented huge amounts of contents and data may not be practical enough and abusively time-consuming, specially when researchers have the chance to look up different alternative or competing resource websites.

At the strict data level, this problematics had already been handled by different initiatives, notably BioDAS (Dowell et al., 2001) for gathering annotation from different interconnected and separate resources, and which was used in platforms such as ENSEMBL or EBI. However this system is going to be discontinued in both locations (EMBL-EBI, 2015; ENSEMBL Blog, 2015) in the near future in favor of either track hubs, an approximation pioneered by UCSC Genome Browser (Rosenbloom et al., 2014), and/or REST webservices.

At the more analysis/processes level, we had BioMoby (Wilkinson & Links, 2002), which acted as a registry of webservices. As in the previous case, it has been progressively abandoned in favor of more generalistic REST approaches, or alternately evolving to Semantic Web ones, such as SADI (Wilkinson et al., 2011).

At the time of writing, I would dare to say that is hardly justifiable any newly published biological resource that may not consider some kind of programmatic access, both for biological databases or analytic web services. Considering the huge amount of data that are available at public databases and also all the information researchers may need to handle in their own studies, this may be the only practical way a wide public can take advantage of the possibilities of Bioinformatics.

Accordingly, some of the applications that we adapted in the previous pages, concretely ProtLoc, TransMem and Bypass, will be soon available as REST services as well.

Although there are initiatives such as BioCatalogue (Bhagat et al., 2010), a repository of data and web services accessible via an API, we might lament that, despite some attempts such as BioXSD (Kalas et al., 2010), there is not any existing successful acknowledged exchange format or schema so far. This is also aggravated by the fact that XML format is not anymore the most common web exchange format, but JSON. Whereas, the former format does have schema widely-used description formats (such as XSD, and before DTD), usage of JSON Schema (Court & Zyp, 2010), an equivalent description specification for JSON, is practically anecdotal so far.

An alternative approximation would be recurring to Semantic Web solutions. Arguably, because its sophistication may have even repelled Bioinformatics technologists in the past, it is worth noting that websites such as BioPortal (Whetzel et al., 2011) are being populated by an increasing amount of ontologies describing very different biological aspects. Specifications such as JSON-LD (Sporny et al., 2013) for exchanging linked data (semantic web information) and flexible content server solutions such as Semantic MediaWiki (Krötzsch et al., 2006), mentioned in one of the accompanying appendices, may be the definitive chance for closing the gap between web formality and everyday practice.

Of course, all these massive analysis web-enabled paradigm described above requires the development of programmable web robots (bots). An alternative option is using integrative platforms, that may hide to non-technical users the inherent complexity at the level of workflow specification (Amstutz, 2015) and management (Goecks et al., 2010; Oinn et al., 2004) or data analysis and visualization (Chelaru et al., 2014).

Apart from the strictly technical aspect of handling data and how they are processed, one key component in their management is their actual annotation. Indeed, no organization of data is possible, and so no database can be built from them, if no kind of annotation is previously done. However, data which can be retrieved from public biological databases have already a huge amount of, what can be called, *metadata* (e. g., tissue/origin or provider) that can be already directly used for annotation purposes. In fact, some researchers have already tried to go one step beyond an extract additional metadada from the social component of the Web (B. M. Good, Tennis, & Wilkinson, 2009; B. Good, 2009).

In any case, in this document we have approached annotation with applications such as Bypass in a more *classical way*: taking into account homology search and the physiochemical properties of the involved molecules. As it can be seen in literature, this is far from being a closed topic, and there is still a huge demand (Ijaq et al., 2015) for these kind of solutions, which are often handled precariously in many laboratories. So far this approximation has been quite successful for finding out possible moonlighting cases in proteins (Hernández, Calvo, et al., 2014; Hernández, Ferragut, et al., 2014).

Among the later improvements in our annotation platform, I would highlight its Gene Ontology integration. By crossing annotations from these very curated ontologies, at the evidence level we may choose (Škunca et al., 2012), we might ideally find out newer predictions or contrast existing ones.

Apart from this, one interesting addition to be included in this platform may be interaction information, which has already been defended to be significant in multitasking discovery of proteins (Gómez et al., 2011).

Concurrently to the works included in this thesis —mostly focused on proteins—, as a consequence of an increase experimental sensitivity, there has been a boom in the knowledge of non-coding RNAs (apart from the already well-known tRNAs and rRNAs). These other non-coding RNAs (ncRNAs), normally classified according to their length and/or location (e.g. lncRNA, exRNA, microRNA, snRNA, etc. (Morris & Mattick, 2014), have been assigned to different processes, notably, RNA silencing (Eggleston, 2009), transcriptional regulation (Anders et al., 2012) and even intercellular communication (Mittelbrunn & Sánchez-Madrid, 2012). Thus, RNAs are being studied as key molecules for better understanding epigenetics (Castel & Martienssen, 2013) and, relatedly, observed Lamarckian-like inheritance (Devanapally et al., 2015).

Coherently, we may wonder whether some of the methodologies introduced in the pages above could be translated (or rather *retrotranslated*) to these molecules, despite RNA Bioinformatics is rather different from that of proteins.

First of all, considering just one dimension, starting from a smaller alphabet (~4 vs ~20 letters), it is more difficult to find in a defined-size region the same meaningful information (e.g. homology), that could be retrieved in a protein counterpart alignment. Secondary and tertiary structural work is also normally managed by different software than that used with proteins (Gardner & Giegerich, 2004; Laing & Schlick, 2011).

In any case, even taking into account these differences, it is worth considering some of the already used informational approaches in proteins for handling the huge outcome of data resulting from projects such as ENCODE (Myers et al., 2011) —with all the ongoing and controversial discussion about the prevalence of what was named junk DNA (Palazzo & Gregory, 2014) and what function really means (Germain et al., 2014)—.

Although we have been mostly dealing with biological data from a broad-view perspective, it must not be forgotten that sometimes some of the more meaningful achievements can arrive focusing on specific aspects. This is actually what it was attained with the bioinformatical characterization of new metallocarboxypeptidase family M14D. This is an example that demonstrates that no novel methodology may be needed to be developed for delving into certain hypotheses, but a deep-enough understanding of both the biological issues and existing applications (in this specific case, Hidden Markov models or homology modeling software). At the time of writing, this very protein family is still in need of further characterization for having a clearer explanation of its diversity at the organism and genomics level, and Bioinformatics can help ongoing experimental practices in this effort.

Undoubtedly, there may be many other little-know protein families, even with human members, that may have not received the deserved scrutiny yet. For handling this, the first thing is ensuring a proper (inter)institutional collaborative environment between wet-lab researchers and bioinformaticians. This may also be promoted at a larger community level by using networking and content-sharing tools such as wikis (Romano et al., 2011). A non-excluding possibility could also be training experimental biologists into Bioinformatics (Fernandes et al., 2012). This can be beneficial for the mutual collaboration and, above all, for the actual research.

As a final sum-up, as it has been explicitly detailed in the previous sections, we can see an increasing need for data integration in the biological and medical fields. Moreover, we can safely foresee this is not going to slow down, for instance, with challenges such as personalized medicine (Fernald et al., 2011).

If it was ever the case, in nowadays Bioinformatics, we can hardly regard anymore data management as a secondary aspect that can be dealt within generic IT practices. As much as possible, it is important to have a holistic enough vision to model biological data and make it accessible in a useful and rich way. This thesis has been written guided with this exploration, adhering as far as it was possible to the scientific dogma of reproducibility and the philosophical direction of openness in the technical practice.

Hopefully this document and its associated works and code can become a useful resource and reference for all those new bioinformaticians that might decide to assume the challenges of biological data generation, management and interpretation.

5 - CONCLUSIONS

Different methodological advances for handling and processing data in a Bioinformatics context are presented within the pages of this work. Except from the initial raw retrieval (e. g., from sequencing machines), the whole succession of data manipulation is considered: ranging from the adaptation of existing resources to the set up of data management environments, public web services, and ultimately their interpretation.

1. A set of tools developed at IBB-UAB were actively rewritten in order to be suitable for present-day computational paradigm. These applications, ProtLoc (cellular location assignation), TransMem (transmembrane stretches prediction), TranScout (transcription factor signature detection) or Bypass (protein functionality annotation), were wrapped into scripting languages (e. g. Perl) so they could be used within Bioinformatics pipelines for performing high-throughput analyses.

2. Those very tools were also modified and adapted so they could be used as websites by the whole scientific community. Likewise, by being deployed as web services as well, they could be more easily integrated into diverse workflows in the future.

3. By adapting those programs into massive analyses pipelines, it was possible to process genome sequence data from different model organisms. Taking advantage of the possibilities of relational database management systems, the resulting outcomes could be fit into a publicly available web databases.

4. As example, a public resource named TrSDB was created based on TranScout previous classification of transcription factors. This was also enriched with ProtLoc and TransMem predictions as discriminant criteria. As additional accompanying results, different candidate sequences were compiled so they could be further examined for a putative transcription factor activity.

5. Using the same approaches as in the previous case, ArchDB, a database of protein structural loops was designed and published. In that example, this was created starting from ArchType —and automatic procedure program for protein loop classification based on loop conformation and geometry— plus a posteriori clustering of different involved protein structure datasets. Among them, it is worth noting a biologically selected group such as protein kinases. That resource represented an important repository especially useful for the refining structural protein models, which are normally problematic at loop regions.

6. Bypass, an application which uses fuzzy logic for highlighting additional functionalities of proteins from PSI-BLAST analyses and their intrinsic physiochemical properties, was ported into a website. Moreover, in order to facilitate its usage by non-technical profiles, it was integrated in a later stage as a part of a sequence data management platform.

7. This latter framework made possible the exploration and study of protein moonlighting, that is, the presence of different biological activities among proteins that were supposed to have an only function in the cell.

8. By using NoSQL database approaches, still relatively unexplored in Bioinformatics, it was possible to host interlinked batch analyses in a coherent way within a web environment. Moreover, in order to empower users for exploring annotation of the included protein sequences, a custom Gene Ontology enrichment method was integrated. This system was also open and flexible enough for accommodating additional webservices to improve annotation of gene products.

9. Additionally to the previous studies, a group of little known proteins, previously mentioned as Nna1-like or ATP/GTP-binding proteins, were characterized by Bioinformatics methods to be part of a distinctive group of metallocarboxypeptidases. This was concretely performed thanks to Hidden Markov Model search of curated alignments against public databases and phylogenetic analyses.
   This novel family of proteins, assigned under M14D group of Merops classification, was also referred as cytosolic carboxypeptidases because of their main cellular location.

10. Thanks to protein homology modeling and active site inspection, it was proved that some members could present enzymatic activity, for instance, notably Tubulinyl-Tyr carboxypeptidase activity. Therefore, they may be key participants in biological processes such as tubulin processing in microtubules and other cell components.

11. Parallely to these latter studies, up to 6 Nna-1 like proteins in mice were described with the help of computational approaches. This protein abundance highlights the relevance of these molecules, still little-know in life processes, representing a promising starting point for future experimental studies.

6 - APPENDICES

6.1. SQLite as convenient in-situ DBMS for analysis

As we have detailed in the main pages of this document, database management systems (DBMS) are truly useful environments for handling huge amounts of data commonly present in Bioinformatics field. However, apart from the obvious flat-file databases, relational and NoSQL DBMS normally require a previous setup consisting in several steps. Moreover, it may also happen that bioinformaticians may not have full administrative privileges in the machines where they perform their analyses. In these circumstances, installing a DBMS can be problematic.

A possible solution is using SQLite (Owens, 2010), as its name suggests, it's a 'lite version' of a typical SQL relational DBMS. Instead of establishing a connection via a UNIX socket or a network port for interacting with databases, databases are handled as individual binary files. One database per file.

As a general example application, at the time of writing, SQLite is for instance the preferred local storage option of browsers such as Firefox when storing browsing history, passwords and other user configuration parameters.

In the same philosophy, SQLite is a convenient in-situ relational DBMS for command-line analyses. It's quite common that both input or output files are simple human readable text files. Although UNIX piping allows to handle IO (input-output) between different applications almost transparently, sometimes is inevitable to recur to intermediary files. And, these intermediary files are perfect candidates to be replaced by database-indexed rows of data.

