PBxplore: a tool to analyze local protein structure and deformability with Protein Blocks

Jonathan Barnoud[1,2,3,4,5,†], Hubert Santuz[1,2,3,4,†], Pierrick Craveur[1,2,3,4,6], Agnel Praveen Joseph[1,2,3,4,7], Vincent Jallu[4,8], Alexandre G. de Brevern[1,2,3,4,*,‡], and Pierre Poulain[1,2,3,4,9,*,‡]

[1]INSERM, U 1134, DSIMB, F-75739 Paris, France.
[2]Univ. Paris Diderot, Sorbonne Paris Cité, UMR-S 1134, F-75739 Paris, France.
[3]Institut National de la Transfusion Sanguine (INTS), F-75739 Paris, France.
[4]Laboratoire d'Excellence GR-Ex, F-75739 Paris, France.
[5]Groningen Biomolecular Sciences and Biotechnology Institute and Zernike Institute for Advanced Materials, University of Groningen, Nijenborgh 7, AG Groningen 9747, The Netherlands.
[6]The Scripps Research Institute, Department of Integrative Structural and Computational Biology, 10550 North Torrey Pines Road, La Jolla, CA 92037, USA.
[7]Birkbeck College, University of London, Malet Street, London WC1E 7HX, UK.
[8]INTS, Platelet Unit, F-75739 Paris, France.
[9]Mitochondria, Metals and Oxidative Stress Group, Institut Jacques Monod, UMR 7592, Univ. Paris Diderot, CNRS, Sorbonne Paris Cité, F-75205 Paris, France.
[†]These authors contributed equally to this work.
[‡]These authors contributed equally to this work.
[*]Corresponding authors: alexandre.de-brevern@inserm.fr, pierre.poulain@univ-paris-diderot.fr

ABSTRACT

Proteins are highly dynamic macromolecules. A classical way to analyze their inner flexibility is to perform molecular dynamics simulations. In this context, we present the advantage to use small structural prototypes, namely the Protein Blocks (PBs). PBs give a good approximation of the local structure of the protein backbone. More importantly, by reducing the conformational complexity of protein structures, they allow analyzes of local protein deformability which cannot be done with other methods and had been used efficiently in different applications. PBxplore is a suite of tools to analyze the dynamics and deformability of protein structures using PBs. It is able to process large amount of data such as those produced by molecular dynamics simulations. It produces various outputs with text and graphics, such as frequencies, entropy and information logo. PBxplore is available at https://github.com/pierrepo/PBxplore and is released under the open-source MIT license.

INTRODUCTION

Proteins are highly dynamic macromolecules (Frauenfelder et al., 1991; Bu and Callaway, 2011). To analyze their inner flexibility, computational biologists often use molecular dynamics (MD) simulations. The quantification of protein flexibility is based on various methods such as Root Mean Square Fluctuations (RMSF) that relies on multiple MD snapshots or Normal Mode Analysis (NMA) that relies on a single structure and focus on quantifying large movements.

Other interesting *in silico* approaches assess protein motions through the protein residue network (Atilgan et al., 2007) or dynamical correlations from MD simulations (Ghosh and Vishveshwara, 2007; Dixit and Verkhivker, 2011). We can also notice the development of the MOdular NETwork Analysis (MONETA), which localizes the perturbations propagation throughout a protein structure (Laine et al., 2012).

Here we use an alternative yet powerful approach based on small prototypes or ”structural alphabets” (SAs). SAs approximate conformations of protein backbones and code the local structures of proteins as one-dimensional sequences (Offmann et al., 2007). Protein Blocks (PBs) (de Brevern et al., 2000) is one of these SAs (de Brevern, 2005; Etchebest et al., 2005; Joseph et al., 2010).

PBs are composed of 16 blocks designed through an unsupervised training performed on a representative non-redundant databank of protein structures (de Brevern et al., 2000). PBs are defined from a set of dihedral angles describing the protein backbone. This property makes PBs interesting conformational prototypes of the local protein structure. PBs are labeled from *a* to *p* (see Fig. 1a). PBs *m* and *d* are prototypes for central α-helix and central β-strand, respectively. PBs *a* to *c* primarily represent β-strand N-caps and PBs *e* and *f*, β-strand C-caps; PBs *g* to *j* are specific to coils, PBs *k* and *l* are specific to α-helix N-caps, and PBs *n* to *p* to α-helix C-caps (de Brevern, 2005). Figure 1 illustrates how a PB sequence is assigned from a protein structure. Starting from the 3D coordinates of the barstar protein (Fig. 1b), the local structure of each amino acid is compared to the 16 PB definitions (Fig. 1a). The most similar protein block is assigned to the residue under consideration (the similarity metric is explained latter in this article). Eventually, assignment leads to the PB sequence represented in Fig. 1c.

By reducing the complexity of protein structure, PBs have been showed to be efficient and relevant in a wide spectrum of applications. To name a few, PBs have been used to analyze protein contacts (Faure et al., 2008), to propose a structural model of a transmembrane protein (de Brevern, 2005), to reconstruct globular protein structures (Dong et al., 2007), to design peptides (Thomas et al., 2006), to define binding site signatures (Dudev and Lim, 2007), to perform local protein conformation predictions (Li et al., 2009; Rangwala et al., 2009; Suresh et al., 2013; Suresh and Parthasarathy, 2014; Zimmermann and Hansmann, 2008), to predict β-turns (Nguyen et al., 2014) and to understand local conformational changes due to mutations of the αIIbβ3 human integrin (Jallu et al., 2012, 2013, 2014).

PBs are also useful to compare and superimpose protein structures with pairwise and multiple approaches (Joseph et al., 2011, 2012), namely iPBA (Gelly et al., 2011) and mulPBA (Léonard et al., 2014), both currently showing best results compared to other superimposition methods. Eventually, PBs lead to interesting results at predicting protein structures from their sequences (Ghouzam et al., 2015, 2016) and at predicting protein flexibility (Bornot et al., 2011; de Brevern et al., 2012).

Our results on biological systems such as, the DARC protein (de Brevern et al., 2005), the human αIIbβ3 integrin (Jallu et al., 2012, 2013, 2014) and the KISSR1 protein (Chevrier et al., 2013), highlighted the usefulness of PBs to understand local deformations of large protein structures. Specially, these analyzes have shown that a region considered as highly flexible through RMSF quantifications, can be seen through PBs as locally highly rigid. This unexpected behavior is explained by a local rigidity, surrounded by deformable regions (Craveur et al., 2015). To go further, we recently used PBs to analyze long-range allosteric interactions in the Calf-1 domain of αIIb integrin (Goguet et al., 2017). To our knowledge, the only other related approach based on SA to assess local deformation is GSATools (Pandini et al., 2013), it is specialized in the analysis of functional correlations between local and global motions, and the mechanisms of allosteric communication.

