



**University
of Cyprus**

DEPARTMENT OF BIOLOGICAL SCIENCES

Computational approaches for the identification
of LIR-motifs in selective autophagy receptor
and adaptor proteins

IOANNA KALVARI

**A Dissertation Submitted to the University of Cyprus in Partial
Fulfillment of the Requirements for the Degree of Doctor of Philosophy**

December 2018

IOANNA KALVARI

VALIDATION PAGE

Doctoral Candidate: Ioanna Kalvari

Doctoral Thesis Title: Computational approaches for the identification of LIR-motifs in selective autophagy receptor and adaptor proteins

*The present Doctoral Dissertation was submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy at the **Department of Biological Sciences** and was approved on the 11th of December by the members of the **Examination Committee**.*

Examination Committee:

Research Supervisor: Dr. Vasilis I. Promponas, Assistant Professor _____
(Name, position and signature)

Committee Member: Dr. Paris Skourides, Associate Professor _____
(Name, position and signature)

Committee Member: Dr. Pantelis Georgiades, Associate Professor _____
(Name, position and signature)

Committee Member: Dr. Miguel Andrade, Professor _____
(Name, position and signature)

Committee Member: Dr. Costas Bouyioukos, Assistant Professor _____
(Name, position and signature)

DECLARATION OF DOCTORAL CANDIDATE

The present doctoral dissertation was submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy of the University of Cyprus. It is a product of original work of my own, unless otherwise mentioned through references, notes, or any other statements.

.....Full Name of Doctoral Candidate

.....Signature

IOANNA KALVARI

ΠΕΡΙΛΗΨΗ

Η μακροαυτοφαγία (ή αυτοφαγία) αποτελεί εξελικτικά συντηρημένο ευκαρυωτικό καταβολικό μηχανισμό κυτταρικής ομοιόστασης. Υπό συνθήκες stress δημιουργούνται αυτοφαγοσώματα, διπλομεμβρανικά κυστίδια απομόνωσης κυτταροπλασματικού υλικού που οδηγείται σε αποικοδόμηση στα λυσοσώματα/κενοτόπια “ανακυκλώνοντας” δομικά στοιχεία του κυττάρου. Η αυτοφαγία συχνά εκτελείται επιλεκτικά, υποβοηθούμενη από πρωτεΐνες-υποδοχείς που προσδέουν τα φορτία με εξειδικευμένες αλληλεπιδράσεις. Ταυτόχρονα, μέσω ενός βραχέως γραμμικού μοτίβου (LIR-motif) προσδέουν πρωτεΐνες της οικογένειας Atg8, που βρίσκονται ομοιοπολικά συνδεδεμένες στη μεμβράνη του αυτοφαγοσώματος. Επιπλέον, πρωτεΐνες-προσαρμογείς αλληλεπιδρούν με τις Atg8 με LIR-motifs για την επιτέλεση άλλων αυτοφαγικών λειτουργιών. Κατά την περίοδο έναρξης αυτής της διδακτορικής διατριβής είχαν χαρακτηριστεί 25 περίπου πρωτεΐνες-υποδοχείς/προσαρμογείς της αυτοφαγίας και τα LIR-motifs τους.

Εξετάσαμε τη δυνατότητα ανάπτυξης υπολογιστικών μεθόδων/εργαλείων χαρακτηρισμού LIR-motifs, στοχεύοντας στη διεύρυνση της γνώσης σχετικά με τις πρωτεΐνες υποδοχείς/προσαρμογείς. Συγκεκριμένα, έχοντας υπόψη προηγούμενες προσπάθειες περιγραφής των LIR-motifs, προτείναμε μια γενικευμένη κανονική έκφραση (xLIR) στοχεύοντας στην απόλυτη ευαισθησία. Αναμενόμενα, η προσέγγιση αυτή οδηγεί σε ανίχνευση πλήθους μοτίβων χωρίς βιολογική σημασία. Προκειμένου να μειώσουμε τον αριθμό τους, διατηρώντας ταυτόχρονα υψηλή ευαισθησία ανίχνευσης των βιολογικά σημαντικών μοτίβων, αξιολογήσαμε συστηματικά πληθώρα συμπληρωματικών χαρακτηριστικών. Παρατηρώντας ότι (α) οι πρωτεΐνες της αυτοφαγίας τείνουν να περιέχουν εγγενώς μη δομημένες περιοχές (IDRs), και (β) βραχέα μοτίβα πρόσδεσης συχνά βρίσκονται σε IDRs, αρχικά επιβεβαιώσαμε ότι ισχύουν στο σύνολο αναφοράς και τις εφαρμόσαμε ως φίλτρο, βελτιώνοντας σημαντικά την ειδικότητα. Δείξαμε επίσης ότι η πιθανοθεωρητική αναπαράσταση των βιολογικά λειτουργικών LIR-motifs μέσω PSSMs αυξάνει περισσότερο την ειδικότητα, οδηγώντας σε προβλέψεις υψηλότερης ακρίβειας. Βασιζόμενοι στα παραπάνω, αναπτύξαμε την πρώτη σχετική μέθοδο στη βιβλιογραφία, η οποία διατίθεται ελεύθερα για χρήση στην ερευνητική κοινότητα (διαδικτυακή εφαρμογή iLIR).

Στοχεύοντας να κατανοήσουμε σε βάθος τις σχέσεις της αμινοξικής ακολουθίας και των δομικών χαρακτηριστικών των πρωτεϊνών με λειτουργικά LIR-motifs και να βελτιώσουμε περαιτέρω την απόδοση της iLIR: (α) Μελετήσαμε συστηματικά διάφορες πηγές δεδομένων που αφορούν IDRs. Προτείνουμε πολύ-κριτηριακές προβλέψεις, που μπορούν να

χρησιμοποιηθούν σε διαφορετικές εφαρμογές, στοχεύοντας σε υψηλότερη ειδικότητα ή ευαισθησία. (β) Πραγματοποιήσαμε συστηματική συλλογή πειραματικά προσδιορισμένων τρισδιάστατων δομών πρωτεϊνών της οικογένειας Atg8 και LIR-motifs. Μετά από προεπεξεργασία των δεδομένων για τον καθορισμό των περιοχών δέσμευσης των LIR-motifs, εκτελέσαμε πειράματα αγκυροβόλησης πεπτιδίων στις δομές “στόχους”, καταδεικνύοντας ότι μπορούμε με επιτυχία να αναγνωρίζουμε περιπτώσεις ειδικότητας αλληλεπίδρασης των LIR-motifs με συγκεκριμένα ομόλογα της Atg8. Αναπτύξαμε μια εξειδικευμένη βάση δεδομένων για την καταχώρηση και περαιτέρω ανάλυση των αποτελεσμάτων, η οποία θα διατεθεί σύντομα προς χρήση.

Στο ταχύτατα αναπτυσσόμενο αυτό ερευνητικό πεδίο είναι παρακινδυνευμένο να κάνει κανείς επιτυχημένες προβλέψεις των εξελίξεων σε βάθος χρόνου. Το γεγονός ότι σήμερα (συχνά με τη βοήθεια μεθόδων που αναπτύχθηκαν σε αυτή τη διατριβή) έχει πολλαπλασιαστεί η γνώση μας για τους υποδοχείς/προσαρμογείς της επιλεκτικής μακροαυτοφαγίας δημιουργεί νέες προοπτικές. Η αύξηση των διαθέσιμων δεδομένων αναφοράς επιτρέπει την ανάπτυξη εξελιγμένων μεθόδων πρόβλεψης (π.χ. βασισμένων σε τεχνικές μηχανικής μάθησης) που, σε συνδυασμό με δεδομένα μεταγραφομικής, μπορούν να προσφέρουν νέα γνώση για τους μηχανισμούς ρύθμισης της επιλεκτικής αυτοφαγίας σε διαφορετικούς κυτταρικούς τύπους, ιστούς και αναπτυξιακά στάδια. Παράλληλα, ανακαλύψεις νέων μοριακών οντοτήτων που εμπλέκονται ενεργά στην επιλεκτική αυτοφαγία (π.χ. ncRNAs) αναμένεται να μας προσφέρουν “εκπλήξεις” αλλά και νέο υλικό και πεδίο δράσης για πειραματισμό, τόσο στο εργαστήριο όσο και *in silico*.

ABSTRACT

Macroautophagy (hereinafter autophagy) is a catabolic, cellular homeostasis mechanism conserved throughout the eukaryotes. Under stress conditions, double membraned vesicles (autophagosomes) isolate cytoplasmic material, eventually targeted to the lysosome/vacuole for degradation, thus recycling structural blocks for use by the cell. Selective modes of autophagy are facilitated by receptor proteins capable of binding specific cargos via cargo-specific interactions. These receptors bind to members of the Atg8 protein family (conjugated to the autophagosome membrane) via short linear motifs (LIR-motifs). Furthermore, protein adaptors interact with Atg8 proteins via LIR-motifs for performing other autophagic functions. At the initiation of this PhD project approximately 25 selective autophagy receptors/adaptors had been characterized along with their LIR-motifs.

We set to develop computational methods and tools for characterizing LIR-motifs, aiming to broaden our knowledge on selective autophagy receptors/adaptors. Based on previous attempts to describe LIR-motifs, we propose a generic regular expression (xLIR) aspiring to achieve absolute sensitivity. Expectedly, this approach leads to many false positive hits without any biological relevance. We systematically examined additional sequence-derived features to reduce false positives.

Knowing that:

- a) autophagy proteins are enriched in intrinsically disordered regions (IDRs), and
- b) short linear motifs are often found in IDRs

we confirm these observations in our reference autophagy receptor/adaptor dataset and, consequently, apply these principles as filters, leading to increased specificity. We also demonstrate that using a profile representation of LIR-motifs (in the form of a PSSM) further increases specificity, yielding high quality predictions. This work led to the first method of its kind reported in the literature, now freely available for use by the research community via the iLIR web server.

In our quest for deeper understanding the relationships between aminoacid sequences and the structural features of proteins with functional LIR-motifs (and to further improve iLIR efficiency):

- a) We systematically studied different data resources regarding IDRs, proposing multi-scheme predictions, each suited for different applications aiming at higher specificity/sensitivity.
- b) We compiled a comprehensive collection of experimentally determined 3D-structures of Atg8 proteins and LIR-motifs. Following data pre-processing for defining the LIR-motif interaction interfaces, we conduct peptide docking experiments, illustrating that this approach is useful for predicting LIR::Atg8 binding-specificity. We develop a specialized database for storing this information, facilitating downstream analyses, which we plan to make freely available for use.

It is difficult to make successful long-term predictions in a rapidly developing field like autophagy. The increasing number of selective autophagy receptors/adaptors discovered (often using the methods/tools developed for this thesis) opens exciting research perspectives. Access to substantially broader reference datasets enables (or, better, requires) development of more sophisticated methods/tools (e.g. based on machine-learning techniques) to successfully capture hidden dependencies between sequence-features and functional properties of LIR-motifs. Combined with the increasing availability of -omics data, we envisage cutting-edge research towards elucidating regulation of autophagy in different cell types, tissues and developmental stages. In addition, the discovery of novel molecular entities (e.g. ncRNAs) with active roles in autophagy guarantees further ‘surprises’ but also new data material and research directions for experimentation, *in vivo*, *in vitro* or *in silico*.

Acknowledgements

First and foremost, I would like to thank my supervisor Prof. Vasilis J. Promponas for giving me the opportunity to join his lab, which I personally consider a privilege. Prof. Promponas is an incredibly brilliant and talented scientist with a broad and very impressive spectrum of knowledge. I must admit I have always admired his unbelievable enthusiasm and impeccable competence with which he takes in new science and technologies coming his way. Information that he skilfully manifests into many original and fascinating project ideas, like the inception of my PhD project.

Apart from a great scientist my deep appreciation towards him stems from his high qualities as a person. Most importantly his tremendous patience, continuous support, his dedication and selfless attitude. The last couple of months he has been working along my side very long and late hours making sure that I bring this project to completion. For that and many more I will be sincerely and wholeheartedly forever grateful.

I would also like to thank the members of my examining committee Prof. Miguel Andrade from Johannes Gutenberg University Mainz, Prof. Costas Bouyioukos from Paris Diderot University and our two internal examiners from the University of Cyprus Prof. Paris Skourides and Prof. Pantelis Georgiades. Especially for being kind and considerate enough to accept my PhD thesis for review very close to my defence date.

A big thank you should also go to the Department of Computer Science for the provision of computational resources used for running big portion of the docking experiments. Special thanks go to Prof. George Pallis for granting me access to the resources of the Department of Computer Sciences, and Mr. Thanasis Tryfonos in particular, for providing technical support.

For the last three years I have been part of an incredible team at the European Bioinformatics Institute (EMBL-EBI) under the supervision of Dr. Rob D. Finn and Dr. Anton I. Petrov. I would like to express my deep appreciation and respect for all their patience and continuous support. I would not have made it without all the flexible working arrangements they agreed to the last month, which allowed me to bring my PhD thesis to completion. For that and all the opportunities I have been given so far, I am truly grateful.

Last but not least I would like to thank my family and my friends for standing by my side throughout this long journey, making every step of the way lighter and more fun. Pursuing a Doctorate of Philosophy is not an easy task and as an athlete I personally believe that almost 90% of the credit should go to the people who work behind the scenes. I am feeling very fortunate being part of a larger pack composed of amazing people. You are my heroes.

IOANNA KALVARI

Table of Contents

Table of Contents

1	INTRODUCTION	1
	AUTOPHAGY	1
1.1	1
1.2	SELECTIVE MACROAUTOPHAGY	2
1.3	INTRINSIC DISORDER AND LIR-MOTIFS.....	5
1.4	HYPOTHESIS AND OBJECTIVES.....	6
2	COMPUTATIONAL STEPS TOWARDS THE CHARACTERIZATION AND IDENTIFICATION OF LIR-MOTIFS	9
2.1	PREFACE.....	9
2.1.1	<i>The Atg8 protein family</i>	9
2.1.2	<i>Selective autophagy receptor and adaptor proteins</i>	10
2.2	DATA AND METHODS	12
2.2.1	<i>Data</i>	12
2.2.2	<i>Methods</i>	13
2.3	RESULTS.....	19
2.3.1	<i>Combining the predictive power of xLIR and Anchors</i>	23
2.3.2	<i>Using profile-based methods to identify functional LIR-motifs</i>	25
2.3.3	<i>Validating xLIR, anchors and PSSM with independent datasets</i>	27
2.3.4	<i>Assembling everything into a unified resource: the iLIR webserver</i>	29
2.4	CONCLUSIONS	34
3	INTRINSIC DISORDER AS A MEANS FOR THE IDENTIFICATION OF GENUINE LIR-MOTIFS	37
3.1	PREFACE.....	37
3.1.1	<i>Intrinsically disordered proteins</i>	37
3.2	DATA AND METHODS	37
3.2.1	<i>Data</i>	37
3.2.2	<i>Methods</i>	39
3.4	RESULTS.....	46
3.4.1	<i>In seek of the optimal predictive method and disorder threshold</i>	48
3.4.2	<i>Assessing the power of MobiDB over IUPRED2A and SPOT-disorder</i>	52
3.4.3	<i>Scrutinizing the potential of disorder binding regions in the determination of genuine LIRs</i> .	54
3.4.4	<i>Assessing the efficacy of multi-scheme predictors</i>	56
3.4.5	<i>Independent validation</i>	64
3.4.6	<i>A comparison to existing tools</i>	69
3.5	CONCLUSIONS	73
4	ILIR3D: DELVING INTO SELECTIVE AUTOPHAGY STRUCTURAL DATA	75
4.1	PREFACE.....	75
4.2	DATA AND METHODS	76
4.2.1	<i>Data</i>	76
4.2.2	<i>Methods</i>	80
4.3	RESULTS.....	88
4.3.1	<i>The iLIR3D MySQL database</i>	88
4.3.2	<i>Learning from template structures</i>	92
4.3.3	<i>A comprehensive set of experiments</i>	98
4.3.4	<i>Availability</i>	99
4.4	CONCLUSION.....	99
5	DISCUSSION AND FUTURE GOALS	102
6	REFERENCES.....	106
7	SUPPLEMENT	117

7.1	DIZSCAN.PY CODE	124
7.2	CONSENSUS_DISORDER_CALCULATOR.PY CODE.....	151
7.3	ANCHOR2_SCANNER.PY CODE	158
7.4	SPOT_SCANNER.PY CODE.....	170

IOANNA KALVARI

Table of Figures

FIGURE 1. THE THREE DIFFERENT TYPES OF AUTOPHAGY IN MAMMALS	1
FIGURE 2. THE MULTIFACETED VIEW OF AUTOPHAGY.	4
FIGURE 3. THE “EVOLUTION” OF THE NOTION OF A LIR/AIM/GIM MOTIF.	5
FIGURE 4. THREE OF THE RECEPTORS BELONGING TO THE ATG8 FAMILY IN COMPLEX WITH A LIR-MOTIF PEPTIDE.....	10
FIGURE 5. THE xLIR-PSSM.....	17
FIGURE 6. GRAPHICAL REPRESENTATION OF THE xLIR-PSSM	18
FIGURE 7. PSSM SCORE DISTRIBUTIONS FOR DIFFERENT CLASSES OF HEXAPEPTIDES.	26
FIGURE 8. HOME PAGE OF THE ILIR WEBSERVER.	30
FIGURE 9. ILIR SERVER USER INTERFACE.	31
FIGURE 10. ILIR RESULTS PAGE.	33
FIGURE 11. THE FULL COLLECTION OF PRE-RAN EXAMPLES AS THEY APPEAR ON THE ILIR WEBSITE.	34
FIGURE 12. CALCULATION OF DISORDER IN CALRETICULIN LIR-MOTIF DDWDFL AT POSITIONS 198-203.	40
FIGURE 13. THE FLOWCHART OF THE <i>DIZSCAN</i> ALGORITHM.	41
FIGURE 14. CONSTRUCTION OF THE CONSENSUS DISORDER STRING (cdSTR) OF THE LIR-MOTIF OF OPTINEURIN.....	43
FIGURE 15. BALANCED ACCURACY (%) ACHIEVED WITH MULTI-SCHEME PREDICTOR xLIR+A2 D PX CAPTURED AT VARIOUS PSSM THRESHOLDS.....	59
FIGURE 16. BALANCED ACCURACY (%) ACHIEVED WITH MULTI-SCHEME PREDICTOR xLIR+A2 D PX CAPTURED AT VARIOUS PSSM THRESHOLDS.....	60
FIGURE 17. MULTI-SCHEME METHOD COMPARISON.	62
FIGURE 18. ILIR RESULTS FOR HUMAN PLECKSTRIN HOMOLOGY DOMAIN-CONTAINING FAMILY M MEMBER	67
FIGURE 19. THE DOCKING GRIDS OF THE ATG8 FAMILY.	82
FIGURE 20. PROTEIN-PROTEIN DOCKING EXAMPLE.	85
FIGURE 21. MYSQL QUERY THAT RETRIEVES THE TEMPLATE STRUCTURE RESULTS PRESENTED IN TABLE 20.....	89
FIGURE 22. ILIR3D RELATIONAL MODEL CREATED USING MYSQL WORKBENCH BY APPLICATION OF REVERSE ENGINEERING. ..	90
FIGURE 23. QUERY EXAMPLES FOR THE COMPUTATION OF THE TP, TN, FP, FN VALUES FOR DISORDER PREDICTIONS.	92
FIGURE 24. MYSQL QUERY SNIPPET FOR THE RETRIEVAL OF FYCO1/ATG8 DOCKING SCORES.....	96
FIGURE 25. BOXPLOT REPRESENTATION OF THE DISTRIBUTIONS OF SCORES OF FYCO1 PEPTIDES DOCKED AGAINST THE ATG8 FAMILY.	98

Table of Tables

TABLE 1. SELECTIVE AUTOPHAGY RECEPTOR AND ADAPTOR PROTEINS WITH EXPERIMENTALLY DETERMINED LIR-MOTIFS.	12
TABLE 2. SELECTIVE AUTOPHAGY RECEPTOR/ADAPTOR PROTEINS WITH EXPERIMENTALLY VERIFIED LIR-MOTIFS.	14
TABLE 3. SEQUENCES USED IN THIS STUDY.	22
TABLE 4. AMINO ACID RESIDUE BACKGROUND DISTRIBUTION.	24
TABLE 5. VALIDATION OF xLIR AND cLIR MOTIF-BASED PREDICTORS.	25
TABLE 6. VALIDATION OF THE PSSM METHOD AS A PREDICTOR OF LIR-MOTIFS.	27
TABLE 7. A COLLECTION OF 52 PROTEINS WITH THEIR EXPERIMENTALLY VALIDATED LIR-MOTIFS.....	48
TABLE 8. DISORDER RESULTS AS COMPUTED FROM MOBiDB 3.0.0 DATA.	50
TABLE 9. CLASSIFICATION OF LIR-MOTIFS USING DISORDER DATA FROM MOBiDB.	51
TABLE 10. COMPARISON OF IUPRED2A, SPOT AND MOBiDB.	53
TABLE 11. COMPARING THE EFFICACY OF ANCHOR AND ANCHOR2 ON DIFFERENT PREDICTIVE SCHEMES.	55
TABLE 12. MULTI-SCHEME PREDICTORS APPLIED ON THE 47 LIR-MOTIFS COLLECTED BY ALEMU ET AL.....	57
TABLE 13. MULTI-SCHEME PREDICTOR RESULTS ON THE COMPLETE DATASET.	63
TABLE 14. NEW PROTEINS AND THEIR CORRESPONDING VERIFIED LIR MOTIFS.....	65
TABLE 15. CLASSIFICATION OF NOVEL LIR-MOTIFS BASED ON 3 DIFFERENT PREDICTION SCHEMES	68
TABLE 16. hfAIM AIM PREDICTIONS ON THE PROTEIN COLLECTION OF ALEMU ET AL.	71
TABLE 17. ILIR AND hfAIM PREDICTIVE POWER ASSESSMENT.	72
TABLE 18. PROTEINS OF THE ATG8 FAMILY, HEREIN “RECEPTORS”, FOUND IN TEMPLATE STRUCTURES.....	77
TABLE 19. SHORT DESCRIPTIONS OF THE TABLES COMPOSING THE ILIR3D DATABASE.	91
TABLE 20. TOP SCORING CONFORMATIONS OF THE TEMPLATE STRUCTURES.....	94
TABLE 21. SELECTIVE AUTOPHAGY RECEPTOR AND ADAPTOR PROTEIN STRUCTURES.....	123

Abbreviations

AIM: Atg8-family Interacting Motif

Atg: Autophagy-related

cdSTR: Consensus Disorder String

CGI: Common Gateway Interface

CLI: Command Line Interface

cLIR: Canonical LIR-motif

dSTR: Disorder String

FN: False Negative

FP: False Positive

GABARAP: Gamma-AminoButyric Acid Receptor Associated Protein

GIM: GABARAP Interaction Motif

HTTP: Hypertext Transfer Protocol

IDP: Intrinsically Disordered Protein

IDR: Intrinsically Disordered Region

LDS: LIR Docking Site

LIR: LC3 Interacting Region

LIRCP: LIR Containing Protein

MAP1LC3/LC3: Microtubule-associated protein 1 light chain 3

MSA: Multiple Sequence Alignment

NMR: Nuclear Magnetic Resonance

PSSM: Position Specific Scoring Matrix

REGEX: Regular Expression

RMSD: Root Mean Square Deviation

SAR: Selective Autophagy Receptor

SLIM: Short Linear Motifs

SMART: Simple Modular Architecture Research Tool

SQL: Structured Query Language

SQSTM1: Sequestosome 1

TN: True Negative

TP: True Positive

URL: Uniform Resource Locator

xLIR: Extended LIR-motif

RBP: RNA-binding protein

Publications directly related to this thesis

Kalvari, Ioanna, Stelios Tsompanis, Nitha C. Mulakkal, Richard Osgood, Terje Johansen, Ioannis P. Nezis, and Vasilis J. Promponas. 2014. “iLIR: A Web Resource for Prediction of Atg8-Family Interacting Proteins.” *Autophagy* 10 (5): 913–25.

Kalvari Ioanna, Chatzichristofi Agathangelos and Promponas V.J., 2018 “Intrinsically disordered regions in selective macroautophagy receptors and adaptors”, in preparation

Kalvari Ioanna and Promponas V.J., 2018 “iLIR3D: a database to explore macroautophagy in 3D”, in preparation

Other publications

Kalvari, Ioanna, Joanna Argasinska, Natalia Quinones-Olvera, Eric P. Nawrocki, Elena Rivas, Sean R. Eddy, Alex Bateman, Robert D. Finn, and Anton I. Petrov. 2018. “Rfam 13.0: Shifting to a Genome-Centric Resource for Non-Coding RNA Families.” *Nucleic Acids Research* 46 (D1): D335–42.

Kalvari, Ioanna, Eric P. Nawrocki, Joanna Argasinska, Natalia Quinones-Olvera, Robert D. Finn, Alex Bateman, and Anton I. Petrov. 2018. “Non-Coding RNA Analysis Using the Rfam Database.” *Current Protocols in Bioinformatics / Editorial Board, Andreas D. Baxevanis ... [et Al.]* 62 (1): e51.

The RNACentral Consortium. 2017. “RNACentral: A Comprehensive Database of Non-Coding RNA Sequences.” *Nucleic Acids Research* 45 (D1): D128–34.

The RNACentral Constortium. 2018. “RNACentral: A Hub of Information for Non-Coding RNA Sequences.” *Nucleic Acids Research*, November.
<https://doi.org/10.1093/nar/gky1034>.

Kannas, C. C., K. G. Achilleos, Z. Antoniou, C. A. Nicolaou, C. S. Pattichis, I. Kalvari, I. Kirmitzoglou, and V. J. Promponas. 2012. “A Workflow System for Virtual Screening in Cancer Chemoprevention.” In 2012 IEEE 12th International Conference on Bioinformatics Bioengineering (BIBE), 439–46.

Kannas, Christos C., Ioanna Kalvari, George Lambrinidis, Christiana M. Neophytou, Christiana G. Savva, Ioannis Kirmizoglou, Zinonas Antoniou, et al. 2015. "LiSIs: An Online Scientific Workflow System for Virtual Screening." *Combinatorial Chemistry & High Throughput Screening* 18 (3): 281–95.

Antoniades, Athos, Ioanna Kalvari, Constantinos Pattichis, Neil Jones, Paul M. Matthews, Enrico Domenici, and Pierandrea Muglia. 2009. "Discovering Genetic Polymorphism Associated with Gene Expression Levels across the Whole Genome." *Conference Proceedings: ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference 2009*: 5466–69.

IOANNA KALVARI

1 Introduction

1.1 Autophagy

Autophagy is an essential conserved catabolic biological process through which the cell maintains energy homeostasis and protects itself against pathogens. This is achieved via the breakdown of cytosolic material at the lytic compartments of the cell, which is the vacuole in plants and fungi and the lysosome in higher eukaryotes. To put it in simple terms, one can see autophagy as the recycling machinery of the cell (Yang and Klionsky 2009; White et al. 2015).

There are three different types of autophagy known in mammals: microautophagy, chaperone mediated autophagy and macroautophagy. Microautophagy involves the invagination of the lysosomal membrane and at the same time engulfing cytosolic material, which will be broken down once completely secluded. Chaperone mediated autophagy, as the name denotes, is coordinated via heat shock cognate 70 proteins and their co-chaperones, tethering proteins to the lysosome via a KFERQ like motif (Wirawan et al. 2012).

Macroautophagy is more distinctive compared to the other two types of autophagy in the sense that it requires an intermediate double membrane vesicle called the autophagosome, to transport cytosolic material to the lysosome for degradation (Wirawan et al. 2012).

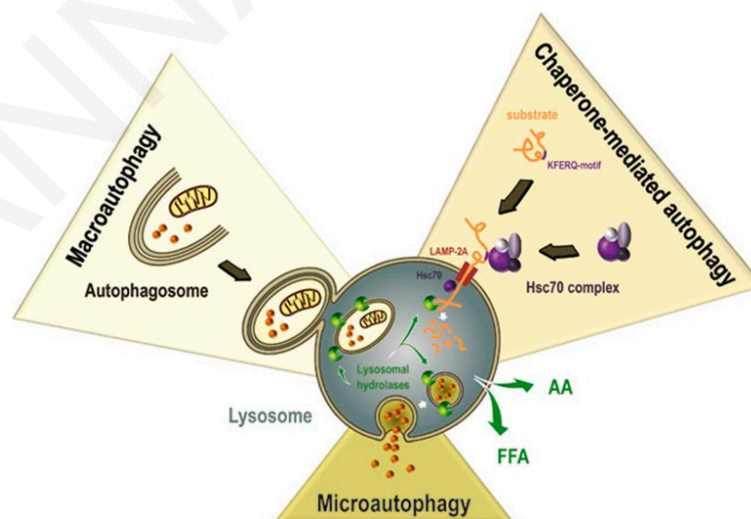


Figure 1. The three different types of autophagy in mammals

This figure was obtained from (Wirawan et al. 2012).

1.2 Selective macroautophagy

Macroautophagy (or simply autophagy) is known to be induced by stress or nutrient starvation leading to the degradation of cytosolic material to the lysosome/vacuole, resulting in the generation of “fresh” building blocks such as amino acids for protein synthesis (Onodera & Ohsumi 2005).

Autophagy was identified as a cellular response to nutrient starvation (Scott et al. 2004; Rubinsztein et al. 2011) and although it was initially considered to be a bulk process, where cytoplasmic material is recycled in an unselective manner, relevant work over the years showed that it can happen in a highly selective manner. A wide range of different cargo degraded by this process includes from single protein molecules or protein aggregates (Lamark & Johansen 2012; Lamark et al. 2017; Zaffagnini et al. 2018) to damaged organelles like mitochondria (mitophagy), endoplasmic reticulum (reticulophagy) and chloroplasts (chlorophagy – in plants) (Palikaras et al. 2018; Avin-Wittenberg & Fernie 2014), peroxisomes (pexophagy) (Marshall & Vierstra 2018) and even pathogens (xenophagy) (Knodler & Celli 2011).

Upon nutrient starvation, this biological mechanism starts with the generation of the double membrane organelle – the phagophore – near the endoplasmic reticulum, a process known as the nucleation. The phagophore then elongates, sequestering cytosolic material and, finally, closes forming a completely structured vesicle known as the autophagosome (closure). In a final step, the autophagosome travels to the lysosome (or the vacuole in plants and fungi) with which it fuses to form the autolysosome (Parzych & Klionsky 2014) (Zaffagnini & Martens 2016). Its constituents will then be degraded and recycled material is released back to the cytoplasm to be reused by the cell (Yang and Klionsky 2009; White et al. 2015).

Selective autophagy is orchestrated by specialised proteins called selective autophagy receptor (SARs) and adaptor proteins (Pankiv et al. 2007). SARs recognize and tether cargo from the cytoplasm to the phagophore and all the way to the lysosome in a selective manner (Stolz et al. 2014; Johansen & Lamark 2011; Rogov et al. 2017). Recruitment of cargo to the phagophore is enabled via interaction with proteins of the Atg8 family, located on the inner (receptors) and outer (adaptor) membranes of the phagophore (Rogov et al. 2014).

Atg8 proteins were first identified in yeast, where they also get their name from, and although there is only one gene encoded in *Saccharomyces cerevisiae*, more than one homologs are expressed in higher eukaryotes. For instance, there are 4 distinct Atg8 homologs expressed in the human genome (Shpilka et al. 2011):

1. Microtubule-associated proteins 1A/1B light chains 3A, 3B and 3C (MAP1LC3A, MAP1LC3B, MAP1LC3C)
2. Gamma-aminobutyric acid receptor-associated protein (GABARAP)
3. Gamma-aminobutyric acid receptor-associated protein-like 1 (GABARAPL1/GEC1)
4. Gamma-aminobutyric acid receptor-associated protein-like 2 (GABARAPL2/GATE-16)

The proteins of the Atg8 family and selective autophagy receptors and adaptors all together constitute the key players of the autophagic apparatus. Members of the Atg8 family are ubiquitin-like proteins (C-terminal), but do not share any similarities with Ubiquitins at sequence level and their N-terminal contains two consecutive α -helices, which is also what distinguishes them between ubiquitin proteins (Noda et al. 2010; Shpilka et al. 2011).

The interaction between Atg8 and selective autophagy receptor and adaptor proteins is facilitated through short linear motifs (SLIMs) often named as LIRs (Pankiv et al. 2007), AIMs (Noda et al. 2010), or GIMs (Rogov et al. 2014) based on their species of origin or their preference towards a certain type of Atg8 homolog. From this point onwards, we will collectively refer to these motifs as LIR-motifs, unless we specify otherwise.

LIR-motifs bind to the two conserved hydrophobic pockets of the Atg8 proteins – the W-site and L-site named after the amino acids firstly identified to interact with – by adopting an extended β -strand conformation, forming a parallel intermolecular β -sheet with that of the Atg8 proteins (Rogov et al. 2014).

Apart from autophagy's central role in the survival of the cell, this intracellular procedure is also known to be implicated in many biological pathways and mechanisms such as apoptosis (Mukhopadhyay et al. 2014), innate immunity (Boyle & Randow 2013), development e.g. embryogenesis (Mizushima & Levine 2010; Qu et al. 2007) and ageing (Mizushima 2007; Rubinsztein et al. 2011). In its defective form, autophagy can result in serious diseases from neurodegeneration (e.g. Alzheimer's (Uddin et al. 2018), Parkinson's (Lynch-Day et al.

