Bioinformatics, 31(11), 2015, 1716–1723
doi: 10.1093/bioinformatics/btv047
Advance Access Publication Date: 31 January 2015
Original Paper

OXFORD

Sequence analysis

KAPPA, a simple algorithm for discovery and clustering of proteins defined by a key amino acid pattern: a case study of the cysteine-rich proteins

Valentin Joly and Daniel P. Matton*

Institut de Recherche en Biologie Végétale, Département de Sciences biologiques, Université de Montréal, 4101 rue Sherbrooke Est, Montréal, QC H1X 2B2, Canada

*To whom correspondence should be addressed.
Associate Editor: Burkhard Rost

Received on August 26, 2014; revised on November 10, 2014; accepted on January 21, 2015

Abstract

Motivation: Proteins defined by a key amino acid pattern are key players in the exchange of signals between bacteria, animals and plants, as well as important mediators for cell–cell communication within a single organism. Their description and characterization open the way to a better knowledge of molecular signalling in a broad range of organisms, and to possible application in medical and agricultural research. The contrasted pattern of evolution in these proteins makes it difficult to detect and cluster them with classical sequence-based search tools. Here, we introduce *Key Aminoacid Pattern-based Protein Analyzer* (KAPPA), a new multi-platform program to detect them in a given set of proteins, analyze their pattern and cluster them by comparison to reference patterns (*ab initio* search) or internal pairwise comparison (*de novo* search).

Results: In this study, we use the concrete example of cysteine-rich proteins (CRPs) to show that the similarity of two cysteine patterns can be precisely and efficiently assessed by a quantitative tool created for KAPPA: the κ-score. We also demonstrate the clear advantage of KAPPA over other classical sequence search tools for *ab initio* search of new CRPs. Eventually, we present *de novo* clustering and subclustering functionalities that allow to rapidly generate consistent groups of CRPs without a seed reference.

Availability and implementation: KAPPA executables are available for Linux, Windows and Mac OS at http://kappa-sequence-search.sourceforge.net.

Contact: dp.matton@umontreal.ca

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

In recent years, novel types of proteins defined by a key amino acid pattern—often referred to as 'X-rich proteins'—have emerged as important and diversified actors of molecular signalling in animals and plants. Although glycine-rich proteins (Mousavi and Hotta, 2005), proline-rich peptides (Scocchi *et al.*, 2011) and leucine-rich repeats-containing proteins (Bella *et al.*, 2008) are nowadays the subject of increasing research, cysteine-rich proteins (CRPs) still remain the most extensively studied ones (Edstam *et al.*, 2011; Giacomelli *et al.*, 2012; Hanks *et al.*, 2005; Silverstein *et al.*, 2005).

Although traditional sequence search tools were successfully used to characterize the diversity of CRPs in various organisms, implementation of numerous manual data curation steps was always necessary, with the drawback of being both tedious and time-consuming. Moreover, manual intervention always introduces the risk of a subjective, hence, potentially skewed or biased analysis.

In this article, we introduce *Key Aminoacid Pattern-based Protein Analyzer* (KAPPA), a new automated sequence search program dedicated to the discovery and clustering of 'X-rich proteins', and we assess its performance on plant CRPs.

Although quite heterogeneous, CRPs share three common features: (i) a small size (50–200 amino acids); (ii) the presence of a signal peptide at their N-termini to allow secretion; (iii) a mature protein comprising six or more cysteines (usually from 6 to 14). They are involved in a wide range of functions in living organisms. For instance, antimicrobial peptides (AMPs) such as defensins are broadly studied for their role in human innate immunity against viruses, bacteria and fungi (Pasupuleti *et al.*, 2012), and in plant responses to pathogens (Odintsova and Egorov, 2012). Plant–bacteria symbiotic equilibrium relies on CRPs as well (van de Velde *et al.*, 2010), while venoms from snakes, spiders and scorpions have also adopted the CRPs as neurotoxins and myonecrotic agents (Wong and Belov, 2012). Thus, characterization of these communication mediators opens the way to future applications for human health and agronomy (de Souza Cândido *et al.*, 2014).

In addition to their inter-organism communication functions, CRPs appear to be important messengers for cell–cell signalling within a single individual. They are particularly involved in the control of developmental processes, such as seed, root and stomata development (Marshall *et al.*, 2011), but also in sexual reproduction in plants (Chevalier *et al.*, 2011, 2013; Higashiyama, 2010) and in animals (Koppers *et al.*, 2011).

Numerous families of CRPs like defensins, thionins, albumins, snakins, lipid-transfer proteins, rapid alkalinization factors (RALFs), etc. have already been described, each of them being characterized by a precise cysteine spacing pattern. In several plant species, these CRPs can represent up to 2–3% of the total genome (Silverstein *et al.*, 2007). Describing their diversity and evolution now becomes a major stake in plant, animal and microbe biology.

The structural particularities of CRPs are a challenging issue for bioinformaticians: on the one hand, the cysteine backbone—which governs maintenance of disulfide bonds, hence, the 3D structure of proteins—is highly conserved in a given CRP family, even between distant species. For instance, all defensins share a γ-core and a cysteine-stabilized (CS) $\alpha\beta$ motif (Zhu *et al.*, 2005). On the other hand, the remaining residues in the sequence—involved in fine and specific recognition functions—can exhibit a fast evolutive speed, often underlined by a positive selection. Hence, global sequence identity can be extremely low in a given CRP family (Supplementary Fig. S1). As shown in our study, this dual evolutive pattern makes it difficult to discover paralogues and orthologues of already known CRPs with classical *ab initio* sequence similarity search tools.