Despite the versatility of PBs and the large spectrum of their applications, PBs lack a uniform and easy-to-use toolkit to assign PB sequences from 3D structures, and to analyze these sequences. The only known implementation is a an old C program not publicly available and not maintained anymore. Such tool not being available reduces the availability of the PBs for studies where they would be meaningful.

We thus propose PBxplore, a tool to analyze local protein structure and deformability using PBs. It is available at https://github.com/pierrepo/PBxplore. PBxplore can read PDB structure files (Bernstein et al., 1977), PDBx/mmCIF structure files (Bourne et al., 1997), and MD trajectory formats from most MD engines, including Gromacs MD topology and trajectory files (Lindahl et al., 2001; van der Spoel et al., 2005). Starting from 3D protein structures, PBxplore assigns PBs sequences; computes a local measurement of entropy, a density map of PBs along the protein sequence and a WebLogo-like representation of PBs.

In this paper, we first present the principle of PBxplore, then its different tools, and finally a step-by-step user-case with the β3 subunit of the human platelet integrin αIIbβ3.

DESIGN AND IMPLEMENTATION

PBxplore is written in Python (van Rossum, 1995; Software, 2010; Bassi, 2007). It is compatible with Python 2.7, and with Python 3.4 or greater. It requires the Numpy Python library for array manipulation

(Ascher et al., 1999), the matplotlib library for graphical representations, and the MDAnalysis library for molecular dynamics simulation files input (Michaud-Agrawal et al., 2011; Gowers et al., 2016). Optionally, PBxplore functionalities can be enhanced by the installation and the use of WebLogo (Crooks et al., 2004) to create sequence logos.

PBxplore is available as a set of command-line tools and as a Python module. The command-line tools allow for an easy integration of PBxplore in existing analysis pipelines. These programs can be linked up together to carry out the most common analyses on PB sequences to provide insights on protein flexibility. In addition, the PBxplore Python library provides an API to access its core functionalities which allows the integration of PBxplore in Python programs and workflows, and the extension of the method to suit new needs.

PBxplore is released under the open-source MIT license (Open Source Initiative, 2014). It is available on the software development platform GitHub (GitHub, 2007) at https://github.com/pierrepo/PBxplore.

The package contains unit and regression tests and is continuously tested using Travis CI (Travis CI, 2015). An extensive documentation is available on Read the Docs (Holscher et al., 2010) at https://pbxplore.readthedocs.io.

Installation

The easiest way to install PBxplore is through the Python Package Index (PyPI):

```
pip install --user pbxplore
```

It will ensure all required dependencies are installed correctly.

Command-line Tools

A schematic description of PBxplore command line interface is provided in Fig. 2. The interface is composed of three different programs: `PBassign` to assign PBs, `PBcount` to compute PBs frequency on multiple conformations, and `PBstat` to perform statistical analyses and visualization. These programs can be linked up together to make a structure analysis pipeline to study protein flexibility.

PBassign

The very first task is to assign PBs from the protein structure(s). A PB is associated to each pentapeptide included in the protein sequence. To assign a PB to a residue n, 5 residues are required (residues $n-2$, $n-1$, n, $n+1$ and $n+2$). From the structure of these 5 residues, 8 dihedral angles (ψ and ϕ) are computed, going from the ψ angle of residue $n-2$ to the ϕ angle of residue $n+2$ (de Brevern, 2005). This set of 8 dihedral angles is then compared to the reference angles set of the 16 PBs (de Brevern et al., 2000) using the Root Mean Square Deviation Angle (RMSDA) measure, i.e., an Euclidean distance on angles. PB with the smallest RMSDA is assigned to residue n. A dummy PB Z is assigned to residues for which all 8 angles cannot be computed. Hence, the first two N-terminal and the last two C-terminal residues are always assigned to PB Z.

The program `PBassign` reads one or several protein 3D structures and performs PBs assignment as one PBs sequence per input structure. `PBassign` can process multiple structures at once, either provided as individual structure files, as a directory containing many structure files or as topology and trajectory files issued from MD simulations. Note that PBxplore is able to read any trajectory file format handled by the MDAnalysis library, yet our tests focused on Gromacs trajectories. Output PBs sequences are bundled in a single file in fasta format.

PBcount

During the course of a MD simulation, the local protein conformations can change. It is then interesting to analyze them through PB description. Indeed, as each PB describes a local conformation, the variability of the PB assigned to a given residue throughout the trajectory indicates some local deformation of the protein structure. Thus, once PBs are assigned, PBs frequencies per residue can be computed.

The program `PBcount` reads PBs sequences for different conformations of the same protein from a file in the fasta format (as outputted by `PBassign`). Many input files can be provided at once. The output data is a 2D matrix of x rows by y columns, where x is the length of the protein sequence and y is the 16 distinct PBs. A matrix element is the count of a given PB at a given position in the protein sequence.

PBstat

The number of possible conformational states covered by PBs is higher than the classical secondary structure description (16 states instead of 3). As a consequence, the amount of information produced by `PBcount` can be complex to handle. Hence, we propose three simple ways to visualize the variation of PBs which occur during a MD simulation.

The program `PBstat` reads PBs frequencies as computed by `PBcount`. It can produce three types of outputs based on the input argument(s). The first two use the matplotlib library and the last one requires the installation of the third-party tool Weblogo (Crooks et al., 2004). `PBstat` also offers two options (`--residue-min` and `--residue-max`) to define a residue frame allowing the user to quickly look at segments of interest. The three graphical representations proposed are:

- *Distribution of PBs.* This feature plots the frequency of each PB along the protein sequence. The output file could be in format .png, .jpg or .pdf. A dedicated colorblind safe color range (Brewer et al., 2013) allows visualizing the distribution of PBs. For a given position in the protein sequence, blue corresponds to a null frequency when the particular PB is never sampled at this position and red corresponds to a frequency of 1 when the particular PB is always found at this position. This representation is produced with the `--map` argument.