Some notable exceptions could be binary files, such as images, but even in these cases, databases can be suitable for storing their file paths.

As stated above, when dealing with the administration of a DBMS is an issue, having a file working as a whole database is really convenient. One database file can be used for replacing multiple intermediary files and copying/moving the database file with the rest of associated scripts and input/output files is easier than in other scenarios. And what is most, it provides a more powerful approach to query for information (via SQL in this case). Indeed, there are existing end-user solutions that are using SQLite in the background for handling huge CSV files for this latter reason (Adaszewski, 2014).

As a caution, despite SQLite accepts many concurrent open connections, there can be file lock issues if the database file is placed in a NFS, which can be a possible situation when submitting analyses in a cluster environment.
On the other hand, even though SQLite is open-source and nowadays present in most Linux distributions, since its outcome is a binary format, this is not necessarily going to be easily readable in all present and future systems. It's noteworthy to mention that plain text files are preferable as input and output files to be kept for the future.

6.1.1. Example application

As a part of a wider project (Gonzalez, unpublished), starting from *Mycoplasma genitalium* proteome analyses, we wanted to find out the distribution of resulting peptides among different phyla, and so elucidate new candidate essential peptides restricted to a certain taxonomic group or even present in all the phyla.

A whole set of n. peptides (starting from a threshold size) was searched against NR and NT NCBI BLAST DBs with PSI-BLAST and TBLASTN respectively allowing up to 20000 possible hits per sequence with a threshold e-value of 10.

BLAST results were stored in XML format so subsequent analyses could be performed, as it is the case of phylum distribution of certain candidate peptides.

At the time of writing, but to be changed soon (NCBI, 2014), BLAST output result do not still provide taxonomy identifiers of the matched hits. So, in order to retrieve that information, since text parsing of the outcome is not reliable enough (not all entries provide organism information in their sequence descriptions), Eutils (Sayers, 2012) was used.

EUtils are a set of web API tools developed by NCBI in order to allow automation and machine access to their DB resources via request instead of downloading and processing whole huge files at first. Although there are already existing initiatives that provide a JSON-friendly interface to Eutils (Loman & Pallen, 2010), it was preferred that the request could also be integrated into the Express framework introduced in previous sections.

In order to hurry the retrieval of information from NCBI webservice, a batch approach (as documented in their documentation) was used and mappings between GI codes and Taxonomy ID codes were stored in a SQLite file database. In that way, already stored entries correspondences didn’t need to be retrieved from an external server again and were accessible to the analysis script workflow from within the local machine.

Moreover, if other similar analyses needed to be done, e. g. from another batch of BLAST results from a different organism, the same database file could be simply reused, copied to the directory where the new analyses are being run. That database file could even be reused in other projects that may need the same mapping correspondence, just taking care to follow the same SQL schema as defined in the previous application.

6.2. MediaWiki as a biological database repository

Wikis, from Hawaiian language “wiki”, quick, were designed as a fast way to create webpages without having to recur to the higher complexity of HTML markup language (and its subsequent styling). From an actual webpage, that very page can be modified and also more pages could be added to the containing website (or deleted from). At the same time, different users can contribute by editing within the same interface provided by the browser, increasing so the possibilities of a collaborative content curation.

This is actually the case of the most well-known wiki, Wikipedia, which has already become the most comprehensive encyclopedia ever made.

Centering on the biological domain, Wikipedia itself is hosting scientifically relevant documentation projects such as Gene Wiki (Good et al., 2012),, a combination of automatic and manual annotation approaches for describing gene and protein functions (at the time of writing only for *Homo sapiens*). Moreover, reference resources such as Xfam (Finn et al., 2010) hosting catalogues of protein or RNA families, are directly exposing their entry descriptions to Wikipedia.

However, apart from these examples, Wikipedia and other Wikimedia Foundation projects (such as WikiSpecies), there are other independent wiki projects that deal with biological knowledge and discovery. In most of the cases, they also use MediaWiki, the software behind Wikipedia. As examples, we have Subtiwiki (Michna et al., 2014) , centered on different aspects of the biology of *Bacillus subtilis* microorganism, WikiPathways (Kelder et al., 2012), an actual encyclopaedia of metabolic pathways from different species or Proteopedia (Prilusky et al., 2011), a visual 3D encyclopaedia of proteins and other biomolecules.

Some of these cases include important customizations that make the wikis depart from considerably from Wikipedia implementations.

Another example of further complexity using MediaWiki is Semantic MediaWiki (Krötzsch et al., 2006; Krötzsch et al., 2007), an extension over the former software, that enables having a database-like layer over a wiki by using a wiki-like syntax, and so allowing users to contribute with structured data in a transparent way.

Even though the underlying foundation of Semantic MediaWiki is the very Semantic Web, it also exposes a close relational-like model, where categories or namespaces act as tables, pages as rows and properties plus values as fields plus values.

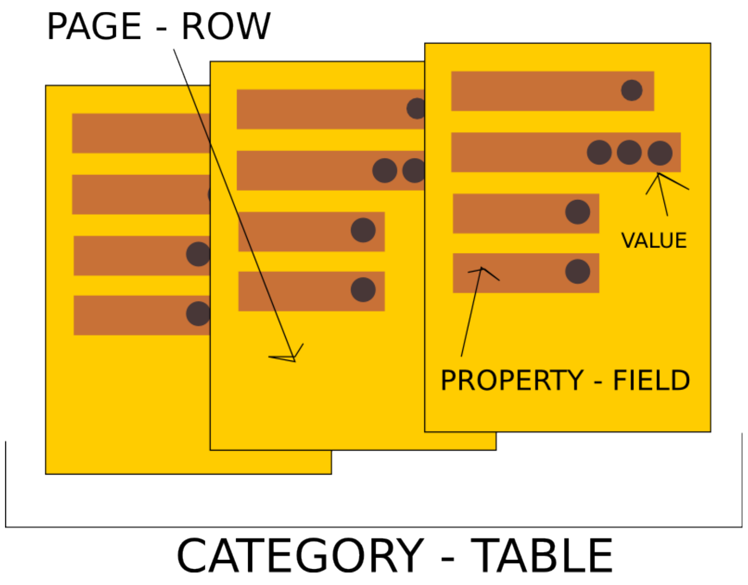


Figure 40. Representation showing how a Semantic MediaWiki deployment can match into a relational database schema.

Likewise, a query language, named ask (in contraposition with SQL in RDBMS), is provided for drawing the relationships from the above-mentioned entities.

```
{{#ask:[[Category:Genes]][[Description::+]]|?Id|?Chrom|?Start|?End|?
Genome|format=table|limit=100}}
```

Its Semantic qualification refers to the fact that the extension also enables wikis that include it to act as nodes of the Semantic Web and export their contents in RDF formats. The Semantic Web envisions a World Wide Web made up of data that might be interlinked (linked data) in a similar fashion as a worldwide database. A paradigmatic example in the biological domain are initiatives such as Bio2RDF (Belleau et al., 2008), SPARQL endpoints of UniProt (The UniProt Consortium, 2014) or the Linked Data platform of Nature Publishing Group (http://data.nature.com).

As example of Semantic MediaWiki powered wikis we have SNPedia (Cariaso & Lennon, 2012), a database of Human SNPs mostly maintained by bots (automated scripts that interact with wiki websites), or PRGDB (Sanseverino et al., 2013), an expert curated of plants resistance genes information.

6.2.1. Example applications

Because of Wikipedia popularity, MediaWiki offers a well-known interface for most end-users (here including scientific community as well). Moreover, it provides a quick and easy way to edit content in a way quite close to WYSWYG. Actually, at the time of writing, Wikipedia is already allowing editing via a WYSWYG interface.

For that reason, once we have some properly stored data (e.g. in a MySQL databases), MediaWiki can provide an accessible interface for presenting those data and their mutual relationships. Starting from different resulting analyses in a parseable text format (e.g. CSV), page entities can be generated, which could be, in turn, hyperlinked (interrelated) following wiki principles and syntax.

As a paradigmatic application example, we have websites as PRGDB (Sanseverino et al., 2013) or GreeNC (Paytuvi et al., 2015, under review), which match a relational database against the above explained Semantic MediaWiki schema methodology.

Bulk data (in formats such as CSV) can be imported in MySQL relational tables and therefore queries can be set up for showing data to be printed for the visiting user. This linkage is thanks to ExternalData MediaWiki extension (Koren, n.d.), actually a custom extension based on the former one which, apart from linking external MySQL instances, it already contains keywords for the involved queries and the resulting fields, abstracting the implicit relations between the involved relational tables.

In order to print out all the different entries (for instance in the latter example case: genes including transcripts and plant organisms), different templates are designed so that they included those predefined keywords in the extension (which can used to call database-stored values). By only using page names as identifier values (such as gene alias and scientific names), we are able to print all pre-designed data in the contained templates.

As an extra benefit, all the Semantic MediaWiki assignments in the templates can actually become potential search points in the wiki. And, at the same time, they can be used for generating filtered lists (e. g., transcripts of a certain organism)
Care must be taken to preserve 1->n relationships. For this, subobjects / internal objects. In relational parlance, subobjects allow to have additional rows to be defined within the spacial context that is normally assigned to a row, that is a page.

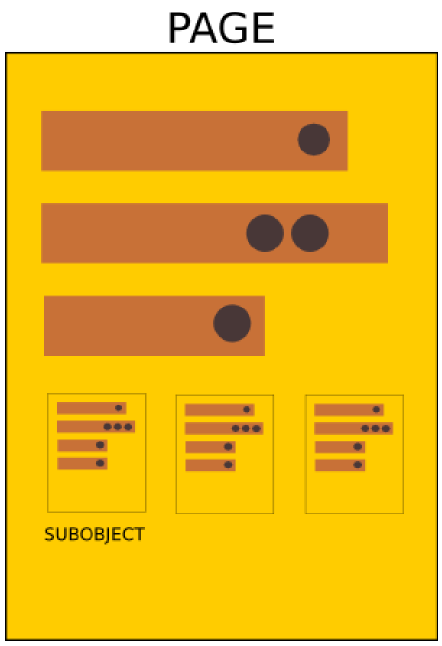


Figure 41. Representation showing how Semantic MediaWiki subobjects fit into a wiki page.

Code of the mentioned extension can be found at:
https://github.com/toniher/mediawiki-BioDB

6.3. Benchmarking different DBMS as storage systems for biological sequences

Throughout many previous pages of this document different applications that make extensive use of diverse DBMS were presented. It must be said that, in some cases, a certain solution was simply chosen because it was the most commonly (if not only) available option at the time of the application design (notably MySQL). On other cases, the platform choice was based upon the underlying structure of the data being considered (e.g., Neo4j, a graph DBMS, for Gene Ontology DAGs, and CouchDB for handling documents derived from BLAST reports, structured hierarchically, but at the same time, with variable elements).

On the other hand, the fact that a DBMS should be supposedly more suitable for a kind of data is not a full assurance that a better result is going to be achieved. Every software has its strong points and issues, and it may be convenient to convert some data to other formats or fit them in a different structure so they can be handled more efficiently in another system, and thus get a better result.

Moreover, what is a better result is also a matter of discussion, or more correctly, it should be more properly stated in light of the processes to be optimised (that is, considering time, disk space, etc.), which are the stakes that are assumed and at what costs are acceptable within some of the steps involved. For instance, in a web service, the waiting time for the user to receive any kind of outcome should be considerably reduced, but this may be at the expense of other processes (e. g., it may be necessary to precompute or cache certain values before, which represents an extra expense in CPU or disk space).

As the web becomes a more common user interface, also for biological analyses, different decisions and trade-offs must be taken into account in this scenario. This is important considering that biological data can normally represent huge files (e. g. GB). Moreover, some analyses cannot always be split into smaller processes (and those actual huge files must be wholly put into memory). This leads to the need of having to plan carefully a proper infrastructure (web server, computer cluster in the background) that may adapt to these restrictions.

Below different technology DBMS and indexing approaches are explored dealing with protein sequences. Concretely, the following functionalities are tested:

- Bulk import of FASTA sequences into a database (or indexing a flat file databases)
- Retrieval of a sequence from the database
- Import of another single sequence into the previously filled database; retrieval of the last sequence just imported and retrieval of another sequence.