In addition, the cysteine backbone itself can evolve—though at a lesser speed—and give rise to new families of CRPs that may be specific to a given taxon. Literature suggests that CRPs have been largely under-predicted (Silverstein *et al.*, 2007); it is therefore essential to be able to detect and cluster CRPs *de novo*, without necessarily relying on a set of reference proteins.

The KAPPA workflow, presented in Supplementary Figure S2, meets this sequence search challenge by extracting and comparing cysteine patterns by means of a quantitative similarity index called κ-score. This mapping step allows to detect CRPs that are similar to reference patterns (*ab initio* search) or to cluster CRPs having similar patterns without relying on a reference (*de novo* search). A BLAST-based subclustering step then allows to refine clustering, analyzing the remaining amino acids of the sequence and to visualize output groups of CRPs graphically.

KAPPA is a free program coded in Python 3 and is executable on most of existing operating systems (Linux, Mac OS, Windows). It supports sequence search parallelization by multithreading. UNIX manual pages and a complete user's guide are also available.

2 Methods

2.1 Cysteine pattern extraction

The first step consists in providing KAPPA with one or several FASTA files containing protein sequences. Several scripts bundled with KAPPA can be used to detect open reading frames (ORFs) in a set of nucleotide sequences (`kappa_findorfs`), translate them into proteins (`kappa_translate`) and predict secretion (`kappa_secretion`). The latter program relies on SignalP (Petersen *et al.*, 2011) and SecretomeP (Bendtsen *et al.*, 2004). After sequence import and FASTA identifier parsing, KAPPA pre-filters proteins to retain only those susceptible to be true CRPs, according to amino acid length (options `-l` and `-L`) and number of cysteines (options `-m` and `-M`). The user then obtain a list of *target proteins*.

For each of them, KAPPA analyzes cysteine spacing and creates a *pattern*, a 1D vector in which each value corresponds to the number of amino acids in a sequence block between two cysteines. The first block corresponds to amino acids before the first cysteine, the last block to those located after the last cysteine. Two consecutive cysteines define an empty block, represented by a zero in the pattern. *Pattern length* describes the number of blocks in a pattern. For example, sequence XCXXXCCXCXX gives pattern $\{1, 3, 0, 1, 2\}$, whose length is 5.

The mapping step then consists in comparing each target pattern to reference patterns called *query patterns*. If an *ab initio* search is performed, the user must provide query patterns in a separate text file (option `-q`). The user can write this file directly or build it from a set of reference proteins with the `kappa_extract_patterns` script. Unlike target patterns, each position in a query pattern comprises two values: a minimum and a maximum number of amino acids. These values can be equal or replaced by letter *n* to define an unknown number of amino acids.

Alternatively, if a *de novo* search is chosen, no reference is required, since an all-versus-all pairwise comparison of patterns will be performed: query patterns are simply target patterns themselves.

2.2 Mapping

To determine the similarity level between a query and a target pattern, KAPPA aligns them in all possible configurations, i.e. considering all possible shifts between them. For each alignment, a κ-score reflecting pattern homology is computed. The maximum κ-score obtained among all possible alignments between the two patterns is retained as the final score.

The κ-score incorporates three indicators varying between 0 and 1 called *block identity*, *key persistence* and *alignment coverage*, which describe different aspects of pattern similarity. Figure 1 presents an example of κ-score calculation for two given patterns.

Block identity I aims at describing conservation of the number of amino acids between query and target aligned blocks. To begin, KAPPA computes local query-target variation δ_k at each aligned position k, as shown in Equation (1), where Q_k^- and Q_k^+ are the minimum and maximum values in the query pattern and T_k the value in the target pattern at position k.

$$\delta_k = \begin{cases} 0 & \text{if} \quad Q_k^- \leq T_k \leq Q_k^+ \\ \dfrac{Q_k^- - T_k}{Q_k^- + T_k} & \text{if} \quad T_k < Q_k^- \\ \dfrac{T_k - Q_k^+}{T_k + Q_k^+} & \text{if} \quad T_k > Q_k^+ \end{cases} \tag{1}$$

Local identity at position k is the difference $1 - \delta_k$. Before being incorporated in the calculation of global identity I, local identity

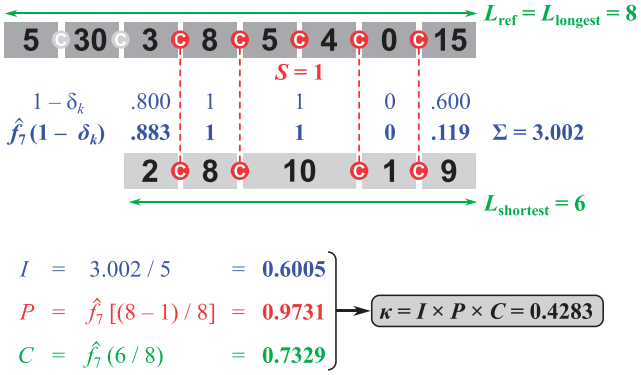

Fig. 1. Example of κ-score calculation between two patterns, with default stringency ($\alpha_I = \alpha_P = \alpha_C = 7$) and one loss or gain of cysteine allowed

values are adjusted with the normalized logistic function $\hat{f}$ described in Equation (2).

$$\hat{f}_\alpha(x) = \frac{f_\alpha(x) - f_\alpha(0)}{f_\alpha(1) - f_\alpha(0)} \quad \text{with} \quad f_\alpha(x) = \frac{1}{1 + e^{2(\alpha - 10x)}} \tag{2}$$

This step allows penalizing low values and favouring high values in a more or less restrictive manner, depending on the value attributed to the user-defined stringency parameter α. The user can thus choose to be stringent (i.e. considering only perfect identities) or more permissive. Stringency values are comprised between 0 (low) and 10 (high). The default is 7.