- *Equivalent number of PBs (N_{eq}).* The N_{eq} is a statistical measurement similar to entropy (Offmann et al., 2007). It represents the average number of PBs sampled by a given residue. N_{eq} is calculated as follows:

  $$N_{eq} = \exp(-\sum_{i=1}^{16} f_x \ln f_x)$$

  where f_x is the probability (or frequency) of the PB x. A N_{eq} value of 1 indicates that only a single type of PB is observed, while a value of 16 is equivalent to a random distribution, i.e. all PBs are observed with the same frequency 1/16. For example, a N_{eq} value around 5 means that, across all the PBs observed at the position of interest, 5 different PBs are mainly observed. If the N_{eq} exactly equals to 5, this means that 5 different PBs are observed in equal proportions (i.e. 1/5).

  A high N_{eq} value can be associated with a local deformability of the structure whereas a N_{eq} value close to 1 means a rigid structure. In the context of structures issued from MD simulations, the concept of deformability / rigidity is independent to the one of mobility. The N_{eq} representation is produced with the `--neq` argument.

- *Logo representation of PBs frequency.* This is a WebLogo-like representation (Crooks et al., 2004) of PBs sequences. The size of each PB is proportional to its frequency at a given position in the sequence. This type of representation is useful to pinpoint PBs patterns. This WebLogo-like representation is produced with the `--logo` argument.

Python Module

PBxplore is also a Python module that more advanced users can embed in their own Python script. Here is a Python 3 example that assigns PBs from the structure of the barstar ribonuclease inhibitor (Lubienski et al., 1994):

```
import urllib.request
import pbxplore as pbx

# Download the pdb file
urllib.request.urlretrieve('https://files.rcsb.org/view/1BTA.pdb', '1BTA.pdb')

# The function pbx.chain_from_files() reads a list of files
# and for each one returns the chain and its name.
for chain_name, chain in pbx.chains_from_files(['1BTA.pdb']):
    # Compute phi and psi angles
    dihedrals = chain.get_phi_psi_angles()
    # Assign PBss
    pb_seq = pbx.assign(dihedrals)
    print('PBs sequence for chain {}:\n{}'.format(chain_name, pb_seq))
```

The documentation contains complete and executable Jupyter notebooks explaining how to use the module. It goes from the PBs assignments to the visualization of the protein deformability using the analysis functions. This allows the user to quickly understand the architecture of the module.

RESULTS

This section aims at giving the reader a quick tour of PBxplore features on a real-life example. We will focus on the $\beta3$ subunit of the human platelet integrin αIIb$\beta3$ that plays a central role in hemostasis and thrombosis. The $\beta3$ subunit has also been reported in cases of alloimmune thrombocytopenia (Kaplan, 2006; Kaplan and Freedman, 2007). We studied this protein by MD simulations (for more details, see references (Jallu et al., 2012, 2013, 2014)).

The $\beta3$ integrin subunit structure (Poulain and de Brevern, 2012) comes from the structure of the integrin complex (PDB 3FCS (Zhu et al., 2008)). Final structure has 690 residues and was used for MD simulations. All files mentioned below are available in the `demo_paper` directory from the GitHub repository (https://github.com/pierrepo/PBxplore/tree/master/demo_paper).

Protein Blocks assignment

The initial file `beta3.pdb` contains 225 structures issued from a single 50 ns MD simulation of the $\beta3$ integrin.

```
PBassign -p beta3.pdb -o beta3
```

This instruction generates the file `beta3.PB.fasta`. It contains as many PB sequences as there are structures in the input `beta3.pdb` file.

Protein Blocks assignment is the slowest step. In this example, it took roughly 80 seconds on a laptop with a quad-core-1.6-GHz processor.

Protein Blocks frequency

```
PBcount -f beta3.PB.fasta -o beta3
```

The above command line produces the file `beta3.PB.count` that contains a 2D-matrix with 16 columns (as many as different PBs) and 690 rows (one per residue) plus one supplementary column for residue number and one supplementary row for PBs labels.

Statistical analysis

Distribution of PBs

```
PBstat -f beta3.PB.count -o beta3 --map
```

Figure 3 shows the distribution of PBs for the $\beta3$ integrin. The color scale ranges from blue (the PB is not found at this position) to red (the PB is always found at this position). The $\beta3$ protein counts 690 residues. This leads to a cluttered figure and prevents getting any details on a specific residue (Fig. 3a). However, it exhibits some interesting patterns colored in red that correspond to series of neighboring residues exhibiting a fixed PB during the entire MD simulation. See for instance patterns associated to PBs *d* and *m* that reveal β-sheets and α-helices secondary structures (de Brevern, 2005).

With a large protein such as this one, it is better to look at limited segments. A focus on the PSI domain (residue 1 to 56) (Jallu et al., 2012; Zhu et al., 2008) of the $\beta3$ integrin was achieved with the command:

```
PBstat -f beta3.PB.count -o beta3 --map --residue-min 1 --residue-max 56
```

Figure 3b shows the PSI domain dynamics in terms of PBs. Interestingly, residue 33 is the site of the human platelet antigen (HPA)-1 alloimmune system. It is the first cause of alloimmune thrombocytopenia in Caucasian populations and a risk factor for thrombosis (Kaplan, 2006; Kaplan and Freedman, 2007). In Fig. 3b, this residue occupies a stable conformation with PB *h*. Residues 33 to 35 define a stable core composed of PBs *h*-*i*-*a*. This core is found in all of the 255 conformations extracted from the MD simulation and then is considered as highly rigid. On the opposite, residue 52 is flexible as it is found associated to PBs *i*, *j*, *k* and *l* corresponding to coil and α-helix conformations.

Equivalent number of PBs

The N_{eq} is a statistical measurement similar to entropy and is related to the flexibility of a given residue. The higher is the value, the more flexible is the backbone. The N_{eq} for the PSI domain (residue 1 to 56) was obtained from the command line:

```
PBstat -f beta3.PB.count -o beta3 --neq --residue-min 1 --residue-max 56
```

The output file `beta3.PB.Neq.1-56` contains two columns, corresponding to the residue numbers and the N_{eq} values. Figure 4a represents the N_{eq} along with the PBs sequence of the PSI domain, as generated by `PBstat`. The rigid region 33-35 and the flexible residue 52 are easily spotted, with low N_{eq} values for the former and a high N_{eq} value for the latter.