2012; Wang et al. 2016)), retinitis pigmentosa (Moreno et al. 2018)), metabolic diseases (Rocchi & He 2015), diseases related with the heart (Martinet et al. 2007; Mei et al. 2015), liver (Ueno & Komatsu 2017) and lungs (Ryter & Choi 2015; Racanelli et al. 2018) as well as cancer (Amaravadi et al. 2016; Santana-Codina et al. 2017; Degenhardt et al. 2006) (Figure 2). With such a complex interplay of autophagy with other cellular processes and external stimuli a better understanding of this mechanism and its course of action seems to be crucial.

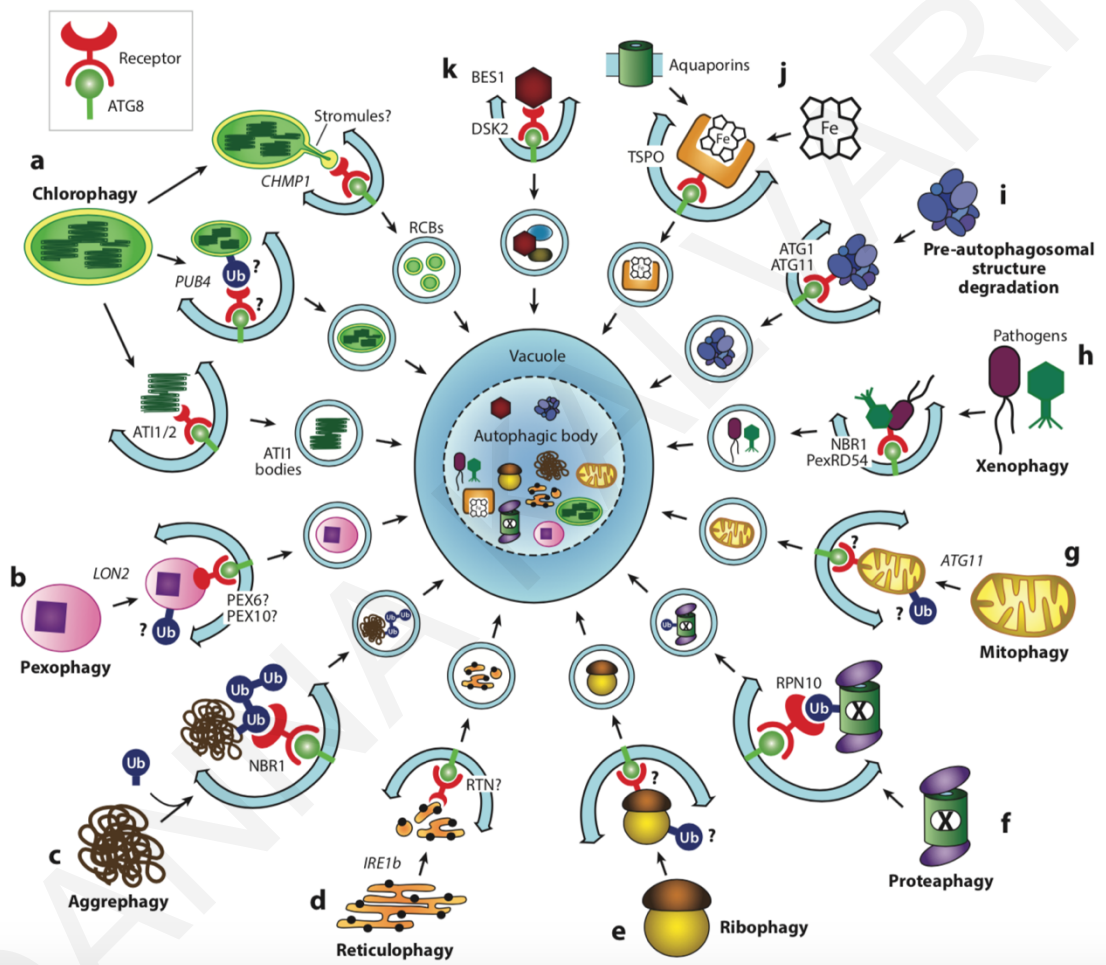


Figure 2. The multifaceted view of autophagy.

The figure was obtained from (Marshall and Vierstra 2018).

In the past decade a great curiosity around this biological mechanism emerged with studies concentrating on the characterization of LIR-motifs through sequence and structural analysis. The latter enabled scientists to define a short linear motif in the form of **WXXL**, where the amino acids tryptophan (W) and leucine (L) were proven to be significant for the interaction of the autophagic proteins with the two hydrophobic binding sites of the LC3 receptor (Birgisdottir et al. 2013; Noda et al. 2008). Later on, Noda et al. re-defined the linear motif by extending it to the form of $X_3X_2X_1[**WY**]X_1X_2[**LIV**]$, suggesting that acidic

residues at the leftmost end of the motif (positions -3 to -1) favoured interaction with the Atg8 receptor, also naming this short peptide AIM for **A**tg8 **I**nteracting **M**otif (Noda et al. 2010).

Another team, in their experimental work towards the identification of LIR-motifs of the ULK complex, Alemu and colleagues also made an effort to devise a consensus LIR-motif to further explore common aspects of these linear peptides. They collected **27** experimentally verified LIR-motifs from the literature. From a multiple sequence alignment composed using the sequences they gathered, the authors proposed the following regular expression: [DE][DEST][WFY][DELIV]X[ILV] (Alemu et al. 2012) and gave rise to a plethora of successive analogous studies. The timeline of the “evolution” of the notion of the AIM/LIR/GIM-motif as drawn by the traces of the pioneers in the field is shown in **Figure 3**.

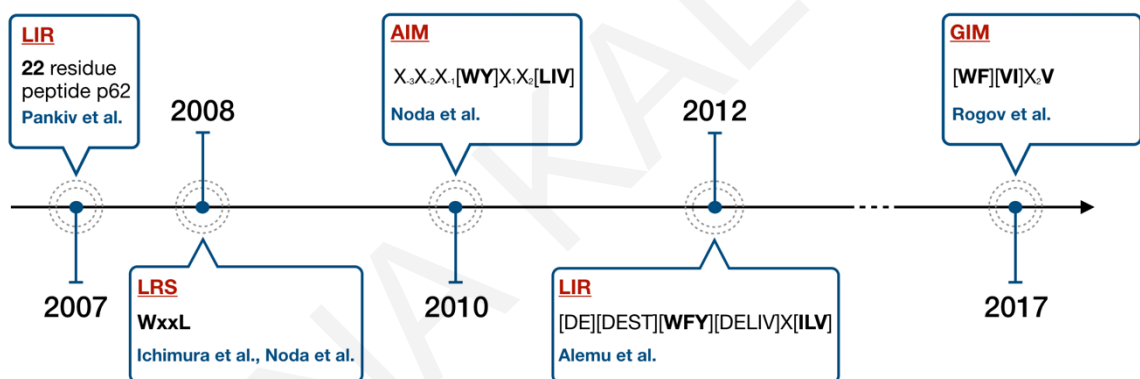


Figure 3. The “evolution” of the notion of a LIR/AIM/GIM motif.

1.3 Intrinsic disorder and LIR-motifs

Intrinsically disordered proteins (IDPs) are proteins which in their free state do not fold into a unique stable conformation (Wright & Dyson 1999). IDPs have been intensively studied during the last two decades and an increasing amount of knowledge continues to accumulate regarding to their possible functions (Wright & Dyson 2015; Dyson & Wright 2005; Oldfield & Dunker 2014; Darling & Uversky 2018). In several cases, a single protein may contain both globular (i.e. well-folded) and disordered (i.e. unstructured) domains, e.g. p53 (Derbyshire et al. 2002; Rowell et al. 2012; Suad et al. 2009).

In the majority of the currently documented cases, the conformation of the LIR-motif is extended when bound to the LIR docking site (LDS) of Atg8 homologs. An intriguing case

is the CLTC LIR-motif, which adopts an α -helical structure (Fotin et al. 2004). If we assume that during its interaction with the LDS a LIR-motif must have an extended conformation, then it would be possible that LIR-motifs may have the characteristics of so-called “chameleon sequences” (Mezei 1998) or “conformational switches” (Tsolis et al. 2013), that is, short sequences found to adopt more than one distinct secondary structure state. Such sequences have been long known to be important in protein aggregation and amyloid formation (Kelly 1996).

Additionally, it has been postulated that the function of LIR-motifs may be facilitated by short-range (with respect to the LIR-motif) conformational changes. Such structural rearrangements could bring this short linear motif in a suitable extended conformation in order to interact with the 2 well-conserved hydrophobic pockets on the surface of Atg8 homologs (Noda et al. 2008; Noda et al. 2010). Combined with the recent observation that autophagy-related proteins are relatively rich in intrinsically disordered regions (Mei et al. 2014), it is possible that the LIR-motifs may adopt the required conformation after switching from a disordered to an ordered state.

1.4 Hypothesis and objectives

Despite the central role of selective autophagy in cell physiology, at the beginning of this project only a few instances of selective autophagy receptors had been experimentally verified and reported in the literature. In addition, throughout the years several groups identified and refined the definition of LIR/AIM motifs. However, there was no systematic manner (e.g. dedicated computational tools) to look for instances of functionally relevant LIR motifs in amino acid sequences. Thus, a molecular biologist that wanted to investigate whether a protein sequence of interest had the potential to interact with an Atg8 protein, would have to manually check for an instance of the LIR motif or tweak existing software to perform this task. In addition, in the absence of automated tools, scanning of complete proteomes for the presence of LIR containing proteins (LIRCPs) was simply impossible.

We examined the efficacy of the consensus regular expression (cLIR) introduced in Alemu et al. (Alemu et al. 2012) and found it to be weak in discriminating LIR instances. The cLIR with a reported sensitivity of only 40.7% would only capture 11 out of the 27 verified LIR-motifs, a simple method which evidently required further improvement. Therefore, we hypothesised that a systematic study of experimentally known LIR-motifs and LIRCPs could facilitate the development of useful tools for the identification of functional LIR-motifs in

protein sequences. Moreover, accumulating structural evidence, started to highlight structural properties of the LIR-motif mediated interactions between LIRCPs and Atg8 proteins (Noda et al. 2008; Noda et al. 2010). In particular, an important observation related these Short Linear Motifs with disorder to order transitions upon binding with their compactly folded partners, proposing intrinsic disorder as a potentially important property of functional LIR-motifs (Noda et al. 2010).

In this study, we focus on the following objectives:

1. Development of *in silico* methods and user-friendly tools for detecting putative LIR-motifs and for providing useful information for downstream prioritizing LIR-motifs for experimental validation. Such a method can facilitate the discovery of *novel selective autophagy receptor/adaptor proteins* across eukaryotes. Towards this goal, it is necessary to delimit the structural and functional properties of functional LIR-motifs and to possibly devise new representations of the LIR-motif.
2. Determination of sequence features of functional LIR-motifs and their (predicted) structural properties. Available experimental data highlight the importance of flexibility in the regions containing functional LIR-motifs, thus systematically investigating the role of intrinsically disordered regions in LIR-motifs is a main focus of this study.
3. Development of tools and databases for exploiting existing structural data for enhancing our understanding of the properties of LIR-mediated interactions between LIRCPs and Atg8 proteins.

With the fulfilment of the above objectives, we also aim to generate new knowledge that may enhance our understanding of autophagy-related protein-protein interactions and open new avenues for research in the elucidation of the molecular mechanisms underlying selective autophagy.

In the following chapters, we present:

- i. The development of the freely available iLIR server which provides an easy way to analyse protein sequences for the presence of LIR motifs. The underlying method is carefully validated in a set of well-studied proteins known to interact with Atg8 proteins.
- ii. The systematic validation of different sources of information (including predictions) of intrinsic disorder as a feature for enhancing LIR-motif prediction methods.

- iii. The development of tools to analyse structural instances of complexes of LIRCPs and Atg8 proteins (including the results of peptide docking experiments) and the development of a specialised database to make these results available to the wider scientific community.

We anticipate that the tools and types of analyses presented in the following sections will be useful in the elucidation of novel players in selective autophagy (receptor and adaptor proteins). Furthermore, it will enable the study of autophagy in species other than human and yeast (where, traditionally, most knowledge regarding this important cellular process has been acquired) and may inspire complementary computational approaches which may facilitate further advances in this rapidly evolving and exciting field of research.

IOANNA KALVARI

2 Computational steps towards the characterization and identification of LIR-motifs

2.1 Preface

2.1.1 The Atg8 protein family

An undoubtedly central role in autophagy hold the proteins belonging to the autophagy-related 8 (Atg8) family, a name deriving from the Atg8 protein primarily identified in *Saccharomyces cerevisiae*. Although there is only one Atg8 protein encoded in the yeast genome, higher eukaryotes come with 4 distinct types:

- Microtubule associated protein 1 light chain 3, known as MAP1LC3 or LC3 and its 4 isoforms including:
 - LC3A, LC3B, LC3B2 and LC3C
- Gamma-aminobutyric acid receptor-associated protein (GABARAP)
- Gamma-aminobutyric acid receptor-associated protein-like 1 (GABARAPL1/GEC1)
- Gamma-aminobutyric acid receptor-associated protein-like 2 (GABARAPL2/GATE16/GEF2)

According to the work of Noda et al. (Noda et al. 2010), who very nicely demonstrated the secondary and tertiary structural aspects of the proteins belonging to the Atg8 family, these proteins comprise a C-terminal domain that resembles ubiquitin-like structures, but without any (or below the detection threshold) similarities at sequence level, and an N-terminal extension with two consecutive α -helices. Their distinction from ubiquitin proteins relies on this conserved and unique feature.

The binding to the adaptor and receptor proteins is achieved via short linear motifs known in the literature as AIMS, LIRs or GIMs, a name driven from the Atg8 homolog they preferably interact with. The LIR-motifs are located on the surface of adaptor and receptor proteins and upon binding with the Atg8 homologs they also interact with the two conserved hydrophobic pockets on the surface of the Atg8 receptors, undertaking an extended β -sheet conformation (**Figure 4**). In particular, their interaction with the LIR motifs is achieved via binding to their 2 hydrophobic pockets named **W-site** and **L-site**, a name given by the amino

acids initially found to be interacting with those, a Tryptophan (Trp - W) and a Leucine (Leu - L) at positions 3 and 6 of the LIR-motif respectively. The adoption of an extended β -strand conformation by the core of the LIR motif facilitates its interaction in a parallel fashion with the second β -strand of the Atg8 target forming an intermolecular β -sheet (Rogov et al. 2014).

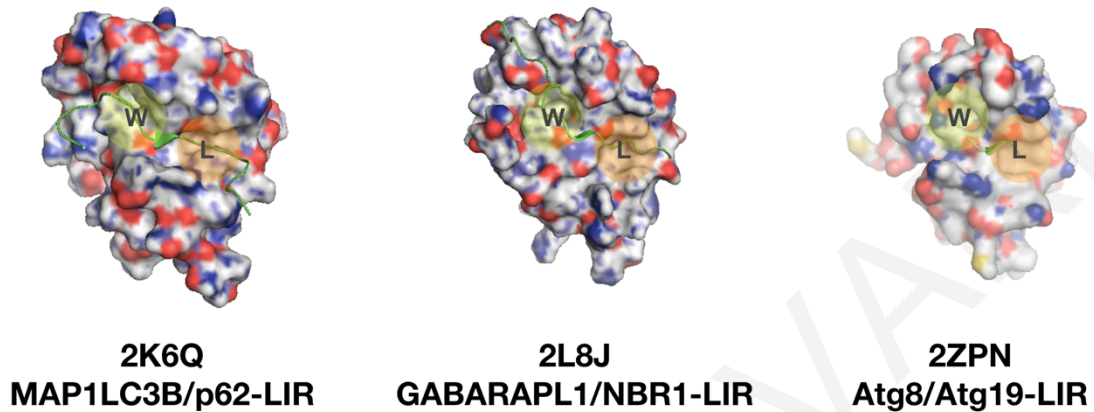


Figure 4. Three of the receptors belonging to the Atg8 family in complex with a LIR-motif peptide

Three of the receptors belonging to the Atg8 family in complex with a LIR-motif peptide. Structure A (PDB id: 2K6Q) is the MAP1LC3B protein in complex with the LIR-motif of p62. Structure B (PDB id: 2L8J) is the GABARAPL1 protein in complex with the LIR-motif of NBR1 and structure C (PDB id: 2ZPN) is the actual Atg8 protein in complex with the LIR peptide of cargo-receptor protein Atg19. In all three structures the LIR peptides (green) take on an extended β -strand conformation at the two hydrophobic pockets of the Atg8 proteins (W-site, L-site; illustrated in yellow and orange respectively).

Until very recently, the majority of studies focused on the characterisation of Atg8 Interacting Motifs (AIMs) (Noda et al. 2010) and LC3 Interacting Regions (LIRs) (Pankiv et al. 2007). Recently, Rogov and colleagues in their structural analysis on binders of the GABARAP isoforms, suggested the classification of the LIR-motifs based on their preferred Atg8 partner and gave emphasis to those motifs found to be specific to the GABARAP proteins, which they named GABARAP Interaction Motifs (GIMs). For the purposes of this study we maintain the term LIR-motif for all motifs regardless of binding specificity or species of origin.

2.1.2 Selective autophagy receptor and adaptor proteins

Autophagy is mediated by adaptor and receptor proteins, that selectively recruit cargo to an enclosed double membrane structure called the autophagosome. The autophagosome will travel carrying all its freight to fuse with a lysosome (or vacuole) to form the autolysosome.

This event causes all intra-autophagosomal components to be broken down by lysosomal hydrolases and released back to the cytosol to be re-used by the cell.

The distinction between receptor and adaptor proteins is based on the way they interact with the proteins of the Atg8 family. Selective autophagy adaptors interact with proteins of the Atg8 family on the outer membrane of the autophagosome and are found to be implicated in many different processes from autophagy initiation to the degradation of materials by the autolysosome. A couple of examples are the ULK1 and ULK2 adaptor proteins participate in autophagosome formation (Kraft et al. 2012), FYCO1 which participates in autophagosome transport (Pankiv et al. 2010), TBC1D5, establishes communication with the endocytic network (Popovic et al. 2012) and PLEKHM1 with a key role in the autophagosome-lysosome fusion (McEwan et al. 2015). In contrast, selective autophagy receptors (SARs) interact with the Atg8 proteins on the inner side of the autophagosome and as a consequence being degraded with the rest of the cytosolic material. Clearly the fate of the adaptor proteins is less “tragic”.

As previously mentioned, interaction with the proteins of the Atg8 family is achieved via short linear motifs, namely LIR-motifs. With respect to structural features, in the majority of the currently documented cases, LIR-motifs are shown to take an extended conformation when bound to the LIR docking site (LDS) of Atg8 homologs. An exception to the standard is the LIR-motif of the Clathrin heavy chain 1 (CLH1_HUMAN), which instead folds into an α -helical structure (Fotin et al. 2004).

Building on that observation and based on the fact that a LIR-motif during its interaction with the LDS must have an extended conformation, then it is highly probable for LIR-motifs to have the characteristics of “chameleon sequences” (Mezei 1998) or conformational switches (Tsolis et al. 2013). Those are found to be short sequences that adopt more than one distinct secondary structure state and have been long known to be important in protein aggregation and amyloid formation (Kelly 1996).

Another assumption is that the function of LIR-motifs may be facilitated by a short-range (with respect to the length of these sequences) of conformational changes. Such structural rearrangements could bring this short linear motif in a suitable extended conformation in order to interact with the 2 well-conserved hydrophobic pockets on the surface of Atg8 homologs (Noda et al. 2008; Noda et al. 2010). Combined with the observation that autophagy-related proteins are relatively rich in intrinsically disordered regions (IDRs) (Mei

et al. 2014), it is possible that the LIR-motifs may adopt the required conformation after switching from a disordered to an ordered state.

In order to test this hypothesis, we scanned all of the proteins containing experimentally verified LIR-motifs (**Table 1**), in search for the presence of intrinsically disordered regions (IDRs). Disorder LIR-motifs were initially determined using the ANCHOR software (Mészáros et al. 2018), a dataset that was later enhanced by incorporating data from MobiDB (Piovesan et al. 2018) and 2 additional disorder prediction tools IUPRED2A (Mészáros et al. 2018) and SPOT-disorder (Hanson et al. 2017) (see next chapter 3).

2.2 Data and Methods

2.2.1 Data

2.2.1.1 Compiling a sequence dataset

The sequences used in this study were obtained from the UniProt Knowledgebase (<https://www.uniprot.org/>) and saved locally in flat files in FASTA format. Access to each sequence was established via their corresponding accession which is used as parameter, search by protein or gene name or keywords.

Name	Species	UniProt Accession	Name	Species	UniProt Accession
ATG4B	<i>H. sapiens</i>	Q9Y4P1	TP53INP2/DOR	<i>H. sapiens</i>	Q8IXH6
ATG13	<i>H. sapiens</i>	O75143	TP53INP1	<i>H. sapiens</i>	Q96A56
Calreticulin	<i>H. sapiens</i>	P27797	TBC1D5 LIR2	<i>H. sapiens</i>	Q92609
Clathrin HC	<i>H. sapiens</i>	Q00610	Stbd1	<i>H. sapiens</i>	O95210
c-Cbl	<i>H. sapiens</i>	P22681	p62	<i>H. sapiens</i>	Q13501
Dvl2	<i>H. sapiens</i>	O14641	NIX	<i>H. sapiens</i>	O60238
FUNDC1	<i>H. sapiens</i>	Q81VP5	FIP200	<i>H. sapiens</i>	Q8TDY2
FYCO1	<i>H. sapiens</i>	Q9BQS8	AtNBR1	<i>A. thaliana</i>	Q9SB64
NBR1	<i>H. sapiens</i>	Q14596	DmATG1B	<i>D. melanogaster</i>	Q8MQJ7
OATL1/TBC1D25	<i>H. sapiens</i>	Q3MII6	ScAtg1	<i>S. cerevisiae</i>	P53104
Optineurin	<i>H. sapiens</i>	Q96CV9	ScAtg3	<i>S. cerevisiae</i>	P40344
ULK1	<i>H. sapiens</i>	O75385	ScAtg19	<i>S. cerevisiae</i>	P35193
ULK2	<i>H. sapiens</i>	Q8IYT8	ScAtg32	<i>S. cerevisiae</i>	P40458

Table 1. Selective autophagy receptor and adaptor proteins with experimentally determined LIR-motifs. Gene or protein names, species of origin and the UniProt accession numbers are displayed.

All sequences underwent manual curation to ensure their validity and the exact position of the LIR-motifs. In cases where searches resulted in multiple hits, we manually selected complete sequences over truncated ones with preference to curated entries matching the sequences reported in the respective literature. The 26 UniProt entries were retrieved using this procedure and are listed in **Table 1**.

In particular, for the LIRCPs studied for the updated definition of the LIR-motif (xLIR) we followed a manual data-cleansing procedure, where instances of LIR-motifs reported in the literature that did not match with UniProt sequences were corrected for downstream analyses (Table 2).

2.2.1.2 Randomised sequence dataset

Randomisation of datasets is necessary in order to eliminate any biased interpretation of the outcome and ensure that what is being observed is not happening at random. For this end, we devised a randomized dataset where randomised versions of the sequences in **Table 2** were generated by shuffling, thus maintaining composition of the peptides using the `shuffleseq` program available from the EMBOS explorer server (<http://emboss.bioinformatics.nl/>).

2.2.2 Methods

2.2.2.1 Intrinsic disorder prediction with ANCHOR software

We hypothesize that several of the genuine LIR-motifs will lie in intrinsically disordered regions which have the potential to become ordered upon interaction with the Atg8 proteins. This seems to be a general property of several SLIMs (Davey et al. 2012). Therefore, we decided to use the ANCHOR software which predicts (using single-sequence information) subsequences flanking or overlapping intrinsically disordered regions - herein called *anchors* - with a high potential to be stabilized upon binding to a target molecule (Mészáros et al. 2018).

2.2.2.2 Revising the LIR-motif regular expression

A first scanning using the canonical LIR (cLIR) `[DE][DEST][WFY][DELIV]X[ILV]` introduced by Alemu et al. (Alemu et al. 2012), revealed its weakness in identifying LIR motifs, with it being able to only recognize 11 out of the 27 experimentally determined LIR motifs (40.7%). Driven by that outcome, we downloaded all protein sequences proposed in their study and manually created a multiple sequence alignment composed of the 27 verified LIR sequences illustrated in **Table 3** under the block of Alemu et al.

Name	Species	UniProt ACC	LIR-motif	LIR Position	LIR Limits	Masked	Masked Residues	References
ATG4B	<i>H. sapiens</i>	Q9Y4P1	DAATLTYDGLRF	8	2-13	N (N)	D	(Satoo et al. 2009)
ATG13	<i>H. sapiens</i>	O75143	GNTDDDFVMIDF	429* (444)	438-449	N (N)	-	(Alemu et al. 2012)
Calreticulin	<i>H. sapiens</i>	P27797	GSLEDDWDFLPP	183* (200)	194-205	Y (Y)	E	(Mohrlüder, Stangler, et al. 2007)
Clathrin HC	<i>H. sapiens</i>	Q00610	VGYPDWIFLLR	513* (514)	508-519	N (N)	-	(Mohrlüder, Hoffmann, et al. 2007)
c-Cbl	<i>H. sapiens</i>	P22681	ASSSFGWLSLDG	802	796-807	Y (Y)	P, S, G, H, D	(Sandilands et al. 2011)
Dvl2	<i>H. sapiens</i>	O14641	EVRDRMWLKIIT	444	438-449	Y (N)	P, R, S	(Gao et al. 2010)
FUNDC1	<i>H. sapiens</i>	Q8IVP5	ESDDDSYEVLDL	18	12-23	N (N)	-	(Liu et al. 2012)
FYCO1	<i>H. sapiens</i>	Q9BQS8	PPDDAVFDIITD	1280	1274-1285	Y (N)	Q, S, E	(Pankiv et al. 2010)
NBR1	<i>H. sapiens</i>	Q14596	SASSEDIYIILP	732	726-737	Y (Y)	E, K	(Kirkin et al. 2009)
OATL1/TBC1D25	<i>H. sapiens</i>	Q3MII6	SPLLEDWDIISP	136	130-141	Y (N)	P, S	(Itoh et al. 2011)
Optineurin	<i>H. sapiens</i>	Q96CV9	GSSEDSFVEIRM	178	172-183	Y (Y)	E	(Wild et al. 2011)
ULK1	<i>H. sapiens</i>	O75385	SCDTRDDFVMVPA	357	351-362	Y (Y)	S, P	(Alemu et al. 2012)
ULK2	<i>H. sapiens</i>	Q8IYT8	SCDTRDDFVLPVPH	353	347-358	Y (Y)	S	(Alemu et al. 2012)
TP53INP2/DOR	<i>H. sapiens</i>	Q8IXH6	EDEVGDWLIIDL	35	29-40	Y (N)	R, P	(Sancho et al. 2012)
TP53INP1	<i>H. sapiens</i>	Q96A56	EKEDEEWILVDF	31	25-36	Y (Y)	E	(Sancho et al. 2012)
TBC1D5 LIR2	<i>H. sapiens</i>	Q92609	SSKDSGFTIVSP	788* (787)	781-792	Y (Y)	S	(Popovic et al. 2012)
Stbd1	<i>H. sapiens</i>	O95210	RVDHEEWEVMPR	203	197-208	Y (N)	S	(Jiang et al. 2011)
p62	<i>H. sapiens</i>	Q13501	SGGDDDTWHLSS	338	332-343	Y (Y)	S	(Pankiv et al. 2007)
NIX	<i>H. sapiens</i>	O60238	AGLNSGWELPM	36	30-41	Y (Y)	N, S	(Novak et al. 2010)
FIP200	<i>H. sapiens</i>	Q8TDY2	DAHTFDFTIPH	702	696-707	Y (N)	E, S	(Alemu et al. 2012)
AtNBR1	<i>A. thaliana</i>	Q9SB64	LCGVSEWDPILE	661	655-666	Y (N)	S	(Svenning et al. 2011)
DmATG1B	<i>D. melanogaster</i>	Q8MQJ7	HEDSDDFVLVVK	391	385-396	Y (Y)	S, Q	(Alemu et al. 2012)
ScAtg1	<i>S. cerevisiae</i>	P53104	RSFEREYVVVEK	391* (429)	423-434	Y (Y)	S, E	(Alemu et al. 2012; Nakatogawa et al. 2012; Kraft et al. 2012)
ScAtg3	<i>S. cerevisiae</i>	P40344	LDGVGDWEDLQD	270	264-275	Y (Y)	D	(Yamaguchi et al. 2010)
ScAtg19	<i>S. cerevisiae</i>	P35193	NEKALTWEEL	412	406-415	Y (Y)	E	(Noda et al. 2008)
ScAtg32	<i>S. cerevisiae</i>	P40458	DSISGGSQAIQP	86	80-91	Y (Y)	S, D	(Okamoto et al. 2009)

Table 2. Selective autophagy receptor/adaptor proteins with experimentally verified LIR-motifs.

Name: protein/gene name. **Species:** the particular species it belongs to. **UniProt ACC:** a unique identifier assigned by UniProtKB. **LIR-motif:** the sequence of the LIR-motif*. **LIR Position:** center of LIR-motif based on Alemu et al. (Alemu et al. 2012) – in parenthesis corrected the position of the LIR-motif on the UniProtKB sequence. **LIR Limits:** start-end positions of the LIR-motif based on the sequence retrieved from UniProtKB. **Masked:** presence of Low Complexity Region (Y/N) – in parenthesis a “binary” value indicating if the LCR overlaps the LIR-motif. **Masked Residues:** residues identified as LCRs – in bold is the residue participating in the overlap. **Reference:** the study in which a particular motif was experimentally verified. *We could not trace the difference with the UniProt entry based on the evidence listed therein. **Calreticulin is known to contain a cleavable signal peptide (residues 1-17). Low complexity regions (in particular, local compositional bias) was detected using CAST with default parameters (Promponas et al. 2000), which for each detected LCR assigns a specific residue type.

With the help of explicitly developed software that loops over the MSA and identifies all distinct amino acids that appear at each column of the alignment, we generated a new more relaxed regular expression. The resulting regular expression is [ADEFGLPRSK][DEGMSTV][WFY][DEILQTV][ADEFHIKLMPTV][ILV], keeping the conserved residues **W, F, Y** at the 3rd position and maintaining the aliphatic amino acids **I, L, V** at the 6th position of the sequence and allowing all possible amino acids at remaining positions. We named this regular expression as the eXtended LIR motif (xLIR-motif).

As expected, this revised regular expression matches all 27 experimentally verified LIR-motifs introduced by Alemu et al. (Alemu et al. 2012) with a 100% sensitivity. At this point one can argue that in proteome-wide scans, this would introduce many spurious hits. In fact, we compute the probability of occurrence of cLIR- and xLIR-motifs in random sequences as 1.8×10^{-6} and 1.5×10^{-3} respectively (see Results) – therefore, many false positive hits are expected to be detected by the xLIR motif. In the following sections we propose additional methods that work in a synergistic manner for the elimination of falsely classified instances.

2.2.2.3 Generation of an xLIR Position Specific Scoring Matrix (PSSM)

Regular expressions are very useful tools for quickly scanning large volume of data in search for meaningful patterns. However, due to their deterministic nature, speed comes at the expense of their expressive power, meaning that a subsequence either matches the regex at hand or not. In the case of allowing almost all possible amino acid alternatives at each position in an attempt to capture as many instances as possible, the pattern on one hand can become more sensitive, but on the other hand it comes with the hazardous drawback that the regular expression becomes saturated.

Another disadvantage is that in the case of LIR-motifs, the short length of the peptides increases the probability of such patterns to occur by chance in long protein sequences, thus resulting in many spurious hits. Imagine using a regular expression to annotate complete proteomes with LIR-motifs. It is anticipated that the saturated xLIR regex would result in numerous hits, the majority of which would be have falsely predicted as such. It is therefore very crucial that more sophisticated methods are employed in order to be able to filter out as many of those falsely annotated LIR regions.

In 2009, Mohrlüder and colleagues used position specific scoring matrices (PSSMs) as a means for detecting LIR-motifs. The PSSM was composed from data coming from phage display screening data of a randomised peptide library (Mohrlüder, Stangler, et al. 2007). A scan of the entire SwissProt database using the PSSM they constructed, resulted in the discovery of calreticulin (CALR) and its interaction with GABARAP. Two other known LIRCPs identified during this process was clathrin heavy chain Hc (CLTC) and BNIP3L/NIX (Mohrlüder, Stangler, et al. 2007).

Building on that idea and driven by the fact that regular expressions could be insufficient in correctly identifying LIR-motifs in an attempt to filter false positives, we constructed a PSSM based on the list of 27 experimentally verified LIR-motifs, in support of the instances predicted by the xLIR motif. This required the creation of a multiple sequence alignment (MSA), which contained all 27 verified LIRs from Alemu et al. (Alemu et al. 2012).