Identity I is simply computed as shown in Equation (3), where L corresponds to the number of positions in the alignment. Stringency on identity α_I can be set with option `-I`.

$$I = \frac{1}{L}\sum_{k=1}^{L} \hat{f}_{\alpha_I}(1 - \delta_k) \tag{3}$$

Because residues located before the first cysteine or after the last one are generally excluded from cysteine pattern analyzes, KAPPA provides options `-n` and `-c` to define a number of N- and C-term blocks to be ignored in the calculation of identity. The default value for both of these options is 1.

Key persistence P accounts for maintenance of the cysteine content itself. Indeed, despite high conservation of cysteine backbones within a given CRP family, gain or loss of cysteine can occur marginally in the course of evolution. Options `-kg` and `-kl` can then be used to specify a number of X-to-C or C-to-X substitutions allowed in the query pattern, respectively. If one or more substitutions need to be simulated by KAPPA to achieve better identity, persistence P will decrease, as shown in Equation (4), where S is the number of substitutions simulated, and L is the length of the pattern where substitutions occurred. As well as identity, persistence is finally adjusted with the $\hat{f}$ function. The stringency α_P applied to persistence adjustment through $\hat{f}$ can be set with option `-P`.

$$P = \hat{f}_{\alpha_P}\left(\frac{L - S}{L}\right) \tag{4}$$

Finally, alignment coverage C describes how wide the alignment is compared to a reference number of blocks L_{ref}. By default, L_{ref} is the size of the longest pattern, so that coverage is maximized only when two patterns with similar size are fully aligned. However, one may want to allow the query pattern to be fully included in the targeted one without making coverage decrease. In this case, option `-i` can be enforced. L_{ref} will then be the size of the query pattern (*ab initio* search) or the one of the shortest pattern (*de novo* search). Raw coverage is adjusted with the $\hat{f}$ function, taking in account a specific stringency level α_C set with option `-C`.

$$C = \hat{f}_{\alpha_C}\left(\frac{L}{L_{ref}}\right) \tag{5}$$

The κ-score is eventually computed as the product of I, P and C. In case of an *ab initio* search, all sequences matching to a given reference cysteine pattern with a κ-score greater than a threshold defined with option `-S` will be assigned to it and exported in a FASTA file.

2.3 Clustering

If a *de novo* search is performed, a two-step clustering immediately follows mapping. Supplementary Figure S3 gives an overview of the clustering process. First, preclusters are formed recursively by connecting all sequences pairs having a κ-score above a threshold defined with option `-S1`. Second, similarity is assessed for all possible pairs of sequences belonging to two different preclusters. For two given preclusters, if more than a certain percentage of these inter-precluster sequence pairs (defined with option `-F`) have a κ-score above a certain threshold (specified with option `-S2`), the two preclusters are fused into one final cluster. Final clusters can result from an unfused precluster as well as from the fusion of two or more preclusters. They are exported in separate FASTA files.

Clusters can be seen as similarity networks. It is then interesting to assess their density (i.e. the number of links between sequences) and compactness (i.e. the level of similarity between sequences). Statistical indicators are computed taking in account all sequence–sequence similarities represented in the cluster, but also more specifically focusing on intra- and inter-sample similarities only. In addition, option `-G` can be enforced to generate network files that can be imported into Cytoscape (Shannon *et al.*, 2003) to graphically visualize clusters.

2.4 Subclustering

Mapping and clustering steps allow assembling families of proteins sharing similar cysteine backbones. However, the remaining elements of the sequence are also of interest to refine this clustering. Indeed, CRPs also contain motifs or domains conserved on a more or less large scale that can be used to define subgroups within clusters. Users may want to split large families of CRPs into smaller, motif-defined subclusters; or to analyse paralogy and orthology relationships within a cluster.

The subclustering step consists in using BLASTp to make pairwise comparisons of all sequences within each group, relying on a network approach that was first developed in clustering tools such as EGN (Halary *et al.*, 2013). Two sequences are grouped into the same subcluster, if BLAST characteristic values pass a user-defined threshold: e-value, percentage of identity, percentage of positive matches, hit length, alignment coverage (options `-sE`, `-sI`/`-sJ`, `-sP`/`-sQ`, `-sL` and `-sC`, respectively). Moreover, the user can use option `-sR` to enforce a reciprocity condition: two sequences will be considered similar if they are reciprocal best or near-best hits.

Here again, several density and compactness indicators are computed for each subcluster and graphical visualization with Cytoscape is possible through option `-G`.

2.5 Optimization

Several options affecting mapping and clustering granularity are to be set by the user. Therefore, finding the optimal combination of

parameters to fit biological reality is a crucial issue. Two kinds of pitfalls especially need attention: (i) the 'snowball effect' taking place when most of the proteins are clustered into the same big group because of too permissive settings and (ii) the accumulation of singletons due to too stringent parameters.

To overcome these problems, the `kappa` script allows the user to provide several values for each option instead of one. The script will then execute KAPPA on the input target proteins, considering all possible combinations of parameters among those provided by the user. The output is a large table presenting, for each combination tested, a broad range of statistical indicators the user can explore to determine the most suitable settings.

2.6 Performance tests

2.6.1 Reference proteins

In experiments described in Sections 3.1 and 3.2, we used reference small families of CRPs from *Arabidopsis* and rice as query proteins. LTP1s (subfamily 1 of non-specific lipid-transfer proteins) were retrieved from Edstam *et al.* (2011). Three OsLTP1s that were obviously not LTP1s in terms of cysteine pattern were removed from the dataset (Os11g02330.1, Os11g02379.1 and Os12g02290.1), because they would have skewed the analysis. True defensins and snakins come from groups CRP0000 and CRP2700 published by Silverstein *et al.* (2007), respectively.