An interesting point, seen in our previous studies, is that the region delimited by residues 33 to 35 was shown to be highly mobile by the RMSF analysis we performed in Jallu et al. (2012) (for more details, see Materials and Methods section in Jallu et al. (2012)). For comparison, RMSF and N_{eq} are represented on the same graph on Fig. 4b. This high mobility was correlated with the location of this region in a loop, which globally moved a lot in our MD simulations. Here, we observe that the region 33-35 is rigid. The high values of RMSF we observed in our previous work were due to flexible residues in the vicinity of the region 33-35, probably acting as hinges (residues 32 and 36–37). Understanding the flexibility of residues 33 to 35 is important since this region defines the HPA-1 alloantigenic system involved in severe cases of alloimmune thrombocytopenia. PBxplore allows discriminating between flexible and rigid residues. The N_{eq} is a metric of deformability and flexibility whereas RMSF quantifies mobility.

Logo representation of PBs frequency

While the N_{eq} analysis focuses on the flexibility of amino acids, the WebLogo-like representation (Crooks et al., 2004) aims at identifying the diversity of PBs and their frequencies at a given position in the protein sequence. With a focus on the PSI domain, the following command line was used:

```
PBstat -f beta3.PB.count -o beta3 --logo --residue-min 1 --residue-max 56
```

Figure 5 represents PBs found at a given position. The rigid region 33-35 is composed of a succession of PBs *h*-*i*-*a* while the flexible residue 52 is associated to PBs *i*, *j*, *k* and *l*. This third representation summarized pertinent information, as shown in Jallu et al. (2013).

CONCLUSION

From our previous works (Jallu et al., 2012, 2013, 2014; Chevrier et al., 2013), we have seen the usefulness of a tool dedicated to the analysis of local protein structures and deformability with PBs. We also showed the relevance of studying molecular deformability in the scope of structures issued from MD simulations. In a very recent study (Goguet et al, 2017), long independent MD simulations were performed for seven variants and one reference structure of the Calf-1 domain of the αIIb human integrin. Simulations were analyzed with PBxplore. Common and flexible regions as well as deformable zones were observed in all the structures. The highest B-factor region of Calf-1, usually considered as most flexible, is in fact a rather rigid region encompassed into two deformable zones. Each mutated structure barely showed any modifications at the mutation sites while distant conformational changes were detected by PBxplore. These results question the relationship between MD simulations and allostery and the role of long range effects on protein structure. In this context, we propose PBxplore, freely available at https://github.com/pierrepo/PBxplore. It is written in a modular fashion that allows embedding in any PBs related Python application.

Software Availability

PBxplore is released under the open-source MIT license (Open Source Initiative, 2014). Its source code can be freely downloaded from the GitHub repository of the project: https://github.com/pierrepo/PBxplore. In addition, the present version of PBxplore (1.3.7) is also archived in the digital repository Zenodo (Barnoud et al., 2017).

REFERENCES

Ascher, D., Dubois, P. F., Hinsen, K., James, J. H., and Oliphant, T. (1999). Numerical Python. Technical report, Lawrence Livermore National Laboratory, Livermore, CA.

Atilgan, A. R., Turgut, D., and Atilgan, C. (2007). Screened Nonbonded Interactions in Native Proteins Manipulate Optimal Paths for Robust Residue Communication. *Biophysical Journal*, 92(9):3052–3062.

Barnoud, J., Santuz, H., de Brevern, A. G., and Poulain, P. (2017). PBxplore (v1.3.7): A program to explore protein structures with Protein Blocks. *Zenodo*.

Bassi, S. (2007). A primer on python for life science researchers. *PLoS Comput. Biol.*, 3(11):e199.

Bernstein, F. C., Koetzle, T. F., Williams, G. J., Meyer, E. F., Brice, M. D., Rodgers, J. R., Kennard, O., Shimanouchi, T., and Tasumi, M. (1977). The Protein Data Bank: A computer-based archival file for macromolecular structures. *J.Mol. Biol.*, 112(3):535–542.

Bornot, A., Etchebest, C., and de Brevern, A. G. (2011). Predicting protein flexibility through the prediction of local structures. *Proteins*, 79(3):839–852.

Bourne, P. E., Berman, H. M., McMahon, B., Watenpaugh, K. D., Westbrook, J. D., and Fitzgerald, P. M. (1997). [30] Macromolecular crystallographic information file. In *Methods in Enzymology*, volume 277, pages 571–590. Elsevier.

Brewer, C., Harrower, M., Sheesley, B., Woodruff, A., and Heyman, D. (2013). ColorBrewer2.

Bu, Z. and Callaway, D. J. (2011). Proteins MOVE! Protein dynamics and long-range allostery in cell signaling. In *Advances in Protein Chemistry and Structural Biology*, volume 83, pages 163–221. Elsevier.

Chevrier, L., de Brevern, A., Hernandez, E., Leprince, J., Vaudry, H., Guedj, A. M., and de Roux, N. (2013). PRR Repeats in the Intracellular Domain of KISS1R Are Important for Its Export to Cell Membrane. *Molecular Endocrinology*, 27(6):1004–1014.

Craveur, P., Joseph, A. P., Esque, J., Narwani, T. J., Noel, F., Shinada, N., Goguet, M., Leonard, S., Poulain, P., Bertrand, O., Faure, G., Rebehmed, J., Ghozlane, A., Swapna, L. S., Bhaskara, R. M., Barnoud, J., Téletchéa, S., Jallu, V., Cerny, J., Schneider, B., Etchebest, C., Srinivasan, N., Gelly, J.-C., and de Brevern, A. G. (2015). Protein flexibility in the light of structural alphabets. *Frontiers in Molecular Biosciences*, 2.

Crooks, G. E., Hon, G., Chandonia, J.-M., and Brenner, S. E. (2004). WebLogo: A Sequence Logo Generator. *Genome Research*, 14(6):1188–1190.

de Brevern, A., Wong, H., Tournamille, C., Colin, Y., Le Van Kim, C., and Etchebest, C. (2005). A structural model of a seven-transmembrane helix receptor: The Duffy antigen/receptor for chemokine (DARC). *Biochimica et Biophysica Acta (BBA) - General Subjects*, 1724(3):288–306.

de Brevern, A. G. (2005). New assessment of a structural alphabet. *In Silico Biology*, 5(3):283–289.

de Brevern, A. G., Bornot, A., Craveur, P., Etchebest, C., and Gelly, J.-C. (2012). PredyFlexy: Flexibility and local structure prediction from sequence. *Nucleic Acids Research*, 40(Web Server issue):W317–322.

de Brevern, A. G., Etchebest, C., and Hazout, S. (2000). Bayesian probabilistic approach for predicting backbone structures in terms of protein blocks. *Proteins*, 41(3):271–287.