A PSSM is a $L \times 20$ scoring matrix based on the amino acid frequencies at every position of a multiple sequence alignment (MSA), where L is the length of the sequences comprising the MSA (**Figure 5**). Each element in a PSSM matrix is a log-odds score representing the appearance of an amino acid in a particular position. Highly frequent amino acids are assigned very high and positive scores, whereas rarely occurring residues are assigned negative values. The strength of this approach relies on the fact that apart from the presence of different amino acid residues in a specific position of the pattern, PSSMs are also able to capture the significance of each residue type occupying a certain position, compensating in a combinatorial model for the weakness of the regular expression. Residues absent from the alignment can be assigned log-odds scores based on the background probabilities encoded in a typical (position in-specific) scoring matrix, or by introducing pseudocounts, which is equivalent to multiplying the probabilities of occurrence of each residue in a specific column of the MSA by a Dirichlet distribution. In order to construct an xLIR specific PSSM we used the stand-alone (command-line) version of PSI-BLAST with default parameters and the MSA of the 27 experimentally verified LIR-motifs as input. The properties of the MSA (and thus the resulting PSSM) are summarized in the sequence logo in **Figure 6-C**.

In principle, we could scan our set of protein candidates with the aforementioned PSSM using a simple script that moves the PSSM along the query sequences in search for highly scoring hexapeptides, irrespectively of the presence of a cLIR- or xLIR-motif. The sliding of the PSSM would happen one residue at a time, meaning that at each iteration the starting point of the PSSM window is at position $p_{n+1} = p_n + 1$, with the final iteration at position p_{final}

= $L-w+1$, where L is the length of the target sequence and w is the size of the sliding window (the length of the PSSM/LIR-motifs), thus not allowing for gaps. However, as the xLIR-motif is designed to be highly sensitive, we decided to only scan LIR-motif candidates with the PSSM in order to optimize consumption of computational resources necessary for scanning (possibly) large sequence datasets. A custom software tool was built to use this PSSM for scanning protein sequences. Since the vast majority of the known LIR-motifs are of length 6, we implement our search procedure by sliding the PSSM along the query sequence with infinite gap-penalty (i.e. without allowing for gaps).

LIR-PSSM

A	0	-2	-4	-2	0	-2
R	1	-2	-4	-2	-1	-4
N	-1	0	-5	-2	-2	-4
D	3	5	-6	2	0	-5
C	-3	-3	-4	-3	-2	-2
Q	-1	-1	-4	1	-1	-4
E	1	2	-5	2	1	-4
G	0	1	-5	-3	-2	-5
H	-2	-2	-3	-2	0	-4
I	-2	-3	-4	1	1	4
L	-1	-3	-3	1	0	4
K	0	-1	-5	-1	0	-4
M	-2	1	-3	-1	2	1
F	1	-4	4	-2	1	-1
P	0	-2	-5	-3	0	-4
S	1	1	-4	-1	0	-3
T	-1	1	-4	1	1	-2
W	-3	-4	11	-3	-3	-3
Y	-2	-3	5	-2	-1	-2
V	-2	-3	-4	2	1	3
	X₂	X₁	X₀	X₁	X₂	X₃

Figure 5. The xLIR-PSSM.

The actual matrix of the PSSM with amino acid log-odds scores for each position of the LIR-motif. The PSSM was constructed based on the MSA of the 27 LIR-motifs from Alemu et al. (Alemu et al. 2012), using the stand-alone version of PSI-BLAST (see text for details).

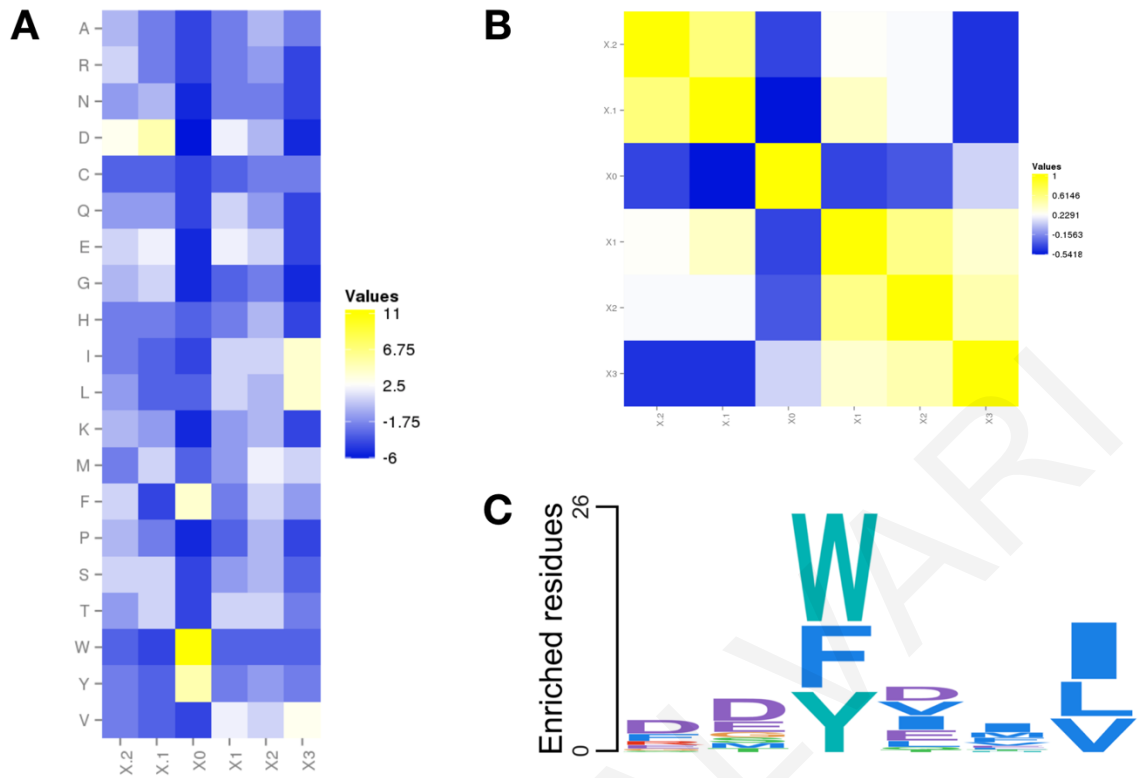


Figure 6. Graphical representation of the xLIR-PSSM

A – Heatmap plot representation of the xLIR-PSSM, where “hot” colours correspond to higher PSSM scores. **B** – Correlation plot showing between position similarities for the xLIR-PSSM. For each PSSM position pair the Euclidean distance serves as the clustering metric. **C** - The sequence logo resulted from the multiply aligned verified LIR-motifs and used to define the xLIR regular expression. The xLIR-PSSM heatmap and correlation plot were generated using <http://www2.heatmapper.ca/> (Babicki et al. 2016), and the sequence logo was generated using the PSSMsearch webserver (Krystkowiak et al. 2018).

2.2.2.4 Metrics for assessing the quality of predictions

In this section, the evaluation of all prediction schemes was performed by calculating the following metrics:

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{FP + TN}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Balanced\ accuracy = \frac{TPR + TNR}{2}$$

The values were computed under the following assumptions:

- A true positive (TP) LIR-motif is a functional LIR motif with experimental evidence reported in the literature
- A true negative (TN) LIR motifs is one whose experimental validation shows they are not functional and predicted as such
- A false positive (FP) LIR-motif is one predicted as functional, but without existing experimental support
- A false negative (FN) LIR-motif the case where experimental evidence proves it is functional, but not predicted as such

2.3 Results

A collective illustration of our results is portrayed in **Table 3**, followed by a thorough analysis and discussion that is organised in distinct sections. The table lists the entire collection of proteins rigorously examined in this study, including their experimentally validated LIR-motifs and their corresponding computational predictions (i.e cLIR, xLIR, Anchor, PSSM). The sections are conversed exhaustively hereunder.

UNIPROT ID	UNIPROT ACC	MOTIF						PSSM score (e-value)	Species
		Sequence	Position	Verified	cLIR	xLIR	Anchor		
Data set from Alemu et al. (Alemu et al. 2012)									
ATG13_HUMAN	O75143	EGFQTV	166–171	No	No	Yes	No	11 (1.5e-01)	Human
		DDFVMI	442–447	Yes	Yes	Yes	Yes	20 (8.4e-03)	Human
Atg1_YEAST	P53104	REYVVV	427–432	Yes	No	Yes	Yes	14 (5.7e-02)	Yeast
Atg32_YEAST	P40458	GSWQAI	84–89	Yes	No	Yes	Yes	17 (2.2e-02)	Yeast
		KEYQSL	235–240	No	No	Yes	No	12 (1.1e-01)	Yeast
		LGYILL	524–529	No	No	Yes	No	10 (2.0e-01)	Yeast
ATG4B_HUMAN** [MM]	Q9Y4P1	LTYDTL	6–11	Yes	No	Yes	No	12 (1.1e-01)	Human
		PMFELV	347–352	No	No	Yes	No	10 (2.0e-01)	Human
		EDFEIL	386–391	No	Yes	Yes	No	17 (2.2e-02)	Human
Atg19_YEAST	P35193	LTWEEL	410–415	Yes	No	Yes	No	18 (1.6e-02)	Yeast
Atg3_YEAST	P40344	GDWEDL	268–273	Yes	No	Yes	No	22 (4.4e-03)	Yeast
BN13L_HUMAN	O60238	SSWVEL	34–39	Yes	No	Yes	Yes	20 (8.4e-03)	Human
		AEFLKV	183–188	No	No	Yes	No	10 (2.0e-01)	Human
CALR_HUMAN	P27797	GGYVKL	107–112	No	No	Yes	No	12 (1.1e-01)	Human
		DEFTHL	166–171	No	No	Yes	No	14 (5.7e-02)	Human
		DDWDFL	198–203	Yes	Yes	Yes	Yes	26 (1.2e-03)	Human

CBL_HUMAN	P22681	DTYQHL	90–95	No	No	Yes	No	14 (5.7e-02)	Human
		LTYDEV	272–277	No	No	Yes	No	11 (1.5e-01)	Human
		FGWLSL	800–805	Yes	No	Yes	Yes	18 (1.6e-02)	Human
		REFVSI	893–898	No	No	Yes	Yes*	13 (7.9e-02)	Human
FUND1_HUMAN	Q8IVP5	DSYEVL	16–21	Yes	Yes	Yes	No	16 (3.0e-02)	Human
		GGFLLL	81–86	No	No	Yes	No	10 (2.0e-01)	Human
OPTN_HUMAN	Q96CV9	DSFVEI	176–181	Yes	Yes	Yes	Yes	15 (4.2e-02)	Human
Q8MQJ7_DROME	Q8MQJ7	ADYLSV	96–101	No	No	Yes	No	14 (5.7e-02)	Drosophila
		DDFVLV	389–394	Yes	Yes	Yes	Yes	17 (2.2e-02)	Drosophila
Q9SB64_ARATH	Q9SB64	RVWVLI	479–484	No	No	Yes	No	15 (4.2e-02)	Arabidopsis
		SEWDPI	659–664	Yes	No	Yes	No	20 (8.4e-03)	Arabidopsis
RBCC1_HUMAN	Q8TDY2	FDKETI	700–705	Yes	No	Yes	Yes	17 (2.2e-02)	Human
SQSTM_HUMAN** [LL]	Q13501	DDWTHL	336–341	Yes	No	Yes	Yes	24 (2.3e-03)	Human
STBD1_HUMAN** [LN]	O95210	EEWEMV	201–206	Yes	Yes	Yes	No	21 (6.1e-03)	Human
T5311_HUMAN	Q96A56	DEWILV	29–34	Yes	Yes	Yes	Yes	20 (8.4e-03)	Human
TBC25_HUMAN	Q3MII6	EVYLSL	95–100	No	No	Yes	No	8 (3.9e-01)	Human
		EDWDII	134–139	Yes	Yes	Yes	No	24 (2.3e-03)	Human
TBCD5_HUMAN	Q92609	KEWEEL	57–62	Yes	No	Yes	No	20 (8.4e-03)	Human
		DDFILI	713–718	No	Yes	Yes	Yes*	17 (2.2e-02)	Human
		SGFTIV	785–790	Yes	No	Yes	Yes	11 (1.5e-01)	Human
T5312_HUMAN	Q8IXH6	DGWLI	33–38	Yes	No	Yes	Yes	21 (6.1e-03)	Human
ULK1_HUMAN	O75385	DDFVMV	355–360	Yes	Yes	Yes	Yes	19 (1.2e-02)	Human
ULK2_HUMAN	Q8IYT8	DDFVLV	351–356	Yes	Yes	Yes	Yes	17 (2.2e-02)	Human
CLH1_HUMAN	Q00610	PDWIFL	512–517	Yes	No	Yes	No	22 (4.4e-03)	Human
		GMFTEL	1315–1320	No	No	Yes	No	11 (1.5e-01)	Human
		EDYQAL	1475–1480	No	No	Yes	No	16 (3.0e-02)	Human
DVL2_HUMAN	O14641	RMWLKI	442–447	Yes	No	Yes	No	18 (1.6e-02)	Human
FYCO1_HUMAN** [MM]	Q9BQS8	ADYQAL	644–649	No	No	Yes	Yes*	15 (4.2e-02)	Human
		AVFDII	1278–1283	Yes	No	Yes	Yes	8 (3.9e-01)	Human
NBR1_HUMAN	Q14596	LSFELL	561–566	No	No	Yes	Yes*	10 (2.0e-01)	Human
		EDYIII	730–735	Yes	Yes	Yes	Yes	17 (2.2e-02)	Human
Additional LIRCPs from Birgisdottir et al. (Birgisdottir et al. 2013)									
BNIP3_HUMAN	Q12983	GSWVEL	16–21	Yes	No	Yes	Yes	19 (1.2e-02)	Human
		AEFLKV	159–164	No	No	Yes	No	10 (2.0e-01)	Human
MK15_HUMAN	Q8TD08	RVYQMI	338–343	Yes	No	Yes	Yes	10 (2.0e-01)	Human
CACO2_HUMAN	Q13137	FMWVTL	72–77	No	No	Yes	No	20 (8.4e-03)	Human
		DILVV	132–136	Yes	No	No	No	N/A	Human
C0H519_PLAF7	C0H519	NDWLLP	103–108	Yes	No	No	No	12 (1.2e-02)	Plasmodium
ATG34_YEAST	Q12292	KVYEKL	194–199	No	No	Yes	No	8 (3.9e-01)	Yeast
		FTWEEI	407–412	Yes	No	Yes	No	20 (8.4e-03)	Yeast
TAXB1_HUMAN	Q86VP1	DMLVV	139–143	Yes	No	No	No	N/A	Human
		ADFDIV	514–519	No	No	Yes	Yes	15 (4.2e-02)	Human
CTNB1_HUMAN	P35222	SHWPLI	502–507	Yes	No	No	No	11 (1.5e-01)	Human
Data set from Behrends et al. (Behrends et al. 2010)									

STK4_HUMAN [MM]	Q13043	EVFDVL	28–33	No	No	Yes	No	9 (2.8e-01)	Human
		GDYEFL	431–436	No	No	Yes	Yes	17 (2.2e-02)	Human
STK3_HUMAN [LM]	Q13188	EVFDVL	25–30	No	No	Yes	No	9 (2.8e-01)	Human
		GDFDFL	435–440	No	No	Yes	Yes	16 (3.0e-02)	Human
RASF5_HUMAN [MN]	Q8WWW0	-	-	N/A	N/A	N/A	N/A	N/A	Human
NEDD4_HUMAN [LL]	P46934	SEYIKL	410–415	No	No	Yes	No	13 (7.9e-02)	Human
		PGWVVL	589–594	No	No	Yes	Yes	19 (1.2e-02)	Human
		ESFEEL	1296–1301	No	Yes	Yes	No	13 (7.9e-02)	Human
A16L1_HUMAN [MM]	Q676U5	DEYDAL	164–169	No	Yes	Yes	Yes	16 (3.0e-02)	Human
TFCP2_HUMAN [LN]	Q12800	-	-	N/A	N/A	N/A	N/A	N/A	Human
SF3A1_HUMAN [LN]	Q15459	PEFEFI	148–153	No	No	Yes	No	13 (7.9e-02)	Human
FNBP1_HUMAN [MN]	Q96RU3	-	-	N/A	N/A	N/A	N/A	N/A	Human
TBC15_HUMAN [LL]	Q8TC07	AEWDMV	96–101	No	No	Yes	No	20 (8.4e-03)	Human
		PGFEVI	295–300	No	No	Yes	No	12 (1.1e-01)	Human
		FSFLDI	540–545	No	No	Yes	No	11 (1.5e-01)	Human
ANFY1_HUMAN [MN]	Q9P2R3	-	-	N/A	N/A	N/A	N/A	N/A	Human
TCPR2_HUMAN [LM]	O15040	GDYIAV	45–50	No	No	Yes	No	14 (5.7e-02)	Human
		AVFQLV	102–107	No	No	Yes	No	5 (1.0e+00)	Human
		AVFVAL	894–899	No	No	Yes	No	7 (5.3e-01)	Human
		DEWEVI	1406–1411	No	Yes	Yes	No	23 (3.2e-03)	Human
ECHA_HUMAN [LM]	P40939	AVFEDL	447–452	No	No	Yes	No	7 (5.3e-01)	Human
NIPS2_HUMAN [MM]	O75323	-	-	N/A	N/A	N/A	N/A	N/A	Human
ATG5_HUMAN [MM]	Q9H1Y0	-	-	N/A	N/A	N/A	N/A	N/A	Human
ATG7_HUMAN [MM]	O95352	SSFQSV	258–263	No	No	Yes	No	10 (2.0e-01)	Human
KPCI_HUMAN [LM]	P41743	-	-	N/A	N/A	N/A	N/A	N/A	Human
EPN4_HUMAN [LM]	Q14677	-	-	N/A	N/A	N/A	N/A	N/A	Human
ATG3_HUMAN [LL]	Q9NT62	-	-	N/A	N/A	N/A	N/A	N/A	Human
DYXC1_HUMAN [LL]	Q8WXU2	AVFLSL	16–21	No	No	Yes	No	6 (7.4e-01)	Human
		AMWETL	81–86	No	No	Yes	No	19 (1.2e-02)	Human
NEK9_HUMAN [LL]	Q8TD19	-	-	N/A	N/A	N/A	N/A	N/A	Human
UBA5_HUMAN [MM]	Q9GZZ9	SDYEKI	66–71	No	No	Yes	No	17 (2.2e-02)	Human
		FDYDKV	103–108	No	No	Yes	No	16 (3.0e-02)	Human
TBD2B_HUMAN [LM]	Q9UPU7	EEWELL	252–257	No	Yes	Yes	Yes	20 (8.4e-03)	Human

KBTB6_HUMAN [LL]	Q86V97	ESFEVL	120–125	No	Yes	Yes	No	13 (7.9e-02)	Human
IPO5_HUMAN [LN]	O00410	ETYENI	31–36	No	Yes	No	No	11 (1.5e-01)	Human
		DGWEFV	655–660	No	No	Yes	No	21 (6.1e-03)	Human
		LSWLPL	997–1002	No	No	Yes	No	16 (3.0e-02)	Human
NCOA7_HUMAN [LM]	Q8NI08	AEYDKL	185–190	No	No	Yes	No	13 (7.9e-02)	Human
		GEWEDL	308–313	No	No	Yes	No	19 (1.2e-02)	Human
		DDFVDL	414–419	No	Yes	Yes	Yes	18 (1.6e-02)	Human
		KSWEII	745–750	No	No	Yes	No	19 (1.2e-02)	Human
KAP0_HUMAN [MM]	P10644	EEFVEV	310–315	No	Yes	Yes	No	13 (7.9e-02)	Human
GYS1_HUMAN [NN]	P13807	-	-	N/A	N/A	N/A	N/A	N/A	Human
KBTB7_HUMAN [LL]	Q8WVZ9	ESFEVL	120–125	No	Yes	Yes	No	13 (7.9e-02)	Human
ATG2A_HUMAN [LM]	Q2TAZ0	PEYTEI	534–539	No	No	Yes	No	13 (7.9e-02)	Human
		EVYESI	828–833	No	No	Yes	No	9 (2.8e-01)	Human
		LEFLDV	1090–1095	No	No	Yes	No	9 (2.8e-01)	Human
FAN_HUMAN [ML]	Q92636	ESFEDL	600–605	No	Yes	Yes	No	12 (1.1e-01)	Human
		LVWDLI	869–874	No	No	Yes	No	13 (7.9e-02)	Human

Table 3. Sequences used in this study.

The data portrayed are divided into 3 distinct segments according to the study in which they were published. The top section refers to the dataset created by Alemu et al. (Alemu et al. 2012), which was used to construct the xLIR-motif and to validate both the cLIR and xLIR motifs. With the term “Verified” we refer to experimentally verified LIR-motifs, whereas “Anchor” refers to intrinsic disorder binding regions predicted by the ANCHOR tool, and found to overlap with a LIR-motif by at least 3 residues (>3). Middle and bottom data blocks derive from the works of Birgisdottir (Birgisdottir et al. 2013) and Behrends (Behrends et al. 2010) and colleagues respectively. Entries marked with a single asterisk (*) correspond to possible spurious xLIR hits, which are also predicted to overlap with *anchors*. A double asterisk (**) denotes that a sample was identifiable in all 3 studies. Since the xLIR-PSSM corresponds to a hexapeptide and is aligned to sequences in a gapless fashion, for the atypical LIR sequences (pentapeptides) of CALCOCO2/NDP52 (CACO2_HUMAN) and TAX1BP1 (TAXB1_HUMAN) their corresponding PSSM scores are marked as “N/A”. The 2 characters in the square brackets accompanying the UniProt IDs are used to distinguish between 3 possible interactions with the GABARAP and MAP1LC3B receptors respectively, as reported in the survey of Behrends and colleagues (Behrends et al. 2010). ‘N’ denotes no binding with the wild-type Atg8 homolog, ‘L’ is used to denote loss of binding with the mutant form, whereas ‘M’ denotes that binding is maintained with the mutant form - i.e. [ML] signifies a case where both the wild-type and the mutated form bind GABARAP, while the wild-type form binds MAP1LC3B and this interaction is abolished in the mutated form. Entries highlighted in red correspond to motifs detected by the xLIR regular expression which at the time of the initial analysis (fall 2013) were considered as false positives, but for which later work has validated that they are genuine LIR motifs: EDFEIL (ATG4B_HUMAN) (Skytte Rasmussen et al. 2017) and DEWEVI (TCPR2_HUMAN) (Stadel et al. 2015) respectively.

2.3.1 Combining the predictive power of xLIR and Anchors

The xLIR matches by design all 27 experimentally verified LIR-motifs at a 100% sensitivity. Positions 1, 2 and 4 ($[ADEFGLPRSK]_1$, $[DEGMSTV]_2$, $[DEILQTV]_4$) are less constrained compared to the cLIR-motif ($[DE]_1$, $[DEST]_2$, $[DELIV]_4$), whereas position 5 is more restricted. To be exact, the 5th position in the cLIR regex is occupied by the wild-card character **X**, which means that this position can be taken by any of the 20 amino acids, whilst in the case of the xLIR regular expression, that particular position can only be occupied by any of the following residues: ADEFHIKLMPTV.

Using the background frequencies for amino acid residues in a then recent version of the UniProt/SwissProt database (**Table 4**) we estimated the probability of occurrence of the cLIR- and xLIR-motif in random sequences drawn from this distribution as 1.8×10^{-6} and 1.5×10^{-3} respectively (Nevill-Manning et al. 1998). This means that overall, the xLIR-motif should be more sensitive but less specific compared to cLIR. In fact, this is the case since (in the same sequence data) the xLIR-motif detects 20 additional subsequences, which can be regarded as false positives for being non-functional as LIRs. As expected, the higher sensitivity of the xLIR-motif comes at the expense of lower specificity and therefore a larger number of bogus hits when examining large datasets (i.e. a complete proteome). In terms of accuracy, the cLIR regex seems to outperform the xLIR with accuracies 61.7% and 57.4% respectively (**Table 5**). A figure that may be misleading due to the imbalanced nature of the dataset, and by imbalanced we mean that the dataset is not comprised by an equal number of functional and non-functional LIRs.

However, the design of the negative dataset that would consist of new motif sequences complying with the xLIR motif, would not permit us to compute meaningful values for specificity and balanced accuracies for the xLIR motif as specificity is estimated at 0% and the balanced accuracy at a borderline value of 50%. In contrast, the specificity and balanced accuracy for cLIR, is estimated at 90% and 65.4% respectively (**Table 5**).

Such a result makes apparent the need to obtain a more unbiased estimate of the false positive rate for both motifs and for that purpose we constructed a sequence dataset composed of randomized (shuffled) versions of the 27 validated LIRCPs. When scanning these sequences with the xLIR and cLIR regular expressions, a number of 23 and 8 hits were reported respectively. It is worth mentioning that this figure for the xLIR-motif is somewhat in agreement with the number of the extra motifs identified in the original dataset (20 matches).

This case does not apply to the cLIR-motif as it deviates significantly from the false positive motifs in the unshuffled sequences with by 4x times.

With respect to intrinsic disorder binding regions, 17 out of the 27 verified LIR-motifs (about 63%) were found to substantially overlap with an *anchor* segment by >3 residues (**Table 3; Table 5**). Even though it is difficult to draw a significant conclusion from such a small dataset, it is worth mentioning that 14 out of 21 LIR-motifs from human LIRCPs (66.7%) overlap with an *anchor*. Interestingly the number of *anchors* discovered in the remaining species namely, *S. cerevisiae*, *D. melanogaster* and *A. thaliana*, is slightly lower (50%). Nevertheless, it seems that the combination of *anchor* prediction and a LIR regex may be a good approach for discriminating genuine (i.e. functional) LIR-motifs. An observation which consequently lead us to the next step of testing the two together.

Residue	Abundance (%)	Residue	Abundance (%)	Residue	Abundance (%)
Ala	8.25	Gly	7.07	Pro	4.70
Arg	5.53	His	2.27	Ser	6.56
Asn	4.06	Ile	5.96	Thr	5.34
Asp	5.45	Leu	9.66	Trp	1.08
Cys	1.37	Lys	5.84	Tyr	2.92
Gln	3.93	Met	2.42	Val	6.87
Glu	6.75	Phe	3.86		

Table 4. Amino acid residue background distribution.

Data regarding the 20 common amino acid residues, calculated from UniprotKB/Swiss-Prot release 2013_04, April 2013; available from the ProtScale tool (<https://web.expasy.org/protscale>).

When using the cLIR regular expression and posing an additional requirement that the functional LIR-motif should overlap with an *anchor* segment, only 8 functional LIRs would be predicted as such (**Table 3; Table 5**), resulting in very low coverage 8/27 or 29.6%. Contrary, the xLIR-motif in combination with *anchor* detection recovers 17 out of the 27 verified LIR-motifs (63.0%) and at the same time eliminates most of the false positives. To be precise, based on this compound criterion, only 4 unverified xLIRs from the human LIRCPs were predicted to be functional LIR-motifs (**Table 3**).

	xLIR	cLIR	xLIR+A	cLIR+A	xLIR+A+P13	xLIR+A P13
TP	27	11	17	8	15	26
TN	0	18	16	19	18	11
FP	20	2	4	1	2	9

FN	0	16	10	19	12	1
Sensitivity (%)	100.00	40.70	63.00	29.60	55.60	96.30
Specificity (%)	0.00	90.00	80.00	95.00	90.00	55.00
ACC (%)	57.40	61.70	70.20	57.40	70.20	78.70
BACC (%)	50.00	65.40	71.50	62.30	72.80	75.70

Table 5. Validation of xLIR and cLIR motif-based predictors.

Different schemes are validated for the prediction of functional LIR-motifs on the set of 26 proteins with validated LIRs described by Alemu and colleagues (Alemu et al. 2012). xLIR and cLIR are based simply on the detection of the xLIR and cLIR motifs, respectively, whereas xLIR+A and cLIR+A require that a functional motif should overlap with an anchor as predicted by the ANCHOR tool. The 2 rightmost columns correspond to xLIR-motifs that overlap with an *anchor* and have a PSSM score > 13 (xLIR+A+P13) and xLIR-motifs that either overlap with an *anchor* or have a PSSM score > 13 (xLIR+A|P13). ACC is for Accuracy (%), and BACC is for Balanced Accuracy (%). For each validation metric the highest recorded value is depicted in bold.

2.3.2 Using profile-based methods to identify functional LIR-motifs

Using the PSSM derived from the 27 experimentally verified LIR-motifs, we scanned the sequences of the 26 verified LIRCPs to investigate whether the PSSM can be used as a more successful means to identify functional LIR-motifs.

On top of the 47 hexapeptides matching the xLIR-motif (27 verified, 20 unverified) we also obtained a score against the PSSM for a total of **18,018** hexapeptides (termed background) stemming from the 26 LIRCP sequences of our reference dataset. More specifically, by “sliding” the PSSM over each sequence one residue at a time, a score for the comparison of the PSSM to the hexapeptide starting at the given sequence position is computed. The median of scores for the 3 classes of hexapeptides (i.e. verified LIRs, unverified LIRs, background) was 18, 12 and -8, respectively and the score distributions indicate significant differences between these classes (**Figure 7**).

To further validate the xLIR-PSSM we performed a randomisation experiment, where hexapeptides were generated on the shuffled dataset of the 26 proteins from Alemu et al. in a similar fashion as done for the “background” hexapeptides, but repeated a 1000 times. This resulted in 18,040,000 PSSM scores after scanning the shuffled sequences with our xLIR-PSSM. The median of this sample equals the median of the background dataset (-8). Furthermore, the randomisation experiment enabled us to compute the corresponding z-scores and p-values at varying PSSM threshold levels (Table 6). From our combined results,

it becomes evident that a xLIR-PSSM score >12 can be a trustworthy computational method for the discrimination between genuine and non-genuine LIR-motifs.

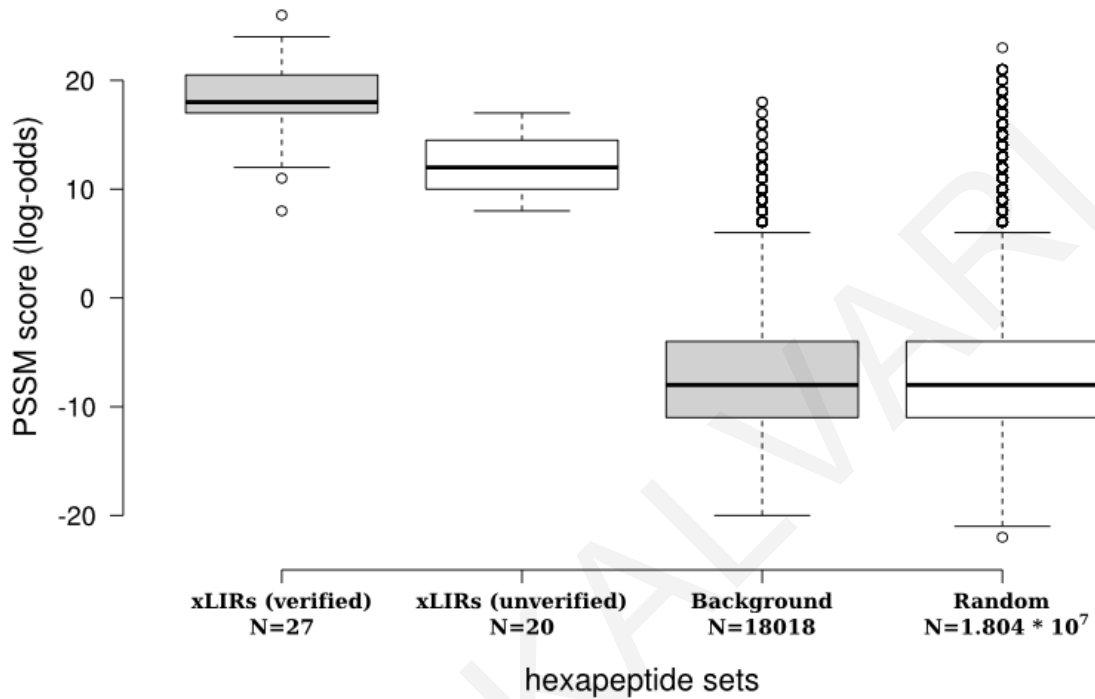


Figure 7. PSSM score distributions for different classes of hexapeptides.

Box-plot representation of PSSM score distributions for xLIR-motifs in the 26 sequences of LIRCPs (verified and unverified), the remaining hexapeptides (“background”) and 1000 randomized versions of the LIRCP dataset. Scores were obtained by evaluating the match of a sliding-PSSM along the sequences in the set of 26 sequences reported by Alemu et al. (Alemu et al. 2012) or simulated datasets. The differences indicated here suggest that the PSSMs may be able to reliably discriminate between functional and non-functional xLIRs. In particular, a Wilcoxon rank sum test with continuity correction demonstrates significant differences between both verified and unverified xLIRs compared to background ($P < 2.2 \times 10^{-16}$ and 1.2×10^{-14} respectively) and verified vs. unverified xLIRs ($P = 6.0 \times 10^{-6}$). Similar trends are observed against the fully randomized dataset (verified vs random: $P < 2.2 \times 10^{-16}$; unverified vs random: $P = 1.2 \times 10^{-14}$), whereas the background and randomized datasets showed no statistically significant differences ($P = 0.06$).