2.6.2 Expected outputs

In Section 3.2, we assessed the quality of outputs from several sequence-based search tools by comparing them to 'expected outputs', i.e. known lipid-transfer proteins (LTPs) and defensin-like proteins (DEFLs). Reference LTPs correspond to those described in *Arabidopsis* and rice by Edstam *et al.* (all subfamilies described in the study) and Silverstein *et al.* (groups CRP3800–4958). Reference DEFLs are those described in Silverstein *et al.* only (groups CRP0000–1520).

2.6.3 Target proteomes

We used the *Arabidopsis thaliana* (Swarbreck *et al.*, 2008) and *Oryza sativa* subsp. *japonica* (Ouyang *et al.*, 2007) proteomes available on line from Phytozome v9.1 (http://www.phytozome.net) as target sets of proteins for assessment of KAPPA performance in Section 3.

2.6.4 Dataset calibration

Studies from which reference proteins were retrieved used older releases of the *Arabidopsis* and rice proteomes. To ensure consistency between our datasets, we did not use reference proteins directly, but the corresponding sequences from the Phytozome proteomes mentioned earlier. In a small minority of cases, no homologue—or only a distant one—was found, since some proteins turned to be obsolete or because they corresponded to pseudogenes rather than true proteins. We simply discarded them. This only affected 'expected outputs', not query proteins.

2.6.5 Software

In Section 3.1, MUSCLE (Edgar, 2004) was used for pairwise and multiple protein alignments; identities were computed on the hit with a homemade script. In Section 3.2, we used HMMER 3.0 (Finn *et al.*, 2011), BLASTp (Altschul *et al.*, 1990), PSI-BLAST and PHI-BLAST (position-specific Iterated and pattern hit initiated BLAST, respectively) (Altschul *et al.*, 1997) from the BLAST+ 2.2.29 suite. All programs were used with default parameters, unless otherwise specified.

3 Results

3.1 Relevance of the κ-score

We first tried to check if the κ-score is relevant to describe conservation of the cysteine backbone within two well-known CRP families: true defensins and snakins described by Silverstein *et al.* (2007) and the LTP1 subfamily described by Edstam *et al.* (2011). In all cases, we analyzed together proteins from *A. thaliana* and *O. sativa*. Multiple alignments of these proteins can be found in Supplementary Figure S1. Within each family, proteins were pairwise aligned with MUSCLE and identity on the alignment was computed. Besides, KAPPA was used to determine the κ-score for each pair of proteins.

As can be seen in Figure 2, pairwise identity can vary to a large extent while the κ-score always remains high within a given CRP family. This holds especially true for LTP1s, which have conserved cysteine spacing. However, defensins include proteins with a more distant pattern, which led to a slight decrease of the κ-score for a few data points. The general conclusion of this test is that the κ-score is a simple but reliable quantitative tool to assess conservation of the cysteine spacing.

3.2 *Ab initio* discovery of CRPs

KAPPA provides an *ab initio* sequence search function, consisting in detecting proteins matching to a given reference cysteine pattern. We compared performance of KAPPA, HMMER, BLASTp, PSI-BLAST and PHI-BLAST in detecting known LTPs and LTP-like proteins in the *Arabidopsis* proteome (Fig. 3 and Supplementary Fig. S4). KAPPA was provided with a consensus cysteine pattern made with `kappa_extract_patterns` after alignment of AtLTP1s with MUSCLE (Supplementary Table S1). HMMER was provided with a position-specific scoring matrix (PSSM) made with `hmmbuild` using the same alignment. Programs from the BLAST+ suite were given, AtLTP1s, as query proteins. In addition, a perfect consensus cysteine pattern of AtLTP1s was given to PHI-BLAST.

As we can see on Figure 3, KAPPA was sensitive because the vast majority (~98%) of the 128 AtLTPs described by Silverstein *et al.* (2007) and Edstam *et al.* (2011) were detected. Only three known AtLTPs were not detected by KAPPA: AT1G05450.1, AT3G63095.1 and AT5G38197.1. The first one was not found simply because it does not contain any cysteine residue. The two latter ones do have cysteines, but their spacing is different from the canonical LTP pattern. Therefore, their κ-score was low and they were discarded by KAPPA. One may conclude that these three proteins were inappropriately described as LTPs in previous studies.

Interestingly, KAPPA also detected 85 new LTP-like proteins. This is mainly due to a better performance with respect to sequence search strategies used in previous studies, but also to the use of a more recent release of the *Arabidopsis* proteome (14 out of the 85 new LTPs were not present in The Arabidopsis Information Resource (TAIR6) database used by Silverstein *et al.*). Sequence identifiers of proteins detected by KAPPA are listed in Supplementary Table S2.

Contrary to other methods, KAPPA also achieved high specificity because the κ-score made it possible to reject non-related sequences efficiently. Here, we have used a 70% threshold, which appeared to be an adequate value to work with divergent CRPs such as LTPs (Supplementary Fig. S5). Not only KAPPA provided the best sensitivity-specificity trade-off, but its execution was also ~9 times faster than the second best method, PSI-BLAST with 3 iterations (Supplementary Fig. S6).