DeLano, W. L. (2002). *The PyMOL Molecular Graphics System*, volume Version 1.5.0.4. Schrödinger, LLC. on World Wide Web http://www.pymol.org.

Dixit, A. and Verkhivker, G. M. (2011). Computational Modeling of Allosteric Communication Reveals Organizing Principles of Mutation-Induced Signaling in ABL and EGFR Kinases. *PLoS Computational Biology*, 7(10):e1002179.

Dong, Q.-w., Wang, X.-l., and Lin, L. (2007). Methods for optimizing the structure alphabet sequences of proteins. *Computers in Biology and Medicine*, 37(11):1610–1616.

Dudev, M. and Lim, C. (2007). Discovering structural motifs using a structural alphabet: Application to magnesium-binding sites. *BMC Bioinformatics*, 8(1):106.

Etchebest, C., Benros, C., Hazout, S., and de Brevern, A. G. (2005). A structural alphabet for local protein structures: Improved prediction methods. *Proteins: Structure, Function, and Bioinformatics*, 59(4):810–827.

Faure, G., Bornot, A., and de Brevern, A. G. (2008). Protein contacts, inter-residue interactions and side-chain modelling. *Biochimie*, 90(4):626–639.

Frauenfelder, H., Sligar, S., and Wolynes, P. (1991). The energy landscapes and motions of proteins. *Science*, 254(5038):1598–1603.

Gelly, J.-C., Joseph, A. P., Srinivasan, N., and de Brevern, A. G. (2011). iPBA: A tool for protein structure comparison using sequence alignment strategies. *Nucleic Acids Research*, 39(suppl):W18–W23.

Ghosh, A. and Vishveshwara, S. (2007). A study of communication pathways in methionyl- tRNA synthetase by molecular dynamics simulations and structure network analysis. *Proceedings of the National Academy of Sciences*, 104(40):15711–15716.

Ghouzam, Y., Postic, G., de Brevern, A. G., and Gelly, J.-C. (2015). Improving protein fold recognition with hybrid profiles combining sequence and structure evolution. *Bioinformatics*, page btv462.

Ghouzam, Y., Postic, G., Guerin, P.-E., de Brevern, A. G., and Gelly, J.-C. (2016). ORION: A web server for protein fold recognition and structure prediction using evolutionary hybrid profiles. *Scientific Reports*, 6(1).

GitHub (2007). GitHub. `https://github.com/`.

Goguet, M., Narwani, T. J., Peterman, R., Jallu, V., and de Brevern, A. G. (2017). In silico analysis of glanzmann variants of calf-1 domain of alphaiib/beta3 integrin revealed dynamic allosteric effect. *Scientific Reports*, 7(8001).

Gowers, R. J., Linke, M., Barnoud, J., Reddy, T. J. E., Melo, M. N., Seyler, S. L., Domaski, J., Dotson, D. L., Buchoux, S., Kenney, I. M., and Beckstein, O. (2016). MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. In Sebastian Benthall and Scott Rostrup, editors, *Proceedings of the 15th Python in Science Conference*, pages 98 – 105.

Holscher, E., Leifer, C., and Grace, B. (2010). Read the Docs.

Jallu, V., Bertrand, G., Bianchi, F., Chenet, C., Poulain, P., and Kaplan, C. (2013). The αIIb p.Leu841Met (Cab3a+) polymorphism results in a new human platelet alloantigen involved in neonatal alloimmune thrombocytopenia. *Transfusion*, 53(3):554–563.

Jallu, V., Poulain, P., Fuchs, P. F. J., Kaplan, C., and de Brevern, A. G. (2012). Modeling and molecular dynamics of HPA-1a and -1b polymorphisms: Effects on the structure of the $b3$ subunit of the αIIb$\beta 3$ integrin. *PloS One*, 7(11):e47304.

Jallu, V., Poulain, P., Fuchs, P. F. J., Kaplan, C., and de Brevern, A. G. (2014). Modeling and molecular dynamics simulations of the V33 variant of the integrin subunit $b3$: Structural comparison with the L33 (HPA-1a) and P33 (HPA-1b) variants. *Biochimie*, 105:84–90.

Joseph, A. P., Agarwal, G., Mahajan, S., Gelly, J.-C., Swapna, L. S., Offmann, B., Cadet, F., Bornot, A., Tyagi, M., Valadié, H., Schneider, B., Etchebest, C., Srinivasan, N., and de Brevern, A. G. (2010). A short survey on protein blocks. *Biophysical Reviews*, 2(3):137–147.

Joseph, A. P., Srinivasan, N., and de Brevern, A. G. (2011). Improvement of protein structure comparison using a structural alphabet. *Biochimie*, 93(9):1434–1445.

Joseph, A. P., Srinivasan, N., and de Brevern, A. G. (2012). Progressive structure-based alignment of homologous proteins: Adopting sequence comparison strategies. *Biochimie*, 94(9):2025–2034.

Kaplan, C. (2006). Neonatal alloimmune thrombocytopenia. In *Thrombocytopenia*, pages 223–244. McCrae KR, taylor & francis group edition.

Kaplan, C. and Freedman, J. (2007). Platelets. In *Platelets*, pages 971–984. Michelson AD, London: Academic Press.

Laine, E., Auclair, C., and Tchertanov, L. (2012). Allosteric Communication across the Native and Mutated KIT Receptor Tyrosine Kinase. *PLoS Computational Biology*, 8(8):e1002661.

Léonard, S., Joseph, A. P., Srinivasan, N., Gelly, J.-C., and de Brevern, A. G. (2014). mulPBA: An efficient multiple protein structure alignment method based on a structural alphabet. *Journal of Biomolecular Structure and Dynamics*, 32(4):661–668.

Li, Q., Zhou, C., and Liu, H. (2009). Fragment-based local statistical potentials derived by combining an alphabet of protein local structures with secondary structures and solvent accessibilities. *Proteins: Structure, Function, and Bioinformatics*, 74(4):820–836.

Lindahl, E., Hess, B., and van der Spoel, D. (2001). GROMACS 3.0: A package for molecular simulation and trajectory analysis. *Journal of Molecular Modeling*, 7(8):306–317.

Lubienski, M. J., Bycroft, M., Freund, S. M., and Fersht, A. R. (1994). Three-dimensional solution structure and 13C assignments of barstar using nuclear magnetic resonance spectroscopy. *Biochemistry*, 33(30):8866–8877.