	Above cutoff	PSSM validation							
PSSM score cutoff	xLIR (verified) N = 27	xLIR (unverified) N = 20	Background N=18018 (randomized, N = 18065)	z-scores	p-values	Sens (%)	Spec (%)	ACC (%)	BACC (%)
9	26	19	93 (85)	3.08	1.04e-03	96.3	5	57.4	50.7
10	26	14	63 (63)	3.27	5.39e-04	96.3	30	68.1	63.2

11	25	11	47 (49)	3.46	2.70e-04	92.6	45	72.3	68.8
12	24	9	28 (32)	3.65	1.31e-04	88.9	55	74.5	72
13	24	8	17 (25)	3.84	6.10e-05	88.9	60	76.6	74.5
14	23	5	13 (16)	4.03	2.76e-05	85.2	75	80.9	80.1
15	22	3	10 (14)	4.22	1.20e-05	81.5	85	83	83.3
16	21	2	4 (11)	4.41	5.07e-06	77.8	90	83	83.9
17	16	0	2 (7)	4.61	2.06e-06	59.3	100	76.6	79.7
18	13	0	0 (5)	4.80	8.09e-07	48.2	100	70.2	74.1

Table 6. Validation of the PSSM method as a predictor of LIR-motifs.

We report the number of hexapeptides with a PSSM score above different threshold values. Peptides from the background dataset scoring above the threshold would be regarded as false positives if there were no restriction to comply with the xLIR-motif. Results for the randomised versions of the 26 verified LIRCPs are displayed in parentheses next to “background” data. Z-values and P-values were generated using the “Random” dataset. For each validation metric the highest recorded value is depicted in bold.

2.3.3 Validating xLIR, anchors and PSSM with independent datasets

As more studies came to the surface, it only made sense to test our methods on new datasets. In 2013, when we were about to publish the iLIR paper, Birgisdottir and colleagues published a list of 7 new LIRCPs with an equal number of experimentally determined LIR-motifs (Birgisdottir et al. 2013). Once again, we extracted all the samples from the papers and downloaded all the sequences from UniProt Knowledgebase in FASTA format. For the analysis we followed exactly the same approach as with the dataset of Alemu et al. (Alemu et al. 2012), as this would allow us to get a more unbiased estimate of how our approach performed.

Interestingly, the cLIR-motif would not match any of these sequences in contrast to xLIR that matched 3 of the 7 experimentally verified LIR-motifs, giving 4 additional “hits”, which can be safely considered as “false positives”. The 4 missed experimentally verified LIRs include the following:

- Human proteins CALCOCO2/NDP52 and TAX1BP1 reported to contain a non-canonical LIR-motif which is only 5 residues long (von Muhlinen et al. 2012; Newman et al. 2012). As expected, no PSSM scores have been computed for these “unconventional” LIR-motifs.
- *Plasmodium falciparum* Atg3 homolog (PfAtg3; UniProt ID: C0H519), with 2 mismatches to the xLIR-motif at positions 1 and 6, with asparagine and proline

occupying those positions respectively. This is however the highest scoring hexapeptide of this sequence against the PSSM (score = 12).

- Human CTNNB1/ β -catenin, also with 2 mismatches to the LIR-motif with a histidine at position 2 and a proline residue occupying position 4. Again, the top-scoring hexapeptide against the PSSM (score = 11).

Notably, none of the aforementioned LIR-motifs is predicted to be an *anchor*.

Another important source of LIRCP-related information, stems from the work of Behrends and colleagues and their effort to decipher the selective autophagy protein-protein interaction network (Behrends et al. 2010). In particular, we focus on the data presented therein in order to unravel the LIR-dependence of interactions of human Atg8 homologs GABARAP and MAP1LC3B with **34** proteins (**Table 3**, bottom).

Briefly, these authors recorded binding of these 34 proteins against the wild type and mutated forms of Atg8 homologs (Y49A, L50A for GABARAP and F52A, L53A for MAP1LC3B). Since the mutated residues lie in the LDS and are considered critical for typical LIR-mediated interactions, maintenance of the interaction after mutation indicates LIR-independent binding, whereas loss of interaction suggests LIR-dependence. Below we summarize the computational results on those proteins showing consistent interaction patterns against both GABARAP and MAP1LC3B.

For 7 of the 9 proteins that demonstrated LIR-independence for both Atg8 homologs (marked as [MM] in **Table 1**) there was at least one match of the xLIR-motif (only 3 for cLIR); interestingly only 2 of these proteins [FYCO1, FYVE and coiled-coil domain containing 1 (FYCO1_HUMAN) and ATG16L1, autophagy related 16-like 1 (*S. cerevisiae*) (A16L1_HUMAN)] had at least one xLIR overlapping with a predicted *anchor*.

Another 8 proteins were shown to interact with both Atg8 homologs in a LIR-dependent manner (marked as [LL] in **Table 3**). Six were detected to have at least an instance of the xLIR-motif (3 with cLIR) of which only 2 overlapped with an anchor: these are the validated LIR-motif of SQSTM1 and the second xLIR match of the E3 ubiquitin-protein ligase NEDD4 (PGWVVL with a PSSM score = 19).

An interesting case is the serine/threonine protein kinase NEK9, which is predicted to have 10 anchor segments, 2 of which overlap with hexapeptides scoring high against the PSSM, albeit the fact that they do not match the xLIR-motif; RGWHTI (positions: 716-721; PSSM score: 19) and DSWCLL (positions: 965-970; PSSM score: 16). Both of these hexapeptides have a single mismatch to the xLIR motif (a His and Cys residue respectively at position 4) and, along with NEDD4, they could be good candidates for further experimental validation. Intriguingly, from all the known LIRCPs with a verified LIR-motif the only protein belonging to this class is SQSTM. Interestingly, the single case in this dataset of a protein not interacting with the wild type Atg8 homologs (GYS1) does not match either the xLIR or the cLIR-motif.

2.3.4 Assembling everything into a unified resource: the iLIR webserver

Driven by our findings that the power of our approach makes a good means for an overall estimate of the genuineness of a new LIR-motif, the next logical step was to develop a resource to make our predictive methods available to the scientific community.

In 2014 we released a new web resource called iLIR, where iLIR stands for “identify LIR”. iLIR is a resource purposely designed to guide autophagy researchers to make rational decisions on which targets to select, rather than providing explicit predictions of putative LIR-motifs. iLIR is freely accessible to the research community via the URL <http://repeat.biol.ucy.ac.cy/iLIR/> and provides a unified resource combining all of our predictive tools in a single, publicly and freely available unit. The iLIR web server was developed following very simple web technologies such as the Common Gateway Interface (CGI) standard protocol, JavaScript/AJAX and Cascade Style Sheets (CSS) for the provision of common formatting between all web pages and also improve content accessibility and web page interactivity.

The Common Gateway Interface (CGI) is part of the Hypertext Transfer Protocol (HTTP) and a simple form of establishing front to back-end communication in a web resource and vice versa. The back-end can be a collection of application scripts, with each script mapping to its dedicated HTML page. CGI is language independent, so the application scripts can be implemented in any language from python and perl to C/C++ for faster processing.

2.3.4.1 iLIR: Home page

A novice iLIR user lands in the home page, where a brief description on the functionalities offered by iLIR are presented (**Figure 8**). Here, hyperlinks are provided to launch a new prediction ([Submit a job](#)) or examine a page with examples pre-ran sequences ([Examples](#)).

iLIR In silico identification of functional LC3 Interacting Region Motifs

iLIR provides easy integrated access to bioinformatics tools aiming to help researchers for assessing whether a protein is a potential functional LIR containing protein based solely on its amino acid sequence. A query sequence is analysed for:

- the presence of a generic, thus sensitive, extended LIR-motif (xLIR-motif), also scored against a PSSM constructed of experimentally validated set of LIR-motifs
- domain architecture based on [SMART](#) and [PFAM](#), and
- predicted anchors, i.e. disordered linear segments with the potential to become ordered following protein interaction.

Results are displayed in a simple graphical form, accompanied with detailed tables. Examples of iLIR output for experimentally validated functional LIR containing proteins may be found in the [Examples Page](#).

Note: noncanonical LIR-motifs, as for example the atypical motif recently discovered in NDP52 ([von Muhlinen et al., 2012](#)), are not currently detected by iLIR.

[iLIR Home](#) [Submit a job](#) [Examples](#)

To get some assistance e-mail: vprobon@ucy.ac.cy

Go to the [Bioinformatics Research Laboratory](#) web site

If you use iLIR please cite: Kalvari I, Tsompanis S, Mulakkal NC, Osgood R, Johansen T, Nezis IP, Promponas VJ. iLIR: A web resource for prediction of Atg8-family interacting proteins. *Autophagy* 2014; 10:166 - 178; [PMID: 24589857](#); [[Open access full text](#)]

Figure 8. Home page of the iLIR webserver.

2.3.4.2 iLIR: Launching a new prediction

When a user makes this selection, a simple input form is dynamically generated by the underlying CGI perl script (iLIR_cgi). The input required by the user is purposely designed to be very simple, knowing that most biologists do not want to deal with many different parameters (whose meaning they often fail to understand!!).

The user only needs to input the sequence of interest in FASTA format either by entering text (typing or copy-and-paste) or by uploading a plain ASCII text file (**Figure 9**). Since iLIR calls external services, only one amino acid sequence is expected by the server. Once the sequence is uploaded or made available in the text box, the user can launch the processing by pressing the submit button.



In silico identification of functional LC3 Interacting Region Motifs

Please paste a single sequence in [FASTA format](#)

```
>sp|Q13501|SQSTM_HUMAN Sequestosome-1 OS=Homo sapiens OX=9606
GN=SQSTM1 PE=1 SV=1
MASLTVKAYLLGKEDAAREIRRF5FCCSPEPEAEAEAAAGPGPCERLLSRVAALFPALRP
GGFQAHYRDEDGDLVAFSSDEELTMAMSYVKDDIFRIYIKEKKECRRDHRPPCAQEAPRN
MVHPNVICDGCNGPVVGTTRYKCSVCPDYDLCVCEGKGLHRGHTKLAFSPFGLHSEGF5
HSRWLRKVKHGHFGWPGWEMGPPGNWSPRPPRAGEARPGPTAESASGPPSEDPSVNF5LKNV
GESVAAALSPLGIEVDIDVEHGGKRSRLTPVSPESSTEEKSSSQPSSCCSDPSKPGGNV
EGATQSLAEQMRKIALESEGRPEEQMESDNC5GGDDDWTHLSSKEVDPSTGELQSLQMP5E
SEGPS5LDPSQEGPTGLKEAALYPHLPPEADPRLIESLSQMLSMGF5DEGGWLT5RL5LQTK
NYDIGAALDTIQYSKHPPPL
```

or upload a FASTA formatted plain text file No file chosen

[iLIR Home](#) [Submit a job](#) [Examples](#)

To get some assistance e-mail: vprobon@ucy.ac.cy

Go to the [Bioinformatics Research Laboratory](#) web site

If you use iLIR please cite: Kalvari I, Tsompanis S, Mulakkal NC, Osgood R, Johansen T, Nezis IP, Promponas VJ. iLIR: A web resource for prediction of Atg8-family interacting proteins. *Autophagy* 2014; 10:166 - 178; [PMID: 24589857](#); [[Open access full text](#)]

Figure 9. iLIR server user interface.

A simple user interface enables sequence data entry in FASTA format either by copy-pasting the sequence in the respective text box or uploading the data via a local FASTA formatted text file. At its current state only a single sequence can be processed at a time.

Initial checks on the sequence are performed and then the iLIR server takes care of executing the pipeline of tools as follows:

- a. ANCHOR prediction: the sequence is submitted to a locally installed instance of the ANCHOR software for the prediction of *anchors*. Anchors are regions within or neighbouring unstructured regions with the potential to undergo a disorder to order conformational change and bind to a globular protein.
- b. Retrieval of domain information: an automatic sequence query is executed against the SMART database (Letunic et al. 2012), resulting in a list of annotated domains and motifs including PFAM domains (Punta et al. 2012).
- c. Detection of homologs with known structure: a remote BLASTP (Blast 1997) query is issued against the Protein Data Bank (Berman et al. 2000) (using the PDB REST API), thus facilitating access to relevant structural data. More specifically all

significant hits with alignments including the reported motif are compiled in a list, linking to the respective PDB entry, and the complete output is also available for further analysis. BLAST parameters are pre-set to E-value cutoff of 0.001 and the BLOSUM62 substitution matrix.

- d. Detection of LIR-motifs: instances of xLIR- and the simpler WxxL-motifs (xx[WFY]xx[VLI]) are scanned throughout the submitted sequence.
- e. Computation of PSSM scores: Whenever a successful hit is recorded, the matched hexapeptide is scored against the position specific scoring matrix developed using the collection of experimentally verified LIR-motifs. The PSSM score is accompanied by an e-value computed using the Karlin-Altschul equation (Karlin and Altschul 1990). The e-value represents the number of random (i.e. unrelated) hexapeptides expected to achieve a score at least as high as the one reported by chance alone.
- f. Output is sent to the web browser for display (see next section).

2.3.4.3 iLIR: Results page

The results are presented in two formats: a graphical illustration of the different motif regions spanning the protein sequence (**Figure 10 - A**), and a series of tables that provide extended information about the identified regions (**Figure 10 - B**). The graphical representation of the protein domains is generated using the **domains graphic generator** used by Pfam (https://pfam.xfam.org/generate_graphic), which provides a clean and familiar depiction to most users. The coloured representation of the various domains in the schematic has as follows: any Pfam domains are displayed in orange, while domains known to be associated with specific classes of selective autophagy LIRCPs are illustrated in green. Other sequence features reported by SMART/PFAM, such as low complexity regions—blue boxes are displayed along the sequence, with detected xLIR-motifs painted in magenta.

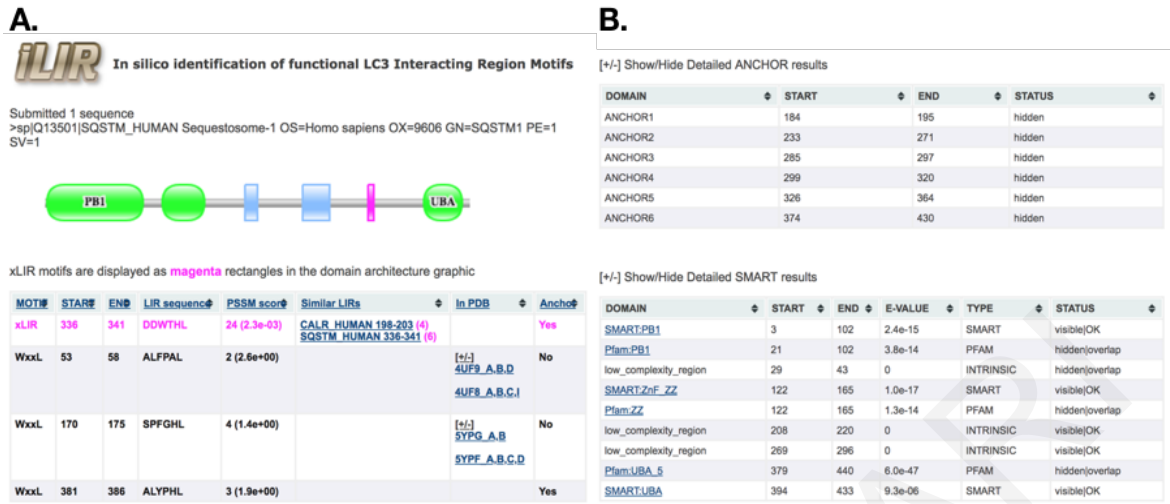


Figure 10. iLIR results page.

The output page of the human SQSTM1 (Uniprot accession: Q13501) is displayed. A graphical representation of the identified domains (A) is accompanied with detailed results from ANCHOR and SMART searches (B). By moving the mouse over any domain/feature on the graphic, a pop-up tip displays further information. It is in the user's discretion if the tables containing further information regarding ANCHOR and SMART regions will remain hidden or not.

For a simple resource like iLIR, the minimal technologies used are sufficient for provisioning the required functionality.

2.3.4.4 iLIR: examples page

To gain a better understanding on what kind of output a user may expect to get from the iLIR server, we have compiled a simple "examples page" (available at the URL: <http://repeat.biol.ucy.ac.cy/iLIR/examples.html>) with static web-pages containing hyperlinks to pre-ran iLIR results on all protein sequences mentioned in Table 1 (Figure 11). For simplicity, the results for WxxL-motifs have been omitted from these pages.



In silico identification of functional LC3 Interacting Region Motifs

Experimentally validated LIR-motif containing proteins, taken from [Alemu et al., 2012](#), analysed by iLIR. The * symbol indicates proteins which were also included in [Behrends et al., 2010](#) (see below).

ATG13_HUMAN	ATG19_YEAST	ATG1_YEAST	ATG32_YEAST	ATG3_YEAST
ATG4B_HUMAN*	BNI3L_HUMAN	CALR_HUMAN	CBL_HUMAN	CLH1_HUMAN
DVL2_HUMAN	FUND1_HUMAN	FYCO1_HUMAN*	NBR1_HUMAN	OPTN_HUMAN
Q8MQJ7_DROME	Q9SB64_ARATH	RBCC1_HUMAN	SQSTM_HUMAN*	STBD1_HUMAN*
T5311_HUMAN	T53I2_HUMAN	TBC25_HUMAN	TBCD5_HUMAN	ULK1_HUMAN
ULK2_HUMAN				

Additional experimentally validated LIR-motif containing proteins, taken from [Birgisdottir, Lamark and Johansen, 2013](#), analyzed by iLIR.

ATG34_YEAST	C0H519_PLAF7	BNIP3_HUMAN	CACO2_HUMAN	CTNB1_HUMAN
MK15_HUMAN	TAXB1_HUMAN			

Human proteins tested for LIR-dependent interactions with GABARAP and MAP1LC3B, taken from [Behrends et al., 2010](#), analyzed by iLIR.

STK4_HUMAN	STK3_HUMAN	RASF5_HUMAN	NEDD4_HUMAN	A16L1_HUMAN
TFCP2_HUMAN	SF3A1_HUMAN	FNBP1_HUMAN	TBC15_HUMAN	ANFY1_HUMAN
TCPR2_HUMAN	ECHA_HUMAN	NIPS2_HUMAN	ATG5_HUMAN	ATG7_HUMAN
KPCI_HUMAN	EPN4_HUMAN	ATG3_HUMAN	DYXC1_HUMAN	NEK9_HUMAN
UBA5_HUMAN	TBD2B_HUMAN	KBTB6_HUMAN	IPO5_HUMAN	NCOA7_HUMAN
KAP0_HUMAN	GYS1_HUMAN	KBTB7_HUMAN	ATG2A_HUMAN	FAN_HUMAN

Figure 11. The full collection of pre-ran examples as they appear on the iLIR website.

The various autophagy proteins are listed by UniProt ID and organised by the source literature in 3 distinct sections as in **Table 1**.

2.4 Conclusions

The work presented in this chapter resulted in the development and provision to the scientific community of a new web resource for the identification of novel LIR-motifs in putative proteins of the autophagic apparatus. iLIR, although nowadays is not the sole available resource, it was the first of its kind when launched in late 2013. A couple of years later the hfAIM server was developed, however with limited usage so far, if judged by the number of citations to the respective paper.

Retrospectively, we speculate that the simplicity of the user interface, combined with the uniqueness of the iLIR web server, has attracted tens of thousands of submissions since the server became available online. In particular, more than 70,000 sequences have been submitted to the iLIR server since becoming publicly available online (fall 2013 till fall

2018). Moreover, we assume that the comprehensive output provided by the iLIR web server provides information that can easily be utilized by experimental biologists aiming to decipher the modes of interaction of putative Atg8/LC3 binding proteins. The detailed output of iLIR provides orthogonal evidence that can be related to structural (e.g. ANCHOR predictions, PDB-homologs) and functional (e.g. SMART/PFAM domains) properties of examined protein sequences. Consequently, the iLIR server has driven the experimental discovery of several new instances of functional LIR-motifs, as seen in a number of papers (<http://goo.gl/yzGUFe>) citing our original publication (Kalvari et al. 2014).

Despite the fact that iLIR was immediately proven to be a useful resource for autophagy researchers and is being continuously used by researchers all over the world, we can already think of improvements for enhancing its performance and providing novel features for an improved user experience.

First, methodological developments may increase the predictive performance of iLIR, thus streamlining efforts for the efficient characterisation of novel autophagy receptor and adaptor proteins. With the current trend of deep machine learning architectures and their applications in several sequence analysis problems in bioinformatics and computational biology (Singh et al. 2018; Wei et al. 2018) this might look like a straightforward option. However, the currently small amount of well characterized data for functional LIR-motifs makes such a scenario sound premature. Nevertheless, our group is already performing preparatory work, where the existing literature corpus on LIR-motifs is manually analyzed for cataloging hopefully all known functional LIR-motifs. This effort will be further assisted by custom, semi-automated biomedical literature mining tools, currently under development in our group (Chadjichristofi and Promponas, work in progress) to extract available information from publications in XML or PDF format. This information needs to be cleansed (remove unrelated instances), extracted from text (and independently validated on its accuracy) and (possibly) organized in a database until we come up with a large enough data set for training and validating machine learning schemes. Before reaching the desired volume, these data may be used for a thorough evaluation of future algorithms performing this task.

Second, the iLIR server could provide richer options and a more interactive graphical user interface. Some novel features we consider for expanding the iLIR server include the possibility of providing a number of different output formats and report alternatively defined LIR-motifs (e.g. hfAIM regular expressions). In particular, more powerful and modernized

technologies should be put into practice, such as REACT (<https://reactjs.org/> - a Javascript library for the construction of user interfaces) and perhaps Django REST framework (<https://www.django-rest-framework.org/>) for the development and provisioning of additional services.

Third, based on several requests made by users of the system, the option to execute batch runs (e.g. scanning a complete proteome) is being taken into consideration. In fact, based on a preliminary analysis of the iLIR server logs, a large fraction of the sequence submissions seems to originate from automated software queries. In addition, in several cases, we have been directly contacted by individual researchers to assist with the analyses of complete proteomes and other large datasets. Possible implementations would be from a simple standalone toolkit with a basic CLI (code distribution with appropriate licence) made available through code versioning systems such as GitHub or Bitbucket, or a more “official” programmatic access to the resource via a REST API, to more advanced and modernised infrastructures employing Cloud technologies (Markstedt 2017; Novella et al. 2018) (e.g. running iLIR as a containerised application on a Kubernetes cluster, with access through user accounts).

3 Intrinsic Disorder as a means for the identification of genuine LIR-motifs

3.1 Preface

3.1.1 Intrinsically disordered proteins

Intrinsically disordered proteins (IDPs) are proteins with no stable secondary or tertiary structure that do not conform to the traditional paradigm of proteins folding into a unique stable conformation (Wright & Dyson 1999). IDPs have been intensively studied during the last two decades and an increasing amount of knowledge accumulates regarding to their possible functions (Wright & Dyson 2015; Dyson & Wright 2005; Oldfield & Dunker 2014; Darling & Uversky 2018). In several cases, a single protein may contain both globular (i.e. well-folded) and disordered (i.e. unstructured) domains.

A large number of prediction tools have been developed to predict intrinsically disordered regions (IDRs) from sequence information (Oldfield & Dunker 2014). In addition, a number of other resources focusing on intrinsic disorder in proteins have been available, as for example DisProt (Piovesan et al. 2017), DIBS (Schad et al. 2018), MobiDB (Piovesan et al. 2018), FuzDB (Miskei et al. 2017).

It is often the case that Short Linear Motifs (SLIMs), such as the LIR-motif, are found within IDRs (Davey et al. 2012), with the flexibility of the disordered region facilitating the motif interaction to a globular partner. Having observed that ANCHOR predictions were very successful in eliminating a significant number of false positives detected by iLIR (Kalvari et al. 2014), we set to investigate whether predictions of IDRs could be used to enhance the discrimination of functional LIR-motifs.

3.2 Data and Methods

3.2.1 Data

3.2.1.1 Sequences

The sequences used in this study are updated versions of the proteins listed in **Table 1** (**Table 3** in this document) from Kalvari et al (Kalvari et al. 2014). As a quality assurance measure all sequences were re-downloaded from UniProt Knowledgebase (The UniProt Consortium

2018) using UniProt accessions (<https://www.uniprot.org/>) and saved in flat files in FASTA format.

3.2.1.2 Disorder Data

Disorder data was obtained from MobiDB v.3.0.0 (<http://mobidb.bio.unipd.it>), a database of protein disorder and mobility annotations (Piovesan et al. 2018). MobiDB incorporates protein disorder data from various databases, which groups them in three categories: **DB** - manually curated disorder data extracted from DisProt (Piovesan et al. 2017), FuzDB (Miskei et al. 2017) and UniProt (The UniProt Consortium 2018) databases, **Predicted** - an ensemble of predicted data from tools like DisEMBL (Linding, Jensen, et al. 2003), ESpritz (Walsh et al. 2012), GlobPlot (Linding, Russell, et al. 2003), IUPred (Mészáros et al. 2018), Jronn (Yang et al. 2005), VSL2b (Peng et al. 2006) with long disorder annotation calculated using MobiDB-lite (Necci et al. 2017), and finally **Indirect** - structural disorder descriptions collected from PDB (Rose et al. 2015) structures.

To compare the power of an aggregating resource like MobiDB as opposed to stand alone predictors, disorder regions were also computed using the SPOT-disorder webserver (<http://sparks-lab.org/server/SPOT-disorder>) (Hanson et al. 2017) (whose authors claim that it performs in par with the top IDP prediction methods), and the most recent release of IUPred (IUPred2A - <https://iupred2a.elte.hu>) (Mészáros et al. 2018) using its command line interface (CLI). ANCHOR2 disordered binding regions were recomputed alongside to evaluate the potential of this revised version. Data extraction as well as the determination of disorder/LIR overlaps was exploited programmatically using explicit software developed in python 2.7.

3.2.1.3 New autophagy proteins and LIR-motifs

More recent studies, in an attempt to further characterise members of the autophagic machinery, introduced new proteins and LIR-motifs (Xie et al. 2016; Rogov et al. 2017; Svenning et al. 2011), which also gave rise to the generation of new LIR prediction tools (Xie et al. 2016). To examine how our optimal methods would behave on a new dataset, we manually selected a small number of samples from the papers of Rogov (Rogov et al. 2017) and Svenning et al. (Svenning et al. 2011) that were not included in our previous experiments. The protein sequences were downloaded from UniProt (The UniProt

Consortium 2018) in FASTA format using Gene ids or simple keywords like “FIP200”. Start and end positions of the LIR-motifs were extracted from the papers and analysed using the iLIR webserver (Kalvari et al. 2014). The samples along with other computationally produced features are summarised in **Table 7**.

3.2.2 Methods

3.2.2.1 Identification of LIR-disorder region overlaps using MobiDB data

Overlaps were computed for each of the 96 motif regions depicted in **Table 7** using custom-made software called *dizscan* (see supplement 7.1). The algorithm takes as input a tab delimited file with each line mapping the UniProt accession numbers of each protein and their corresponding LIRs (sequence, start-end points), along with their experimental status: *verified*, *unverified*. For each line in the input file *dizscan* extracts disordered regions from MobiDB on the fly (using the provided REST API), using UniProt accession numbers to access the data. MobiDB data type can be specified using the option **--type** followed by one of the keywords **all**: incorporates all three types of data indirect, predicted and curated, or **indirect**: MobiDB derived data only, **predicted**: MobiDB predictions or **curated**: to take into account regions deposited in disorder databases by expert curators.

The second step was to search for overlaps with the LIR sequences by taking into account their start and end positions on the peptides. Tracing of disordered residues in the LIR motifs was accomplished by applying simple hashing techniques with the exploitation *hash* data structures. Each LIR motif is represented by a hash, where start-end positions work as keys whose associated values are characters. The characters are in agreement with MobiDB’s naming scheme, where ‘S’ is for structured, ‘D’ is used to denote disorder. All values in the hashes are initialized with question marks ‘?’, to represent an unknown primary state.

Disorder overlaps are determined dynamically, meaning that the disorder state of each value in the dictionary may change over time. There is also a moderate greediness towards disorder, such that it only allows the transitions S -> D, ? -> S, ? -> D. This means that once a residue has been labelled as disordered (‘D’) its status cannot be changed to structured ‘S’ or back to unknown ‘?’, this way preserving as much disorder information as possible.

Finally, the information concealed within each hash is converted to what we hereby call a *disorder string* (**dSTR**), with the same length as the motif sequence. Disorder strings are

used as a simple means of visualisation giving further insight on the per residue disorder status. For instance, the LIR-motif **DDWTHL** of human p62 (UniProt accession: Q13501) in positions 336-341 is classified as completely disordered with a dSTR **DDDDDD** both for curated and predicted data. On the other hand, the resulting dSTR from indirect regions was **??????**, suggesting that no structural information was available for that particular region in MobiDB or that in presence of disorder regions none of those overlapped with the LIR-motifs. In several cases, Indirect, Predicted and Curated data provided by MobiDB can be conflicting. One such case is depicted in **Figure 12**. The flowchart in **Figure 13** provides a graphical representation of the algorithm.

To assess the level of disorderliness of each LIR motif, the percentage of disorder was calculated based on the frequency of ‘D’ characters in the output string. Sensitivity, specificity, accuracy, balanced accuracy (Baldi et al. 2000) and F1-score (Lipton et al. 2014), were calculated on the set of 96 LIR motifs (verified and unverified) and at 6 incremental cutoffs of 16% (1 residue), 33% (2 residues), 50% (3 residues), 66% (4 residues), 83% (5 residues) and 100% (6 residues) (**Table 7**).

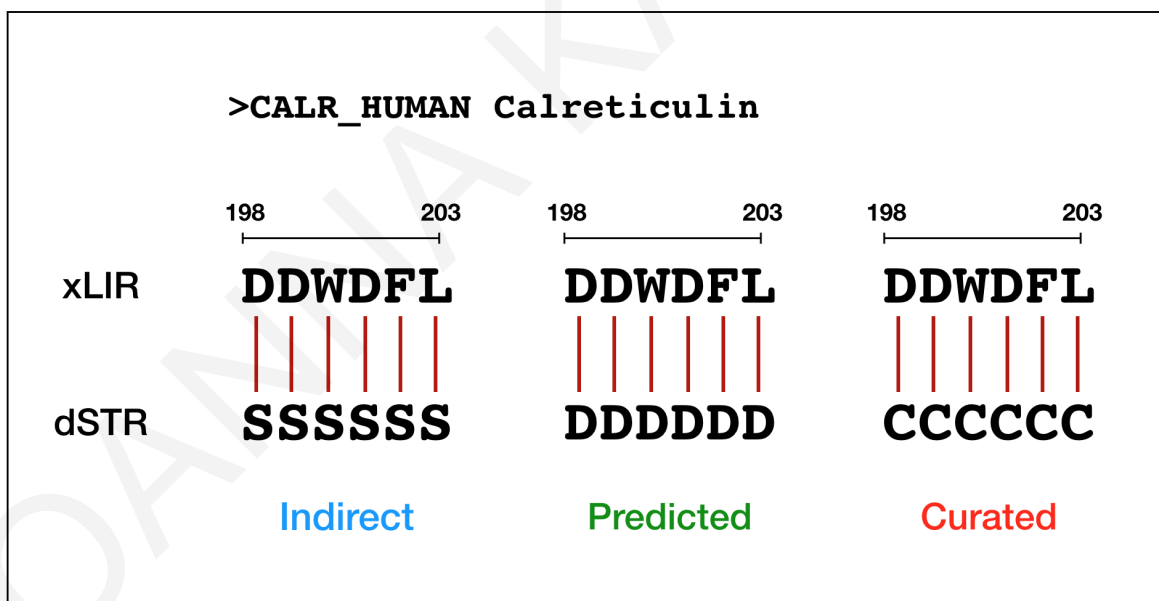


Figure 12. Calculation of disorder in Calreticulin LIR-motif DDWDFL at positions 198-203.

The leftmost schematic shows the disorder string (dSTR) using MobiDB’s Indirect regions (structural data) and the rightmost the dSTR obtained from curated data, with the predicted one in between. This particular example makes apparent how difficult it is to come to a conclusion when dSTRs among the various types of data are contradictory. The ‘C’ characters in the curated dSTR denote conflict.