For sake of homogenization, MD5sum derived header IDs and non-redundant generated files have been used.

Input data
record.id -> record.value

```
>5c23fcce0fb1dd3bd338749415729d87
MRMRGRRLLPIILSLLLIVLLSLCYFSNHLRDSSQSRKNGFLLHLPLETKRNPSNPNTPLSNLLNLTDFHYLLASNVCRKAKREL
LVTSYAGHDALRSAHRQAIPQSKLEEMGLRRVFLLAALPSREHFISQDQLASEQNRFGDLLQGNFIEDYRNLSYKHVMGLKWVSE
ECKKQAKFIIKLDDDIIYDVFHLRRYLETLEVREPGLATSSTLLSGYVLDAKPPIRLRANKWYVSKKEYPQALYPAYLSGWLYVT
NVPTAERIVAEAERMSFFWIDDTWLTGVVRTRLGIPLERHNDWFSANAEFIDCCVRDLKKHNYECEYSVGPNGGDDRLLVEFLHN
VEKCYFDECVKRPVGKSLKETCLAAAKSRPPKHGFPEIKALRLR
```

For most of the DBMS options (that is, all except flat file databases), one convenient, if not realistically mandatory, way to handle the import of many rows of data is to use existing batch options. This basically entails grouping many insert queries into one single request. This approach reduces the time log involved in establishing and closing any connection associated to the request.

```
...
conn = MySQLdb.connect()
c = conn.cursor()

batch = 1000
if len( argv ) > 1:
        batch = int(argv[1])
itera = 0

handle = open( argv[0], "r")
for record in SeqIO.parse(handle, "fasta") :
        c.execute("INSERT INTO SEQS VALUES ('" + str(record.id) + "', '" +
str(record.seq) + "' )")
        itera = itera + 1
        if itera > batch :
                      conn.commit()
                      itera = 0
if itera > 0:
        conn.commit()

handle.close()
```

Figure 42. Example code in Python showing how to do batch insert in MySQL. Unless using a builtin importing method (e. g., LOAD DATA in MySQL), which is normally more efficient, It would be very similar approach for most other DBMS.

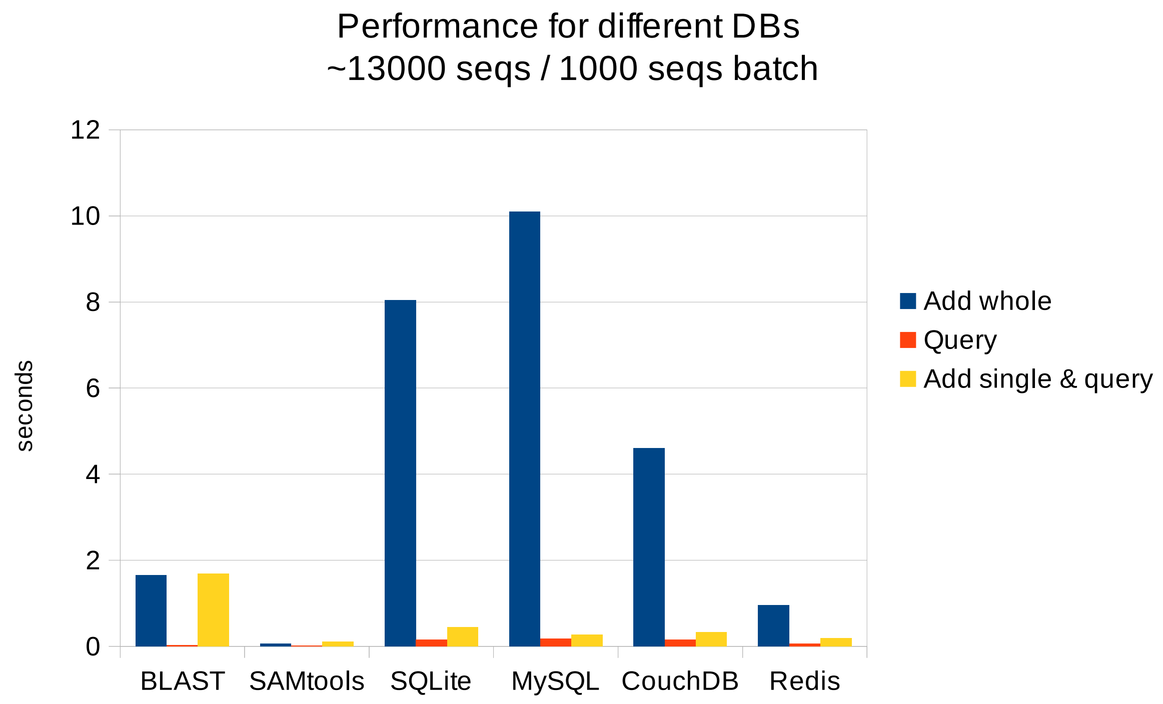


Figure 43. Chart with the performance for different solutions for droso.aa (~13,000 seqs). When using BLAST or SAMtools, no batch can be performed (at the user level) and the file is processed at once.

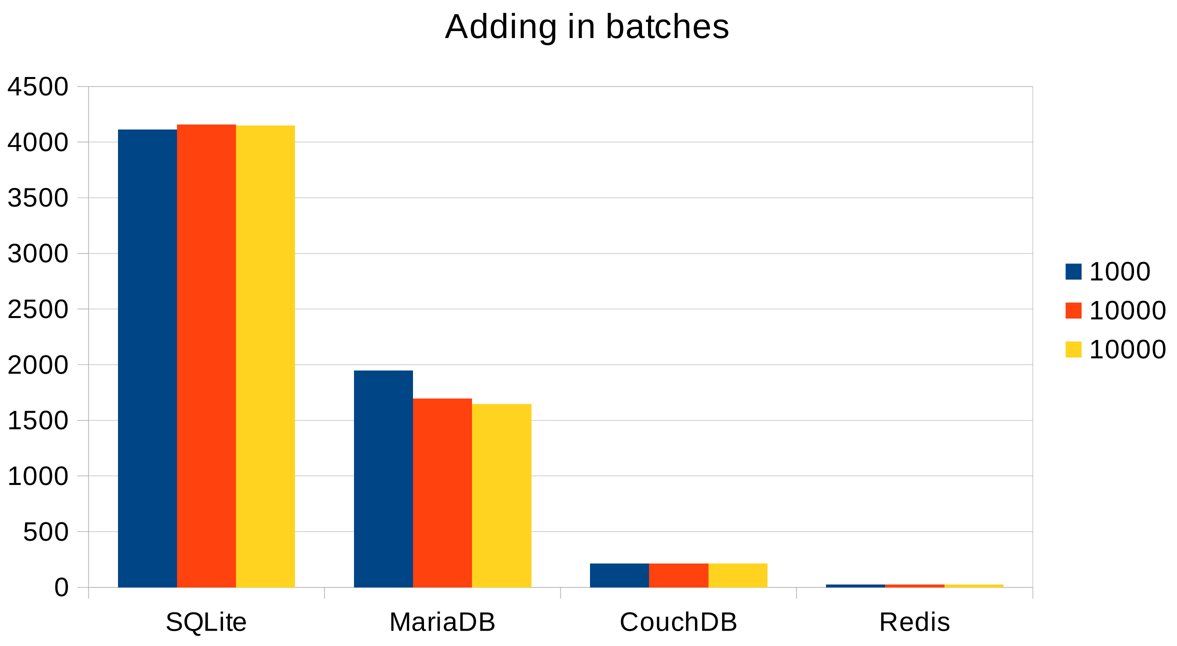


Figure 44. Performance of importing in batches with set of ~ 460,000 seqs (swissprot.aa NCBI)

As it can be seen from the figures above, it would seem that the most optimal solution when bulk-adding, single-adding and also retrieving sequences is Redis.

Since Redis is essentially a key-value store by definition, this outcome could be more or less expectable. However, at the time of writing, management (naming and listing of databases) and storing (persistence) using Redis is not so straightforward as with the other, sometimes more widely known, solutions. However, as results would seem to show, this higher complexity would compensate for the benefits of opting for this DBMS for this kind of data.

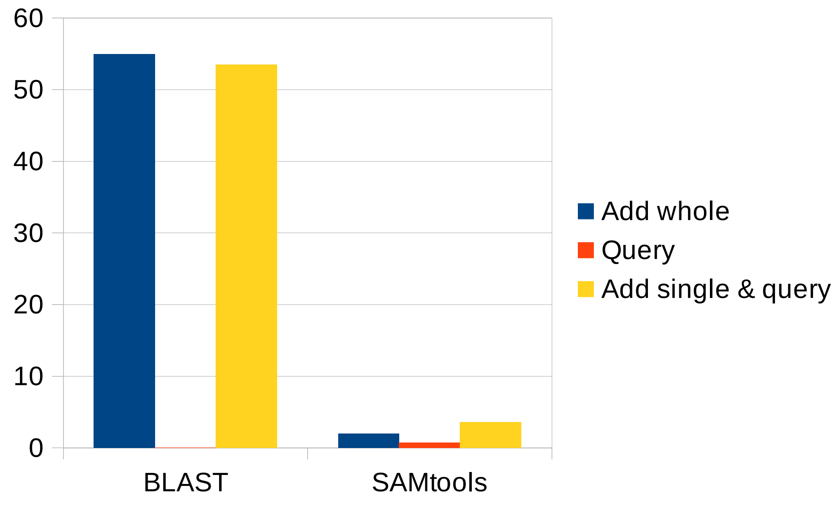


Figure 45. Comparison of performance of flat indexed methods for a bigger DB of ~ 460,000 seqs (swissprot.aa)

As it is evident from the outcome above, when using the flat indexed database options, we see that NCBI-BLAST `makeblastdb` indexing seems to be very convenient when retrieving sequences (via `blastdbcmd`), but it's rather slow when bulk importing. Moreover, when more sequences are appended to an existing flat sequences file, it seems that is necessary to start the indexing process from scratch again. In any case, it's worth to mention that by using this technology, we do not only benefit from retrieval and storing of sequences, but also with homology search capabilities (that is,. BLAST). Hence, this approach seems to be very suitable for webservices with a non-changing (or slowly-changing) amount of sequences.

On the other hand, SAMTools `faidx` function, another flat file database approach, seems to be the other way around. Bulk indexing is relatively fast and specially reindexing after appending new sequences. Compared to `makeblastdb` reindexing, it does not seem to restart the indexing process when new sequences are added. However, the retrieval of sequences by ID is not as responsive when managing a huge DB (at least with MD5sum derived IDs), and so it becomes little convenient to be used within a webservice for these latter cases. Taking into account the fewer analysis possibilities compared to the previous option, it remains as a worth-considering alternative for offline analysis (especially if many sequences are manipulated back and forth from the dataset).

Another interesting aspect is the huge bulk-import performance difference of SQL DBMS tested (MariaDB/MySQL and SQLite at the time of writing), even when using batch method, compared to builtin import methods (that require files to be converted to CSV beforehand). Specially with MariaDB/MySQL is also important to fine-tune some parameters in advance.

However, if very large files are involved, it may be necessary to split original data files into smaller chunks so they can fit into existing computer memory when read and parsed.