Michaud-Agrawal, N., Denning, E. J., Woolf, T. B., and Beckstein, O. (2011). MDAnalysis: A toolkit for the analysis of molecular dynamics simulations. *Journal of Computational Chemistry*, 32(10):2319–2327.

Nguyen, L. A. T., Dang, X. T., Le, T. K. T., Saethang, T., Tran, V. A., Ngo, D. L., Gavrilov, S., Nguyen, N. G., Kubo, M., Yamada, Y., and Satou, K. (2014). Predicting *B*eta-Turns and *B*eta-Turn Types Using a Novel Over-Sampling Approach. *Journal of Biomedical Science and Engineering*, 07(11):927–940.

Offmann, B., Tyagi, M., and de Brevern, A. (2007). Local Protein Structures. *Current Bioinformatics*, 2(3):165–202.

Open Source Initiative (2014). The MIT License (MIT). Technical report.

Pandini, A., Fornili, A., Fraternali, F., and Kleinjung, J. (2013). GSATools: Analysis of allosteric communication and functional local motions using a structural alphabet. *Bioinformatics*, 29(16):2053–2055.

Poulain, P. and de Brevern, A. G. (2012). Model of the Beta3 Subunit of Integrin alphaIIb/beta3. `https://dx.doi.org/10.6084/m9.figshare.104602.v2`.

Rangwala, H., Kauffman, C., and Karypis, G. (2009). svmPRAT: SVM-based Protein Residue Annotation Toolkit. *BMC Bioinformatics*, 10(1):439.

Sevcík, J., Urbanikova, L., Dauter, Z., and Wilson, K. S. (1998). Recognition of RNase Sa by the inhibitor barstar: Structure of the complex at 1.7 A resolution. *Acta Crystallographica. Section D, Biological Crystallography*, 54(Pt 5):954–963.

Software, F. P. (2010). Python Language Reference, version 2.7. Technical report.

Suresh, V., Ganesan, K., and Parthasarathy, K. (2013). A Protein Block Based Fold Recognition Method for the Annotation of Twilight Zone Sequences. *Protein Pept Lett*, 20(3):249–254.

Suresh, V. and Parthasarathy, S. (2014). SVM-PB-Pred: SVM Based Protein Block Prediction Method Using Sequence Profiles and Secondary Structures. *Protein & Peptide Letters*, 21(8):736–742.

Thomas, A., Deshayes, S., Decaffmeyer, M., Van Eyck, M. H., Charloteaux, B., and Brasseur, R. (2006). Prediction of peptide structure: How far are we? *Proteins: Structure, Function, and Bioinformatics*, 65(4):889–897.

Travis CI (2015). Travis CI. `https://travis-ci.org/`.

van der Spoel, D., Lindahl, E., Hess, B., Groenhof, G., Mark, A. E., and Berendsen, H. J. C. (2005). GROMACS: Fast, flexible, and free. *J Comput Chem*, 26(16):1701–1718.

van Rossum, G. (1995). Python tutorial. Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam.

Zhu, J., Luo, B.-H., Xiao, T., Zhang, C., Nishida, N., and Springer, T. A. (2008). Structure of a Complete Integrin Ectodomain in a Physiologic Resting State and Activation and Deactivation by Applied Forces. *Molecular Cell*, 32(6):849–861.

Zimmermann, O. and Hansmann, U. H. E. (2008). LOCUSTRA: Accurate Prediction of Local Protein Structure Using a Two-Layer Support Vector Machine Approach. *Journal of Chemical Information and Modeling*, 48(9):1903–1908.

ACKNOWLEDGEMENTS

This work was supported by grants from National Institute for Blood Transfusion (INTS, France) and Lab of Excellence GR-Ex to JB, PC, SL, APJ, VJ, AGdB and PP, from the Ministry of Research (France), University Paris Diderot, Sorbonne Paris Cité (France), National Institute for Health and Medical Research (INSERM, France) to JB, PC, SL, APJ, AGdB and PP. The labex GR-Ex, reference ANR-11-LABX-0051 is funded by the program ”Investissements d'avenir” of the French National Research Agency, reference ANR-11-IDEX-0005-02. AGdB acknowledges to Indo-French Centre for the Promotion of Advanced Research / CEFIPRA for collaborative grant (number 5302-2). JB was supported by the TOP program of Prof. Marrink, financed by the Netherlands Organisation for Scientific Research (NWO).

AUTHOR CONTRIBUTIONS

PP and AGdB conceived the project. PP, JB and HS wrote the software. AGdB, PC, APJ and VJ improved and tested the software. All authors reviewed the manuscript.

COMPETING INTERESTS

The authors declare that they have no competing interests.