BLASTp and PSI-BLAST could easily find the LTP-like proteins presenting a high sequence identity to input AtLTP1s. However,

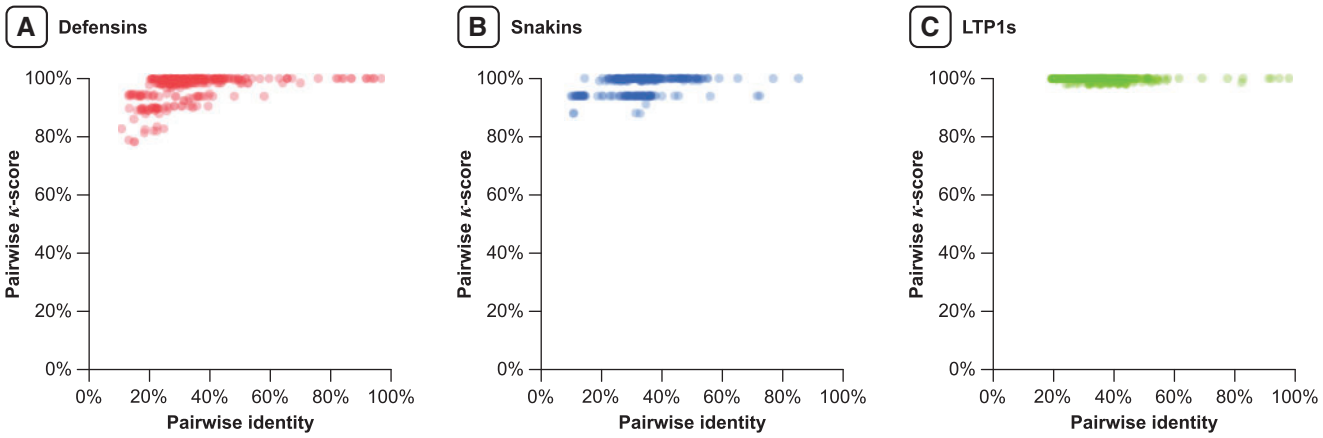

Fig. 2. Comparison of all-versus-all pairwise identities and κ-scores within three reference CRP families. KAPPA was used with default parameters, except for defensins for which one gain or loss of cysteine was allowed

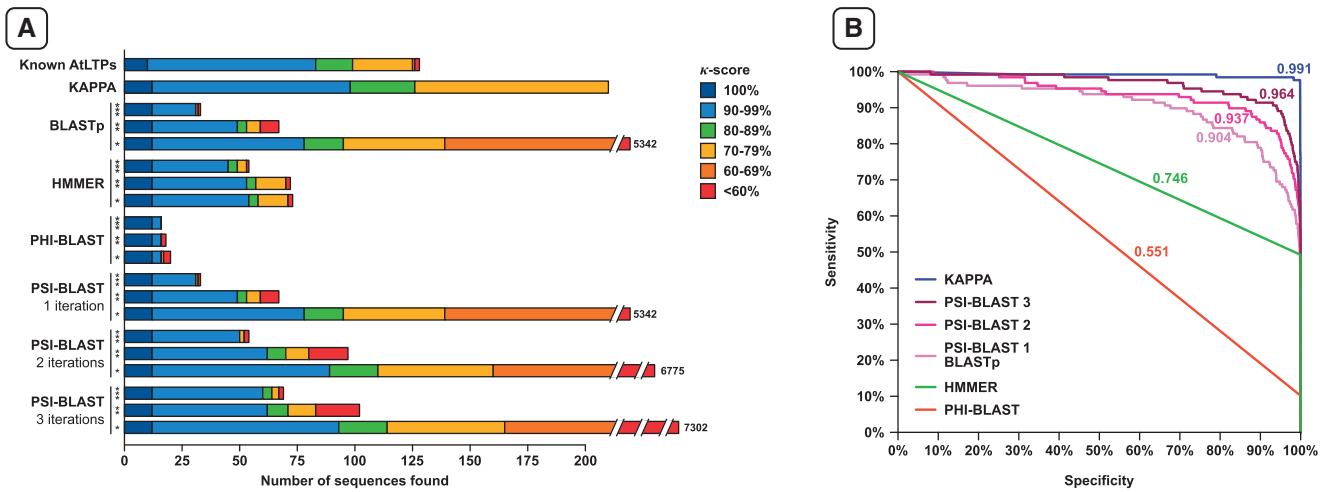

Fig. 3. Assessment of *ab initio* sequence search performance for KAPPA and other programs. (**A**) Comparison of outputs from different programs. AtLTP1s were used as input sequences to parse the *A. thaliana* proteome available in Phytozome 9.1. KAPPA was used with parameters specified in Supplementary Table S1. Other programs were used with default parameters and three levels of stringency based on e-value: 10^{-3} (***), 1 (**) or 1000 (*). PSI-BLAST was run with 1, 2 and 3 iterations. Colours refer to the κ-score of the output sequences with respect to the consensus pattern of AtLTP1s. Known AtLTPs described by Silverstein *et al.* (2007) and Edstam *et al.* (2011) were used to define a minimal expected output. (**B**) Sensitivity-specificity ROC plot comparing performances of KAPPA and other programs, using known AtLTPs as reference true sequences. Sensitivity refers to the true positive rate while specificity corresponds to the true negative rate. Values indicated on the graph correspond to the area under the curve

more dissimilar LTP-like proteins could not be retrieved unless the stringency level was decreased. In this situation, a high number of non-related proteins accumulated in the output since they have the same low sequence similarity to AtLTP1s. Supplementary Figure S4C shows that the κ-score correctly discriminated LTP-like proteins from other sequences, including LTP-like with a low sequence identity, whereas the BLASTp e-value did not make a difference. Supplementary Figures S4D–G show this was the same for outputs from other programs.

PHI-BLAST and HMMER appeared to be way more efficient in detecting target LTP-like CRPs without accumulating non-related proteins. However, they were also much more restrictive. Although they were able to give more weight to cysteine residues in the similarity search, both tools seemed unable to deal with variation in cysteine number and spacing. In contrast, KAPPA took advantage of a more or less stringent calculation of block identity I and key residue persistence P to address this issue.