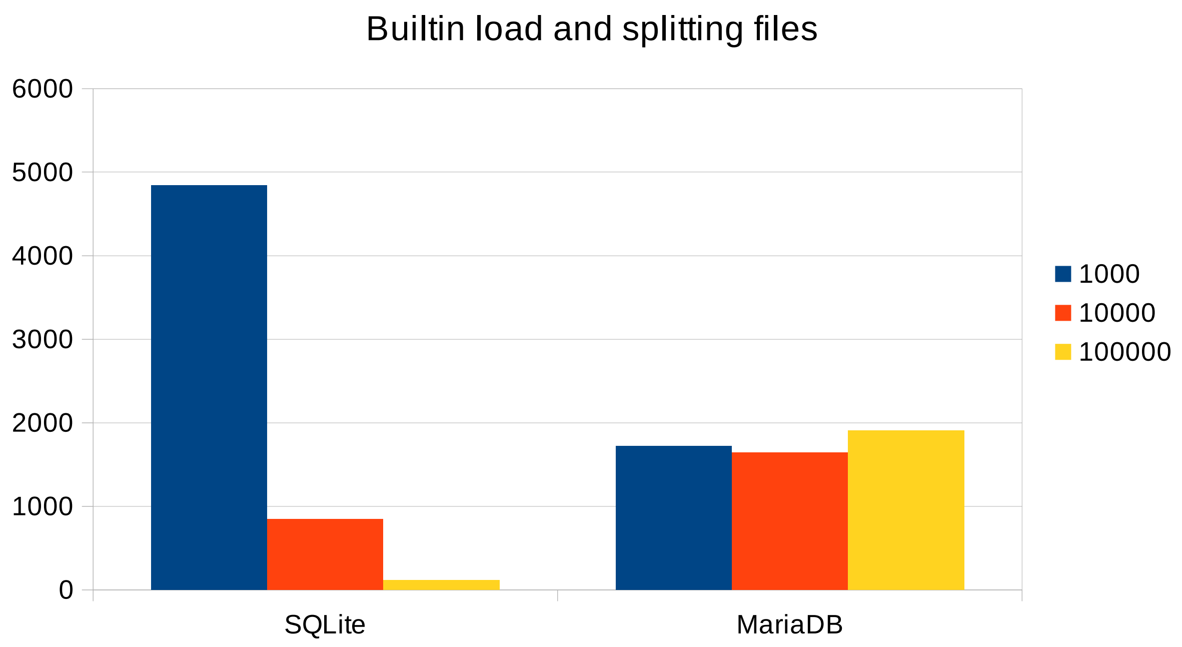


Figure 46. Performance of using builtin import methods (LOAD) for relational databases previously splitting files and converting into CSV ~ 460,000 seqs (swissprot.aa NCBI)

In any case, both single-add and retrieval are rather fast, and so both would be suitable methods to be used in web contexts (more MariaDB than SQLite).

CouchDB, a platform used for storing sequences in a previous chapter of this document, offers us an average performance when bulk-uploading, which can be improved dealing with batch chunks size, but its rather fast when single-adding and retrieving sequences, so not surprisingly, it’s a convenient platform also for web environments.

In conclusion, the benchmarks above, which can be easily extended in the future to other database platforms (e.g. Postgres or MongoDB) are a simple way to have a quick outlook of which solutions are better for a certain kind of sequence manipulation problems (e.g., user-submission via the web vs massive offline data analysis).

In many cases, different solutions can perform the same role under the same restrictions and other criteria, such as compatibility with other parts of the developed framework or the desire to use less systems because of system administration simplicity, may be as well important in the end.

At the time of writing, tests are being ported so they can be run within Docker instances. By doing this, third parties can use these tests with their own datasets and without having to install all necessary database software by themselves.

Code and reference datasets of the benchmarks can be found at:
https://github.com/toniher/biodb-benchmark

7 - BIBLIOGRAPHY

Adaszewski, S. (2014). Mynodbcsv: lightweight zero-config database solution for handling very large CSV files. *PloS One*, *9*(7), e103319.

Agrawal, G. (2008). Supporting high performance bioinformatics flat-file data processing using indices. In *2008 IEEE International Symposium on Parallel and Distributed Processing* (pp. 1–8).

Aguilar, D., Oliva, B., Aviles, F. X., & Querol, E. (2002). TranScout: prediction of gene expression regulatory proteins from their sequences. *Bioinformatics*, *18*(4), 597–607.

Alfa, C. E., & Hyams, J. S. (1991). Microtubules in the fission yeast Schizosaccharomyces pombe contain only the tyrosinated form of alpha-tubulin. *Cell Motility and the Cytoskeleton*, *18*(2), 86–93.

Aloy, P., Cedano, J., Oliva, B., Avilés, F. X., & Querol, E. (1997). "TransMem": a neural network implemented in Excel spreadsheets for predicting transmembrane domains of proteins. *Computer Applications in the Biosciences : CABIOS*, *13*(3), 231–4.

Altman, R. B. (2009). Bioinformatics & Computational Biology = same? No. Retrieved May 9, 2015, from https://rbaltman.wordpress.com/2009/02/18/bioinformatics-computational-biology-same-no/

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, *215*(3), 403–10.

Altschul, S. F., & Koonin, E. V. (1998). Iterated profile searches with PSI-BLAST--a tool for discovery in protein databases. *Trends in Biochemical Sciences*, *23*(11), 444–7.

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, *25*(17), 3389–402.

Amstutz, P. (2015). Common Workflow Language. Retrieved May 10, 2015, from https://common-workflow-language.github.io/#/

Anders, G., Mackowiak, S. D., Jens, M., Maaskola, J., Kuntzagk, A., Rajewsky, N., … Dieterich, C. (2012). doRiNA: A database of RNA interactions in post-transcriptional regulation. *Nucleic Acids Research*, *40*(D1).

Anderson, J. C., Lehnardt, J., & Slater, N. (2010). *CouchDB: The Definitive Guide*. *Substance and alcohol actionsmisuse* (Vol. 5).

ArangoDB GmbH. (n.d.). ArangoDB - the multi-model NoSQL DB. Retrieved April 14, 2015, from https://www.arangodb.com/

Aravind, L., Iyer, L. M., Wellems, T. E., & Miller, L. H. (2003). Plasmodium biology: genomic gleanings. *Cell*, *115*(7), 771–85.

Argaraña, C. E., Barra, H. S., & Caputto, R. (1978). Release of [14C]tyrosine from tubulinyl-[14C]tyrosine by brain extract. Separation of a carboxypeptidase from tubulin-tyrosine ligase. *Molecular and Cellular Biochemistry*, *19*(1), 17–21.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., … Sherlock, G. (2000). Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature Genetics*, *25*(1), 25–9.

Bairoch, A., & Apweiler, R. (2000). The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Research*, *28*(1), 45–8.

Bairoch, A., Apweiler, R., Wu, C. H., Barker, W. C., Boeckmann, B., Ferro, S., … Yeh, L.-S. S. L. (2005). The Universal Protein Resource (UniProt). *Nucleic Acids Research*, *33*(Database issue), D154–9.

Baker, D., & Sali, A. (2001). Protein structure prediction and structural genomics. *Science*, *294*(5540), 93–6.

Bard, J. B. L., & Rhee, S. Y. (2004). Ontologies in biology: design, applications and future challenges. *Nature Reviews Genetics*, *5*(3), 213–222.

Baron Schwartz, Peter Zaitsev, V. T. (2012). *High Performance MySQL: Optimization, Backups, and Replication* (3rd ed.). O'Reilly.

Bartkuhn, M., & Renkawitz, R. (2008). Long range chromatin interactions involved in gene regulation. *Biochimica et Biophysica Acta*, *1783*(11), 2161–6.

Barton, M. (2014). nucleotides · genome assembler benchmarking. Retrieved April 14, 2015, from http://nucleotid.es/

Bateman, A., & Bycroft, M. (2000). The structure of a LysM domain from E. coli membrane-bound lytic murein transglycosylase D (MltD). *Journal of Molecular Biology*, *299*(4), 1113–9.

Bateman, A., Coin, L., Durbin, R., Finn, R. D., Hollich, V., Griffiths-Jones, S., … Eddy, S. R. (2004). The Pfam protein families database. *Nucleic Acids Research*, *32*(Database issue), D138–41.

Bayés, A., Comellas-Bigler, M., Rodríguez de la Vega, M., Maskos, K., Bode, W., Aviles, F. X., … Vendrell, J. (2005). Structural basis of the resistance of an insect carboxypeptidase to plant protease inhibitors. *Proceedings of the National Academy of Sciences of the United States of America*, *102*(46), 16602–7.

Belleau, F., Nolin, M. A., Tourigny, N., Rigault, P., & Morissette, J. (2008). Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. *Journal of Biomedical Informatics*, *41*(5), 706–716.

Berezniuk, I., Vu, H. T., Lyons, P. J., Sironi, J. J., Xiao, H., Burd, B., … Fricker, L. D. (2012). Cytosolic carboxypeptidase 1 is involved in processing α- and β-tubulin. *Journal of Biological Chemistry*, *287*(9), 6503–6517.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., … Bourne, P. E. (2000). The Protein Data Bank. *Nucleic Acids Research*, *28*(1), 235–42.

Bernstein, L. S., Ramineni, S., Hague, C., Cladman, W., Chidiac, P., Levey, A. I., & Hepler, J. R. (2004). RGS2 binds directly and selectively to the M1 muscarinic acetylcholine receptor third intracellular loop to modulate Gq/11α signaling. *Journal of Biological Chemistry*, *279*(20), 21248–56.

Bhagat, J., Tanoh, F., Nzuobontane, E., Laurent, T., Orlowski, J., Roos, M., … Goble, C. A. (2010). BioCatalogue: a universal catalogue of web services for the life sciences. *Nucleic Acids Research*, *38*(Web Server issue), W689–94.

Biodados. (n.d.). MaLe-PSI-BLAST. Retrieved April 15, 2015, from http://malepsiblast.sourceforge.net/

Birney, E., Andrews, D., Caccamo, M., Chen, Y., Clarke, L., Coates, G., … Hubbard, T. J. P. (2006). Ensembl 2006. *Nucleic Acids Research*, *34*(Database issue), D556–61.

Bloom, K. (2004). Microtubule composition: cryptography of dynamic polymers. *Proceedings of the National Academy of Sciences of the United States of America*, *101*(18), 6839–40.

Böck, A., Forchhammer, K., Heider, J., Leinfelder, W., Sawers, G., Veprek, B., & Zinoni, F. (1991). Selenocysteine: the 21st amino acid. *Molecular Microbiology*, *5*(3), 515–520.

Bonet, J., Planas-Iglesias, J., Garcia-Garcia, J., Marin-Lopez, M. a., Fernandez-Fuentes, N., & Oliva, B. (2013). ArchDB 2014: structural classification of loops in proteins. *Nucleic Acids Research*, 1–5.

Bosc, C., Cronk, J. D., Pirollet, F., Watterson, D. M., Haiech, J., Job, D., & Margolis, R. L. (1996). Cloning, expression, and properties of the microtubule-stabilizing protein STOP. *Proceedings of the National Academy of Sciences of the United States of America*, *93*(5), 2125–30.

Brenner, S. E., Koehl, P., & Levitt, M. (2000). The ASTRAL compendium for protein structure and sequence analysis. *Nucleic Acids Research*, *28*(1), 254–6.

Burke, D. F., Deane, C. M., & Blundell, T. L. (2000). Browsing the SLoop database of structurally classified loops connecting elements of protein secondary structure. *Bioinformatics*, *16*(6), 513–9.

Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K., & Madden, T. L. (2009). BLAST+: architecture and applications. *BMC Bioinformatics*, *10*, 421.

Camon, E., Magrane, M., Barrell, D., Binns, D., Fleischmann, W., Kersey, P., … Apweiler, R. (2003). The Gene Ontology Annotation (GOA) project: implementation of GO in SWISS-PROT, TrEMBL, and InterPro. *Genome Research*, *13*(4), 662–72.

Cariaso, M., & Lennon, G. (2012). SNPedia: a wiki supporting personal genome annotation, interpretation and analysis. *Nucleic Acids Research*, *40*(Database issue), D1308–12.

Carter, P., Andersen, C. A. F., & Rost, B. (2003). DSSPcont: Continuous secondary structure assignments for proteins. *Nucleic Acids Research*, *31*(13), 3293–5.

Castel, S. E., & Martienssen, R. a. (2013). RNA interference in the nucleus: roles for small RNAs in transcription, epigenetics and beyond. *Nature Reviews. Genetics*, *14*(2), 100–12.

Cedano, J., Aloy, P., Pérez-Pons, J. A., & Querol, E. (1997). Relation between amino acid composition and cellular location of proteins. *Journal of Molecular Biology*, *266*(3), 594–600.

Chandonia, J.-M., Hon, G., Walker, N. S., Lo Conte, L., Koehl, P., Levitt, M., & Brenner, S. E. (2004). The ASTRAL Compendium in 2004. *Nucleic Acids Research*, *32*(Database issue), D189–92.

Chang, J. (2015). Core services: Reward bioinformaticians. *Nature*, *520*(7546), 151–152.