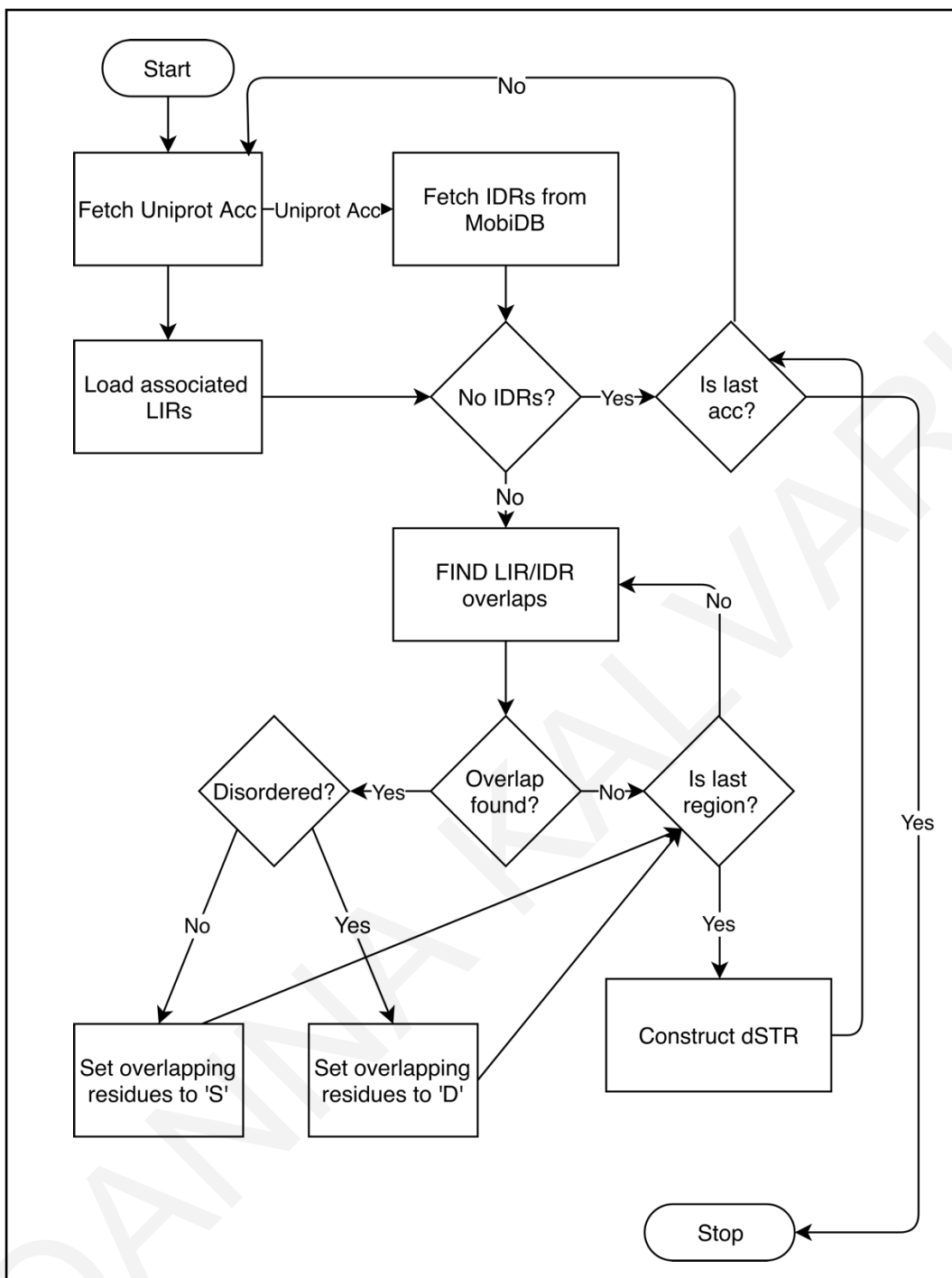


Figure 13. The flowchart of the *dizscan* algorithm.

The above flowchart is a graphical representation of the *dizscan* algorithm described in detailed herein. For the generation of the flowchart we used draw.io software (<https://www.draw.io>)

3.2.2.2 Calculating MobiDB consensus disorder

Alongside MobiDB's consensus data, we devised another script called *consensus_disorder_calculator* (see supplement 7.2), which combines all computed dSTRs regardless of their data type origin (predicted, indirect, curated). This is somewhat a "binary" calculator meaning that each residue position can only be assigned one of Disorder ('D') or Structured ('S') tags, while the initial '?' characters are being ignored.

Visualise a multiple sequence alignment of all dSTRs, the idea is to identify between the maximum count of Ds and Ss in each column and assign that as the final indicator of structure or disorder for that specific position. For that purpose, the algorithm starts by loading all dSTRs in a unified structure - a hash - mapping all LIR accessions to their corresponding list of pre-calculated dSTRs. The next step is to take each individual LIR accession and construct temporary hashes, with key-value pairs that will serve as counters for the 'D' and 'S' characters. For instance, the residue positions 336-341 of the LIR sequence of SQSTM_HUMAN would be the keys, each of which is associated with a nested "binary" hash keeping record of the occurrence of Ds and Ss, each initially set to zero: 0.

The two final steps include the generation of the consensus dSTR (cdSTR) and the computation of the disorder percentage (**Figure 14**). Having computed the counts of Ds and Ss in each column, the one with the highest value (majority rule) is appointed as the disorder status in that column. The procedure continues until all columns have been evaluated resulting to the final consensus disorder string (cdSTR). Ties are resolved in a conservative fashion making a decision in favour of ordered structure ('S'). This also compensates our previous greediness towards favouring disorder during the construction of dSTRs. Finally, the percentage of disorder is calculated based on the frequency of 'D' characters in the cdSTR exactly as described in the previous section.

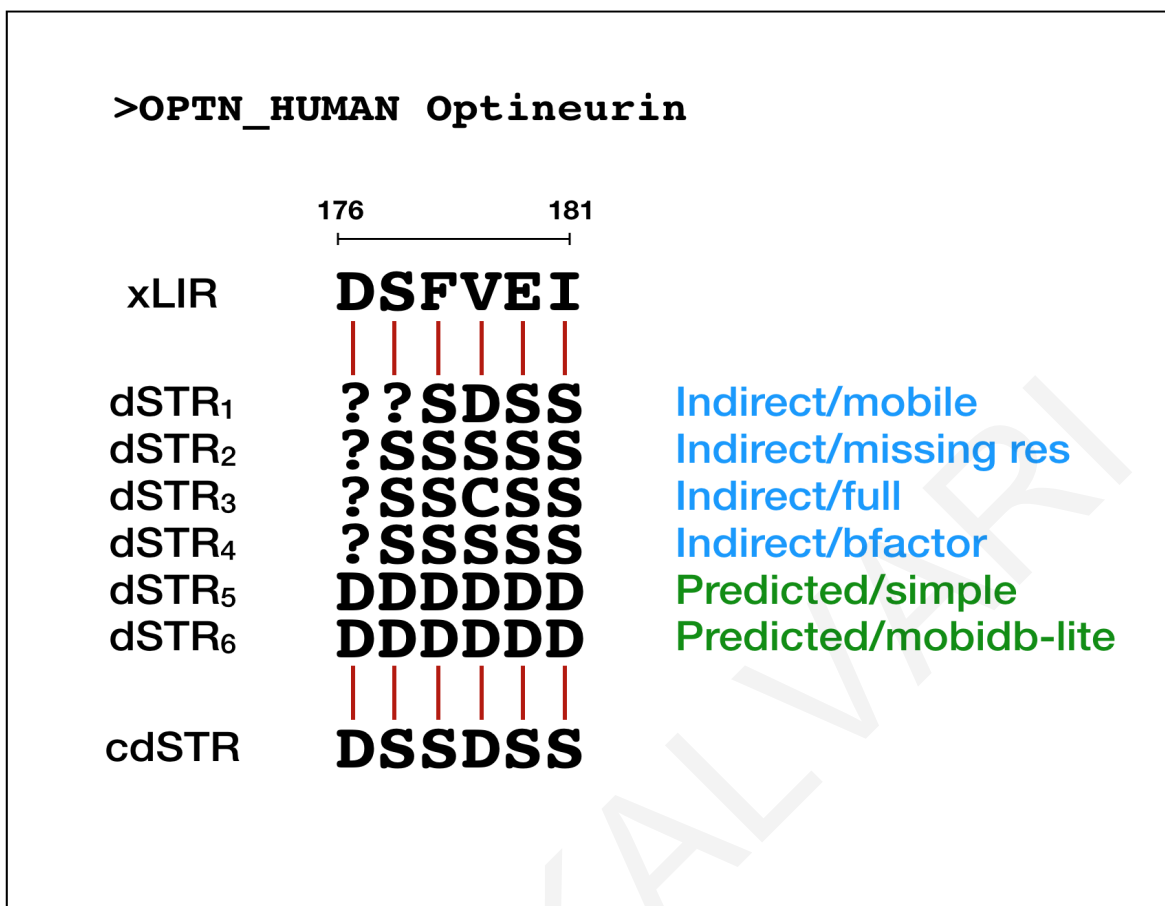


Figure 14. Construction of the consensus disorder string (cdSTR) of the LIR-motif of Optineurin.

In this particular case the consensus disorder string derived from two types of data: Indirect/structural and Predicted. Once again ‘D’ denotes disorder and ‘S’ structure, whereas ‘C’ is used by MobiDB to represent conflicts among resources or methods and are handled as missing data (‘?’). A multiple sequence alignment of the precalculated dSTRs shows variability in each column. The final cdSTR is the majority vote between ‘D’ and ‘S’ states, ignoring missing information. Under this scheme, optineurin cdSTR appears to have two disordered residues at positions 176 and 179, while the rest of the peptide is structured.

As another “consensus-like” but less radical approach, was the selection of the dSTR with the highest disorder. This was achieved by applying a rather rudimentary method which consisted of grouping together all dSTRs belonging to a specific LIR-motif and selecting the one with the maximum disorder percentage (MAX(all)). The findings of the aforementioned methods are juxtaposed (Table 7) and thoroughly analysed in the following sections.

3.2.2.3 Incorporating IUPred2A/Anchor2 disorder predictions

In conjunction with MobiDB consensus disorder data, we scanned our datasets for intrinsic disorder regions using IUPred2A, an intrinsic disorder predictor which identifies disorder in proteins using an energy estimation approach to calculate the interaction potential of amino acids, by capturing the physicochemical properties of IDPs (Mészáros et al. 2018).

IUPred2A predictions of disordered regions were generated for all 52 proteins listed in **Table 7**, and disorder overlaps were identified with a new explicitly developed python script (2.7) called *anchor2_scanner*. Prior to scanning, disordered regions were determined in all proteins using the command line version of IUPred2A (<https://iupred2a.elte.hu>). The output files were then supplied as input to *anchor2_scanner* (see supplement 7.3).

The algorithm works in a similar manner as *dizscan* - making use of has structures - but the scanning process is significantly simpler as there only can be one match for each LIR-motif, whereas the case of MobiDB - dealing with regions from multiple resources - was slightly more challenging.

Here, a residue belonging to a LIR-motif is tagged as disordered ('D') if its IUPred2A score is ≥ 0.5 , otherwise the residue is considered to be structured ('S'). *anchor2_scanner* follows the same notion as *dizscan* constructing a **dSTR** with the final disorder percentage calculated according to the frequency of 'D' and 'S' characters in the string. In addition to IUPred2A default score (0.5) disorder was also computed, capturing disorder at lower values of 0.2-0.4. Performance metrics were once again calculated for the 6 different thresholds of 16%, 33%, 50%, 66%, 83% and 100% disorder. Disorder binding regions were computed using the **--anchor2** option, while **--iupred2** option - as the name suggests - scans for disorder regions in general.

3.2.2.4 Annotating LIR-motifs with disorder using SPOT-disorder

Since MobiDB predicted data also incorporate predictions from IUPRED (Mészáros et al. 2018), we wanted to compare our results to a completely independent tool. For that reason we turned to a newly published disorder prediction tool SPOT-disorder (Hanson et al. 2017) that employs contemporary methods, deep bidirectional long short-term memory recurrent neural networks. In their paper Hanson J et al. (Hanson et al. 2017) showcase that their

algorithm supersedes all other methods compared in their benchmark, including those comprised in MobiDB.

To evaluate the performance of this method on our dataset, we searched our 52 proteins for disorder regions using SPOT-disorder webserver (Hanson et al. 2017) and processed the output in a similar manner as the previous methods with the notation followed by SPOT-disorder (see supplement 7.4). Similarly, SPOT-disorder marks any disordered residues with ‘D’ characters but uses ‘O’ for order instead of ‘S’ for structure.

3.2.2.5 Quality assessment of the predictions

To evaluate the performance of our disorder prediction strategies and in order for the results to be comparable to what discussed in (Kalvari et al. 2014), we followed an analogous approach by calculating the numbers of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), examining disorder at six incremental levels of 16% (1 residue), 33% (2 residues), 50% (3 residues), 66% (4 residues), 83% (5 residues) and 100% (6 residues) disorder. In particular, to evaluate the effectiveness of X% disorder in a given sequence, a LIR-motif supported by experimental validation is considered a true positive (TP) if its dSTR disorder percentage is $\geq X\%$. In the same setting a LIR-motif is classified as TN if there is no experimental evidence in the literature and the disorder level is below X%. In a similar manner, we consider as a FP a LIR-motif with no experimental evidence and disorder of X% or higher and as FN we label the verified LIRs with predicted disorder lower than X%.

With respect to the preceding assumptions and in accordance with the work presented in the previous chapter, we evaluated our methods using the same metrics in addition to the following 2 metrics:

$$Precision = \frac{TP}{TP + FP}$$

$$F1 - score = \frac{2TP}{2TP + FP + FN}$$

3.4 Results

Disorder data retrieved from MobiDB were programmatically examined for overlaps with the 96 LIR-motifs listed in **Table 3** (see code supplement 7.2). Disorder was computed using the start-end positions of each LIR-motif. To assess the power of the various predictive methods we evaluated the level of disorder at meaningful thresholds reflecting disorder at residue scale. For instance, a LIR-motif with 1 disordered residue corresponds to 16% disorder, 2 disordered residues to 33% disorder and so forth up to a 100% disorder indicating a completely disordered peptide. Along these lines, values for each of the quality assessment metrics were generated at 16%, 33%, 50%, 66%, 83% and 100% disorder by utilizing the information encapsulated in the dSTRs and cdSTRs for the consensus scheme. The findings of our different strategies are discussed below.

Uniprot Id	MOTIF		Position	Verified	cLIR	xLIR	Anchor2	PSSM score (e-value)	cdSTR	Disorder percentage (%)
	UniProt Accession	Sequence								
Alemu et al. (Alemu et al. 2012)										
ATG13_HUMAN	O75143	EGFQTV	166-171	0	0	1	0	11 (1.5E-01)	SSSSSS	0
	O75143	DDFVMI	442-447	1	1	1	1	20 (8.4E-03)	DDDDDD	100
Atg19_YEAST	P35193	LTWEEL	410-415	1	0	1	0	18 (1.6E-02)	DDSSSS	33
Atg1_YEAST	P53104	REYVVV	427-432	1	0	1	1	14 (5.7E-02)	DDDDDD	100
Atg32_YEAST	P40458	GSWQAI	84-89	1	0	1	1	17 (2.2E-02)	DSSSSS	16
	P40458	KEYQSL	235-240	0	0	1	0	12 (1.1E-01)	SSSSSS	0
	P40458	LGYILL	524-529	0	0	1	0	10 (2.0E-01)	DDSSDD	66
Atg3_YEAST	P40344	GDWEDL	268-273	1	0	1	0	22 (4.4E-03)	SSSDDD	50
ATG4B_HUMAN	Q9Y4P1	LTYDTL	6-11	1	0	1	0	12 (1.1E-01)	DDSSSS	33
	Q9Y4P1	PMFELV	347-352	0	0	1	0	10 (2.0E-01)	SSSSSS	0
	Q9Y4P1	EDFEIL	386-391	1	1	1	0	17 (2.2E-02)	DDSSSS	33
BNI3L_HUMAN	O60238	SSWVEL	34-39	1	0	1	1	20 (8.4E-03)	DDDDDD	100
	O60238	AEFLKV	183-188	0	0	1	0	10 (2.0E-01)	SSSSSS	0
CALR_HUMAN	P27797	GGYVKL	107-112	0	0	1	0	12 (1.1E-01)	CCCCCC	0
	P27797	DEPTHL	166-171	0	0	1	0	14 (5.7E-02)	CCCCCC	0
	P27797	DDWDFL	198-203	1	1	1	1	26 (1.2E-03)	CCCCCC	0
CBL_HUMAN	P22681	DTYQHL	90-95	0	0	1	0	14 (5.7E-02)	SSSSSS	0
	P22681	LTYDEV	272-277	0	0	1	0	11 (1.5E-01)	SSSSSS	0
	P22681	FGWLSL	800-805	1	0	1	1	18 (1.6E-02)	DDDDDD	100
	P22681	REFVSI	893-898	0	0	1	1	13 (7.9E-02)	SSSSSS	0
CLH1_HUMAN	Q00610	PDWIFL	512-517	1	0	1	0	22 (4.4E-03)	SSSSSS	0
	Q00610	GMFTEL	1315-1320	0	0	1	0	11 (1.5E-01)	SSSSSS	0
	Q00610	EDYQAL	1475-1480	0	0	1	0	16 (3.0E-02)	SSSSSS	0
DVL2_HUMAN	O14641	RMWLKI	442-447	1	0	1	0	18 (1.6E-02)	SSSSSS	0
FUND1_HUMAN	Q8IVP5	DSYEVV	16-21	1	1	1	0	16 (3.0E-02)	SSSSSS	0
	Q8IVP5	GGFLLL	81-86	0	0	1	0	10 (2.0E-01)	SSSSSS	0
FYCO1_HUMAN	Q9BQS8	ADYQAL	644-649	0	0	1	1	15 (4.2E-02)	DDDDDD	100
	Q9BQS8	AVFDII	1278-1283	1	0	1	1	8 (3.9E-01)	SSSSSS	0
NBR1_HUMAN	Q14596	LSFELL	561-566	0	0	1	1	10 (2.0E-01)	SSSSSS	0
	Q14596	EDYIII	730-735	1	1	1	1	17 (2.2E-02)	DDSSSS	33
OPTN_HUMAN	Q96CV9	DSFVEI	176-181	1	1	1	1	15 (4.2E-02)	DSSSSS	16
Q8MQJ7_DROME	Q8MQJ7	ADYLSV	96-101	0	0	1	0	14 (5.7E-02)	SSSSSS	0
	Q8MQJ7	DDFVLV	389-394	1	1	1	1	17 (2.2E-02)	DDDDDD	100

Q9SB64_ARATH	Q9SB64	RVWVLI	479-484	0	0	1	0	15 (4.2E-02)	SSSSSS	0
	Q9SB64	SEWDPI	659-664	1	0	1	0	20 (8.4E-03)	SSSSSS	0
RBCC1_HUMAN	Q8TDY2	FDFTETI	700-705	1	0	1	1	17 (2.2E-02)	DDDDDD	100
SQSTM_HUMAN	Q13501	DDWTHL	336-341	1	0	1	1	24 (2.3E-03)	DDDDDD	100
STBD1_HUMAN	Q95210	EEWEMV	201-206	1	1	1	0	21 (6.1E-03)	DDDDDD	100
T53I1_HUMAN	Q96A56	DEWILV	29-34	1	1	1	1	20 (8.4E-03)	DDDDDD	100
T53I2_HUMAN	Q8IXH6	DGWLII	33-38	1	0	1	1	21 (6.1E-03)	DDDDDD	100
TBC25_HUMAN	Q3MII6	EVYLSL	95-100	0	0	1	0	8 (3.9E-01)	SSSSSS	0
	Q3MII6	EDWDII	134-139	1	1	1	0	24 (2.3E-03)	DDDDDD	100
TBCD5_HUMAN	Q92609	KEWEEL	57-62	1	0	1	0	20 (8.4E-03)	SSSSSS	0
	Q92609	DDFILI	713-718	0	1	1	1	17 (2.2E-02)	DDDDDD	100
	Q92609	SGFTIV	785-790	1	0	1	1	11 (1.5E-01)	DDDDDD	100
ULK1_HUMAN	Q75385	DDFVMV	355-360	1	1	1	1	19 (1.2E-02)	DDDDDD	100
ULK2_HUMAN	Q8IYT8	DDFVLV	351-356	1	1	1	1	17 (2.2E-02)	DDDDDD	100
Birgisdottir et al. (Birgisdottir et al. 2013)										
ATG34_YEAST	Q12292	KVYEKL	194-199	0	0	1	0	8 (3.9E-01)	SSSSSS	0
	Q12292	FTWEEI	407-412	1	0	1	0	20 (8.4E-03)	DDDDDD	100
BNIP3_HUMAN	Q12983	GSWVEL	16-21	1	0	1	1	19 (1.2E-02)	DDDDDD	100
	Q12983	AEFLKV	159-164	0	0	1	0	10 (2.0E-01)	DDDDDD	100
C0H519_PLAF7	C0H519	NDWLLP	103-108	1	0	0	0	12 (1.2E-02)	SSSSSS	0
CACO2_HUMAN	Q13137	FMWVTL	72-77	0	0	1	0	20 (8.4E-03)	SSSSSS	0
	Q13137	DILVV	132-136	1	0	0	0	0 (0)	SSSSSS	0
CTNB1_HUMAN	P35222	SHWPLI	502-507	1	0	0	0	11 (1.5E-01)	SSSSSS	0
MK15_HUMAN	Q8TD08	RVYQMI	338-343	1	0	1	1	10 (2.0E-01)	DDDDDD	100
TAXB1_HUMAN	Q86VP1	DMLVV	139-143	1	0	0	0	0 (0)	DDDDDD	100
	Q86VP1	ADFDIV	514-519	0	0	1	1	15 (4.2E-02)	DDDDDD	100
Behrends et al. (Behrends et al. 2010)										
A16L1_HUMAN	Q676U5	DEYDAL	164-169	0	1	1	1	16 (3.0E-02)	DDDDSS	66
ATG2A_HUMAN	Q2TAZ0	PEYTEI	534-539	0	0	1	0	13 (7.9E-02)	DDSDDD	83
	Q2TAZ0	EVYESI	828-833	0	0	1	0	9 (2.8E-01)	SSSSSS	0
	Q2TAZ0	LEFLDV	1090-1095	0	0	1	0	9 (2.8E-01)	SSSSSS	0
ATG7_HUMAN	Q95352	SSFQSV	258-263	0	0	1	0	10 (2.0E-01)	SSSSSS	0
DYXC1_HUMAN	Q8WXU2	AVFLSL	16-21	0	0	1	0	6 (7.4E-01)	SSSDDD	50
	Q8WXU2	AMWETL	81-86	0	0	1	0	19 (1.2E-02)	SSSSSS	0
ECHA_HUMAN	P40939	AVFEDL	447-452	0	0	1	0	7 (5.3E-01)	SSSSSS	0
FAN_HUMAN	Q92636	ESFEDL	600-605	0	1	1	0	12 (1.1E-01)	DDDDDD	100
	Q92636	LVWDLI	869-874	0	0	1	0	13 (7.9E-02)	SSSSSS	0
IPO5_HUMAN	Q00410	ETYENI	31-36	0	1	0	0	11 (1.5E-01)	DDDDDD	100
	Q00410	DGWEFV	655-660	0	0	1	0	21 (6.1E-03)	DDDDDD	100
	Q00410	LSWLPL	997-1002	0	0	1	0	16 (3.0E-02)	SSSSSS	0
KAP0_HUMAN	P10644	EEFVEV	310-315	0	1	1	0	13 (7.9E-02)	SSSSSS	0
KBTB6_HUMAN	Q86V97	ESFEVL	120-125	0	1	1	0	13 (7.9E-02)	SSSSSS	0
KBTB7_HUMAN	Q8WVZ9	ESFEVL	120-125	0	1	1	0	13 (7.9E-02)	SSSSSS	0
NCOA7_HUMAN	Q8NI08	AEYDKL	185-190	0	0	1	0	13 (7.9E-02)	DDDDDD	100
	Q8NI08	GEWEDL	308-313	0	0	1	0	19 (1.2E-02)	DDDDDD	100
	Q8NI08	DDFVDL	414-419	0	1	1	1	18 (1.6E-02)	DDDDDD	100
	Q8NI08	KSWEII	745-750	0	0	1	0	19 (1.2E-02)	DDDDDD	100
NEDD4_HUMAN	P46934	SEYIKL	410-415	0	0	1	0	13 (7.9E-02)	DDDDDD	100
	P46934	PGWVVL	589-594	0	0	1	1	19 (1.2E-02)	DDDDDD	100
	P46934	ESFEEL	1296-1301	0	1	1	0	13 (7.9E-02)	SSSSSS	0
SF3A1_HUMAN	Q15459	PEFEFI	148-153	0	0	1	0	13 (7.9E-02)	DDDDDD	100
STK3_HUMAN	Q13188	EVFDVL	25-30	0	0	1	0	9 (2.8E-01)	SSSSSS	0
	Q13188	GDFDFL	435-440	0	0	1	1	16 (3.0E-02)	DSSSSS	16
STK4_HUMAN	Q13043	EVFDVL	28-33	0	0	1	0	9 (2.8E-01)	SSSSSS	0
	Q13043	GDYEFLL	431-436	0	0	1	1	17 (2.2E-02)	SSSSSS	0
TBC15_HUMAN	Q8TC07	AEWDMV	96-101	0	0	1	0	20 (8.4E-03)	DDDDDD	100
	Q8TC07	PGFEVI	295-300	0	0	1	0	12 (1.1E-01)	DDSDSS	50
	Q8TC07	FSFLDI	540-545	0	0	1	0	11 (1.5E-01)	SSSSSS	0
TBD2B_HUMAN	Q9UPU7	EEWELL	252-257	0	1	1	1	20 (8.4E-03)	DDDDDD	100
TCPR2_HUMAN	O15040	GDYIAV	45-50	0	0	1	0	14 (5.7E-02)	SSSSSS	0

	O15040	AVFQLV	102-107	0	0	1	0	5 (1.0E-00)	SSSDDD	50
	O15040	AVFVAL	894-899	0	0	1	0	7 (5.3E-01)	SSSSSS	0
	O15040	DEWEVI	1406-1411	1	1	1	0	23 (3.2E-03)	DDDDDD	100
UBA5_HUMAN	Q9GZZ9	SDYEKI	66-71	0	0	1	0	17 (2.2E-02)	DDSSSS	33
	Q9GZZ9	FDYDKV	103-108	0	0	1	0	16 (3.0E-02)	SSSSSS	0

Table 7. A collection of 52 proteins with their experimentally validated LIR-motifs.

Disorder percentage is calculated based on occurrence of ‘D’ characters in cdSTRs. Verified column indicates whether a LIR-motif is functional (verified=1) or non-functional (verified=0), which is a result of literature curation. The values in columns xLIR and Anchor2 indicate whether a LIR-motif is discoverable (value=1) by the tool or not (value=0). PSSM scores and e-values were computed using iLIR webserver (Kalvari et al. 2014).

3.4.1 In seek of the optimal predictive method and disorder threshold

This section focuses solely on consensus disorder data retrieved from MobiDB v.3.0.0 and aims at determining the method that best fits our data. We hereby examine the potential of the three methods MobiDB-simple, Consensus, MAX(all) and disorder being used as another variable in the equation towards discriminating genuine LIR-motifs.

MobiDB makes available two consensus predicted schemes by default, mobidb-lite and simple. Although the two methods aggregate predicted data from the exact same tools and in similar fashion, each approach captures disorder by employing different thresholds and strategies. For instance, mobidb-simple is less stringent by allowing a residue to be appointed as disordered (‘D’) if only half of the tools are agreement ($\geq 50\%$), whereas mobidb-lite is slightly more strict requiring that at least 6 out of 8 tools moving the bar to 75% and up (Damiano Piovesan, personal communication). On top of that mobidb-lite also includes a post-processing step which filters out short regions, therefore further investigation was required to choose the best for our dataset.

With respect to our samples, mobidb-lite lacked information for a small number of LIR-motifs, possibly due to its filtering of short regions. One-on-one comparison to mobidb-simple revealed a rather fixed nature when alternating between thresholds. Meaning that mobidb-lite reached a Balanced Accuracy of about 55% with a very low F1 score of 0.23 (0.27 at max) and did not deviate much from those values. This outcome was also the worst amongst all methods and therefore excluded from further analysis.

The predictive power of the remaining three methods MobiDB-simple, consensus disorder and Max(all) was evaluated at six distinct thresholds of 16%, 33%, 50%, 66%, 83% and

100% disorder encapsulated in disorder strings (dSTRs) and consensus dSTRs (cdSTRs). The results are depicted in **Table 8**.

From balanced accuracy and F1-scores it is evident that MobiDB-simple outperforms the two other methods at all different thresholds, reaching a peak of about **74%** Balanced Accuracy (BACC) and an F1-score of **0.68** at 100% disorder. With respect to the other metrics, we observe an oscillation between the leading method at different thresholds. For example, the consensus method prevails over the other two at the lowest threshold (16%) in terms of specificity, precision as well as Accuracy, and MAX(all) in terms of sensitivity. However, MobiDB-simple persists in achieving the highest BACC and F1 values. Balanced accuracy and F1-score are metrics that balance precision and recall given an uneven set, therefore deemed to be more representative of our dataset and will be used extensively to compare different predictive schemes in the following.

Once we selected MobiDB-simple as the utmost method for disorder prediction in LIR-motifs, the next step was the in-depth exploration of False Positive and False Negative predictions, going through each case one by one (**Table 9**). From these results it becomes apparent that MobiDB-derived data cannot yield a perfect discrimination between functional and non-functional LIRs – at least not as a sole parameter. For instance, MobiDB-simple annotates all non-functional LIRs with a 100% disorder, which under the assumption that a completely disordered peptide is also a functional LIR, falsely classify those as such. Thus, additional parameters (or other proxies to the disorderliness of LIR-motifs) need to be taken into consideration when searching for genuine LIR-motifs.

	1+ residues			2+ residues			3+ residues			4+ residues			5+ residues			6 residues		
	Disorder >= 16%			Disorder >= 33%			Disorder >= 50%			Disorder >= 66%			Disorder >= 83%			Disorder 100%		
	MAX(all)	MobiDB simple	Consensus	MAX(all)	MobiDB simple	Consensus	MAX(all)	MobiDB simple	Consensus	MAX(all)	MobiDB simple	Consensus	MAX(all)	MobiDB simple	Consensus	MAX(all)	MobiDB simple	Consensus
TP	35	30	26	35	30	24	33	30	20	29	30	19	18	29	19	13	29	19
TN	11	32	36	15	32	37	21	33	38	29	36	41	40	38	43	46	40	44
FP	49	28	24	45	28	23	39	27	22	31	24	19	20	22	17	14	20	16
FN	1	6	10	1	6	12	3	6	16	7	6	17	18	7	17	23	7	17
Sensitivity (%)	97.22	83.33	72.22	97.22	83.33	66.67	91.67	83.33	55.56	80.56	83.33	52.78	50.00	80.56	52.78	36.11	80.56	52.78
Specificity (%)	18.33	53.33	60.00	25.00	53.33	61.67	35.00	55.00	63.33	48.33	60.00	68.33	66.67	63.33	71.67	76.67	66.67	73.33
Precision (%)	41.67	51.72	52.00	43.75	51.72	51.06	45.83	52.63	47.62	48.33	55.56	50.00	47.37	56.86	52.78	48.15	59.18	54.29
Accuracy (%)	47.92	64.58	64.58	52.08	64.58	63.54	56.25	65.63	60.42	60.42	68.75	62.50	60.42	69.79	64.58	61.46	71.88	65.63
Balanced Accuracy (%)	57.78	68.33	66.11	61.11	68.33	64.17	63.33	69.17	59.44	64.44	71.67	60.56	58.33	71.94	62.22	56.39	73.61	63.06
F1 - score	0.58	0.64	0.60	0.60	0.64	0.58	0.61	0.65	0.51	0.60	0.67	0.51	0.49	0.67	0.53	0.41	0.68	0.54

Table 8. Disorder results as computed from MobiDB 3.0.0 data.

The three algorithms described in methods were tested for their predictive power on the collection of verified and non-verified LIR-motifs of Alemu et al (Alemu et al. 2012) as illustrated in **Table 7**. Evaluation was carried out based on six metrics: Sensitivity, Specificity, Precision, Accuracy (ACC), Balanced Accuracy (BACC) and F1-score with the top score for each metric represented in bold.