FIGURE LEGENDS

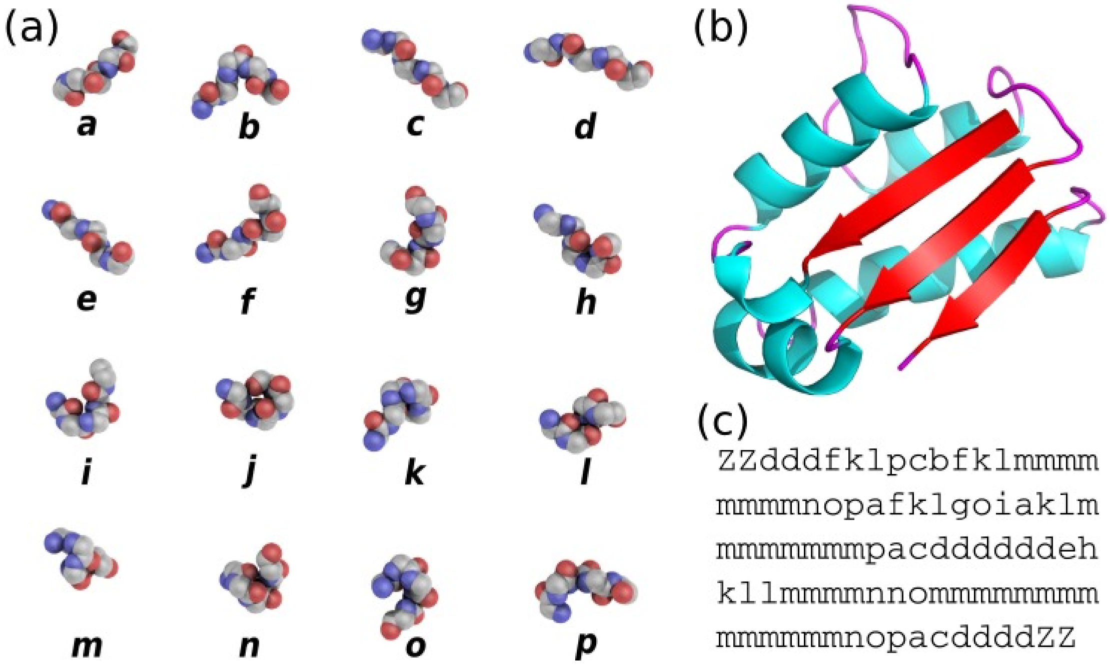

Figure 1. (a) The 16 protein blocks (PBs) represented in balls with carbon atoms in gray, oxygen atoms in red and nitrogen atoms in purple (hydrogen atoms are not represented). (b) The barstar protein (PDB ID 1AY7 (Sevcík et al., 1998)) represented in cartoon with alpha-helices in blue, beta-strands in red and coil in pink. These representations were generated using PyMOL software (DeLano, 2002) (c) PBs sequence obtained from PBs assignment. Z is a dummy PB meaning that no PB can be assigned to this position.

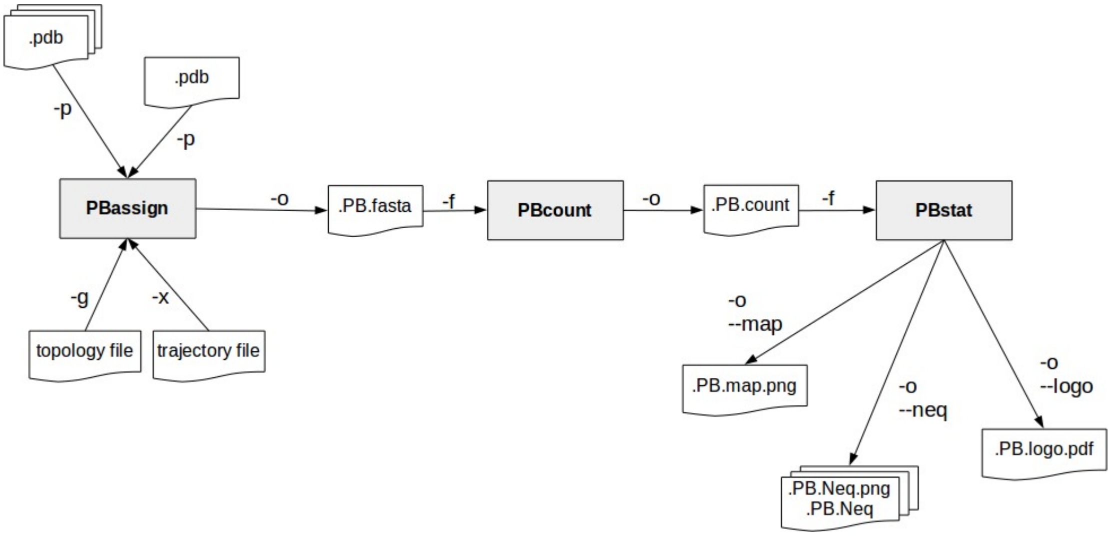

Figure 2. PBxplore is based on 3 programs that can be chained to build a structure analysis pipeline. Main input file types (.pdb, MD trajectory, MD topology), output files (.fasta, .png, .Neq, .pdf) and parameters (beginning with a single or double dash) are indicated.

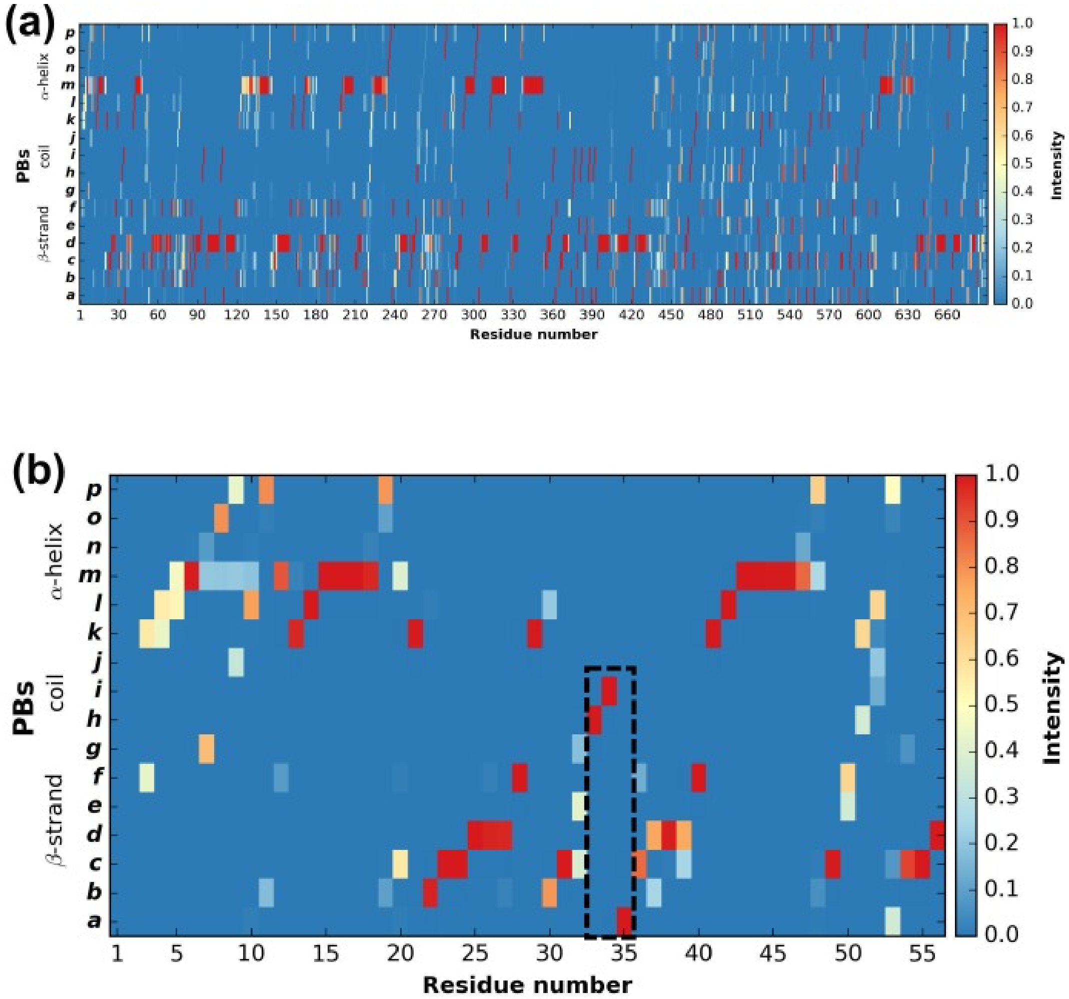

Figure 3. Distribution of PBs for the $\beta3$ integrin along the protein sequence. On the x-axis are found the 690 position residues and on the y-axis the 16 consecutive PBs from a to p (the two first and two last positions associated to ”Z” have no assignment). (a) For the entire protein. (b) For the PSI domain only (residues 1 to 56). The dashed zone pinpoints residue 33 to 35.