Chatzou, M. (2014). Nextflow - Reproducibility in Science - Nextflow meets Docker. Retrieved April 14, 2015, from http://www.nextflow.io/blog/2014/nextflow-meets-docker.html

Chelaru, F., Smith, L., Goldstein, N., & Bravo, H. C. (2014). Epiviz: interactive visual analytics for functional genomics data. *Nature Methods*, *11*(9), 938–940.

Cho, J. H., Kim, D. H., Lee, K. J., & Choi, K. Y. (2001). The role of Tyr248 probed by mutant bovine carboxypeptidase A: insight into the catalytic mechanism of carboxypeptidase A. *Biochemistry*, *40*(34), 10197–203.

Christianson, D. W., & Lipscomb, W. N. (1986). X-ray crystallographic investigation of substrate binding to carboxypeptidase A at subzero temperature. *Proceedings of the National Academy of Sciences of the United States of America*, *83*(20), 7568–72.

Cock, P. J. A., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., … de Hoon, M. J. L. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, *25*(11), 1422–3.

Coleman, J. E., Williams, K. R., King, G. C., Prigodich, R. V, Shamoo, Y., & Konigsberg, W. H. (1986). Protein chemistry-nuclear magnetic resonance approach to mapping functional domains in single-stranded DNA binding proteins. *Journal of Cellular Biochemistry*, *32*(4), 305–26.

Conesa, A., & Götz, S. (2008). Blast2GO: A comprehensive suite for functional analysis in plant genomics. *International Journal of Plant Genomics*, *2008*, 619832.

Contín, M. A., & Arce, C. A. (2000). Tubulin carboxypeptidase/microtubules association can be detected in the distal region of neural processes. *Neurochemical Research*, *25*(1), 27–36.

Corpas, M., Jimenez, R., Carbon, S. J., García, A., Garcia, L., Goldberg, T., … Hermjakob, H. (2014). BioJS: an open source standard for biological visualisation - its status in 2014. *F1000Research*, *3*, 55.

Court, G., & Zyp, K. (2010). A JSON Media Type for Describing the Structure and Meaning of JSON Documents. *IEFT*.

Crooks, G. E., Hon, G., Chandonia, J. M., & Brenner, S. E. (2004). WebLogo: A sequence logo generator. *Genome Research*, *14*(6), 1188–1190.

Cuthbertson, J. M., Doyle, D. A., & Sansom, M. S. P. (2005). Transmembrane helix prediction: a comparative evaluation and analysis. *Protein Engineering, Design & Selection : PEDS*, *18*(6), 295–308.

De La Vega Otazo, M. R., Lorenzo, J., Tort, O., Avilés, F. X., & Bautista, J. M. (2013). Functional segregation and emerging role of cilia-related cytosolic carboxypeptidases (CCPs). *FASEB Journal*, *27*(2), 424–431.

Devanapally, S., Ravikumar, S., & Jose, A. M. (2015). Double-stranded RNA made in C. elegans neurons can enter the germline and cause transgenerational gene silencing. *Proceedings of the National Academy of Sciences*, *112*(7), 201423333.

Devos, D., & Valencia, A. (2000). Practical limits of function prediction. *Proteins: Structure, Function and Genetics*, *41*(1), 98–107.

Dietmann, S., Fernandez-Fuentes, N., & Holm, L. (2002). Automated detection of remote homology. *Current Opinion in Structural Biology*, *12*(3), 362–7.

Donate, L. E., Rufino, S. D., Canard, L. H., & Blundell, T. L. (1996). Conformational analysis and clustering of short and medium size loops connecting regular secondary structures: a database for modeling and prediction. *Protein Science : A Publication of the Protein Society*, *5*(12), 2600–16.

Dowell, R. D., Jokerst, R. M., Day, A., Eddy, S. R., & Stein, L. (2001). The distributed annotation system. *BMC Bioinformatics*, *2*, 7.

Dudley, J. T., & Butte, A. J. (2009). A quick guide for developing effective bioinformatics programming skills. *PLoS Computational Biology*, *5*(12), e1000589.

Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. *Analysis*.

Durinck, S., Moreau, Y., Kasprzyk, A., Davis, S., De Moor, B., Brazma, A., & Huber, W. (2005). BioMart and Bioconductor: A powerful link between biological databases and microarray data analysis. *Bioinformatics*, *21*(16), 3439–3440.

Eberhard Karls University. (n.d.). SNNS- Stuttgart Neural Network Simulator. Retrieved April 14, 2015, from http://www.ra.cs.uni-tuebingen.de/SNNS/

Edwards, D. (2010). What is DevOps? Retrieved from http://dev2ops.org/2010/02/what-is-devops/

Eggleston, A. K. (2009). RNA silencing. *Nature*, *457*(7228), 395.

Eisenberg, D., Lüthy, R., & Bowie, J. U. (1997). VERIFY3D: assessment of protein models with three-dimensional profiles. *Methods in Enzymology*, *277*, 396–404.

EMBL-EBI. (2015). DAS services to retire. Retrieved May 10, 2015, from http://www.ebi.ac.uk/about/news/service-news/das-services-to-retire

ENSEMBL Blog. (2015). Ensembl DAS support ceasing at the end of 2015. Retrieved May 10, 2015, from http://www.ensembl.info/blog/2015/03/27/ensembl-das-support-ceasing-at-the-end-of-2015/

Erck, C., Peris, L., Andrieux, A., Meissirel, C., Gruber, A. D., Vernet, M., … Wehland, J. (2005). A vital role of tubulin-tyrosine-ligase for neuronal organization. *Proceedings of the National Academy of Sciences of the United States of America*, *102*(22), 7853–8.

Espadaler, J., Fernandez-Fuentes, N., Hermoso, A., Querol, E., Aviles, F. X., Sternberg, M. J. E., & Oliva, B. (2004). ArchDB: automated protein loop classification as a tool for structural genomics. *Nucleic Acids Research*, *32*(Database issue), D185–D188.

Estébanez-Perpiñá, E., Bayés, A., Vendrell, J., Jongsma, M. A., Bown, D. P., Gatehouse, J. A., … Reverter, D. (2001). Crystal structure of a novel mid-gut procarboxypeptidase from the cotton pest Helicoverpa armigera. *Journal of Molecular Biology*, *313*(3), 629–38.

Falquet, L., Pagni, M., Bucher, P., Hulo, N., Sigrist, C. J. A., Hofmann, K., & Bairoch, A. (2002). The PROSITE database, its status in 2002. *Nucleic Acids Research*, *30*(1), 235–8.

Federhen, S. (2012). The NCBI Taxonomy database. *Nucleic Acids Research*, *40*(D1).

Feng, W., Shi, Y., Li, M., & Zhang, M. (2003). Tandem PDZ repeats in glutamate receptor-interacting proteins have a novel mode of PDZ domain-mediated target binding. *Nature Structural Biology*, *10*(11), 972–8.

Fernald, G. H., Capriotti, E., Daneshjou, R., Karczewski, K. J., & Altman, R. B. (2011). Bioinformatics challenges for personalized medicine. *Bioinformatics*.

Fernandes, P., Jain, P., & Moita, C. (2012). Training experimental biologists in bioinformatics. *Advances in Bioinformatics*.

Fernandez-Fuentes, N., Oliva, B., & Fiser, A. (2006). A supersecondary structure library and search algorithm for modeling loops in protein structures. *Nucleic Acids Research*, *34*(7), 2085–97.

Fernandez-Fuentes, N., Querol, E., Aviles, F. X., Sternberg, M. J. E., & Oliva, B. (2005). Prediction of the conformation and geometry of loops in globular proteins: testing ArchDB, a structural classification of loops. *Proteins*, *60*(4), 746–57.

Fernandez-Gonzalez, A., La Spada, A. R., Treadaway, J., Higdon, J. C., Harris, B. S., Sidman, R. L., … Zuo, J. (2002). Purkinje cell degeneration (pcd) phenotypes caused by mutations in the axotomy-induced gene, Nna1. *Science*, *295*(5561), 1904–6.

Fetrow, J. S. (1995). Omega loops: nonregular secondary structures significant in protein function and stability. *The FASEB Journal : Official Publication of the Federation of American Societies for Experimental Biology*, *9*(9), 708–717.

Fette, I., & Melnikov, A. (2011). The websocket protocol. *RFC*, *1*, 1–71. Retrieved from http://tools.ietf.org/html/rfc6455):

Finn, R. D., Clements, J., & Eddy, S. R. (2011). HMMER web server: Interactive sequence similarity searching. *Nucleic Acids Research*, *39*(SUPPL. 2).

Finn, R. D., Mistry, J., Tate, J., Coggill, P., Heger, A., Pollington, J. E., … Bateman, A. (2010). The Pfam protein families database. *Nucleic Acids Research*, *38*(Database issue), D211–22.

Fiser, A., Do, R. K., & Sali, A. (2000). Modeling of loops in protein structures. *Protein Science : A Publication of the Protein Society*, *9*(9), 1753–73.

Fiser, A., & Sali, A. (2003a). Modeller: generation and refinement of homology-based protein structure models. *Methods in Enzymology*, *374*, 461–91.

Fiser, A., & Sali, A. (2003b). ModLoop: automated modeling of loops in protein structures. *Bioinformatics*, *19*(18), 2500–1.

Franzoni, C., & Sauermann, H. (2014). Crowd science: The organization of scientific research in open collaborative projects. *Research Policy*, *43*(1), 1–20.

Fritz-Wolf, K., Schnyder, T., Wallimann, T., & Kabsch, W. (1996). Structure of mitochondrial creatine kinase. *Nature*, *381*(6580), 341–5.

García-Sáez, I., Reverter, D., Vendrell, J., Avilés, F. X., & Coll, M. (1997). The three-dimensional structure of human procarboxypeptidase A2. Deciphering the basis of the inhibition, activation and intrinsic activity of the zymogen. *The EMBO Journal*, *16*(23), 6906–13.

Gardell, S. J., Craik, C. S., Hilvert, D., Urdea, M. S., & Rutter, W. J. Site-directed mutagenesis shows that tyrosine 248 of carboxypeptidase A does not play a crucial role in catalysis. *Nature*, *317*(6037), 551–5.

Gardner, P. P., & Giegerich, R. (2004). A comprehensive comparison of comparative RNA structure prediction approaches. *BMC Bioinformatics*, *5*(1), 140.

Gaston, M. A., Zhang, L., Green-Church, K. B., & Krzycki, J. A. (2011). The complete biosynthesis of the genetically encoded amino acid pyrrolysine from lysine. *Nature*, *471*(7340), 647–650.

Germain, P. L., Ratti, E., & Boem, F. (2014). Junk or functional DNA? ENCODE and the function controversy. *Biology & Philosophy*.

Gilks, W. R., Audit, B., De Angelis, D., Tsoka, S., & Ouzounis, C. A. (2005). Percolation of annotation errors through hierarchically structured protein sequence databases. *Mathematical Biosciences*, *193*(2), 223–34.

Gille, C., Birgit, W., & Gille, A. (2014). Sequence alignment visualization in HTML5 without Java. *Bioinformatics*, *30*(1), 121–2.

Goecks, J., Nekrutenko, A., & Taylor, J. (2010). Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology*, *11*(8), R86.

Gómez, A., Cedano, J., Espadaler, J., Hermoso, A., Piñol, J., & Querol, E. (2008). Prediction of protein function improving sequence remote alignment search by a fuzzy logic algorithm. *Protein Journal*, *27*(2), 130–139.

Gómez, A., Hernández, S., Amela, I., Piñol, J., Cedano, J., & Querol, E. (2011). Do protein-protein interaction databases identify moonlighting proteins? *Molecular bioSystems*, *7*(8), 2379–2382.

Gomis-Rüth, F. X., Companys, V., Qian, Y., Fricker, L. D., Vendrell, J., Avilés, F. X., & Coll, M. (1999). Crystal structure of avian carboxypeptidase D domain II: a prototype for the regulatory metallocarboxypeptidase subfamily. *The EMBO Journal*, *18*(21), 5817–26.

Good, B. (2009). Strategies for amassing, characterizing, and applying third-party metadata in bioinformatics. *Quality*, (April), 1–225.

Good, B. M., Clarke, E. L., De Alfaro, L., & Su, A. I. (2012). The Gene Wiki in 2011: Community intelligence applied to human gene annotation. *Nucleic Acids Research*, *40*(D1).