UniProt id	Sequence	Positions	Verified	PSSM (e-value)	xLIR	dSTR	cdSTR
False Positives							
BNIP3_HUMAN	AEFLKV	159-164	0	10 (2.0E-01)	1	DDDDDD	DDDDDD
CALR_HUMAN	GGYVKL	107-112	0	12 (1.1E-01)	1	DDDDDD	CCCCCC
FAN_HUMAN	ESFEDL	600-605	0	12 (1.1E-01)	1	DDDDDD	DDDDDD
FYCO1_HUMAN	ADYQAL	644-649	0	15 (4.2E-02)	1	DDDDDD	DDDDDD
IPO5_HUMAN	ETYENI	31-36	0	11 (1.5E-01)	0	DDDDDD	DDDDDD
IPO5_HUMAN	DGWEFV	655-660	0	21 (6.1E-03)	1	DDDDDD	DDDDDD
KAP0_HUMAN	EEFVEV	310-315	0	13 (7.9E-02)	1	DDDDDD	SSSSSS
NCOA7_HUMAN	AEYDKL	185-190	0	13 (7.9E-02)	1	DDDDDD	DDDDDD
NCOA7_HUMAN	GEWEDL	308-313	0	19 (1.2E-02)	1	DDDDDD	DDDDDD
NCOA7_HUMAN	DDFVDL	414-419	0	18 (1.6E-02)	1	DDDDDD	DDDDDD
NCOA7_HUMAN	KSWEII	745-750	0	19 (1.2E-02)	1	DDDDDD	DDDDDD
NEDD4_HUMAN	SEYIKL	410-415	0	13 (7.9E-02)	1	DDDDDD	DDDDDD
NEDD4_HUMAN	PGWVVL	589-594	0	19 (1.2E-02)	1	DDDDDD	DDDDDD
SF3A1_HUMAN	PEFEFI	148-153	0	13 (7.9E-02)	1	DDDDDD	DDDDDD
STK3_HUMAN	EVFDVL	25-30	0	9 (2.8E-01)	1	DDDDDD	SSSSSS
STK3_HUMAN	GDFDFL	435-440	0	16 (3.0E-02)	1	DDDDDD	DSSSSS
TAXB1_HUMAN	ADFDIV	514-519	0	15 (4.2E-02)	1	DDDDDD	DDDDDD
TBC15_HUMAN	AEWDMV	96-101	0	20 (8.4E-03)	1	DDDDDD	DDDDDD
TBCD5_HUMAN	DDFILI	713-718	0	17 (2.2E-02)	1	DDDDDD	DDDDDD
TBD2B_HUMAN	EEWELL	252-257	0	20 (8.4E-03)	1	DDDDDD	DDDDDD
False Negatives							
CACO2_HUMAN	DILVV	132-136	1	N/A	0	?????	SSSSSS
CLH1_HUMAN	PDWIFL	512-517	1	22 (4.4E-03)	1	??????	SSSSSS
CTNB1_HUMAN	SHWPLI	502-507	1	11 (1.5E-01)	0	??????	SSSSSS
DVL2_HUMAN	RMWLKI	442-447	1	18 (1.6E-02)	1	??????	SSSSSS
FUND1_HUMAN	DSYEVL	16-21	1	16 (3.0E-02)	1	DDDD??	SSSSSS
Q9SB64_ARATH	SEWDPI	659-664	1	20 (8.4E-03)	1	??????	SSSSSS
TBCD5_HUMAN	KEWEEL	57-62	1	20 (8.4E-03)	1	??????	SSSSSS

Table 9. Classification of LIR-motifs using disorder data from MobiDB.

List of LIR-motifs that were falsely categorized as functional LIRs (False Positives) or falsely predicted as non-functional (False Negatives).

With respect to False Negatives what can be observed is that almost all dSTRs are in their initial state ‘?????’ at 0% disorder, apart from the dSTR of FUND1_HUMAN LIR-motif (DSYEVL) at 66% disorder (DDDD??). This is due to the stringent disorder filter of 100% that we apply, but based on the results in **Table 8**, a lower threshold of 66% disorder (4+ residues) would have come with the expense of additional False positives.

3.4.2 Assessing the power of MobiDB over IUPRED2A and SPOT-disorder

MobiDB (Piovesan et al. 2018) is a composite database combining disorder proteomic data and mobility annotations from a wide range of resources including IUPred. In the previous section we explored the predictive power of MobiDB-simple based on a set of metrics which we computed at different thresholds. In this segment, we analyse its potential in opposition to stand alone methods: the newest IUPred, namely IUPred2A (Mészáros et al. 2018) and a newly published tool called SPOT-disorder (Hanson et al. 2017). To evaluate the performance of each tool, we tested their ability to correctly distinguish the genuine LIRs out of the collection of 96 LIRs listed in **Table 7**.

In opposition to ModiDB-simple and SPOT-disorder (used with their default options), IUPRED2A was examined more thoroughly by experimenting with several other scores beyond the default value of 0.5 suggested by its authors. This process revealed that a score of 0.3 is perhaps more suitable for our dataset and was the one selected for further analysis. We hereby evaluate the strength of each method based on the six selected thresholds of 16%, 33%, 50%, 66%, 83% and 100% disorder for all aforementioned metrics. The performance of the three methods are juxtaposed in **Table 10**.

Metrics	1+ residues		2+ residues		3+ residues		4+ residues		5+ residues		6 residues		
	Disorder >= 16		Disorder >= 33		Disorder >=50		Disorder >= 66		Disorder >= 83		Disorder 100		
	IUPRED2A (0.3)	SPOT	IUPRED2A (0.3)	SPOT	IUPRED2A (0.3)	SPOT	IUPRED2A (0.3)	SPOT	IUPRED2A (0.3)	SPOT	IUPRED2A (0.3)	SPOT	MobiDB simple
TP	31	17	29	15	29	13	27	13	26	12	24	12	29
TN	35	46	36	46	38	47	38	48	42	50	42	51	40
FP	25	14	24	14	22	13	22	12	18	10	18	9	20
FN	5	19	7	21	7	23	9	23	10	24	12	24	7
Sensitivity (%)	86.11	47.22	80.56	41.67	80.56	36.11	75.00	36.11	72.22	33.33	66.67	33.33	80.56
Specificity (%)	58.33	76.67	60.00	76.67	63.33	78.33	63.33	80.00	70.00	83.33	70.00	85.00	66.67
Precision (%)	55.36	54.84	54.72	51.72	56.86	50.00	55.10	52.00	59.09	54.55	57.14	57.14	59.18
Accuracy (%)	68.75	65.63	67.71	63.54	69.79	62.50	67.71	63.54	70.83	64.58	68.75	65.63	71.88
Balanced Accuracy (%)	72.22	61.94	70.28	59.17	71.94	57.22	69.17	58.06	71.11	58.33	68.33	59.17	73.61
F1-score	0.67	0.51	0.65	0.46	0.67	0.42	0.64	0.43	0.65	0.41	0.62	0.42	0.68

Table 10. Comparison of IUPRED2A, SPOT and MobiDB.

Performance of MobiDB-simple, IUPRED2A and SPOT-disorder as calculated on the 96 LIR-motifs presented in **Table 7**. Assessment is conducted based on six incremental thresholds 16%, 33%, 50%, 66%, 83% and 100% disorder with optimum values in each test case represented in bold.

Giving emphasis to Balanced accuracy and F1-scores one can notice that IUPRED2A_{0.3} outperforms MobiDB-simple almost at all disorder thresholds apart from the case of complete disorder (100%). That is where MobiDB-simple is once again the best method with a 74% BACC and a 0.68 F1-score. The difference between IUPRED2A and MobiDB-simple is very small with a 72% BACC/0.67 F1-score and a 68% BACC/0.64 F1-score respectively. SPOT appears to be the weakest with a BACC of only 62% and an F1-score of 0.51 calculated at 16% disorder, which continues to gradually downdrift as the level of disorder increases and consequently thrown out of competition.

With respect to the other four metrics SPOT does not fall far behind IUPRED2A_{0.3}, but a very interesting observation is that IUPRED2A_{0.3} is more sensitive, whereas SPOT is more specific and this trend persists at all thresholds.

Coming back to IUPRED2A and MobiDB-simple, one can confidently accept MobiDB-simple as the optimal method overtaking all other methods at all tested thresholds. However, since the difference between the two is not large, we re-assess both for their contribution in a multi-scheme predictor in the sections that follow.

3.4.3 Scrutinizing the potential of disorder binding regions in the determination of genuine LIRs

In 2014, driven by the observation that proteins involved in autophagy are highly abundant in intrinsically disordered regions (Mei et al. 2014), we investigated the possibility of LIR-motifs undergoing a disorder to order conformational change upon binding to Atg8 homologs. For that purpose, we used the ANCHOR software (Meireles et al. 2010) to search for such regions in our collection of proteins. A residue with score over **0.5** (by default) was considered to belong to a disorder binding region with motif classified as such if the constraint that at least 66% of LIR-motif being disordered was met (4 out of 6 residues).

With the release of the new IUPred2A software (Mészáros et al. 2018) a new revised version of the ANCHOR tool became available referred to as ANCHOR2. According to Meszaros et al., ANCHOR2 underwent major revision which lead to better results. This current version has been modified to take into account interactions with globular domains as well as local disorder sequence environment and was re-trained using a new dataset from DIBS database (Schad et al. 2018). From their findings it is evident that ANCHOR2 outperforms its

predecessor in all tested scenarios and for that reason we herein assess its power on our dataset. For this purpose, new *anchors* were generated for all 52 proteins with LIR-motifs that are listed in **Table 7**. We also went a step further by testing disorder prediction at lower scores to investigate whether an alternative could fit our dataset better than the default. We examined three alternative thresholds for ANCHOR2 scores in addition to the default 0.5 used by IUPRED2A 0.2, 0.3 and 0.4, and which pinpointed **0.3** as another candidate for annotating LIR-motif residues with *anchors*.

In order to determine the optimum *anchor* score for our dataset, the two anchor schemes were assessed for their predictive strength in combination with other parameters such as the characterisation of LIR-motifs by xLIR (xLIR=1) and its corresponding PSSM score. Our findings showed that *anchor* predictions with the default score of **0.5 (Anchor2_{0.5})** was more efficient, reaching a Balanced Accuracy of 78% and an F1-score of 0.85 as opposed to a 72% BACC and 0.83 F1 achieved with *anchor* score of 0.3 (**Anchor2_{0.3}**). Consequently **Anchor2_{0.5}** was selected to further compare its efficacy to its former version as contributor to the multi-scheme predictors. The results are depicted in **Table 11**.

	xLIR	xLIR + A	xLIR + A2	xLIR + A + P13	xLIR + A2 + P13	xLIR + A P13	xLIR + A2 P13
TP	27	17	12	15	11	26	26
TN	0	16	18	18	18	11	11
FP	20	4	1	2	1	9	8
FN	0	10	16	12	17	1	2
Sensitivity (%)	100.00	62.96	42.86	55.56	39.29	96.30	92.86
Specificity (%)	0.00	80.00	94.74	90.00	94.74	55.00	57.89
Precision (%)	57.45	80.95	92.31	88.24	91.67	74.29	76.47
Accuracy (%)	57.45	70.21	63.83	70.21	61.70	78.72	78.72
Balanced Accuracy (%)	50.00	71.48	68.80	72.78	67.01	75.65	75.38
F1-score (%)	0.73	0.71	0.59	0.68	0.55	0.84	0.84

Table 11. Comparing the efficacy of ANCHOR and ANCHOR2 on different predictive schemes.

In order for the results to be comparable to those in Table 3 of the iLIR paper (Kalvari et al. 2014) both versions were reassessed on the subset of proteins introduced by Alemu et al, but using the same schemes we studied before. ANCHOR appears to supersede ANCHOR2 in terms of accuracy (ACC), balanced accuracy (BACC) and F1-score in all schemes although in the case of xLIR+AX|P13, the difference in balanced accuracy is only marginal and both versions of the tool give an F1 score of 0.84. One can argue that since the old version of

ANCHOR seems to be better than its successor, that updating to the new version would not come with a positive outcome, however both of the tools give an F1 score of 0.84. If we are to compare the two data-wise, the difference is only an additional FN for ANCHOR2 at the expense of a single FP. Since ANCHOR2 is the latest version and the one that is most likely to be maintained for the foreseeable future, it is also the one we will be including in any further analysis.

3.4.4 Assessing the efficacy of multi-scheme predictors

In our previous work we showed that combining different metrics resulted in more powerful LIR prediction schemes. Supporting evidence is the inclusion of *anchors* and *PSSMs* as an enhancing means for discriminating genuine LIR-motifs reaching a maximum of 75.7% BACC (**Table 5** (Kalvari et al. 2014)). With the refinement of ANCHOR (ANCHOR2), **xLIR+A2|P13** BACC went up to **78%** (~2% improvement) with an F1-score of **0.85**, an even greater better performance compared to MobiDB-simple₁₀₀ with an increased BACC and F1-score by **5%** and **0.17** respectively.

In this final section we investigated the relevance of intrinsic disorder as a predictor of autophagy LC3 interacting regions. Carrying over from the previous section, we examined whether incorporating intrinsic disorder in the multi-scheme logic equation would further improve its predictive power. **Table 12** lists an expanded version of all predictive schemes presented in **Table 5** from Kalvari et al (Kalvari et al. 2014), with the addition of intrinsic disorder. When discussing about multi-scheme predictors we refer to simple logic (e.g. **xLIR+A2+D|P13**) equations that combine binarized LIR-motif descriptors such as *anchors*, *disordered* etc.

	xLIR	xLIR+D	xLIR+A2+D	xLIR+A2 D	xLIR+A2+D P13	xLIR+A2 D P13
TP	27	23	12	23	26	28
TN	0	16	18	16	11	10
FP	20	3	1	3	8	9
FN	0	5	16	5	2	0
Sensitivity (%)	100.00	82.14	42.86	82.14	92.86	100.00
Specificity (%)	0.00	84.21	94.74	84.21	57.89	52.63
Precision (%)	57.45	88.46	92.31	88.46	76.47	75.68
Accuracy (%)	57.45	82.98	63.83	82.98	78.72	80.85
Balanced Accuracy (%)	50.00	83.18	68.80	83.18	75.38	76.32
F1-score	0.73	0.85	0.59	0.85	0.84	0.86

Table 12. Multi-scheme predictors applied on the 47 LIR-motifs collected by Alemu et al.

Assessment of their power in discriminating functional LIR-motifs was conducted based on Sensitivity, Specificity, Precision, Accuracy, Balanced Accuracy (BACC) and F1-score. A2 represents the latest version of ANCHOR, D is for disorder and P13 for pssm scores > 13. The top scores in each row are marked in bold.

Overall our findings suggest that disorder is a good indicator of genuine LIR-motifs reaching a balanced accuracy of 83% for schemes **xLIR+D** and **xLIR+A2|D**, surpassing our previous top score by 5%. Both predictors appear to perform exactly the same with nicely balanced Specificity and Sensitivity of 84% and 82% respectively, suggesting that intrinsic disorder in LIR-motifs is a critical feature. Further evidence that justify this conclusion can be derived by comparing the two aforementioned schemes with predictor **xLIR+A2+D**. This latter case appears to be the weakest of the combinational schemes being in favour of Specificity (~

95%) with a balanced accuracy of 69% and an F1-score of 0.6. Disorder (D) in xLIR+A2|D seems to be overpowering A2, with that particular logical equation giving the exact same results with xLIR+D obliterating *anchors* (A2) completely. However, it is very likely that a larger collection of samples the two tools can better compensate for one another with A2 picking up cases that disorder eludes and vice versa.

Looking into PSSM scores, the inclusion of P13 (PSSM > 13) to the schemes (xLIR+A2|D|P13) seems to have had a negative effect, an outcome which is contradictory to our previous findings. With respect to balanced accuracy, it still performs better with a slight improvement of 0.6% compared to our old optimal case xLIR+A|P13 (Kalvari et al. 2014). Its 7% declination from the 83% of the best scheme (xLIR+A2|D) came with 100% Sensitivity - similar to using xLIR solely - but with a 53% gain in Specificity. Although with a lower balanced accuracy of 76%, this scheme comes with an F1-score of 0.86, which is also the best across all tests performed. This raises the question whether there are other ways to fine tune this multi-scheme predictor to achieve even higher scores.

Building on that notion, the PSSM score is the only descriptor that is still parameterizable, meaning that it is the only one not tested for an optimum value. To look into PSSM alternatives, we computed all metrics on the same dataset we worked with before using the multi-scheme predictor xLIR+A2|D|PX, where X parametrized different PSSM thresholds tested. We captured its performance at six incremental PSSM scores starting from P13, P14 up to P18, where PX denotes a PSSM score > X. The Balanced accuracy for both disorder methods mobidb-simple and IUPRED2A (iupred2) computed at the six distinct PSSM scores is illustrated in **Figure 15**.

xLIR+A2|D|PX BACC captured at different PSSM thresholds

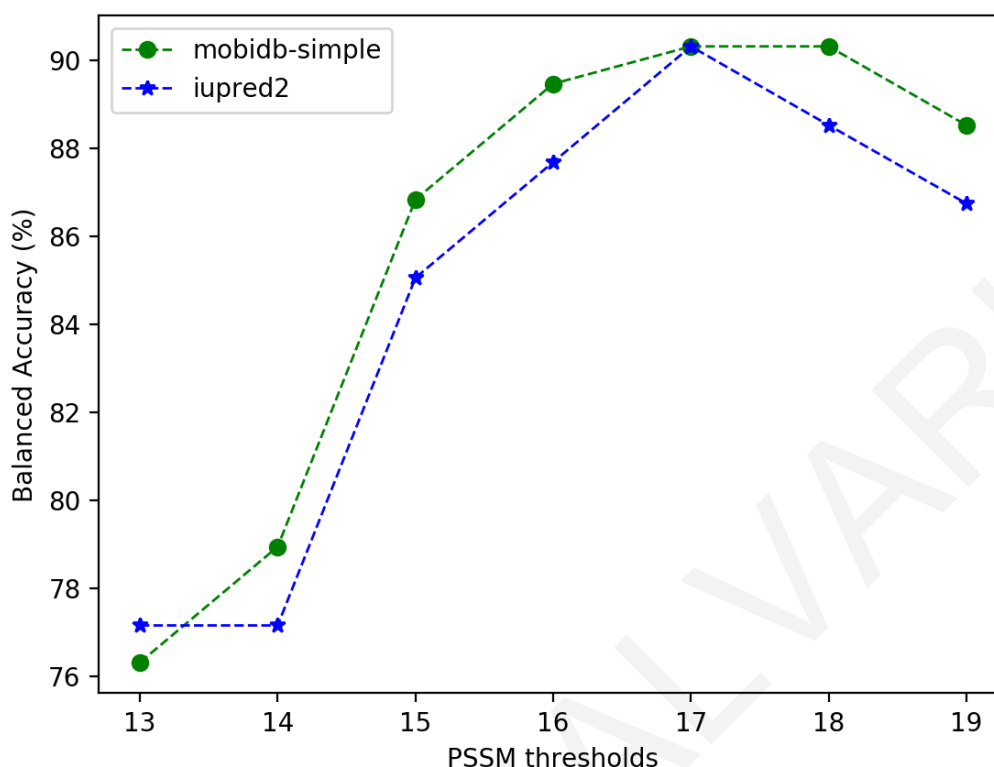


Figure 15. Balanced Accuracy (%) achieved with multi-scheme predictor xLIR+A2|D|PX captured at various PSSM thresholds.

All scores were calculated based on the 47 xLIRs detected in the sequences stemming from the paper of Alemu and colleagues (Alemu et al. 2012). IUPRED2A scores are in blue and mobidb-simple BACC is presented in green.

From the results portrayed in **Figure 15** what is apparent is that, overall, mobidb-simple has a better effect than IUPRED2A when used synergistically in the multi-scheme predictor, although at P13 IUPRED2A seems more favourable with a BACC of 77.16% over the 76.32% achieved with mobidb-simple. At PSSM > 14 the balanced accuracy of mobidb-simple begins to increase and precedes up to P16 (89.47% vs 87.69%). At P17 both tools reach a peak value of 90% BACC, a value that mobidb-simple preserves at P18, followed by a downdrift at P19. This raises another question of whether P17 is the optimum PSSM score. To answer this question, the multischeme predictor xLIR+A2|D|PX was applied on the entire collection of 96 LIRs (Table 1) and all metrics were re-computed at all different PSSM scores. The results are depicted in **Figure 16**.

xLIR+A2|D|PX BACC captured at different PSSM thresholds

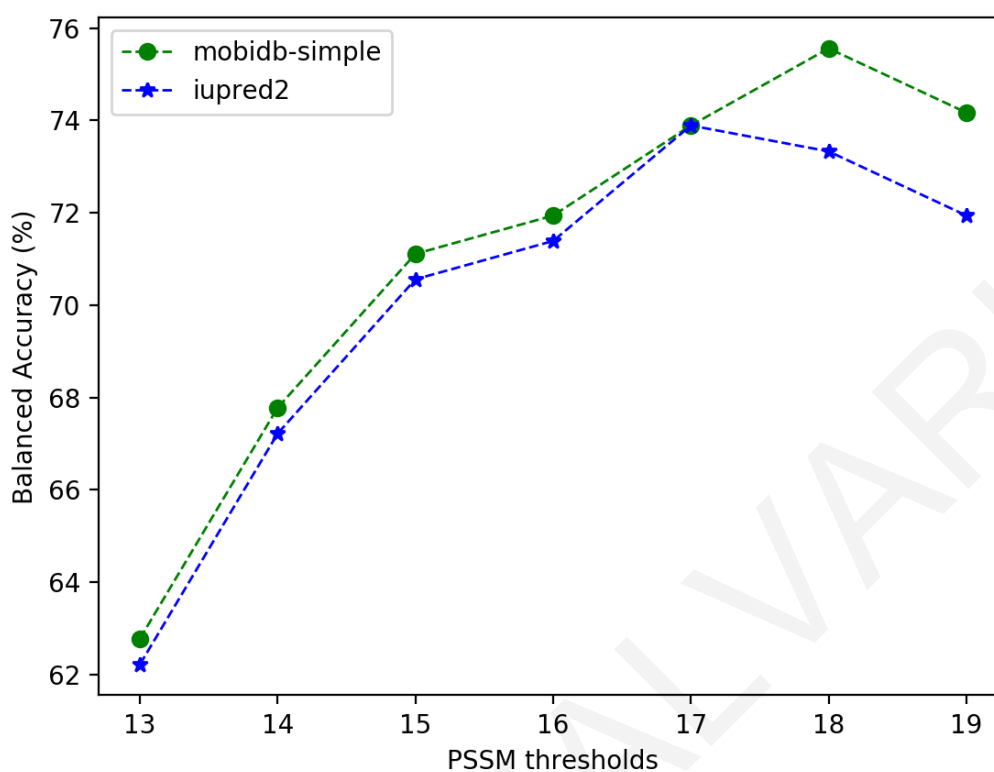


Figure 16. Balanced Accuracy (%) achieved with multi-scheme predictor xLIR+A2|D|PX captured at various PSSM thresholds.

All scores were calculated based on the entire collection of 96 LIR-motifs listed in **Table 7**. IUPRED2A BACC values are in blue and mobidb-simple BACC is presented in green.

These new results illustrate the strength of mobidb-simple in helping to distinguish genuine LIR-motifs. The multi-scheme predictor scheme in collaboration with mobidb-simple once again outperforms the one with IUPRED2A, but both schemes start at lower BACCs of 63% and 62% respectively. It also became clearer the PSSM score at which each method is at peak. For instance, if IUPRED2A were to be used in the multi-scheme predictor, then the PSSM threshold at which the predictor is at its utmost performance would be > 17 reaching a BACC of 74% and F1-score of 0.69. MobiDB-simple, as we also observed in the results depicted in **Figure 16**, reaches its peak performance at a PSSM threshold > 18 with a 75.6% BACC and F1-score of 0.7. This slight improvement in balanced accuracy comes with two additional True Negatives which were previously falsely characterized as functional LIR-motifs. A significant outcome in studies that take the identification of LIR-motifs at proteome scale.

Before closing, it is important to discuss a very intriguing outcome of the comparison of the two multi-scheme predictors xLIR+A2+D|PX and xLIR+A2|D|PX at different PSSM scores. At PSSM scores > 13 , the difference in balanced accuracy and F1 scores between the two methods is marginal (**Table 12**), therefore we went a step further by examining those values at different PSSM cutoffs. Although xLIR+A2|D|PX outperformed xLIR+A2+D|PX for a PSSM score > 13 when tested on both datasets of Alemu et al. (for comparison with iLIR results) and the complete set of LIR-motifs (Table 1), a result that also persisted when we tried different PSSM scores on the Alemu dataset (**Figure 17-left**), this same experiment yielded the opposite outcome on the entire collection of LIR-motifs (**Figure 17-right**).

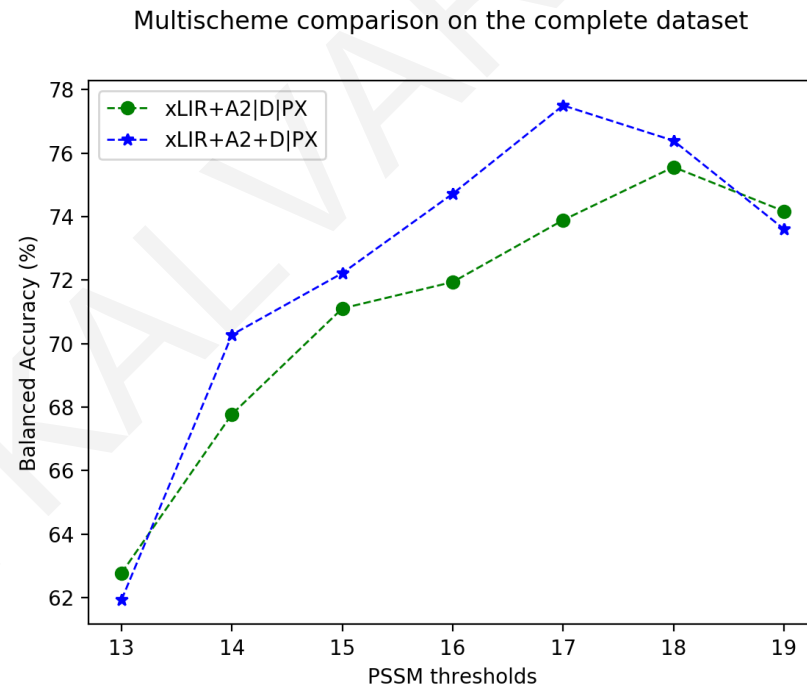
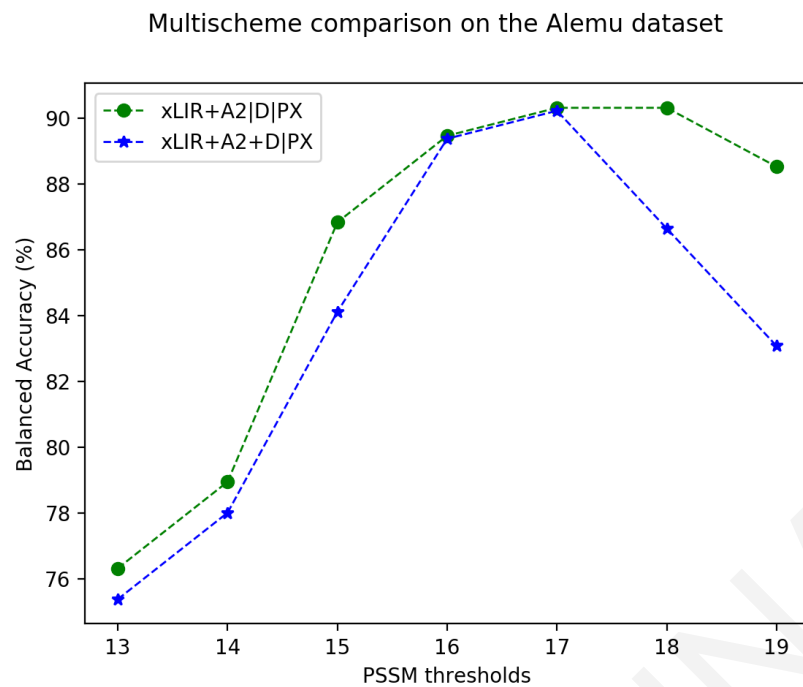


Figure 17. Multi-scheme method comparison.

The performance of the two multi-scheme predictors **xLIR+A2+D|PX** and **xLIR+A2|D|PX** was tested on the Alemu (left) and complete (right) datasets of LIR-motifs, where balanced accuracy is measured at different PSSM thresholds.

What can be observed from the right chart in **Figure 17-right**, is that the multi-scheme predictor **xLIR+A2+D|PX** performs better than the **xLIR+A2|D|PX** for all PSSM scores > 14, reaching a maximum BACC of 77.5% (F1: 0.72) over 73.8% (F1: 0.69) at PSSM >17 respectively. Detailed results for PSSM >17 are presented in **Table 13**.

A possible explanation to this result is that **xLIR+A2+D|PX** requires that both ANCHOR2 and MobiDB-simple predict a LIR-motif as functional, giving more power to ANCHOR2 which we previously saw that the number of correctly identified LIR-motifs was low, hence controlling the outcome of the logical equation A2+D. This means that the outcome relies on the PSSM score solely.

	xLIR+A2+D P17	xLIR+A2 D P17
TP	27	31
TN	48	37
FP	12	23
FN	9	5
Sensitivity (%)	75.00	86.11
Specificity (%)	80.00	61.67
Precision (%)	69.23	57.41
Accuracy (%)	78.13	70.83
Balanced Accuracy (%)	77.50	73.89
F1-score	0.72	0.69

Table 13. Multi-scheme predictor results on the complete dataset.

Comparison of the performance of the multi-scheme predictors **xLIR+A2+D|P17** and **xLIR+A2|D|P17** on the complete dataset of experimentally verified LIR-motifs.

It is important to understand that the purpose of this study is to provide the users with tools and strategies to identify LIR-motifs in putative proteins and various filtering methods based of the requirements of each experiment. The users should use their discretion in selecting the optimal parameters that best fit their needs, taking into account whether specificity or sensitivity is more important. For example, it is expected that when scanning a complete proteome for identifying selective autophagy receptors, choosing a scheme tuned for high

specificity will provide an accurate list of proteins. On the other hand, when scanning a particular protein for candidate LIR-motifs, a high sensitivity scheme will provide a larger number of candidates which can be rationally prioritized using additional features, e.g. those provided by the iLIR server.

3.4.5 Independent validation

Ever since the publication of the iLIR (Kalvari et al. 2014) webserver, new studies exploring the world of selective autophagy came to the surface. Some branched out to other types of autophagy that also resulted in the production of new tools, an example of which is the hfAIM (Xie et al. 2016) web resource that focuses on locating LIR-motifs in proteins participating in pexophagy. To divert from the computational side of things, significant were the experimental surveys that were able to validate functional LIRs that previous studies failed to detect (Skytte Rasmussen et al. 2017; Stadel et al. 2015) and finally studies that introduced novel proteins to the research community (Rogov et al. 2017).

With the best predictive scheme in hand, the next step was to test it on new LIR-motifs. We manually hand-picked four candidate protein sequences with experimentally verified functional LIRs from previous works (Rogov et al. 2017) (Svenning et al. 2011). The sequence of each protein specimen was manually downloaded from the UniProtKB Knowledgebase (<https://www.uniprot.org/>) (The UniProt Consortium 2018) and iLIR webserver was used to search the sequences for an xLIR match, and to generate the PSSM scores used in the multi-scheme predictor. *Anchors* and disorder strings (dSTR, cdSTR) were generated using the tools described in the methods. The results are presented in **Table 14**.

UniProt ID	UniProt Accession	Sequence	Position	Verified	xLIR	Anchor2	PSSM score (e-value)	dSTR	cdSTR	Author	Prediction
C0Z2C5_ARATH	C0Z2C5	REYVLV	358-363	1	1	0	13 (7.90E-02)	DDDDDD	DDDDDD	Svenning et al.	TP
JMY_HUMAN	Q8N9B5	SDWVAV	11-16	1	1	0	22 (4.40E-03)	DDDDDD	DDDDDD	Rogov et al.	TP
		FSFQDL	233-238	0	1	0	11 (1.50E-01)	DDDDDD	DDDDDD	Rogov et al.	FP
		GMWTVL	265-270	0	1	0	18 (1.60E-02)	DDDDDD	DDDDDD	Rogov et al.	FP
		KGVEEV	329-334	0	1	0	12 (1.10E-01)	DDDDDD	DDDDDD	Rogov et al.	FP
		ESFTLL	945-950	0	1	0	11 (1.50E-01)	DDDDDD	DDDDDD	Rogov et al.	FP
PKHM1_HUMAN	Q9Y4G2*	DEWVNV	633-638	1	0	0	19 (1.20E-02)	DDDDDD	DSSSSS	Rogov et al.	FN
RETR1_HUMAN	Q9H6L5	ESWEVI	152-157	0	1	0	20 (8.40E-03)	??DDDD	SSDDDD	Rogov et al.	FP
		LSYLLL	219-224	0	1	0	10 (2.00E-01)	??????	SSSSSS	Rogov et al.	TN
		DDFELL	453-458	1	1	1	18 (1.60E-02)	DDDDDD	DDDDDD	Rogov et al.	TP

Table 14. New proteins and their corresponding verified LIR motifs.

xLIR, PSSM scores, e-values, LIR sequence and positions were generated using iLIR web server (Kalvari et al. 2014). Anchors (Anchor2), dSTRs and cdSTRs were generated using the tools described in methods. *The sequence Q9Y4G2 does not conform to the xLIR motif but only to the most generic WxxL-motif, yet we include it for completeness.

Assuming a wet lab researcher was interested in studying these four sequences:

1. **C0Z2C5_ARATH**: she would be unambiguously pointed to the correct functional motif.
2. **JMY_HUMAN**: among the 5 detected xLIR motifs with intrinsic disorder prediction, the top-scoring against the PSSM is the correct one, which would be prioritized.
3. **PKHM1_HUMAN**: the xLIR motif fails to recognize the functional LIR motif. However, among the 15 WxxL motifs detected by iLIR, the top scoring one is the genuine LIR motif (see **Figure 18**).
4. **RETR1_HUMAN**: the genuine LIR-motif –even though slightly outscored by another xLIR-motif in the PSSM comparison– still has a high PSSM score, and is the only xLIR-motif overlapping an ANCHOR prediction as well as it is predicted to be completely disordered.

All in all, in all of the above cases, even though the predictions are far from perfect, our multi-scheme analysis approach provides useful information for the prioritization of candidate LIR motifs for downstream experimentation.

