PIANA: protein interactions and network analysis

Ramon Aragues, Daniel Jaeggi and Baldo Oliva

ABSTRACT
Summary: We present a software framework and tool called Protein Interactions And Network Analysis (PIANA) that facilitates working with protein interaction networks by (1) integrating data from multiple sources, (2) providing a library that handles graph-related tasks and (3) automating the analysis of protein–protein interaction networks. PIANA can also be used as a stand-alone application to create protein interaction networks and perform tasks such as predicting protein interactions and helping to identify spots in a 2D electrophoresis gel.

Availability: PIANA is under the GNU GPL. Source code, database and detailed documentation may be freely downloaded from http://sbi.imim.es/piana.

Contact: ramon.aragues@upf.edu; boliva@imim.es

1 INTRODUCTION
The analysis of protein interaction networks is fundamental to the understanding of cellular processes (Salwinski and Eisenberg, 2003; Yook et al., 2004). Furthermore, protein interaction networks are being used in tasks such as assignment of function to uncharacterized proteins (Huynen et al., 2003) and searching for remote similarities between proteins (Espadaler et al., 2005a). Some tools developed to visualize and analyze protein–protein interaction networks are Cytoscape (Shannon et al., 2003), Osprey (Breitkreutz et al., 2003), VisANT (Hu et al., 2005) and ProViz (Iragne et al., 2005). Most of these tools focus on visualizing the networks, while a few of them have analytic capabilities.

Protein Interactions And Network Analysis (PIANA) is a software framework that integrates data from multiple sources into a single repository, creates interaction networks, predicts novel interactions and performs automatic analyses. PIANA is different from most other tools in that (1) it is also a framework on which developers can base their applications, (2) it integrates most protein and interaction databases into a single repository and (3) it performs analyses not provided by other tools.

2 PIANA ARCHITECTURE
PIANA has been implemented as a collection of python modules that can be used separately as libraries or as a stand-alone application through a user interface.

The database module
The database module consists of a MySQL database and a library used as an interface to the database. A limited version of a PIANA MySQL database containing interactions from DIP (Salwinski et al., 2004) and interactions predicted from sequence/structure distant patterns (Espadaler et al., 2005b) can be downloaded from our website.

The parsing module
PIANA includes parsers for the main protein databases [UniProt (Bairoch et al., 2005), NCBI GenBank (Benson et al., 2005)] and for protein interaction repositories such as DIP, STRING (von Mering et al., 2003), MIPS (Pagel et al., 2005), BIND (Alfarano et al., 2005) and HPRD (Peri et al., 2003). PIANA can also parse flat text files and interaction data that follow the HUPO PSI MI standard (Hermjakob et al., 2004). Moreover, PIANA provides parsers for databases such as COG (Tatusov et al., 2003), GO (Ashburner et al., 2000) and SCOP (Murzin et al., 1995). These databases contain information that PIANA uses when performing the analyses.

The network module
PIANA implements classes and methods for working with networks. Moreover, PIANA has methods specifically designed for biological networks such as clustering proteins by their molecular function and visualizing the networks in formats appropriate for biological analysis.

3 PIANA CAPABILITIES
Data integration
PIANA accepts as input most types of protein database identifiers and contains cross-references between them. Therefore, interactions from different sources can be integrated into a single network. Currently, the type of input and output protein database identifiers accepted by PIANA are UniProt entry names and accession numbers, gene names, NCBI GenBank gi, EMBL, PDB, PIR and the protein sequence.

Creation of protein–protein interaction networks
Usually, a list of proteins of interest is given as input. PIANA searches in its database for interactions where these proteins are linked (i.e. interactions) and nodes (i.e. protein interaction partners) to the network until a given depth is reached, where depth is defined as the number of interacting steps taken from the original proteins. Internally, a protein interaction network is represented as a set of nodes (proteins) connected by edges (interactions). The networks can be visualized in different formats, mainly tables that describe in detail each interaction and DOT files, which can be used to produce network images. PIANA also has the...
possibility of applying output filters such as highlighting proteins that perform specific functions or identifying proteins in the network whose genes have been found over- or under-expressed in a microarray experiment.

**Predicting novel interactions**
PIANA transfers interactions between proteins that share a given property. For example, PIANA predicts interactions using ‘interologs’ (Yu et al., 2004) by means of COG codes. In a similar way, SCOP codes can be used to transfer interactions between proteins that share a domain family.

**Finding ‘interaction distance’ between proteins**
PIANA can obtain lists of proteins that are at a certain interaction distance (i.e. minimum number of edges separating two proteins) from another protein, which can be useful for tasks such as searching for remote similarities between proteins (Espadaler et al., 2005a).

**Matching spots from electrophoresis experiments**
PIANA can be used to help identify spots in a 2D electrophoresis gel. Spots not identified by mass spectrometry are putatively assigned to proteins in the network by comparing their molecular weights and isoelectric points.

**Clustering proteins by their GO terms**
Networks can become very complex and hence, clustering methods are needed to facilitate their interpretation. PIANA provides a library for applying agglomerative hierarchical clustering to protein interaction networks. For example, using the annotation provided by GO, PIANA groups into clusters those proteins in the network that have similar biological processes or molecular functions. The distance function used for this clustering is based on the length of the path between the GO terms in the GO hierarchical tree. The stop condition is set by the user by means of two thresholds: minimum similarity accepted in order to group two clusters and minimum distance from the terms in the cluster to the GO root term.

**Extending PIANA**
New functionalities can be added to PIANA by extending the current python classes. Moreover, PIANA implements a class called PianaApi that can be used from other Python programs to work with interaction networks.

**4 EXAMPLE**
We illustrate the use of PIANA with two genes (MMP1 and LTBP1) that have been found to mediate breast cancer metastasis to lung (Minn et al., 2005). First of all, we create a PIANA configuration file where we set (1) the input parameters (e.g. input proteins and network depth), (2) the output parameters (e.g. type of protein identifiers to be used) and (3) the PIANA commands to execute (e.g. create network for the proteins and predict interactions based on interologs). Then, we run PIANA with the configuration file as an argument. Figure 1 shows the protein interaction network for MMP1 and LTBP1 (a) before and (b) after adding predictions based on interologs. A detailed PIANA example using all the genes from (Minn et al., 2005) and performing an in-depth analysis of the interaction network can be found at http://sbi.imim.es/piana/example.html.

Furthermore, PIANA has been previously used for the study of biological pathways in breast cancer cells (España et al., 2005).

**5 FUTURE WORK**
Future plans for PIANA include the annotation of proteins based on network motifs, prediction of protein structure using interactions (Espadaler et al., 2005b) and developing a reliability score for interactions. We intend as well to introduce algorithms that split proteins into the domains that perform the interactions.

**ACKNOWLEDGEMENTS**
The authors thank J. Planas, P. Boixeda, B. Gregori and L. Salwinski for their helpful comments. R.A. is supported by a grant from the Spanish Ministerio de Ciencia y Tecnología (MCyT, BIO2002-03609). This work has been supported by grants from Fundación Ramón Areces, from the Spanish Ministerio de Educación y Ciencia (MEC, BIO2002-00533), the ‘Programa Gaspar de Portolá (DURSI)’, and by EU grant INFOBIOMED-NoE (IST-507585).

**Conflict of Interest**: none declared.

**REFERENCES**