Good, B. M., Tennis, J. T., & Wilkinson, M. D. (2009). Social tagging in the life sciences: characterizing a new metadata resource for bioinformatics. *BMC Bioinformatics*, *10*, 313.

Greene, A. C., Giffin, K. A., Greene, C. S., & Moore, J. H. (2015). Adapting bioinformatics curricula for big data. *Briefings in Bioinformatics*, bbv018–.

Greer, C. A., & Shepherd, G. M. (1982). Mitral cell degeneration and sensory function in the neurological mutant mouse Purkinje cell degeneration (PCD). *Brain Research*, *235*(1), 156–61.

Guex, N., & Peitsch, M. C. (1997). SWISS-MODEL and the Swiss-PdbViewer: an environment for comparative protein modeling. *Electrophoresis*, *18*(15), 2714–23.

Gunasekaran, K., Ma, B., & Nussinov, R. (2003). Triggering loops and enzyme function: identification of loops that trigger and modulate movements. *Journal of Molecular Biology*, *332*(1), 143–59.

Gundersen, G. G., Khawaja, S., & Bulinski, J. C. (1987). Postpolymerization detyrosination of alpha-tubulin: a mechanism for subcellular differentiation of microtubules. *The Journal of Cell Biology*, *105*(1), 251–64.

Hallak, M. E., Rodriguez, J. A., Barra, H. S., & Caputto, R. (1977). Release of tyrosine from tyrosinated tubulin. Some common factors that affect this process and the assembly of tubulin. *FEBS Letters*, *73*(2), 147–50.

Harris, A., Morgan, J. I., Pecot, M., Soumare, A., Osborne, A., & Soares, H. D. (2000). Regenerating motor neurons express Nna1, a novel ATP/GTP-binding protein related to zinc carboxypeptidases. *Molecular and Cellular Neurosciences*, *16*(5), 578–96.

He, G. P., Muise, A., Li, A. W., & Ro, H. S. (1995). A eukaryotic transcriptional repressor with carboxypeptidase activity. *Nature*, *378*(6552), 92–6.

Hermoso, A., Aguilar, D., Aviles, F. X., & Querol, E. (2004). TrSDB: a proteome database of transcription factors. *Nucleic Acids Research*, *32*(Database issue), D171–D173.

Hermoso, A., Espadaler, J., Enrique Querol, E., Aviles, F. X., Sternberg, M. J. E., Oliva, B., … Fernandez-Fuentes, N. (2007). Including Functional Annotations and Extending the Collection of Structural Classifications of Protein Loops (ArchDB). *Bioinformatics and Biology Insights*, *1*, 77.

Hernández, S., Calvo, A., Ferragut, G., Franco, L., Hermoso, A., Amela, I., … Cedano, J. (2014). Can bioinformatics help in the identification of moonlighting proteins? *Biochemical Society Transactions*, *42*(6), 1692–7.

Hernández, S., Ferragut, G., Amela, I., Perez-Pons, J., Piñol, J., Mozo-Villarias, A., … Querol, E. (2014). MultitaskProtDB: a database of multitasking proteins. *Nucleic Acids Research*, *42*(Database issue), D517–20.

Hoersch, S., Leroy, C., Brown, N. P., Andrade, M. A., & Sander, C. (2000). The GeneQuiz web server: protein functional analysis through the Web. *Trends in Biochemical Sciences*, *25*(1), 33–5.

Holt, B. (2011a). *MapReduce Views in CouchDB*. *Selling*.

Holt, B. (2011b). *Writing and Querying MapReduce Views in CouchDB*. “O’Reilly Media, Inc.”

Homolog.us. (2012). Bioinformatics and Computational Biology – same? Si, Si, Si. Retrieved May 9, 2015, from http://www.homolog.us/blogs/blog/2012/08/22/bioinformatics-and-computational-biology-same-si-si-si/

Hourdou, M. L., Guinand, M., Vacheron, M. J., Michel, G., Denoroy, L., Duez, C., … Ghuysen, J. M. (1993). Characterization of the sporulation-related gamma-D-glutamyl-(L)meso-diaminopimelic-acid-hydrolysing peptidase I of Bacillus sphaericus NCTC 9602 as a member of the metallo(zinc) carboxypeptidase A family. Modular design of the protein. *The Biochemical Journal*, *292 (Pt 2*, 563–70).

Huang, H., Reed, C. P., Zhang, J. S., Shridhar, V., Wang, L., & Smith, D. I. (1999). Carboxypeptidase A3 (CPA3): a novel gene highly induced by histone deacetylase inhibitors during differentiation of prostate epithelial cancer cells. *Cancer Research*, *59*(12), 2981–8.

Hunter, S., Apweiler, R., Attwood, T. K., Bairoch, A., Bateman, A., Binns, D., … Yeats, C. (2009). InterPro: the integrative protein signature database. *Nucleic Acids Research*, *37*(Database issue), D211–5.

Ijaq, J., Chandrasekharan, M., Poddar, R., Bethi, N., & Sundararajan, V. S. (2015). Annotation and curation of uncharacterized proteins- challenges. *Frontiers in Genetics*, *6*, 119.

Jensen, L. J., Gupta, R., Blom, N., Devos, D., Tamames, J., Kesmir, C., … Brunak, S. (2002). Prediction of human protein function from post-translational modifications and localization features. *Journal of Molecular Biology*, *319*(5), 1257–65.

Johnson, L. N., Lowe, E. D., Noble, M. E., & Owen, D. J. (1998). The Eleventh Datta Lecture. The structural basis for substrate recognition and control by protein kinases. *FEBS Letters*, *430*(1-2), 1–11.

Jones, D. T. (1999). GenTHREADER: an efficient and reliable protein fold recognition method for genomic sequences. *Journal of Molecular Biology*, *287*(4), 797–815.

Jones, D. T., & Swindells, M. B. (2002). Getting the most from PSI-BLAST. *Trends in Biochemical Sciences*, *27*(3), 161–4.

Jones, D. T., Taylor, W. R., & Thornton, J. M. (1992). The rapid generation of mutation data matrices from protein sequences. *Computer Applications in the Biosciences : CABIOS*, *8*(3), 275–82.

Kabsch, W., & Sander, C. (1983). Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, *22*(12), 2577–637.

Kalas, M., Puntervoll, P., Joseph, A., Bartaseviciūte, E., Töpfer, A., Venkataraman, P., … Jonassen, I. (2010). BioXSD: the common data-exchange format for everyday bioinformatics web services. *Bioinformatics*, *26*(18), i540–6.

Kalinina, E., Biswas, R., Berezniuk, I., Hermoso, A., Aviles, F. X., & Fricker, L. D. (2007). A novel subfamily of mouse cytosolic carboxypeptidases. *FASEB Journal : Official Publication of the Federation of American Societies for Experimental Biology*, *21*(3), 836–50.

Karplus, P. A., & Schulz, G. E. (1985). Prediction of chain flexibility in proteins. *Naturwissenschaften*, *72*(4), 212–213.

Kato, C., Miyazaki, K., Nakagawa, A., Ohira, M., Nakamura, Y., Ozaki, T., … Nakagawara, A. (2004). Low expression of human tubulin tyrosine ligase and suppressed tubulin tyrosination/detyrosination cycle are associated with impaired neuronal differentiation in neuroblastomas with poor prognosis. *International Journal of Cancer*, *112*(3), 365–375.

Kawasaki, H., & Kretsinger, R. H. (1995). Calcium-binding proteins 1: EF-hands. *Protein Profile*, *2*(4), 297–490.

Kelder, T., Van Iersel, M. P., Hanspers, K., Kutmon, M., Conklin, B. R., Evelo, C. T., & Pico, A. R. (2012). WikiPathways: Building research communities on biological pathways. *Nucleic Acids Research*, *40*(D1).

Kim, H., & Lipscomb, W. N. (1990). Crystal structure of the complex of carboxypeptidase A with a strongly bound phosphonate in a new crystalline form: comparison with structures of other complexes. *Biochemistry*, *29*(23), 5546–55.

Kim, S. T., Shirai, H., Nakajima, N., Higo, J., & Nakamura, H. (1999). Enhanced conformational diversity search of CDR-H3 in antibodies: role of the first CDR-H3 residue. *Proteins*, *37*(4), 683–696.

King, R. D., Wise, P. H., & Clare, A. (2004). Confirmation of data mining based predictions of protein function. *Bioinformatics*, *20*(7), 1110–1118.

Kinsella, R. J., Kähäri, A., Haider, S., Zamora, J., Proctor, G., Spudich, G., … Flicek, P. (2011). Ensembl BioMarts: A hub for data retrieval across taxonomic space. *Database*, *2011*.

Koonin, E. V, Makarova, K. S., & Aravind, L. (2001). Horizontal gene transfer in prokaryotes: quantification and classification. *Annual Review of Microbiology*, *55*, 709–42.

Koren, Y. (n.d.). Extension:External Data - MediaWiki. Retrieved April 14, 2015, from https://www.mediawiki.org/wiki/Extension:External_Data

Kotyk, A. (1999). IUPAC-IUBMB Joint Commission on Biochemical Nomenclature (JCBN) and Nomenclature Committee of IUBMB (NC-IUBMB). Newsletter 1999. *Folia Microbiologica*, *44*(3), 243–6.

Krampis, K., Booth, T., Chapman, B., Tiwari, B., Bicak, M., Field, D., & Nelson, K. E. (2012). Cloud BioLinux: pre-configured and on-demand bioinformatics computing for the genomics community. *BMC Bioinformatics*, *13*(1), 42.

Krötzsch, M., Vrandečić, D., & Völkel, M. (2006). Semantic MediaWiki. *The Semantic Web-ISWC 2006*, 935–942.

Krötzsch, M., Vrandečić, D., Völkel, M., Haller, H., & Studer, R. (2007). Semantic Wikipedia. *Web Semantics*, *5*(4), 251–261.

Kulakovskiy, I. V, Medvedeva, Y. A., Schaefer, U., Kasianov, A. S., Vorontsov, I. E., Bajic, V. B., & Makeev, V. J. (2013). HOCOMOCO: a comprehensive collection of human transcription factor binding sites models. *Nucleic Acids Research*, *41*(Database issue), D195–202.

Kumar, R., Jain, S., Kumari, B., & Kumar, M. (2014). Protein sub-nuclear localization prediction using SVM and Pfam domain information. *PloS One*, *9*(6), e98345.

Kyte, J., & Doolittle, R. F. (1982). A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology*, *157*(1), 105–32.

Lafanechère, L., Courtay-Cahen, C., Kawakami, T., Jacrot, M., Rüdiger, M., Wehland, J., … Margolis, R. L. (1998). Suppression of tubulin tyrosine ligase during tumor growth. *Journal of Cell Science*, *111 (Pt 2*, 171–181.

Laing, C., & Schlick, T. (2011). Computational approaches to RNA structure prediction, analysis, and design. *Current Opinion in Structural Biology*, *21*(3), 306–18.

Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., … Szustakowki, J. (2001). Initial sequencing and analysis of the human genome. *Nature*, *409*(6822), 860–921.

Landis, S. C., & Mullen, R. J. (1978). The development and degeneration of Purkinje cells in pcd mutant mice. *The Journal of Comparative Neurology*, *177*(1), 125–43.

Laskowski, R. A., Rullmannn, J. A., MacArthur, M. W., Kaptein, R., & Thornton, J. M. (1996). AQUA and PROCHECK-NMR: programs for checking the quality of protein structures solved by NMR. *Journal of Biomolecular NMR*, *8*(4), 477–86.

LaVail, M. M., Blanks, J. C., & Mullen, R. J. (1982). Retinal degeneration in the pcd cerebellar mutant mouse. I. Light microscopic and autoradiographic analysis. *The Journal of Comparative Neurology*, *212*(3), 217–30.

Layne, M. D., Endege, W. O., Jain, M. K., Yet, S. F., Hsieh, C. M., Chin, M. T., … Lee, M. E. (1998). Aortic carboxypeptidase-like protein, a novel protein with discoidin and carboxypeptidase-like domains, is up-regulated during vascular smooth muscle cell differentiation. *The Journal of Biological Chemistry*, *273*(25), 15654–60.