Another problem may be due to the fact we used proteins stemming from the same LTP subfamily and from one single species. In this situation, residues that are common to all input proteins can not only be cysteine characteristic of all LTPs, but also other residues that are specific to AtLTP1s only. Although the first ones are expected to be highly conserved in other LTP subfamilies and in other species, the latter may have changed more rapidly in the course of evolution. Yet, the algorithms to which we compared KAPPA gave the same weight to all of these residues, which can limit their performance in detecting more distant LTP-like proteins.

KAPPA's efficiency in *ab initio* sequence search was also demonstrated taking other examples. We first looked for AtLTP1s in rice (Supplementary Fig. S7), and then turned to study other families of plant CRPs described by Silverstein *et al.* (2007): defensins (Supplementary Figs. S8 and S9) and snakins (Supplementary Figs. S10 and S11). Finally, KAPPA was tested with other key residues: 2008 human and 1917 mouse proteins containing the extended

glycine zipper (EGZ) motif described by Kim *et al.* (2005) were discovered (Supplementary Figs. S12 and S13). Moreover, KAPPA could also efficiently find all members of the small proline-rich protein 2 (SPRR2) family (Cabral *et al.*, 2001) in human and mouse proteomes (Supplementary Figs. S14 and S15).

As shown in Supplementary Figure S16, BLASTp, PSI-BLAST and HMMER could perform as well as KAPPA when studying protein families displaying high sequence identity (e.g. snakins, SPRR2s). PHI-BLAST also efficiently detected target proteins when the key residue spacing is highly conserved (e.g. snakins, EGZs). However, KAPPA's performance was equal or superior to other programs in all cases, combining high sensitivity and specificity. KAPPA thus appears to be a reliable, all-purpose tool to detect any type of protein displaying a key amino acid pattern, with a clear advantage over existing methods when dealing with families displaying extensive sequence divergence or small variations in the key residue spacing.

3.3 *De novo* clustering of CRPs

Once potential CRPs are detected by the pre-filtering function or by a reference-guided *ab initio* search, it may be of interest to split them into clusters defined by precise cysteine spacing. Indeed, Edstam *et al.* (2011) described 79 LTPs in *A. thaliana* and divided them into nine clusters and three singletons. Likewise, Silverstein *et al.* (2007) found 131 new LTPs fragmented into 23 clusters and 8 singletons. In both of these studies, clustering is performed in a more or less arbitrary way, taking into account not only pattern but also sequence similarity. Thus, this experimental procedure can be long, tedious and partially subjective.

KAPPA introduces the possibility of clustering proteins (i) *de novo*, i.e. without relying on a set of reference patterns; (ii) in an automated fashion and (iii) using clear quantitative criteria. As a first step, the κ-score can be used to perform a pattern-based clustering of proteins. Second, the remaining sequences can be used to launch a subclustering step based on classical sequence similarity values (e.g. percentage of identities and positives).

Because KAPPA detected 210 LTPs in the previous section, we tested its ability to divide them into clusters that would be consistent (high intra-group similarity) and distinct from each other (lower inter-group similarity). The *de novo* clustering function was used as follows: first, all pairs of proteins connected with a κ-score ~97% were recursively bound to form preclusters, which were fused to form final clusters if ≥90% of inter-group sequence pairs had a κ-score above 95%. These settings, chosen with the KAPPA optimization function, yielded 21 LTP clusters and 46 singletons.

Figure 4A compares the structure of AtLTPs clusters formed by KAPPA, Edstam *et al.* (2011) and Silverstein *et al.* (2007). The stringent criteria we used to build KAPPA clusters led to a high intra-cluster pairwise κ-score with a narrow distribution that is clearly distinct from a low, wide distribution of inter-cluster κ-scores. Cluster homogeneity is still clear for Edstam's clusters, but starts to be questionable for Silverstein's clusters, because a notable proportion of intra-cluster pattern similarities are lower than inter-cluster similarities.

Figure 4B and Supplementary Table S3 shed light on the relative homogeneity between clusters generated automatically by KAPPA and manually by Edstam *et al.*: KAPPA perfectly reformed clusters AtLTP1, AtLTP2, AtLTPc and AtLTPe and extended the two first with new proteins (in grey). Clusters AtLTPd and AtLTPg were fractioned into 2 and 3 KAPPA clusters, respectively. Attention paid to protein alignments (Supplementary Fig. S17) clearly shows this separation is supported by differences in the cysteine spacing. Moreover, 3 of the 4 Edstam's singletons (AtLTPx, in black) could now be assigned to clusters.

The nine KAPPA clusters represented in Figure 4B gathered 136 proteins (65% of the total) and contained on average 15.1 proteins per cluster. These proteins are quite close to the original AtLTP1s