It is worth mentioning that the validation presented here is by no means comprehensive, as this would require an exhaustive screen of all newly reported proteins with experimentally verified LIR-motifs in the current literature. In fact, we have recently compiled data (Kalvari, Chadjichristofi and Promponas, unpublished data) about dozens of newly reported instances of LIR-motifs - a number of which were discovered based on iLIR predictions. However, a time-consuming manual verification for annotating entries based on literature evidence and cleansing of these data is necessary prior to availability of this dataset for proper analysis. The small dataset analysed here highlights that even though the prediction accuracy of existing and novel prediction schemes developed in this work is not perfect, they can provide valuable guidelines to experimental scientists for rational design of experiments for identifying novel instances of functionally important LIR-motifs.

MOTIF	START	END	LIR sequence	PSSM score	Similar LIRs	In PDB	Anchor
WxxL	31	36	KQYVSL	10 (2.0e-01)		[+/-] 2CXL_A 2CXF_A	No
WxxL	87	92	PVFWPL	2 (2.6e+00)		[+/-] 2CXL_A 2CXF_A	No
WxxL	119	124	RAWLRL	14 (5.7e-02)		[+/-] 2CXL_A 2CXF_A	No
WxxL	132	137	ECYLKL	8 (3.9e-01)		[+/-] 2CXL_A 2CXF_A	No
WxxL	516	521	KSFRVV	7 (5.3e-01)			No
WxxL	530	535	NPFRGL	1 (3.6e+00)			No
WxxL	548	553	GIWKEL	12 (1.1e-01)			No
WxxL	592	597	GRFELV	7 (5.3e-01)			No
WxxL	633	638	DEWVNV	19 (1.2e-02)	T5311_HUMAN 29-34 (4)		No
WxxL	743	748	PSFFKI	7 (5.3e-01)			No
WxxL	766	771	ALWRDL	10 (2.0e-01)			No
WxxL	838	843	IGFSFV	6 (7.4e-01)			No
WxxL	848	853	CAFSGL	0 (5.0e+00)			No
WxxL	907	912	SLYEHV	8 (3.9e-01)			No
WxxL	929	934	GDYLGL	13 (7.9e-02)			No

Figure 18. iLIR results for human Pleckstrin homology domain-containing family M member

The top scoring detected WxxL-motif (score: 19) against the xLIR PSSM corresponds to the functional LIR-motif of human Pleckstrin (Uniprot acc: Q9Y4G2) and additionally has 4 conserved positions compared to the verified LIR-motif of T5311.

It turns out that this particular set of proteins is a good example to demonstrate the efficacy of the multi-scheme predictor. Interestingly, not all motifs match the xLIR regular expression. As the xLIR regular expression derives mostly from human proteins, one would expect this to occur for the case of the plant protein **C0Z2C5_ARATH**. This shows the weakness of the xLIR method and once again signifies the necessity of involving additional characteristics. This is the reason why the iLIR server also reports the more generic WxxL motif and we introduced the *anchors* and the PSSM scores in the past.

Moving on to Anchors, C0Z2C5_ARATH **REYVLV** and JMY_HUMAN **SDWVAV** are not predicted to switch from a disordered to order state upon binding to partners, a case that would result in 2 false negative predictions if anchors were an essential characteristic of functional LIRs e.g. xLIR+A2. However, PSSM scores in the multi-scheme predictor **xLIR+A2|P13** make up for this by successfully picking up all genuine LIRs (**Table 15**). The problem is that with those correctly classified LIRs the PSSM also falsely collects 2 false positives too. This is because almost all of these LIR-motifs appear to be completely disordered, but even if we eliminate this parameter, the PSSM scores of 2 of the unverified LIRs RETR1_HUMAN **ESWEVI** and JMY_HUMAN **GMWTVL** are quite high, therefore they would still be falsely characterised as functional.

One solution to this problem would be to add more constraints to the predictors. For instance, setting upper and lower thresholds to the PSSM scores to filter out outliers. Although this could work for this scenario, there are two cases in the dataset. Unverified LIRs with very high PSSM scores and verified LIRs whose PSSM score is very low. For example, FYCO1_HUMAN **AVFDII** and MK15_HUMAN **RVYQMI** with PSSM scores 8 and 10 respectively. Another case is the case of atypical LIRs (CACO2_HUMAN **DILVV**, TAXB1_HUMAN **DMLVV**) for which PSSMs are unavailable.

	xLIR+A2	xLIR+A2 P13	xLIR+A2 D P13
TP	1	3	3
TN	6	4	1
FP	0	2	5
FN	3	1	1

Table 15. Classification of novel LIR-motifs based on 3 different prediction schemes

TP, TN, FP and FN values computed based on the Verified, xLIR, Anchor2, PSSM-score and dSTR values in Table 14, using the 3 multi-scheme predictors xLIR+A2, xLIR+A2|P13, xLIR+A2|D|P13. The final result is presented under column “Prediction” in Table 14.

It is evident that this is not a one fits all case. As more experimental data become available the better the results of the predictors will be, but it is also expected that more complex methodologies will be required to classify such instances, such as machine learning algorithms that will learn from the data and be able to evaluate multiple parameters at a time.

3.4.6 A comparison to existing tools

The release of the iLIR resource in 2013 paved the way for the development of new resources. In fact about 2 years later a web server called hfAIM (<http://bioinformatics.psb.ugent.be/hfAIM/>) - for high fidelity AIM - made its way out to the scientific community, providing additional computational methods for the identification of Atg8 Interacting Proteins (AIPs), that is selective autophagy receptors and adaptors, with a particular focus in plants (Xie et al. 2016). Their methodology applies more stringent rules requiring that acidic amino acids (Asp (D), Glu (E)) occupying the X₋₁ and X₊₁ positions surrounding the F/W/Y position of the core AIM X₋₁[F/W/Y-X₊₁-X-L/I/V] defined by Schreiber et al. (Schreiber & Peter 2014). These amino acid residues seem to increase the fidelity of the AIM Containing Protein (ACP) interaction with the Atg8 protein (Noda et al. 2008; Wild et al. 2011). Following this notion, they compiled a collection of experimentally verified AIMs (Table S1 in their supplementary material), which resulted in the generation of 5 regular expressions in the form of X₋₂X₋₁[F/W/Y]X₊₁X₊₂[L/I/V] with acidic amino acids occupying positions X₋₁ and X₊₁.

The 5 regular expressions are the following:

- A. hfAIM1: X[DE][DE][WFY][ADCQEIGNLMFPSTWYV]X[LIV]
- B. hfAIM2:
[DE][DE][ADCQEIGNLMFPSTWYV][WFY][ADCQEIGNLMFPSTWYV]X[LIV]
- C. hfAIM3: XX[ADCQEIGNLMFPSTWYV][WFY][DE][DE][LIV]
- D. hfAIM4: [DE]X[DE][WFY][ADCQEIGNLMFPSTWYV]X[LIV]
- E. hfAIM5: XX[DE][WFY][DE]X[LIV]

A comparative analysis they conducted revealed that their approach was able to detect AIMs with a higher specificity compared to the iLIR. As follow-up study and in order to be able to further directly assess this outcome, we used the hfAIM resource to identify LIR-motifs on the protein collection of Alemu et al. The results from the hfAIM scan are presented in the following table.

UniProt ACC	UniProt ID	Verified	Range	Sequence	hfAIM-1	hfAIM-2	hfAIM-3	hfAIM-4	hfAIM-5	#Y
Q8MQJ7	Q8MQJ7_DROME	Y	[388,394]	SDDFVLV	Y	N	N	N	N	1
O75143	ATG13_HUMAN	Y	[441,447]	HDDFVMI	Y	N	N	N	N	1
Q9Y4P1	ATG4B_HUMAN	Y	[6,11]	LTYDTL	N	N	N	N	N	0
		Y	[385,391]	DEDFEIL	Y	Y	N	Y	Y	4
P27797	CALR_HUMAN	N	[165,171]	DDEFTHL	Y	Y	N	Y	N	3
		Y	[197,203]	EDDWDFL	Y	Y	N	Y	Y	4
Q00610	CLH1_HUMAN	Y	[512,517]	PDWIFL	N	N	N	N	N	0
		N	[1147,1153]	SGNWEEL	N	N	Y	N	N	1
		N	[1293,1299]	RGYFEEL	N	N	Y	N	N	1
		N	[1474,1480]	EEDYQAL	Y	Y	N	Y	N	3
O14641	DVL2_HUMAN	N	[61,67]	DQDFGVV	N	N	N	Y	N	1
		Y	[442,447]	RMWLKI	N	N	N	N	N	0
Q8TDY2	RBCC1_HUMAN	N	[602,608]	LCDFEPL	N	N	N	N	Y	1
		Y	[699,705]	TFDFETI	N	N	N	N	Y	1
		N	[910,916]	DNEFALV	N	N	N	Y	N	1
		N	[1000,1006]	IQEFEKV	N	N	N	N	Y	1
Q8IVP5	FUND1_HUMAN	Y	[15,21]	DDSYEVL	N	Y	N	N	N	1
Q9BQS8	FYCO1_HUMAN	Y	[1278-1283]	AVFDII	N	N	N	N	N	0
Q14596	NBR1_HUMAN	Y	[729,735]	SEDYIII	Y	N	N	N	N	1
O60238	BNI3L_HUMAN	Y	[34-39]	SSWVEL	N	N	N	N	N	0
Q3MII6	TBC25_HUMAN	Y	[133,139]	LEDWDII	Y	N	N	N	Y	2
		N	[262,268]	SREYEQL	N	N	N	N	Y	1
Q96CV9	OPTN_HUMAN	Y	[175,181]	EDSFVEI	N	Y	N	N	N	1
O95210	STBD1_HUMAN	Y	[200,206]	HEEWEMV	Y	N	N	N	Y	2
Q92609	TBCD5_HUMAN	Y	[56,62]	RKEWEEL	N	N	Y	N	Y	2
		N	[712,718]	SDDFILI	Y	N	N	N	N	1
		Y	[785,790]	SGFTIV	N	N	N	N	N	0
Q96A56	T53I1_HUMAN	Y	[28,34]	DDEWILV	Y	Y	N	Y	N	3
Q8IXH6	T53I2_HUMAN	Y	[33,38]	DGWLII	N	N	N	N	N	0
O75385	ULK1_HUMAN	Y	[354,360]	TDDFVMV	Y	N	N	N	N	1
Q8IYT8	ULK2_HUMAN	Y	[350,356]	TDDFVLV	Y	N	N	N	N	1

P22681	CBL_HUMAN	N	[111,117]	ENEYFRV	N	N	N	Y	N	1
		N	[271,277]	FLTYDEV	N	N	Y	N	N	1
		Y	[800,805]	FGWLSL	N	N	N	N	N	0
Q13501	SQSTM_HUMAN	Y	[335,341]	DDDWTHL	Y	Y	N	Y	N	3
Q9SB64	Q9SB64_ARATH	Y	[658,664]	VSEWDPI	N	N	N	N	Y	1
P53104	ATG1_YEAST	Y	[426,432]	EREYVVV	N	N	N	Y	N	1
P35193	ATG19_YEAST	N	[225,231]	YHDYERL	N	N	N	N	Y	1
		Y	[409,415]	ALTWEEL	N	N	Y	N	N	1
P40344	ATG3_YEAST	N	[199,205]	EQMFEDI	N	N	Y	N	N	1
		Y	[267,273]	VGDWEDL	N	N	Y	N	Y	2
P40458	ATG32_YEAST	Y	[84,89]	GSWQAI	N	N	N	N	N	0

Table 16. hfAIM AIM predictions on the protein collection of Alemu et al.

hfAIM1-hfAIM5 correspond to the hfAIM regular expressions and ‘Y’ (Yes) indicates a positive hit - hfAIM captures a particular LIR-motif, whilst ‘N’ (No) denotes no matches. Column #Y captures the number of hfAIM models reporting a positive hit (Y).

To compare the two resources, iLIR vs the predictive power of hfAIM, we juxtapose the results of the 5 hfAIM regular expressions to the results of the best iLIR schemes **xLIR+A|P13** discussed in the iLIR paper (Kalvari et al. 2014) and our top multi-scheme **xLIR+A2|D|PX** presented in this chapter at PSSM scores 13 and 17. In addition to the 5 regular expressions introduced by Xie et al. (Xie et al. 2016), we computed a sixth column (#Y) counting the number of hfAIM regular expressions with a match (Y) on each single LIR-motif in the Alemu dataset. Our findings are presented in **Table 17**.

	hfAIM-1	hfAIM-2	hfAIM-3	hfAIM-4	hfAIM-5	hfAIM-any	xLIR	xLIR+A P13	xLIR+A2 D P13
TP	11	6	3	5	8	19	27	26	28
TN	11	13	25	9	10	0	0	11	10
FP	3	2	4	5	4	14	20	9	9
FN	17	21	10	23	20	9	0	1	0
Sensitivity (%)	39.29	22.22	23.08	17.86	28.57	67.86	100.00	96.30	100.00
Specificity (%)	78.57	86.67	86.21	64.29	71.43	0.00	0.00	55.00	52.63
Accuracy (%)	52.38	45.24	66.67	33.33	42.86	45.24	57.45	78.72	80.85
Balanced Accuracy (%)	58.93	54.44	54.64	41.07	50.00	33.93	50.00	75.65	76.32
F1	0.52	0.34	0.30	0.26	0.40	0.62	0.73	0.84	0.86

Table 17. iLIR and hfAIM predictive power assessment.

hfAIM-1 to hfAIM-5 are the predictions of each regular expression provided by the hfAIM web server. hfAIM-any this is a union consensus of the 5 hfAIM methods, which evaluates to a positive LIR-motif prediction if any of the methods hfAIM1 to hfAIM5 predict an instance of a putative functional AIM-motif. The total number of LIR-motif instances considered for hfAIM- and xLIR-based predictions differ, since they rely on the additional pattern introduced by the different regular expressions, which are by definition considered as false positives.

Before going into comparing the two tools (hfAIM, iLIR) it is very important to mention that the results are not directly comparable. hfAIM predictions rely on the 5 regular expressions of length 7, whereas the iLIR service identifies LIR-motifs based on a single regular expression xLIR of length 6, which is more sensitive, but less specific than the hfAIM regular expressions, but eliminates spurious hits with the application of various filters (e.g. PSSM, ANCHOR).

By looking at the TP and FN instances in **Table 17**, what can be observed is that each hfAIM regular expression is of high specificity (> 64%) but of low sensitivity (< 40%, in some cases lower than 25%). When combined in an OR fashion (hfAIM-any) approximately 70% of experimentally verified LIR-motifs can be detected. A relatively good outcome considering that these patterns were initially designed to target peroxisomal autophagic proteins at a great extent. True negatives are specific to the hfAIM regular expressions with every method having its own search-space, and therefore not directly comparable to those of xLIR-based tools.

With respect to the hfAIM web server, although it provides the option for scanning sequences of interest with user-defined patterns, it reports results for each pattern independently. While they allow for multiple sequences to be submitted in each run, the results are provided in separate files. Although this feature is currently not supported by the iLIR web server, it requires that the hfAIM users have at least some programming experience in order to be able to integrate (and prioritize) predictions from the different motifs. Even for a single protein, more than one of the hfAIM patterns may match the same part of the sequence and it is not straightforward (at least not for an average wet biologist) to combine all these results under a single prediction per sequence.

3.5 Conclusions

Our findings show that intrinsic disorder data is a relatively good indicator of genuine LIR-motifs, achieving a 73% of balanced accuracy when used on its own to distinguish between functional and non-functional LIR-motifs from the entire collection presented in **Table 7**. This outcome appears to be 2% lower compared to the multi-scheme predictor, but when combined with other parameters like *anchors* and PSSM predictions, the power of the resulting multi-scheme predictor gives a balanced accuracy that is increased by 1% compared to the one previously introduced in the iLIR paper (Kalvari et al. 2014).

With respect to that dataset, an upgrade to the latest version of disorder binding region predictor ANCHOR2 increased the predictive power of the model by about 3% (78% balanced accuracy). Building on the multi-scheme predictor by incorporating disorder data from MobiDB100_{simple} increased balanced accuracy nearly by 1% (78.95%), a score which increased the number of correctly identified LIR-motifs and at the same time eliminated any false negatives completely.

It seems that ANCHOR2 in presence of Disorder data does not have an effect on balanced accuracy. Evidence to this are the logical equations $\mathbf{xLIR+D}$ and $\mathbf{xLIR+A2|D}$, both of which result in a balanced accuracy of **83** percent. However, since disorder (**D**) is the optimal, this multi-scheme predictor cannot be improved any further. A work around this limitation is the selection of multi-scheme predictor $\mathbf{xLIR+A2|D|PX}$, where **PX** is a parameterizable PSSM score for fine tuning the predictor. In fact, testing a range of PSSM values from P13 to P18, it seems that the best performance for this particular dataset was $\text{PSSM} > 17$. Balanced accuracy under this scheme reaches a maximum value of about 76% on the complete dataset, whilst for the proteins of Alemu and colleagues (Alemu et al. 2012), balanced accuracy is even higher at 88%.

A more balanced and richer dataset will allow us to understand how such prediction schemes would behave under those circumstances. but that one was only tested on the dataset retrieved from the work of Alemu at al. (Alemu et al. 2012).

4 iLIR3D: Delving into selective autophagy structural data

4.1 Preface

In the previous chapters we discussed the development of new tools for the identification of novel LIR-motifs in putative selective autophagy receptor/adaptor proteins. We further improved the prediction accuracy of our tools by incorporating intrinsic disorder data and by devising new multi-scheme predictors that we thoroughly assessed on the efficacy of our methods in our complete dataset of experimentally validated LIR-motifs (**Table 3**), but also for the prediction of novel instances.

The next logical step was to turn to structural data as an alternative predictive method, but also to acquire new knowledge and better understanding of the mechanism of selective macroautophagy. For this purpose, we compiled a dataset of protein structures based on existing data and ran a considerable amount of baseline docking experiments that are further supported by additional experiments of in-house produced decoy sets.

To make our data available to the scientific community we designed and developed a MySQL database that is provided in the form of a MySQL dump. The database can easily be built up and manipulated via database management tools such as MySQL Workbench. The tables were populated with data and metadata accumulated from all the different studies discussed throughout this document with broad usage capabilities.

As a possible use of this dataset we envisage the systematic study of the specificity of known LIR-motifs to different Atg8 homologs (Rogov et al. 2017). Furthermore, peptides from the decoy set with unexpectedly high docking scores may indicate alternative modes of interaction (e.g. via α -helical coiled-coils as in the case of (Mandell et al. 2014)). This chapter provides more details on the structure and contents of the iLIR3D database and demonstrates the potential of such a data resource with real examples.

4.2 Data and Methods

4.2.1 Data

4.2.1.1 Creating a collection of 3D structures

The first task was to create a collection of 3D structures of Atg8 homologs, either isolated or in complex with bound LIR-motifs. For this purpose we searched the RCSB PDB (Berman et al. 2000) (<https://www.rcsb.org/>) with keywords: “ATG8”, “Autophagy”, “GABARAP” and “MAP1LC3” in search for structures of Atg8 homologs and selective autophagy receptor and adaptor proteins. In addition, relevant PDB entries referring to the interaction of a LIR-motif with an Atg8 homolog were also manually retrieved from relevant publications in the biomedical literature. Secondly, to ensure the completeness of the dataset we further automatically retrieved from the UniProt Knowledgebase (The UniProt Consortium 2018) (UniprotKB - <http://www.uniprot.org>) any PDB IDs associated with each protein accession listed in **Table 3**. Then all structures were downloaded programmatically from the RCSB Protein Databank (Berman et al. 2000) (PDB - <http://www.rcsb.org>). Manual curation was conducted as a quality assurance measure, a procedure which discarded any non-relevant structures.

The remaining protein structures (N=40) correspond to PDB entries with an Atg8 protein – or their mammalian homologs (GABARAP, GABARAPL1, GABARAPL2, MAP1LC3, MAP1LC3A, MAP1LC3B, MAP1LC3C) –, which can either be single or in complex with selective autophagy receptor/adaptor proteins, bound via a LIR-motif. From these structures, we further select those entries bound to a LIR-motif cargo (N=21); these will serve to initially identify the LIR-motif binding regions for the definition of the 3D volumes that the docking experiments will target. In addition, these structures provide information about the binding conformation of LIR-motifs (ligands). It is worth mentioning that most of these cases refer to engineered versions of the LIRCPs, e.g. a LIR-motif (possibly with flanking residues) co-crystallized with the Atg8 protein or a construct of the LIR-motif fused to the Atg8 protein via a flexible linker. The structures are listed in **Table 18**.

No.	Receptor Type	PDB ID	Chain	Species	Taxonomy ID	Method	Resolution
1	GABARAP	3DOW	A	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	2.3
2	GABARAP	4XC2	C	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	1.9
3	GABARAPL1	5LXH	C	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	1.58
4	GABARAPL1	5LXI	D	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	1.44
5	GABARAPL1	2L8J	A	<i>Homo sapiens</i>	9606	SOLUTION NMR	N/A
6	MAP1LC3A	5CX3	A	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	2.3
7	MAP1LC3A	3WAN	B	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	1.77
8	MAP1LC3A	3WAN	A	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	1.77
9	MAP1LC3B	2K6Q	A	<i>Rattus norvegicus</i>	10116	SOLUTION NMR	N/A
10	MAP1LC3B	2LUE	A	<i>Homo sapiens</i>	9606	SOLUTION NMR	N/A
11	MAP1LC3B	2ZZP	B	<i>Rattus norvegicus</i>	10116	X-RAY DIFFRACTION	2.05
12	MAP1LC3B	2Z0D	B	<i>Rattus norvegicus</i>	10116	X-RAY DIFFRACTION	1.9
13	MAP1LC3B	2Z0E	B	<i>Rattus norvegicus</i>	10116	X-RAY DIFFRACTION	1.9
14	MAP1LC3B	5D94	A	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	1.53
15	MAP1LC3B	2ZJD	A	<i>Rattus norvegicus</i>	10116	X-RAY DIFFRACTION	1.56
16	MAP1LC3C	3WAP	A	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	3.1
17	MAP1LC3C	3VWV	B	<i>Homo sapiens</i>	9606	X-RAY DIFFRACTION	2.5
18	Atg8	4EOY	C	<i>Plasmodium falciparum</i>	5833	X-RAY DIFFRACTION	2.22
19	Atg8	3VXW	A	<i>Saccharomyces cerevisiae</i>	4932	X-RAY DIFFRACTION	3
20	Atg8	2ZPN	C	<i>Saccharomyces cerevisiae</i>	4932	X-RAY DIFFRACTION	2.7
21	Atg8	2ZPN	B	<i>Saccharomyces cerevisiae</i>	4932	X-RAY DIFFRACTION	2.7

Table 18. Proteins of the Atg8 family, herein “receptors”, found in template structures.

All 21 receptors participating in the docking experiments. The structures come from 4 distinct species including Human and can be further divided into 6 categories based on receptor type (e.g. GABARAP, MAP1LC3B, etc.)

The following sections provide detailed information regarding each data category, pre-processing algorithms and tools utilised in each step.

4.2.1.1.1 Template structures

Template structures are complexes of Atg8 homologs bound to LIR-motif peptides. These pairs are also the ones used for the calculation of the the receptor binding site. For that purpose, the molecules composing the complex structure are separated to the **receptor** (Atg8) structure and its corresponding **ligand** (LIR peptide) to be put back together by the protein-protein docking algorithm. The scores deriving from the docking of **receptor** and its native **ligand** are used as reference for an accurate interpretation of any downstream analyses. There are **24** such structures in total, however we were able to calculate the binding

site for only **20** of those due to structural data artefacts (i.e. modified sequences). In all template structures which were determined by NMR we have arbitrarily chosen only the first model reported in the respective PDB file.

4.2.1.1.2 Ligands (3DLIRs)

Contrary to receptors, the collection of ligands, that is 3DLIR-motifs, is significantly larger. The number of structures we retrieved is about 10 times the number of “good” receptor structures, including both X-RAY crystallography as well as Nuclear Magnetic Resonance (NMR) samples. This is because we have a larger number of protein candidates (64 in **Table 3**) compared to the few Atg8 homologs of the respective species. In addition to that, several of these proteins often come with multiple three-dimensional structures in the PDB, starting with a minimum of 1 structure per protein up to a maximum of 22 structures, as is the case of E3 ubiquitin-protein ligase CBL (CBL_HUMAN). It is important to mention that structure availability was very limited for the case of ligands too.

Back in 2016 when we last updated this dataset, there were 3D structures available for only 34 out of the 64 proteins (53%), but more may have been deposited to RSCB PDB ever since. The total number of SARs structures is **182**, from which we manage to extract **246** 3DLIR-motifs. Preliminary docking experiments revealed that the length of the ligand was influencing the docking scores significantly, therefore setting a constraint that 3DLIR-motifs are at least 6 residues long was essential, therefore any 3D peptides of shorter than 6 residues were filtered out, including the non-canonical LIR-motifs of CACO2_HUMAN and TAXB1_HUMAN. This reduced the number of ligands exploited in the docking experiments to **211**.

The ligand structures were computationally collected in accordance with **Table 3**, using the REST APIs from both resources UniProt and RCSB PDB using in-house code developed in python 2.7. Ligand extraction methodology and issues we stumbled upon during the collection process are thoroughly described in the sections that follow.

4.2.1.2 Collecting useful protein metadata

The final step was to devise a set of metadata that would help us better organize the docking results in a such a way that they can be easily utilized by the end users. The set of metadata comprises data extracted from the UniProt and the RCSB PDB database, as for example species per chain in the structure, LIRCP/Atg8 interactions, structure resolution in Angstrom

(Å), methodology applied (NMR, X-ray crystallography), function of molecule, PubMed ID, source (i.e. UniProt, PDB). We also generated additional information, such as labelling a structure as “Template” or not (1,0), tags like the filename or numbering of the LIR-motifs of a particular protein, start-end positions on the model sequence the LIR peptide was extracted from etc.

4.2.1.3 Decoy Set generation

Decoy sets are used in virtual screening experiments to investigate whether a docking algorithm is able to discriminate between genuine and non-genuine ligands. As a quality assurance measure and in order to ensure that the scores of docking results diverged from what is observed at random, we devised a small collection of sensible decoy sets.

For the generation of the decoy sets we searched the RSCB PDB for human proteins matching the xLIR regular expression pattern, and reduced redundancy at the 30% sequence identity level. This process retrieved a total of **1507** human structures. The protein structures were downloaded using the aforementioned tools and they were further processed for the generation of the decoy sets used in this study. Generation of the decoy sets was done programmatically using the method described in 4.2.2.1.1.

The resulting decoy sets can be divided in the following three categories:

1. iLIR ligands: A total of **12** ligands (3D peptides) extracted from the **1507** human proteins. Two constraints were applied in this case:
 - a. The 3DLIR matches the xLIR regular expression
 - b. The 3DLIR matches the LIR-motifs illustrated in **Table 1** exactly (100% identity).
2. Random dataset: A total of **3215** 6-residue peptides randomly extracted from the collection of proteins we retrieved from RSCB PDB using custom in-house code. To limit the size of this dataset only 2 3DLIRs were extracted from each of the **1507** human proteins. Extraction was enabled using a random number generator that produced random start-end hexapeptide coordinates at any position within the protein sequence at hand.

3. xLIR peptides: A total of **564** 6-residue peptides that match the xLIR motif with no restriction on matching any of the verified and unverified LIR-motifs listed in **Table 3**. This means there is chance that this dataset also includes the hexapeptides from decoy dataset 1.

4.2.2 Methods

4.2.2.1 Computational methods for data extraction from UniProt KB and RSCB PDB databases

4.2.2.1.1 A computational method for 3D structure retrieval from RSCB PDB

Three-dimensional structures were retrieved from RSCB PDB with a custom-made script developed in python 2.7. Structure retrieval is achieved using UniProt KB and RSCB PDB REST APIs with simple utilization of widely used python libraries **httplib** and **requests**. An http request to the UniProt REST API retrieves the correct metadata of a particular protein in text (.txt) format. The algorithm then parses the text by searching for the labels “DR” and “PDBsum;”, which contain the PDB IDs corresponding to a specific protein. If available, a list of PDB IDs is created.

Following the PDB ID extraction, the script then does another http call to RSCB PDB’s REST API and fetches the corresponding PDB file, which then saves locally at a specified destination directory. The process repeats for every UniProt accession listed in **Table 3**.

4.2.2.1.2 Metadata extraction

Metadata are very important for the correct interpretation of the data as well as the results. For this reason and the necessities of this project, specialized software has been developed in Python 2.7 for the collection of useful and relevant metadata from related resources such as UniProt (The UniProt Consortium 2018) and RSCB PDB (Rose et al. 2015). The algorithm uses RSCB PDB and UniProt REST APIs to gather and extract metadata per UniProt entry, which is then imported in the database.

4.2.2.2 Atg8 receptor binding site calculation

Defining the search space of the Atg8 binding site was one of the most crucial steps for the protein-protein docking experiments to be efficient and successful. A predefined search space minimizes the search space to just the area of interest and at the same time eliminates the chance for misplaced ligands.

The binding site has been calculated for the Atg8 receptors including mammalian homologs and their isoforms for all complexes in the PDB template structures. That is, all native receptors in complex with one of their LIRCP binding partner.

This process comprises two steps:

1. Identification of the interface of the protein-protein interaction
2. Translation of the interface residue coordinates into x,y,z coordinates of the search area

Identification of the interface residues for each of template was achieved by manually submitting relevant PDB structures to the PDBePISA (Krissinel & Henrick 2007) web server - a tool for the exploration of macromolecular interfaces (http://www.ebi.ac.uk/msd-srv/prot_int/cgi-bin/piserver) - allowing only for 10% of buried area at most for each residue participating in the interface. This ensures that the entire receptor search space is captured and that there will be no limitations on the rotational grid.

This tool returns the interfacial residues for each input, which were manually extracted for all template structures. We only took into account the participating residues from the receptor side. The reason why we did this is because in the presence of the receptor interface residues the ligand residues become redundant and I will explain why this is true with the following example. Imagine that the receptor is a mass of clay onto which we press down an object, in such a way that when we pull the two apart, the 3D shape of the object is imprinted on the clay. The interface residues of the receptor are 3D descriptors of the formed cavity. The actual 3D coordinates that allow us to calculate the centre of the grid and the volume of the search space. Visual examples are depicted in **Figure 19**.

The docking grid was calculated using explicitly developed in-house software. All the residues that participate in the interface are provided to the tool as a string of integers that are separated by commas ','. The algorithm converts the string of integers to a list and then reads and parses the corresponding PDB file and calculates the average of the x,y,z

coordinates of each residue specified in the list - interface residue - resulting in a single x,y,z triplet that defines the center of the docking grid. All these data (interfacial residues and the grid center coordinates) are collected and stored in the database.

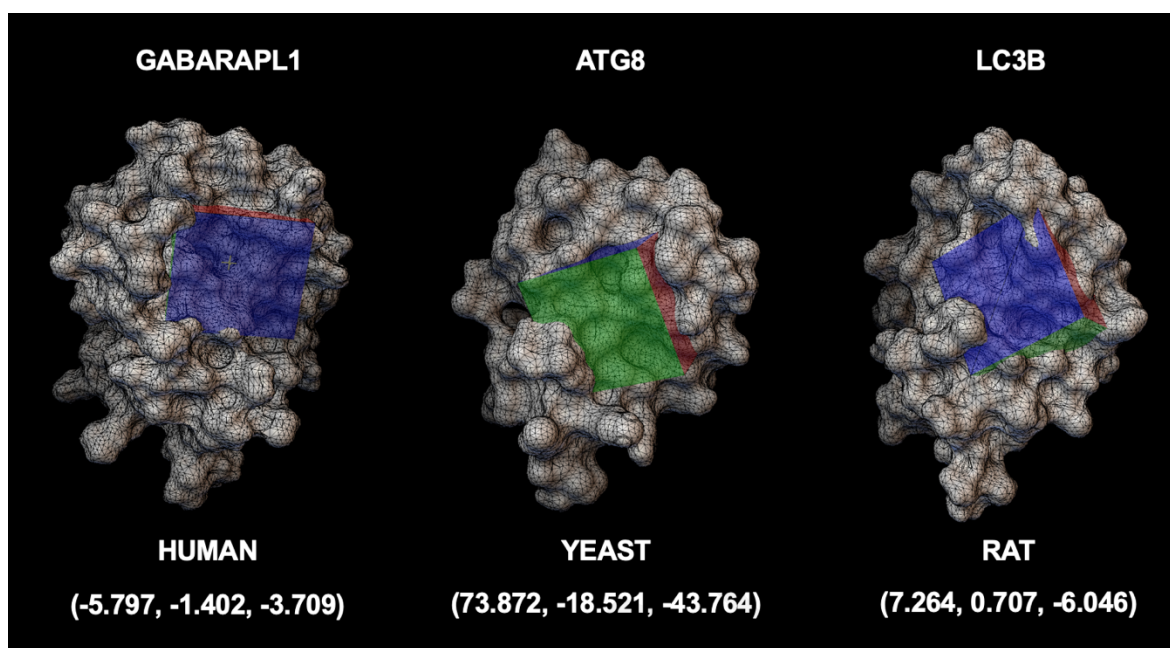


Figure 19. The docking grids of the Atg8 family.

Binding sites of the yeast Atg8 receptor and its two homologs, human GABARAPL1 and rat MAP1LC3B. The coloured cubes define the binding site area with the exact x,y,z coordinates provided in the parenthesis.