Lei, Y., Xin, X., Morgan, D., Pintar, J. E., & Fricker, L. D. (1999). Identification of mouse CPX-1, a novel member of the metallocarboxypeptidase gene family with highest similarity to CPX-2. *DNA and Cell Biology*, *18*(2), 175–85.

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., … Durbin, R. (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, *25*(16), 2078–2079.

Li, W., Jaroszewski, L., & Godzik, A. (2002). Tolerating some redundancy significantly speeds up clustering of large protein databases. *Bioinformatics*, *18*(1), 77–82.

Liakopoulos, T. D., Pasquier, C., & Hamodrakas, S. J. (2001). A novel tool for the prediction of transmembrane protein topology based on a statistical analysis of the SwissProt database: the OrienTM algorithm. *Protein Engineering*, *14*(6), 387–390.

Lo Conte, L., Brenner, S. E., Hubbard, T. J. P., Chothia, C., & Murzin, A. G. (2002). SCOP database in 2002: refinements accommodate structural genomics. *Nucleic Acids Research*, *30*(1), 264–7.

Lok, C. (2010). Literature mining: Speed reading. *Nature*, *463*(7280), 416–8.

Loman, N. J., & Pallen, M. J. (2010). EntrezAJAX: direct web browser access to the Entrez Programming Utilities. *Source Code for Biology and Medicine*, *5*, 6.

Mackey, A. (2002). Relational Modeling of Biological Data: Trees and Graphs. Retrieved from http://archive.oreilly.com/pub/a/network/2002/11/27/bioconf.html

Manyam, G., Payton, M. A., Roth, J. A., Abruzzo, L. V, & Coombes, K. R. (2012). Relax with CouchDB--into the non-relational DBMS era of bioinformatics. *Genomics*, *100*(1), 1–7.

Martin, A. C. R. (2004). PDBSprotEC: a Web-accessible database linking PDB chains to EC numbers via SwissProt. *Bioinformatics*, *20*(6), 986–8.

Mathelier, A., Zhao, X., Zhang, A. W., Parcy, F., Worsley-Hunt, R., Arenillas, D. J., … Wasserman, W. W. (2014). JASPAR 2014: an extensively expanded and updated open-access database of transcription factor binding profiles. *Nucleic Acids Research*, *42*(Database issue), D142–7.

Matys, V., Fricke, E., Geffers, R., Gössling, E., Haubrock, M., Hehl, R., … Wingender, E. (2003). TRANSFAC: transcriptional regulation, from patterns to profiles. *Nucleic Acids Research*, *31*(1), 374–8.

McGuffin, L. J., Bryson, K., & Jones, D. T. (2000). The PSIPRED protein structure prediction server. *Bioinformatics*, *16*(4), 404–5.

McGuffin, L. J., & Jones, D. T. (2003). Improvement of the GenTHREADER method for genomic fold recognition. *Bioinformatics*, *19*(7), 874–81.

McLysaght, A., Hokamp, K., & Wolfe, K. H. (2002). Extensive genomic duplication during early chordate evolution. *Nature Genetics*, *31*(2), 200–4.

Mellor, C. (n.d.). Kryder's law craps out: Race to UBER-CHEAP STORAGE is OVER • The Register. Retrieved April 14, 2015, from http://www.theregister.co.uk/2014/11/10/kryders_law_of_ever_cheaper_storage_disproven/

Meyer, A., & Schartl, M. (1999). Gene and genome duplications in vertebrates: the one-to-four (-to-eight in fish) rule and the evolution of novel gene functions. *Current Opinion in Cell Biology*, *11*(6), 699–704.

Michna, R. H., Commichau, F. M., Tödter, D., Zschiedrich, C. P., & Stülke, J. (2014). SubtiWiki-A database for the model organism Bacillus subtilis that links pathway, interaction and expression information. *Nucleic Acids Research*, *42*(D1).

Mirny, L., & Shakhnovich, E. (2001). Evolutionary conservation of the folding nucleus. *Journal of Molecular Biology*, *308*(2), 123–9.

Mittelbrunn, M., & Sánchez-Madrid, F. (2012). Intercellular communication: diverse structures for exchange of genetic information. *Nature Reviews Molecular Cell Biology*.

Mizuguchi, K., Deane, C. M., Blundell, T. L., Johnson, M. S., & Overington, J. P. (1998). JOY: protein sequence-structure representation and analysis. *Bioinformatics*, *14*(7), 617–23.

Molloy, J. C. (2011). The Open Knowledge Foundation: open data means better science. *PLoS Biology*, *9*(12), e1001195.

Morris, K. V, & Mattick, J. S. (2014). The rise of regulatory RNA. *Nature Reviews. Genetics*, *15*(6), 423–37.

Mount, D. W. (2008). Comparison of the PAM and BLOSUM amino acid substitution matrices. *Cold Spring Harbor Protocols*, *3*(6).

Mulder, N. J., Apweiler, R., Attwood, T. K., Bairoch, A., Barrell, D., Bateman, A., … Zdobnov, E. M. (2003). The InterPro Database, 2003 brings increased coverage and new features. *Nucleic Acids Research*, *31*(1), 315–8.

Mullen, R. J., Eicher, E. M., & Sidman, R. L. (1976). Purkinje cell degeneration, a new neurological mutation in the mouse. *Proceedings of the National Academy of Sciences of the United States of America*, *73*(1), 208–12.

Myers, R. M., Stamatoyannopoulos, J., Snyder, M., Dunham, I., Hardison, R. C., Bernstein, B. E., … Good, P. J. (2011). A user's guide to the Encyclopedia of DNA elements (ENCODE). *PLoS Biology*, *9*(4).

NCBI. (2014). NCBI requests feedback on proposed BLAST XML specification update. Retrieved March 30, 2014, from http://www.ncbi.nlm.nih.gov/news/03-17-2014-blast-xml-feedback/

Nicholas, K., Nicholas, H., & Deerfield, D. (1997). {GeneDoc: analysis and visualization of genetic variation}. *EMBNEW. NEWS*, *4*.

Ning, Z., Cox, A. J., & Mullikin, J. C. (2001). SSAHA: a fast search method for large DNA databases. *Genome Research*, *11*(10), 1725–9.

Ochman, H., Lawrence, J. G., & Groisman, E. A. (2000). Lateral gene transfer and the nature of bacterial innovation. *Nature*, *405*(6784), 299–304.

Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., … Li, P. (2004). Taverna: A tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, *20*(17), 3045–3054.

Okin, J. R. (2005). The Internet Revolution: The Not-for-dummies Guide to the History, Technology, And Use of the Internet.

Oliva, B., Bates, P. A., Querol, E., Avilés, F. X., & Sternberg, M. J. (1997). An automated classification of the structure of protein loops. *Journal of Molecular Biology*, *266*(4), 814–30.

Oliva, B., Bates, P. A., Querol, E., Avilés, F. X., & Sternberg, M. J. (1998). Automated classification of antibody complementarity determining region 3 of the heavy chain (H3) loops into canonical forms and its application to protein structure prediction. *Journal of Molecular Biology*, *279*(5), 1193–210.

Osborne, B. I. (2011). HOWTO:OBDA Flat databases - BioPerl. Retrieved April 14, 2015, from http://www.bioperl.org/wiki/HOWTO:OBDA_Flat_databases

Otero, A., Rodríguez de la Vega, M., Tanco, S., Lorenzo, J., Avilés, F. X., & Reverter, D. (2012). The novel structure of a cytosolic M14 metallocarboxypeptidase (CCP) from Pseudomonas aeruginosa: a model for mammalian CCPs. *FASEB Journal : Official Publication of the Federation of American Societies for Experimental Biology*, *26*(9), 3754–64.

Ouzounis, C. A., & Karp, P. D. (2002). The past, present and future of genome-wide re-annotation. *Genome Biology*, *3*(2), COMMENT2001.

Owens, M. (2010). *The definitive guide to SQLite*. *tThe Definitive Guide to SQLite*.

Palazzo, A. F., & Gregory, T. R. (2014). The Case for Junk DNA. *PLoS Genetics*, *10*(5).

Pareja-Tobes, P. (2012). Finding the lowest common ancestor of a set of NCBI taxonomy nodes with Bio4j - bio4j. Retrieved from http://bio4j.com/blog/2012/02/finding-the-lowest-common-ancestor-of-a-set-of-ncbi-taxonomy-nodes-with-bio4j/

Pareja-Tobes, P., Tobes, R., Manrique, M., Pareja, E., & Pareja-Tobes, E. (2015). *Bio4j: a high-performance cloud-enabled graph-based data platform*. *bioRxiv*. Cold Spring Harbor Labs Journals.

Pearson, W. R., & Lipman, D. J. (1988). Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the United States of America*, *85*(8), 2444–2448.

Pei, J., & Grishin, N. V. (2001). AL2CO: calculation of positional conservation in a protein sequence alignment. *Bioinformatics*, *17*(8), 700–12.

Pieper, U., Eswar, N., Braberg, H., Madhusudhan, M. S., Davis, F. P., Stuart, A. C., … Sali, A. (2004). MODBASE, a database of annotated comparative protein structure models, and associated resources. *Nucleic Acids Research*, *32*(Database issue), D217–22.

Prilusky, J., Hodis, E., Canner, D., Decatur, W. A., Oberholser, K., Martz, E., … Sussman, J. L. (2011). Proteopedia: A status report on the collaborative, 3D web-encyclopedia of proteins and other biomolecules. *Journal of Structural Biology*, *175*(2), 244–252.

Punta, M., Coggill, P. C., Eberhardt, R. Y., Mistry, J., Tate, J., Boursnell, C., … Finn, R. D. (2012). The Pfam protein families database. *Nucleic Acids Research*, *40*(Database issue), D290–301.

Qiu, J. (2014). China goes back to basics on research funding. *Nature*, *507*(7491), 148–9.

Rawlings, N. D., Morton, F. R., & Barrett, A. J. (2006). MEROPS: the peptidase database. *Nucleic Acids Research*, *34*(Database issue), D270–2.

Rees, D. C., & Lipscomb, W. N. (1981). Binding of ligands to the active site of carboxypeptidase A. *Proceedings of the National Academy of Sciences of the United States of America*, *78*(9), 5455–9.

Rego, N., & Koes, D. (2015). 3Dmol.js: molecular visualization with WebGL. *Bioinformatics*, *31*(8), 1322–4.

Roberts, R. J. (2001). PubMed Central: The GenBank of the published literature. *Proceedings of the National Academy of Sciences of the United States of America*, *98*(2), 381–2.

Rodriguez de la Vega, M., Sevilla, R. G., Hermoso, A., Lorenzo, J., Tanco, S., Diez, A., … Avilés, F. X. (2007). Nna1-like proteins are active metallocarboxypeptidases of a new and diverse M14 subfamily. *FASEB Journal : Official Publication of the Federation of American Societies for Experimental Biology*, *21*(3), 851–865.

Rojas, R. (1996). *Neural Networks*. *Neural Networks* (Vol. 7).

Romano, P., Giugno, R., & Pulvirenti, A. (2011). Tools and collaborative environments for bioinformatics research. *Briefings in Bioinformatics*, *12*(6), 549–561.

Rose, A. S., & Hildebrand, P. W. (2015). NGL Viewer: a web application for molecular visualization. *Nucleic Acids Research*.

Rosenbloom, K. R., Armstrong, J., Barber, G. P., Casper, J., Clawson, H., Diekhans, M., … Kent, W. J. (2014). The UCSC Genome Browser database: 2015 update. *Nucleic Acids Research*, *43*(Database issue), D670–81.

Rother, K. (2005). Introduction to PyMOL. *Methods In Molecular Biology Clifton Nj*, *635*(8), 0–32.

Round, J. E., & Campbell, A. M. (2013). Figure facts: encouraging undergraduates to take a data-centered approach to reading primary literature. *CBE Life Sciences Education*, *12*(1), 39–46.

Rubinstein, A., & Chor, B. (2014). Computational thinking in life science education. *PLoS Computational Biology*, *10*(11), e1003897.

Russell, R. B., Sasieni, P. D., & Sternberg, M. J. (1998). Supersites within superfolds. Binding site similarity in the absence of homology. *Journal of Molecular Biology*, *282*(4), 903–18.