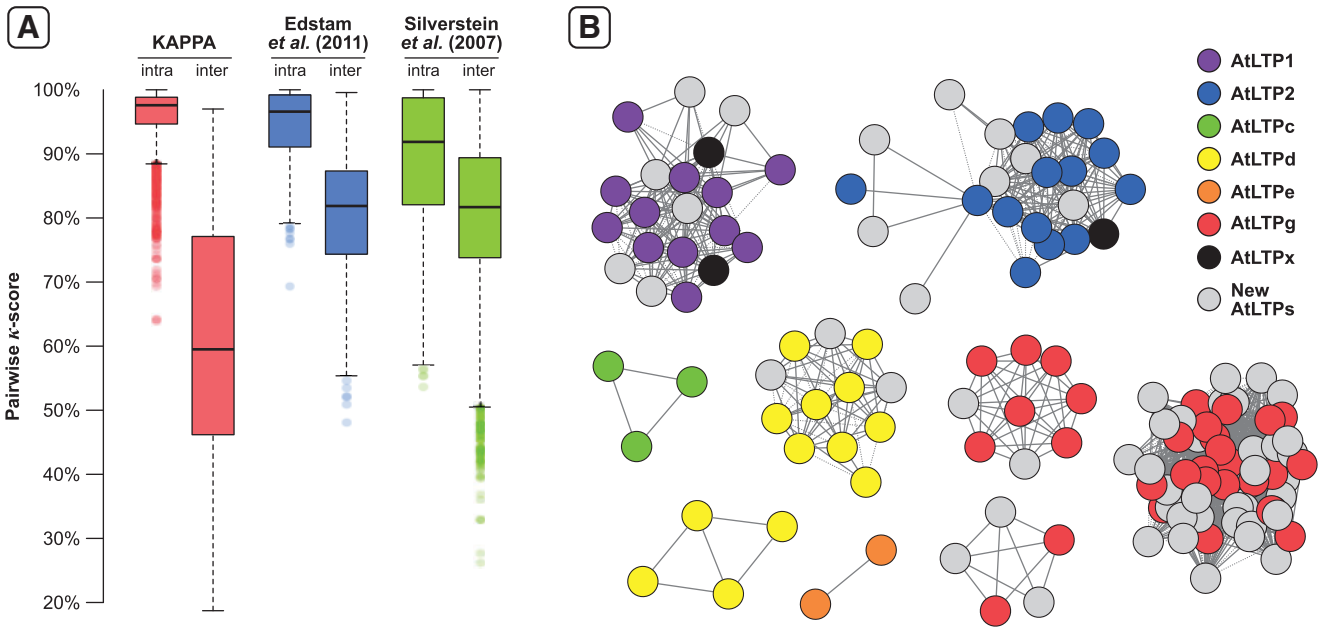

Fig. 4. Compared clustering performance on AtLTPs. (**A**) Distributions of κ-scores for all possible pairs of sequences belonging to the same cluster (intra) or to two different clusters (inter) in AtLTPs clusters generated by KAPPA (210 LTPs, 21 clusters and 46 singletons), Edstam *et al.* (79 LTPs, 6 clusters and 3 singletons) and Silverstein *et al.* (131 LTPs, 23 clusters and 8 singletons). (**B**) Graphical view of the nine KAPPA clusters containing at least one of the 79 LTPs from Edstam *et al.* (2011) generated with Cytoscape. Coloured nodes correspond to LTPs described by KAPPA and Edstam *et al.*; grey nodes represent LTPs newly described by KAPPA. Edge length reflects pattern similarity: the longer the edge, the smaller the pairwise κ-score

that were used as queries for the *ab initio* search, with a mean κ-score of 91.0% with respect to the AtLTP1 cysteine pattern.

The remaining 74 proteins were, however, distributed into 12 smaller clusters (2.2 proteins per cluster) and 48 singletons. Their isolated status probably reflects their more distant nature as LTP-like proteins. Indeed, they all correspond to LTPs that were not described by Edstam *et al.* (except one AtLTPg) and their mean κ-score with respect to the AtLTP1 pattern was much lower (75.7%).

Together, these results suggest that the KAPPA clustering function is able to reproduce and improve previous, manually formed groups.

4 Discussion

By providing an automated pipeline specifically dedicated to the evolutive specificities of proteins defined by a key amino acid pattern, like CRPs, KAPPA fills a gap in the landscape of sequence search computational tools. Supplementary Table S4 presents salient advantages of KAPPA over more conventional approaches.

The κ-score appears to be a robust, quantitative and objective tool to describe cysteine pattern similarity and explore all its aspects: block identity *I* accounts for subtle modifications of cysteine spacing within a given family, while pattern persistence *P* allows to investigate emergence of new groups of CRPs within a given taxon.

Numerous families of CRPs have been detected and characterized so far in a small number of model organisms. Moreover, the number and availability of new sequenced proteomes is quickly growing, especially for non-model organisms. Considering the functional importance of CRPs in plants and animals, there is an increasing need to characterize their orthologues in these new proteomes.

Focusing on the cysteine spacing with the κ-score, the KAPPA *ab initio* search function made it possible not only to detect new members of CRP families within reference species themselves, but also orthologues belonging to more distant species. Furthermore, the *ab initio* search is also flexible, since mapping parameters, especially stringency options, enable users to choose an optimal research framework. One can indeed decide to be strict in the similarity search, using a high κ-score threshold or high stringency parameters, or more permissive and detect new '-like' proteins.

Traditional sequence search approaches such as BLASTp do not make a difference between key residues and the rest of the sequence; hence, true homologues of input CRPs are intermingled with non-related, low similarity sequences. Although HMMER and PSI/PHI-BLAST can give more weight to cysteines and other conserved residues, they are less performant in dealing automatically with extensive divergence of blocks between cysteines, and with fine modifications of the cysteine spacing itself. This is why previous reports dealing with CRPs always implemented curation steps relying on manual review of data or home-made scripts to obtain a consistent final protein dataset.

Though often leading to correct results, this time-consuming approach is suboptimal when dealing with large-scale, proteome-wide studies. KAPPA addresses this issue by providing a fast and accurate analysis pipeline based on quantitative criteria. Moreover, the optimization functionality makes it easy to determine the best parameters for a given sequence search experiment.

Besides providing an automated pipeline, KAPPA also innovates by relieving the user of the need for reference cysteine patterns. Indeed, the *de novo* search and clustering function applied to a whole proteome can uncover totally new families of CRPs without a reference.

In the challenging case of proteins defined by a key amino acid pattern, KAPPA provides a new and accurate detection method over HMMER-based sequence search strategies implemented in previous CRP studies (Silverstein *et al.*, 2007; Edstam *et al.*, 2011) and in gene-finding pipelines such as SPADA (Zhou *et al.* 2013).