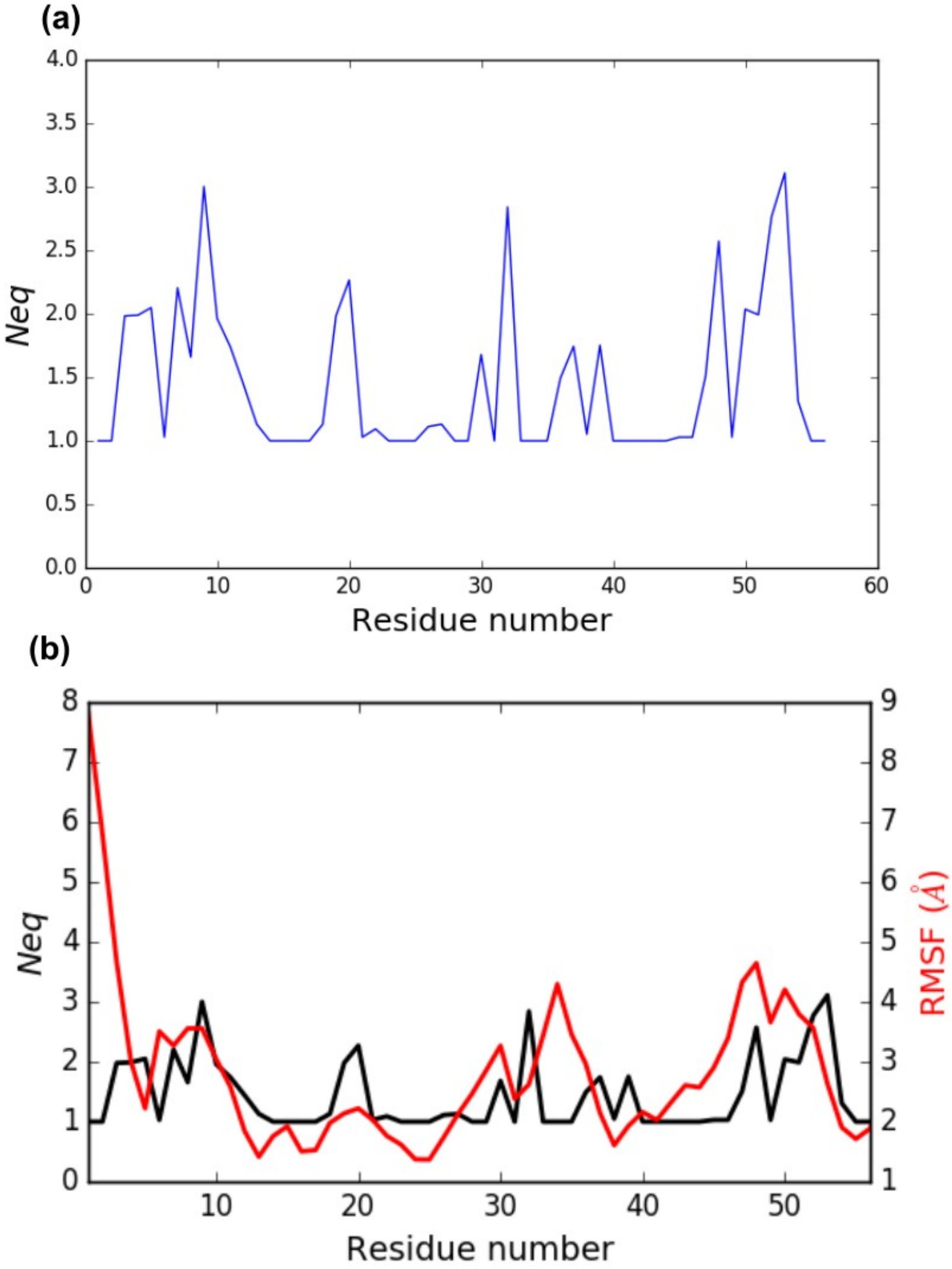

Figure 4. (a) N_{eq} versus residue number for the PSI domain (residues 1 to 56). (b) Comparison between RMSF and N_{eq}.

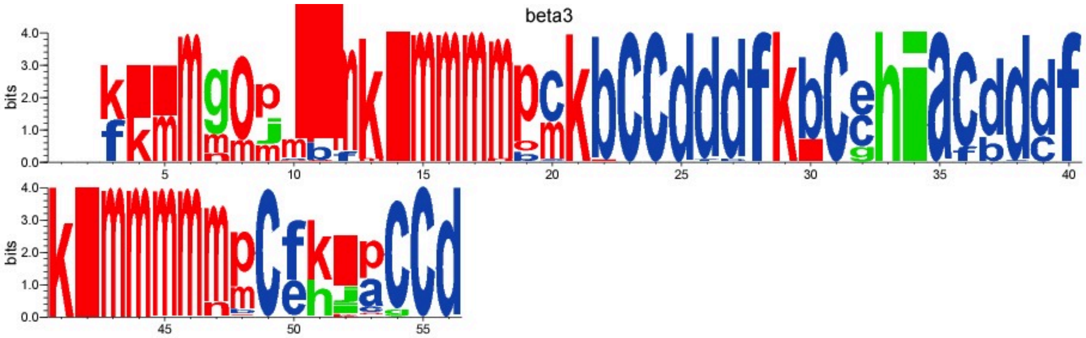

Figure 5. WebLogo-like representation of PBs for the PSI domain of the $\beta 3$ integrin. PBs in red roughly correspond to α-helices, PBs in blue to β-sheets and PBs in green to coil.