Sali, A., & Blundell, T. L. (1990). Definition of general topological equivalence in protein structures. A procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *Journal of Molecular Biology*, *212*(2), 403–428.

Sali, A., & Blundell, T. L. (1993). Comparative protein modelling by satisfaction of spatial restraints. *Journal of Molecular Biology*, *234*(3), 779–815.

Sali, A., Potterton, L., Yuan, F., van Vlijmen, H., & Karplus, M. (1995). Evaluation of comparative protein modeling by MODELLER. *Proteins*, *23*(3), 318–26.

Sánchez, R., & Sali, A. (1997). Evaluation of comparative protein structure modeling by MODELLER-3. *Proteins*, *Suppl 1*, 50–8.

Sanders, G. L., & Shin, S. S. S. (2001). Denormalization effects on performance of RDBMS. *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*.

Sanseverino, W., & Ercolano, M. R. (2012). In silico approach to predict candidate R proteins and to define their domain architecture. *BMC Research Notes*, *5*, 678.

Sanseverino, W., Hermoso, A., D'Alessandro, R., Vlasova, A., Andolfo, G., Frusciante, L., … Ercolano, M. R. (2013). PRGdb 2.0: towards a community-based database model for the analysis of R-genes in plants. *Nucleic Acids Research*, *41*(D1), D1167–D1171.

Saraste, M., Sibbald, P. R., & Wittinghofer, A. (1990). The P-loop--a common motif in ATP- and GTP-binding proteins. *Trends in Biochemical Sciences*, *15*(11), 430–4.

Sayers, E. (2012). E-utilities Quick Start Entrez Programming Utilities Help Entrez Programming Utilities Help. *The Journal of Systems and Software*, *85*, 1930–1952.

Schatz, M. C., Langmead, B., & Salzberg, S. L. (2010). Cloud computing and the DNA data race. *Nature Biotechnology*, *28*(7), 691–3.

Schauer, K., & Stingl, K. (2009). "Guilty by association" - Protein-protein interactions (PPIs) in bacterial pathogens. *Genome Dynamics*.

Schenk, P. W., & Snaar-Jagalska, B. E. (1999). Signal perception and transduction: the role of protein kinases. *Biochimica et Biophysica Acta*, *1449*(1), 1–24.

Shapiro, L., & Harris, T. (2000). Finding function through structural genomics. *Current Opinion in Biotechnology*, *11*(1), 31–5.

Shi, J., Blundell, T. L., & Mizuguchi, K. (2001). FUGUE: sequence-structure homology recognition using environment-specific substitution tables and structure-dependent gap penalties. *Journal of Molecular Biology*, *310*(1), 243–57.

Shindyalov, I. N., & Bourne, P. E. (2000). An alternative view of protein fold space. *Proteins*, *38*(3), 247–60.

Sigrist, C. J. A., Cerutti, L., De Castro, E., Langendijk-Genevaux, P. S., Bulliard, V., Bairoch, A., & Hulo, N. (2009). PROSITE, a protein domain database for functional characterization and annotation. *Nucleic Acids Research*, *38*(SUPPL.1).

Škunca, N., Altenhoff, A., & Dessimoz, C. (2012). Quality of computationally inferred gene ontology annotations. *PLoS Computational Biology*, *8*(5).

Smith, T. F., & Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, *147*(1), 195–197.

Snell, E. A., Brooke, N. M., Taylor, W. R., Casane, D., Philippe, H., & Holland, P. W. H. (2006). An unusual choanoflagellate protein released by Hedgehog autocatalytic processing. *Proceedings. Biological Sciences / The Royal Society*, *273*(1585), 401–7.

Socket.IO Contributors. (n.d.). Socket.IO. Retrieved April 14, 2015, from http://socket.io/

Sonnhammer, E. L., von Heijne, G., & Krogh, A. (1998). A hidden Markov model for predicting transmembrane helices in protein sequences. *Proceedings / ... International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, *6*, 175–82.

Sporny, M., Kellogg, G., & Lanthaler, M. (2013). JSON-LD 1.0 -A JSON-based Serialization for Linked Data. Retrieved from http://www.w3.org/TR/2013/CR-json-ld-20130910/

Stajich, J. E., Block, D., Boulez, K., Brenner, S. E., Chervitz, S. A., Dagdigian, C., … Birney, E. (2002). The Bioperl toolkit: Perl modules for the life sciences. *Genome Research*, *12*(10), 1611–8.

Stajich, J. E., & Lapp, H. (2006). Open source tools and toolkits for bioinformatics: significance, and where are we? *Briefings in Bioinformatics*, *7*(3), 287–96.

Stebbings, L. A., & Mizuguchi, K. (2004). HOMSTRAD: recent developments of the Homologous Protein Structure Alignment Database. *Nucleic Acids Research*, *32*(Database issue), D203–7.

StrongLoop. (n.d.). Express. Retrieved from http://expressjs.com/

Suber, P. (2002). Open access to the scientific journal literature. *Journal of Biology*, *1*(1), 3.

Tainer, J. A., Thayer, M. M., & Cunningham, R. P. (1995). DNA repair proteins. *Current Opinion in Structural Biology*, *5*(1), 20–6.

Tendulkar, A. V, Joshi, A. A., Sohoni, M. A., & Wangikar, P. P. (2004). Clustering of protein structural fragments reveals modular building block approach of nature. *Journal of Molecular Biology*, *338*(3), 611–29.

The Gene Ontology Consortium. (2013). Gene Ontology Annotations and Resources. *Nucleic Acids Research*, *41*, D530–535.

The Jmol Team. (2007). Jmol: an open-source Java viewer for chemical structures in 3D. Retrieved from http://jmol.sourceforge.net/

The UniProt Consortium. (2014). Activities at the Universal Protein Resource (UniProt). *Nucleic Acids Research*, *42*(Database issue), D191–8.

Thompson, J. D., Gibson, T. J., Plewniak, F., Jeanmougin, F., & Higgins, D. G. (1997). The CLUSTAL_X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. *Nucleic Acids Research*, *25*(24), 4876–82.

Tian, W., & Skolnick, J. (2003). How well is enzyme function conserved as a function of pairwise sequence identity? *Journal of Molecular Biology*, *333*(4), 863–82.

Too, C. K., Vickaryous, N., Boudreau, R. T., & Sangster, S. M. (2001). Identification and nuclear localization of a novel prolactin and cytokine-responsive carboxypeptidase D. *Endocrinology*, *142*(3), 1357–67.

Tuimala, J. (2006). *A primer to phylogenetic analysis using the PHYLIP package*. *Espoo Finland Center for Scientific Computing Ltd* (Vol. 6).

UCSC Genome. (2015). UCSC Genome Bioinformatics: Data File Formats. Retrieved April 14, 2015, from http://genome.ucsc.edu/FAQ/FAQformat.html

Van Dongen, S. (2008). Graph Clustering Via a Discrete Uncoupling Process. *SIAM Journal on Matrix Analysis and Applications*.

Venclovas, C., & Margelevicius, M. (2005). Comparative modeling in CASP6 using consensus approach to template selection, sequence-structure alignment, and structure assessment. *Proteins*, *61 Suppl 7*, 99–105.

Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., … Zhu, X. (2001). The sequence of the human genome. *Science*, *291*(5507), 1304–1351.

Ventura, S., Villegas, V., Sterner, J., Larson, J., Vendrell, J., Hershberger, C. L., & Avilés, F. X. (1999). Mapping the pro-region of carboxypeptidase B by protein engineering. Cloning, overexpression, and mutagenesis of the porcine proenzyme. *The Journal of Biological Chemistry*, *274*(28), 19925–33.

Verhey, K. J., & Gaertig, J. (2007). The tubulin code. *Cell Cycle*.

Vihinen, M., Torkkila, E., & Riikonen, P. (1994). Accuracy of protein flexibility predictions. *Proteins: Structure, Function, and Genetics*, *19*(2), 141–149.

Von Heijne, G. (1990). The Signal peptide. *Journal of Membrane Biology*, *115*, 195–201.

Walter, C. (2005). Kryder's Law. *Scientific American*, *293*(2), 32–33.

Wan, S., Mak, M.-W., & Kung, S.-Y. (2012). mGOASVM: Multi-label protein subcellular localization based on gene ontology and support vector machines. *BMC Bioinformatics*, *13*(1), 290.

Wang, T., Parris, J., Li, L., & Morgan, J. I. (2006). The carboxypeptidase-like substrate-binding site in Nna1 is essential for the rescue of the Purkinje cell degeneration (pcd) phenotype. *Molecular and Cellular Neurosciences*, *33*(2), 200–13.

Wehenkel, A., & Janke, C. (2014). Towards elucidating the tubulin code. *Nature Cell Biology*, *16*(4), 303–5.

Wehland, J., & Weber, K. (1987). Turnover of the carboxy-terminal tyrosine of alpha-tubulin and means of reaching elevated levels of detyrosination in living cells. *Journal of Cell Science*, *88 (Pt 2)*, 185–203.

Wei, S., Segura, S., Vendrell, J., Aviles, F. X., Lanoue, E., Day, R., … Fricker, L. D. (2002). Identification and characterization of three members of the human metallocarboxypeptidase gene family. *The Journal of Biological Chemistry*, *277*(17), 14954–64.

Whetzel, P. L., Noy, N. F., Shah, N. H., Alexander, P. R., Nyulas, C., Tudorache, T., & Musen, M. A. (2011). BioPortal: enhanced functionality via new Web services from the National Center for

Biomedical Ontology to access and use ontologies in software applications. *Nucleic Acids Research*, *39*(Web Server issue), W541–5.

Wierenga, R. K., Terpstra, P., & Hol, W. G. (1986). Prediction of the occurrence of the ADP-binding beta alpha beta-fold in proteins, using an amino acid sequence fingerprint. *Journal of Molecular Biology*, *187*(1), 101–7.

Wilkinson, M. D., & Links, M. (2002). BioMOBY: an open source biological web services proposal. *Briefings in Bioinformatics*, *3*(4), 331–41.

Wilkinson, M. D., Vandervalk, B., & McCarthy, L. (2011). The Semantic Automated Discovery and Integration (SADI) Web service Design-Pattern, API and Reference Implementation. *Journal of Biomedical Semantics*, *2*(1), 8.

Wintjens, R. T., Rooman, M. J., & Wodak, S. J. (1996). Automatic classification and analysis of alpha alpha-turn motifs in proteins. *Journal of Molecular Biology*, *255*(1), 235–53.

Wlodawer, A., Miller, M., Jaskólski, M., Sathyanarayana, B. K., Baldwin, E., Weber, I. T., … Kent, S. B. (1989). Conserved folding in retroviral proteases: crystal structure of a synthetic HIV-1 protease. *Science*, *245*(4918), 616–21.

Wojcik, J., Mornon, J. P., & Chomilier, J. (1999). New efficient statistical sequence-dependent structure prediction of short to medium-sized protein loops based on an exhaustive loop classification. *Journal of Molecular Biology*, *289*(5), 1469–90.

Wu, C. H., Yeh, L. S., Huang, H., Arminski, L., Castro-Alvear, J., Chen, Y., … Barker, W. C. (2003). The protein information resource. *Nucleic Acids Research*.

Xin, X., Day, R., Dong, W., Lei, Y., & Fricker, L. D. (1998). Identification of mouse CPX-2, a novel member of the metallocarboxypeptidase gene family: cDNA cloning, mRNA distribution, and protein expression and characterization. *DNA and Cell Biology*, *17*(10), 897–909.

Yang, A.-S., & Wang, L. (2003). Local structure prediction with local structure-based sequence profiles. *Bioinformatics*, *19*(10), 1267–74.

Yang, J., & Russell, J. (2011). Full-Text Search with InnoDB. Retrieved May 1, 2015, from http://www.drdobbs.com/database/full-text-search-with-innodb/231902587

Yona, G., & Levitt, M. (2002). Within the twilight zone: a sensitive profile-profile comparison tool based on information theory. *Journal of Molecular Biology*, *315*(5), 1257–75.

Zhang, J., Haider, S., Baran, J., Cros, A., Guberman, J. M., Hsu, J., … Kasprzyk, A. (2011). BioMart: A data federation framework for large collaborative projects. *Database*, *2011*.