Implementation of KAPPA on available proteomes opens the way to a better and quicker understanding of the diversity, evolution and functions these peculiar proteins, as shown for the CRPs, EGZ proteins and SPRR2s.

Acknowledgements

We would like to thank Maxime Caumartin for his kind review of the KAPPA source code and user's guide.

Funding

This work was supported by the Fonds de Recherche du Québec—Nature et Technologies (FRQ-NT) and the Canadian Natural Sciences and Engineering Research Council (NSERC).

Conflict of Interest: none declared.

References

Altschul,S.F. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.,* **215**, 403–410.

Altschul,S.F. *et al.* (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.,* **25**, 3389–3402.

Bella,J. *et al.* (2008) The leucine-rich repeat structure. *Cell. Mol. Life Sci.,* **65**, 2307–2333.

Bendtsen,J.D. *et al.* (2004) Feature based prediction of non-classical and leaderless protein secretion. *Protein Eng. Des. Sel.,* **17**, 349–356.

Cabral,A. *et al.* (2001) Structural organization and regulation of the small proline-rich family of cornified envelope precursors suggest a role in adaptive barrier function. *J. Biol. Chem.,* **276**, 19231–19237.

Chevalier,E. *et al.* (2011) Cell-cell communication and signalling pathways within the ovule: from its inception to fertilization. *New Phytol.,* **192**, 13–28.

Chevalier,E. *et al.* (2013) ScRALF3, a secreted RALF-like peptide involved in cell-cell communication between the sporophyte and the female gametophyte in a solanaceous species. *Plant J.,* **73**, 1019–1033.

de Souza Cândido,E. *et al.* (2014) The use of versatile plant antimicrobial peptides in agribusiness and human health. *Peptides,* **55C**, 65–78.

Edgar,R.C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.,* **32**, 1792–1797.

Edstam,M.M. *et al.* (2011) Evolutionary history of the non-specific lipid transfer proteins. *Mol. Plant,* **4**, 947–964.

Finn,R.D. *et al.* (2011) HMMER web server: interactive sequence similarity searching. *Nucleic Acids Res.,* **39**(Web server issue), W29–W37.

Giacomelli,L. *et al.* (2012) Identification and characterization of the defensin-like gene family of grapevine. *Mol. Plant Microbe. Interact.,* **25**, 1118–1131.

Halary,S. *et al.* (2013) EGN: a wizard for construction of gene and genome similarity networks. *BMC Evol. Biol.,* **13**, 146.

Hanks,J.N. *et al.* (2005) Defensin gene family in Medicago truncatula: structure, expression and induction by signal molecules. *Plant Mol. Biol.,* **58**, 385–399.

Higashiyama,T. (2010) Peptide signaling in pollen-pistil interactions. *Plant Cell Physiol.,* **51**, 177–189.

Kim,S. *et al.* (2005) Transmembrane glycine zippers: physiological and pathological roles in membrane proteins. *Proc. Natl Acad. Sci. USA,* **102**, 14278–14283.

Koppers,A.J. *et al.* (2011) The role of cysteine-rich secretory proteins in male fertility. *Asian J. Androl.,* **13**, 111–117.

Marshall,E. *et al.* (2011) Cysteine-rich peptides (CRPs) mediate diverse aspects of cell-cell communication in plant reproduction and development. *J. Exp. Bot.,* **62**, 1677–1686.

Mousavi,A. and Hotta,Y. (2005) Glycine-rich proteins: a class of novel proteins. *Appl. Biochem. Biotechnol.,* **120**, 169–174.

Odintsova,T. and Egorov,T. (2012) Plant antimicrobial peptides. In: H.R., Irving and C., Gehring (eds.) *Plant Signaling Peptides, Vol. 16 of Signaling and Communication in Plants*. Springer, Berlin, pp. 107–133.

Ouyang,S. *et al.* (2007) The TIGR rice genome annotation resource: improvements and new features. *Nucleic Acids Res,* **35**(Suppl. 1), D883–D887.

Pasupuleti,M. *et al.* (2012) Antimicrobial peptides: key components of the innate immune system. *Crit. Rev. Biotechnol.,* **32**, 143–171.

Petersen,T.N. *et al.* (2011) SignalP 4.0: discriminating signal peptides from transmembrane regions. *Nat. Methods,* **8**, 785–786.

Scocchi,M. *et al.* (2011) Proline-rich antimicrobial peptides: converging to a non-lytic mechanism of action. *Cell. Mol. Life Sci.,* **68**, 2317–2330.

Shannon,P. *et al.* (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.,* **13**, 2498–2504.

Silverstein,K.A. *et al.* (2005) Genome organization of more than 300 defensin-like genes in Arabidopsis. *Plant Physiol.,* **138**, 600–610.

Silverstein,K.A. *et al.* (2007) Small cysteine-rich peptides resembling antimicrobial peptides have been under-predicted in plants. *Plant J.,* **51**, 262–280.

Swarbreck,D. *et al.* (2008) The arabidopsis information resource (TAIR): gene structure and function annotation. *Nucleic Acids Res.,* **36**(Suppl. 1), D1009–D1014.

van de Velde,W. *et al.* (2010) Plant peptides govern terminal differentiation of bacteria in symbiosis. *Science,* **327**, 1122–1126.

Wong,E.S.W. and Belov,K. (2012) Venom evolution through gene duplications. *Gene,* **496**, 1–7.

Zhou,P. *et al.* (2013) Detecting small plant peptides using SPADA (Small Peptide Alignment Discovery Application). *BMC Bioinf.,* **14**, 335.

Zhu,S. *et al.* (2005) Phylogenetic distribution, functional epitopes and evolution of the CSalphabeta superfamily. *Cell. Mol. Life Sci.*, **62**, 2257–2269.