4.2.2.3 Manual ligand extraction of 3DLIRs

Ligand extraction required manual curation due to the various artefacts that come along with structural data. The problem is that iLIR identifies the LIR motifs on the canonical sequence, which in many cases start-end points on the model sequence (sequence in the PDB structure) may be shifted or missing (short fragment) or even re-engineered for the purposes of an experiment. To ensure the high quality and accuracy of the 3DLIR-motifs, we manually extracted all 3DLIRs using PyMol (DELANO & L 2002) for Education v1.7.4.5 based on the start-end coordinates listed in **Table 3**. As previously mentioned in the preceding data section, the difference in the total number of 3DLIRs (246) and the actual set used in the docking experiments (211), is due to six residue peptide constraint applied to all samples for the following reasons:

1. LIR-motif sequence coordinates generated from the actual protein sequences (UniProtKB FASTA files) do not always match the model sequence in the corresponding structures.
2. Structure mutations: Many of the experiments are designed in order to study specific features which are very often address by bioengineering a molecule at hand and introducing mutations.
3. Short or fragmented model sequences: Many of the PDB structures may only contain small segments or fragmented model sequences rather than entire canonical protein sequence. As a result, this limits the number of 3DLIRs as many of the available structures lacked the model sequence fragment at the position where a LIR was identified.
4. Low structure resolution: This results in worse docking scores which is hard to assess when we need to compare these to the scores of the docked templates.

Following model sequence examination, if the LIR-motif sequence reported by iLIR webserver matches the model subsequence in the exact residue position e.g. SQSTM_HUMAN, positions 336-341, the 3DLIR-motif is extracted using PyMOL's command line interface (CLI) and saved into a separate PDB file to be used as a ligand in the docking experiments.

The following command is an example of the extraction of the 3DLIR-motif of the p62 selective autophagy receptor (SQSTM_HUMAN) using PyMOL:

```
PyMOL> select 2K6Q_LIR1_A_6,2K6Q_A & resi 336-341
```

Where 2K6Q_LIR1_A_6 is the name of the resulting molecule that will also be the filename. 2K6Q is the PDB ID, LIR1 a tag specifying the order in which the LIR-motif was identified in a given sequence by iLIR, A is the chain in the structure and 6 specifies the number of the residues extracted.

In cases where the LIR-motif is found on the model sequence, but with a slight shift compared to the coordinates of the LIR-motif on canonical sequence, the 3DLIR is manually extracted from the corrected positions.

4.2.2.3.1 Computational methods for mass ligand extraction and decoy set generation

Manual extraction of 3DLIRs was feasible due to the relatively small size of the PDB dataset constructed by collecting all PDB structures associated with the UniProt accessions listed in **Table 3**. In the case of the decoy sets, the number of proteins increased significantly, therefore the number of extracted ligands was expected to be even larger, which made it nearly impossible to generate manually.

We devised a new script combining functionality from the broadly used chemical tool openbabel (O'Boyle et al. 2011) and custom-made complementary functions. The script is called *ligand_generator* and was implemented in python 2.7. The script takes as input a directory of protein pdb files, each of which contains a single chain only, and a destination directory where the ligand files will be stored at. With respect to the type of ligands, the algorithm has two methods of ligand extraction defined by two options `--rand` for random and `--regex` for extraction using one of the regular expressions cLIR or xLIR.

Briefly, the process starts by loading the PDB file in an openbabel (O'Boyle et al. 2011) molecule object, which is used to make scanning of the model sequence easier. If the option `--regex` is provided, meaning that the 3DLIR needs to match the xLIR regular expression `[DE][DEST][WFY][DELIV][DERHKSTNQCGPAVILMFYW][ILV]`, then depending on the length of the peptide that we need to generate, the model sequence is scanned from left to right by sliding the window as many residues as its length, meaning that at each iteration the first residue in the window is located at position `index>window_length+1`. If the sequence at the current position of the window matches the regular expression, the start and end positions are stored in a dictionary structure, along with the sequence string, to be extracted at a final step by the *ligand_generator*. This last function parses the PDB file and extracts the sequence segment at the designated positions. Structural coordinates for these peptides are extracted from the PDB file using a custom function in Python. The process completes when all input PDB files have been scanned.

If random ligand generation is selected, the process does not vary much, in a sense that the PDB structure is once again loaded in an openbabel molecule object, but instead of a sliding window a random number generator randomly selects start-end points within the margins of the chain sequence at hand. The extraction of the 3D peptide is done the same way as using the `--regex` option.

4.2.2.3.2 Protein-protein docking using FRODOCK

To investigate whether a 3DLIR-motif is a genuine binder of the Atg8 proteins, we apply protein-protein docking techniques to evaluate the force of the interaction. Such algorithms try to fit two unbound protein structures together, the receptor and the ligand. Given the binding site on the receptor molecule, the ligand is rotated within the defined search space for the optimal position. With every new positioning of the ligand - conformation - within the binding site, the binding energy or a correlation value is measured and reported (**Figure 20**). The best conformation is the one with the highest docking score and the lowest root mean square deviation (RMSD, Å) from the reference ligand (template molecule).

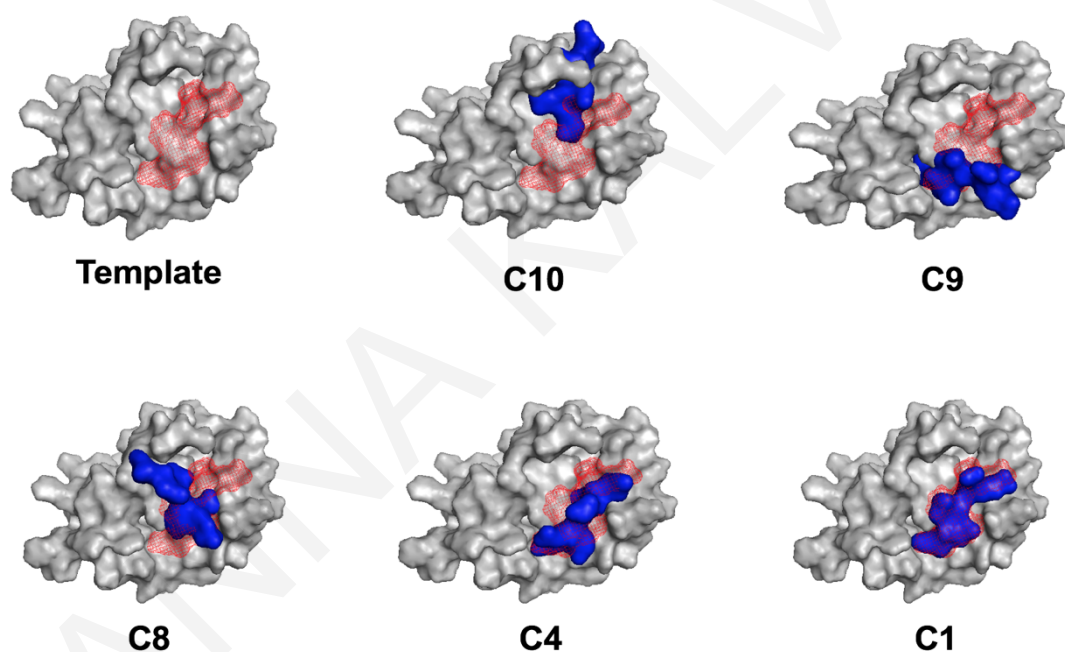


Figure 20. Protein-protein docking example.

An illustration of rigid rotation of the ligand within the receptor binding site in search for the optimal positioning/conformation. The tested ligand is presented in blue, and in red colour we present the orientation of the template molecule. Conformation 1 (C1) is the optimum case with the placement of the tested ligand (blue), almost at the exact same position of its template. The example is a result of the docking of the 3DLIR-motif of Human selective autophagy receptor NBR1 to the binding site of GABARAPL1 (PDB id: 2L8J).

The tool we used for the protein-protein docking experiments is FRODOCK (Garzon et al. 2009), a fast rotational protein-protein docking tool based on global energy minimization and three interaction potentials including electrostatic, van der Waals and desolvation

potentials that was firstly introduced in 2009 (Garzon et al. 2009). Initial experiments were performed using the primary version of the tool, but with the addition of new structures to the data, we also updated to the latest version of FRODOCK v.2.0 (Ramírez-Aportela et al. 2016), which also resulted in higher scores. All the structures (receptors, ligands) prior to docking were further enhanced with the addition of polar hydrogen atoms.

For the docking experiments we used FRODOCK's linux CLI, and implemented a python wrapper to FRODOCK's preprocessing, processing and post-processing tools. The python script is called *pyFrodock* and it comes with 3 distinct options `--ligands`, `--receptors`, and `--pydock`. The first two options are responsible for the execution of all the required pre-processing steps in preparation of the input files for the docking experiments, whilst the latter `--pydock` performs the actual docking using the inputs generated by the other two options. Ligand as well as receptor pre-processing are both compulsory and the output files are organised in distinct directories one for each input PDB file (ligand, receptor). The pre-processing steps include the generation of the three interaction potentials, electrostatic (`_E.ccp4`), van der waals (`_W.ccp4`) and desolvation (`_DS.ccp4`), which are computed in three steps. The python script serves as an abstraction to the various options required for the generation of the files and at the same time simplifies the entire procedure.

FRODOCK (Ramírez-Aportela et al. 2016) includes 4 distinct tools for preprocessing (*frodockgrid*), docking (*frodock*), clustering of conformations (*frodockclust*) and finally for the extraction and visualisation of the results (*frodockview*). Any receptor and ligand structures participating in the docking experiments need to undergo pre-processing with *frodockgrid* for the generation of the required files. The following are the compulsory files for each type of molecule:

- Receptor:
 - Van der Waals potential map
 - Electrostatic potential map
 - Desolvation potential map
- Ligand
 - Desolvation potential map

Finally, `--pydock` based on a text file (.txt) that specifies the list of docking experiments to be performed (receptor vs ligand), conducts the actual docking of the pair and processes the output to report the X top results, where X is the number of conformations specified by

the user. The 3 top conformations (by default) are also saved in PDB format for further manual inspection during the evaluation process.

FRODOCK is Linux based by default, therefore it does not run on other operating systems (OS) such as Mac OS X and Windows. To address this limitation, we developed this as a dockerized application, modernized virtualization techniques that enable software to run on any machine. The dockerized version of *pyFrodock* was tested in a docker container on Mac OS X version 10.13 (Sierra).

4.2.2.3.3 Evaluation Metrics

Similar to the evaluation metrics set we devised to assess our methodologies presented in the previous chapters, we also had to come up with a new set of sensible metrics that would allow us to assess our results. The main aim is to define a set of thresholds for the selection genuine LIR-motifs based on docking scores and comparison of the docked molecule conformation to a template.

For the analysis of outcome of the docking experiments, we will be using the following metrics:

- Docking score MAX, MIN: The maximum and minimum values from the docking scores will allow us to define a range of accepted values for which a docking score can indicate a genuine LIR.
- Mean (MEAN): The central value of the group of docking scores being evaluated.
- Standard deviation (STDEV): A value indicating how close or far the values fall from the mean, where low STDEV shows that the values concentrate around the mean, whilst a high STDEV shows that the values fall far from the mean.
- Average (AVG): An average value of a set of docking scores being examined.
- Root mean square deviation (RMSD): Per residue distance of two superimposed molecules. This will allow us to choose the optimum conformations. The smaller the RMSD the more the docked molecule resembles a template in terms of orientation and interaction with the receptor.

In the following chapter we provide some baseline analyses and preliminary results that may highlight possible uses of the 3DLIR-motif database.

4.3 Results

4.3.1 The iLIR3D MySQL database

The docking experiments resulted in the generation of a large amount of data. In particular, **36,810** receptor-ligand pairs were generated by docking the manually extracted verified and unverified 3DLIRs, and **68,022** additional instances produced from docking the decoys. A total of **104,832** samples, a substantial amount of information that deserves further analysis. With such data volumes it was essential that we developed a resource that would enable us to manage, update and analyse all of that information with the minimum possible effort.

A solution to this problem was the design and development of a relational database in MySQL, which structures all the information in the form of tables and enables data retrieval by association. A good analogy to a database table is that of a class in objective programming. Each table has its own attributes (the columns) that describe a particular entity. For example, the table *sars* contains information about selective autophagy receptors such as UniProt accession, UniProt id, sequence length, taxonomic identifier (tax id), author etc. Table rows or else the records, are more like object instances of a class. Records, like objects, have their own values that define a particular object e.g. protein SQSTM_HUMAN has tax id: 9606, is 440 amino acids long and was obtained from the paper of Alemu et al (Alemu et al. 2012).

The programming language used to manipulate data held in a relational database is called SQL, for Structured Query Language. SQL provides a way of communication between the user and the database through the formation of queries, that is conditional statements for data extraction that aim to answer specific questions. For example, such questions could be “**How many ligands bind the GABARAPL1 receptor with a score > 1200**” or “**How many Atg8 structures are there**” and many others. The following (Figure 21) is a very simple query example that retrieves the template docking scores illustrated in Table 20:

```
template_scores.sql Raw
1  select exp.exp_acc, rec.rec_type, sl.uniprot_id, rec.chain,
2  sld.pdb_id as common_pdb_id, sld.filename, sld.sequence, exp.dock_score
3  from experiment exp, receptor rec, sars_lir_3d sld, sars_lir sl, sars_structure ss
4  where exp.rec_acc=rec.rec_acc
5  and exp.lir_3d_acc=sld.lir_3d_acc
6  and rec.pdb_id=sld.pdb_id
7  and ss.pdb_id=sld.pdb_id
8  and sl.lir_acc=sld.lir_acc
9  and exp.software='FRODOCK2'
10 and exp.conformation=1 and rec.rec_acc in
11 (select rec_acc from receptor where template=1 and
12 x_coord<>0 and y_coord<>0 and z_coord<>0);
```

Figure 21. MySQL query that retrieves the template structure results presented in Table 20.

The power of such tools becomes more apparent upon data analysis. It would be very difficult to combine many different attributes without the help of a relational database. To demonstrate its usefulness, one interesting thing we could look into is the top scoring ligands per receptor type (Atg8, GABARAPL1 etc), for which there is also evidence for its actual interaction with that particular Atg8 homolog. For this purpose, we would have to combine information from the tables *experiment*, *sars*, *sars_lir*, *sars_lir_3d* and *sars_atg8_interaction* using their relationships - foreign key references - as it appears in **Figure 21**.

iLIR3D is a manageable database (nearly 3 MB in size) - excluding the actual output files of the docking experiments - and is organised in 9 tables. **Figure 22** illustrates the database model portraying all its components: the tables, fields and relationships. Short descriptions about the data stored in each table are provided in **Table 19**.

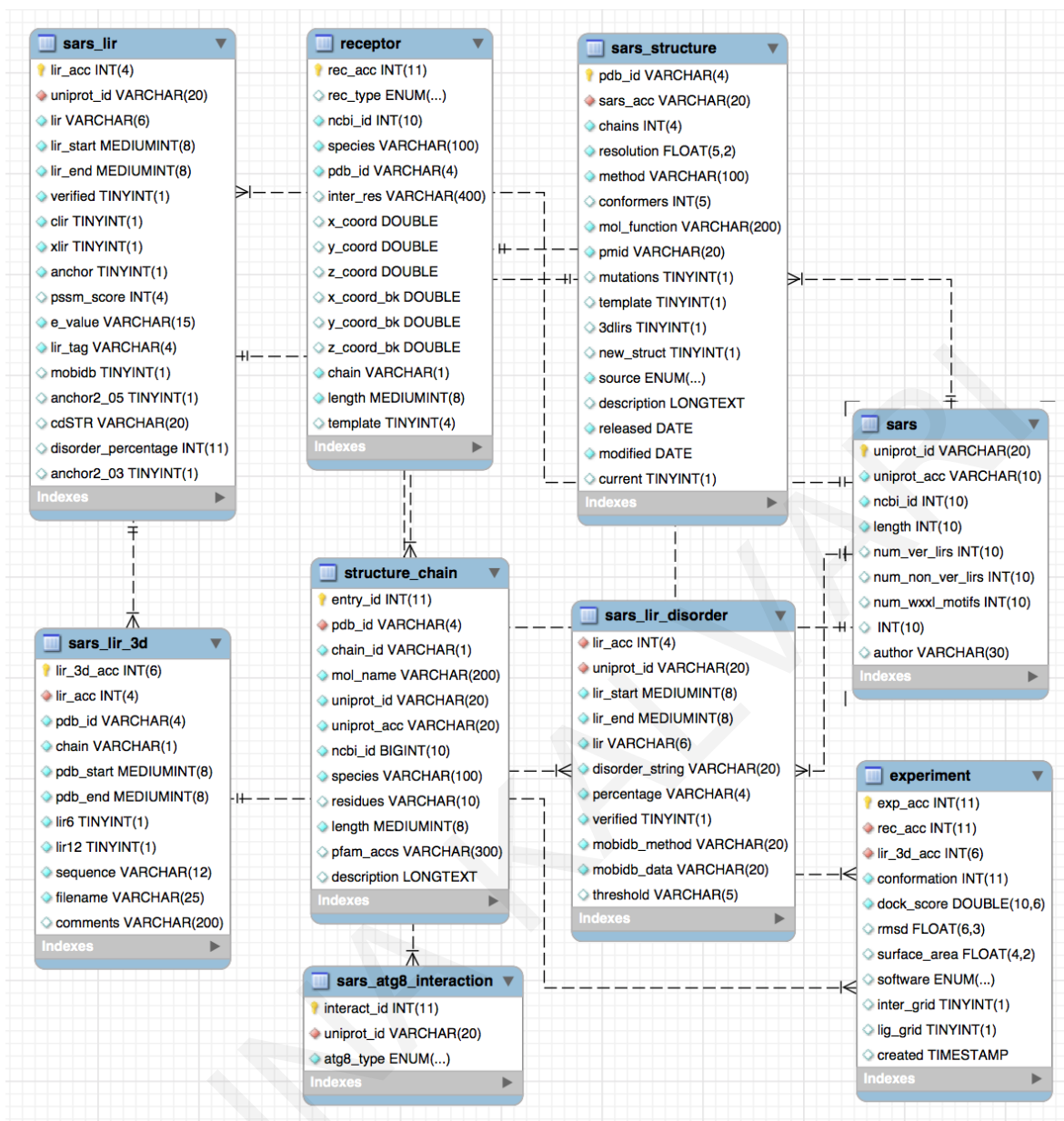


Figure 22. iLIR3D relational model created using MySQL workbench by application of reverse engineering.

The model illustrates the tables composing the database as well as the different fields belonging to each table and the relationships.

Table	Description
receptor	Receptor information including pdb id, chain, receptor type (ATG8, GABARAP, MAP1LC3 etc) as well as the binding site coordinates (x,y,z)
sars	UniProt related information such as UniProt accessions and IDs for all proteins in Table 1 and metadata retrieved from UniProt such as sequence length, tax_id, author etc
sars_atg8_interactions	Atg8 homologs and LIRCPs relationships
sars_lirs	All LIR entries as presented in Table 1
sars_lir_3d	All available 3DLIRs for each LIR in table sars_lir_3d
sars_structures	All 3D structures collected for each of the selective autophagy proteins
structure_chains	Chain metadata per available SARs structure retrieved from UniProt and RSCB PDB
sars_lir_disorder	Disorder predictions generated for each individual case of SARs
experiments	Experiment table holding useful information on each ATG8/SARS docking experiment such as docking score per conformation, RMSD etc

Table 19. Short descriptions of the tables composing the iLIR3D database.

Although the iLIR3D database was initially developed to organise, analyse and provision the structural data, it also became very useful in other areas covered in this study. For example, *sars_lir_disorder* table contains all the disorder data that we generated and helped a lot with the analysis of those results. Computation of the true positives (TP), true negatives (TN), false positives (FP) and false negative (FN) instances, in the majority of cases was achieved using mysql queries. An example of the queries used to compute the aforementioned values based on predicted data produced by MobiDB (Piovesan et al. 2018) using its **simple** method with the constraint that LIR-motifs are 50% composed by disordered residues (**Figure 23**).

```
disorder_queries.sql Raw
1  -- TPs
2  select count(*) from sars_lir_disorder
3  where mobidb_method='simple'
4  and mobidb_data='predicted'
5  and lir_acc < 97
6  and verified=1
7  and percentage >= 50;
8
9  -- TNs
10 select count(*) from sars_lir_disorder
11 where mobidb_method='simple'
12 and mobidb_data='predicted'
13 and lir_acc < 97
14 and verified=0
15 and percentage < 50;
16
17 -- FPs
18 select count(*) from sars_lir_disorder
19 where mobidb_method='simple'
20 and mobidb_data='predicted'
21 and lir_acc < 97
22 and verified=0
23 and percentage >= 50;
24
25 -- FNs
26 select count(*) from sars_lir_disorder
27 where mobidb_method='simple'
28 and mobidb_data='predicted'
29 and lir_acc < 97
30 and verified=1
31 and percentage < 50;
```

Figure 23. Query examples for the computation of the TP, TN, FP, FN values for disorder predictions. The values derive from disorder data from MobiDB's simple method at 50% disorder (percentage < 50).

This database constitutes the stepping stone towards the development of another web-resource, or it could work as a future enhancement of the currently existing iLIR webserver. On top of that, the collection of Atg8 receptors can be used in other projects as well, a subject that is discussed in the following segment.

4.3.2 Learning from template structures

Having separated template complexes to receptors (Atg8 homologs) and ligands (3DLIR-motifs), the next step was to make an attempt to put them back together by employing protein-protein docking algorithms. It was essential that we created a reference set of trusted docking scores of known verified Atg8 binders in order to be able to interpret the results of

any downstream “virtual screening” (VS) experiments. Our reference dataset includes 5 distinct types of Atg8 homologs from 3 species: Human, Rat and yeast, which serve as the receptors and 3DLIR-motifs from 7 distinct LIRCPs originating from the same species, with an additional sample from Mouse (2ZJD).

Preliminary results from docking experiments (**Table 20**) conducted with the molecules of the template structures, although a relatively small dataset, they provide a broad spectrum of examples that demonstrate many of the obstacles that we will need to address for a correct evaluation of the results. The results were split into smaller segments isolating this way the docking scores per type of receptor.

IOANNA KALVARI

Experiment	Receptor Type	Receptor chain	Receptor Species	Uniprot Id	Ligand chain	Ligand Species	PDB ID	LIR-Motif Sequence	LIR tag	Length	3D Motif sequence	Verified	Docking Score
33721	MAP1LC3A	A	Human	FYCO1_HUMAN	E	Human	5CX3	AVFDII	LIR2	6	AVFDII	Yes	1152.35
33731	MAP1LC3A	A	Human	FYCO1_HUMAN	F	Human	5CX3	AVFDII	LIR2	6	AVFDII	Yes	1248.50
33741	MAP1LC3A	A	Human	FYCO1_HUMAN	G	Human	5CX3	AVFDII	LIR2	6	AVFDII	Yes	1188.43
33751	MAP1LC3A	A	Human	FYCO1_HUMAN	H	Human	5CX3	AVFDII	LIR2	6	AVFDII	Yes	1170.81
34351	MAP1LC3B	A	Human	FYCO1_HUMAN	B	Human	5D94	AVFDII	LIR2	6	AVFDII	Yes	1079.11
36801	MAP1LC3B	A	Rat	SQSTM_HUMAN	B	Rat	2K6Q	DDWTHL	LIR1	6	DDWTHL	Yes	1033.15
29891	MAP1LC3B	A	Human	SQSTM_HUMAN	B	Mouse	2ZJD	DDWTHL	LIR1	6	DDWTHL	Yes	1110.16
28801	MAP1LC3B	B	Rat	ATG4B_HUMAN	A	Human	2Z0E	LTYDTL	LIR1	6	LTYDTL	Yes	998.70
28201	MAP1LC3B	B	Rat	ATG4B_HUMAN	A	Human	2Z0D	LTYDTL	LIR1	6	LTYDTL	Yes	958.68
31161	MAP1LC3B	B	Rat	ATG4B_HUMAN	A	Human	2ZZP	LTYDTL	LIR1	6	LTYDTL	Yes	1158.17
32031	MAP1LC3C	B	Human	CACO2_HUMAN	A	Human	3VVW	DILVV	LIR2	5	DILVV	Yes	717.43
32041	MAP1LC3C	B	Human	CACO2_HUMAN	A	Human	3VVW	DILVV	LIR2	5	DILVV	Yes	800.83
27631	GABARAPL1	A	Human	NBR1_HUMAN	B	Human	2L8J	EDYIII	LIR2	6	EDYIII	Yes	1226.46
29911	Atg8	B	Yeast	Atg19_YEAST	E	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	802.03
29921	Atg8	B	Yeast	Atg19_YEAST	F	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	775.44
29931	Atg8	B	Yeast	Atg19_YEAST	G	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	834.26
29941	Atg8	B	Yeast	Atg19_YEAST	H	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	819.03
30501	Atg8	C	Yeast	Atg19_YEAST	E	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	781.45
30511	Atg8	C	Yeast	Atg19_YEAST	F	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	787.74
30521	Atg8	C	Yeast	Atg19_YEAST	G	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	830.80
30531	Atg8	C	Yeast	Atg19_YEAST	H	Yeast	2ZPN	LTWEEL	LIR1	4	WEEL	Yes	804.56
32541	Atg8	A	Yeast	Atg32_YEAST	B	Yeast	3VXW	GSWQAI	LIR1	6	GSWQAI	Yes	1273.54

Table 20. Top scoring conformations of the template structures.

The docking scores of the various Atg8 homologs are presented in distinct segments. Numbering of the LIR tags i.e. LIR1, LIR2 was done based on the location (start-end) of the LIR-motif on the sequence.

To analyse the docking results, one can look into the following aspects:

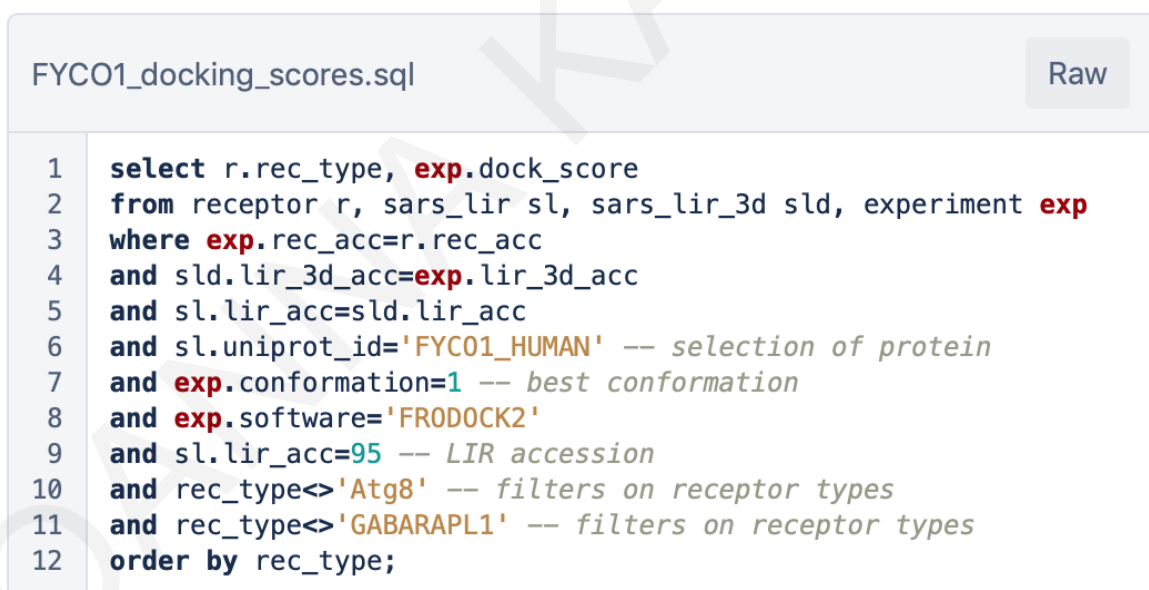
1. Species: We want to compare the scores we get with a receptor/ligand pair of the same species to a pair where the species differs (e.g. docking experiments with MAP1LC3B structures 2ZJD, 2Z0E, 2Z0D, 2ZZP)
2. 3DLIR length: Comparing these scores will allow us to understand how to evaluate the docking scores achieved with the non-canonical LIR-motifs, a case which we could not properly assess using the sequence-based methods presented in the first two chapters of this thesis
3. Ligand chains: In several templates, more than one instances of a 3DLIR-motif may be located (i.e. in different polypeptide chains) and these peptides may be in (slightly) different conformations and could behave differently in peptide docking experiments. Therefore, it is essential to grasp how 3DLIRs extracted from non-native structures would perform with the same receptor. This could be considered equivalent of a novel 3DLIR in the case we want to test its interaction with a protein of the Atg8 family
4. Receptor preference: LIRCPs have preference towards the various Atg8 proteins. Such evidence is the work of Rogov et al. (Rogov et al. 2017), which concentrates on LIR-motifs (GIM) that bind the GABARAP receptors
5. Positioning and orientation in the binding site: The correct amino acids need to interact with the two hydrophobic pockets of the Atg8 binding site. This will require additional visualization software and manual curation
6. Structure resolution: The resolution of the structures is also something that someone could look into. It's important to know what to expect when we have a novel 3DLIR of low resolution docked in a high-resolution binding site and vice versa. Structural data is very limited, so we need to be able to take advantage of as much as possible

4.3.2.1 A real use-case scenario driven by experimental evidence

Driven by the work of Olsvik et al. (Olsvik et al. 2015), which showed that FYCO1 has preference for LC3A and LC3B over GABARAP, we will be using data generated from our docking experiments in an attempt to examine whether we will be able to highlight this preference.

4.3.2.1.1 Forming a MySQL query to fetch the FYCO1 docking scores

The first step was to retrieve the docking scores of pre-ran docking experiments of the FYCO1 functional LIR-motif (AVFDII) to the binding site of MAP1LC3 (MAP1LC3A, MAP1LC3B, MAP1LC3C) and GABARAP structures. The corresponding query is depicted in **Figure 24**.



```
FYCO1_docking_scores.sql Raw
1  select r.rec_type, exp.dock_score
2  from receptor r, sars_lir sl, sars_lir_3d sld, experiment exp
3  where exp.rec_acc=r.rec_acc
4  and sld.lir_3d_acc=exp.lir_3d_acc
5  and sl.lir_acc=sld.lir_acc
6  and sl.uniprot_id='FYCO1_HUMAN' -- selection of protein
7  and exp.conformation=1 -- best conformation
8  and exp.software='FRODOCK2'
9  and sl.lir_acc=95 -- LIR accession
10 and rec_type<>'Atg8' -- filters on receptor types
11 and rec_type<>'GABARAPL1' -- filters on receptor types
12 order by rec_type;
```

Figure 24. MySQL query snippet for the retrieval of FYCO1/Atg8 docking scores.

Conformation=1 restricts the results to only the top conformations, `lir_acc` corresponds to an integer number which is the accession of a LIR-motif in the database where `lir_acc=95` in the `lir_acc` of FYCO1 functional LIR-motif AVFDII. Filtering of the receptors by limiting the results strictly to MAP1LC3 and GABARAP only is achieved with lines 10-11.

The query in **Figure 24**, fetches all top docking scores (`conformation=1`) of FYCO1 functional LIR-motif AVFDII ran using FRODOCK v.2.0 (line 8). Data retrieval requires joint information from 4 iLIR3D database tables: `receptor`, `sars_lir`, `sars_lir_3d` and

experiment based on receptor accession (rec_acc), 3DLIR accession (lir_3d_acc) and LIR-motif accession (lir_acc), which resulted in a total of 45 entries. Strict filtering to MAP1LC3 and GABARAP only is done with the help of lines 10 and 11, where ‘<>’ denotes that the receptor type (rec_type) should not be equal to ‘Atg8’ nor ‘GABARAPL1’. Simple grouping of the results is done by ordering the query outcome based on receptor type (rec_type).

The number of records retrieved per receptor are as follows:

- GABARAP: 5 entries
- MAP1LC3A: 5 entries
- MAP1LC3B: 30 entries
- MAP1LC3C: 5 entries

4.3.2.1.2 Assessing the docking results

The boxplot representation of the distributions of FYCO1 3DLIR docking scores against the 4 “receptors” GABARAP and MAP1LC3A-C (**Figure 25**) shows a preference towards the MAP1LC3 type, a result which is in agreement with work of Olsvik and colleagues (Olsvik et al. 2015). Further into details the computed median values of the docking scores were **800.31**, **1170.81**, **1064.06** and **1092.36** respectively and pairwise Wilcoxon rank sum tests yielded significant differences between the GABARAP and LC3 “receptors” ($p < 0.01$). This result showcases how powerful such a resource can be and suggests the scores of functional 3DLIR peptides could potentially reveal Atg8 homolog preference.

It would be very interesting to investigate whether we can discriminate between functional LIR-motifs via docking experiments, either by docking score comparison to template structures or visual inspection of the formed complex using visualisation tools like PyMOL (<https://pymol.org/2/>) (DELANO & L 2002) or UCSF Chimera (<http://www.cgl.ucsf.edu/chimera/>) (Pettersen et al. 2004). In in the latter case an expert curator/scientist would be looking for correct binding LIR-motif residues to the 2 hydrophobic pockets of the proteins of the Atg8 family.

FYCO1 3DLIR docked against different "receptors"

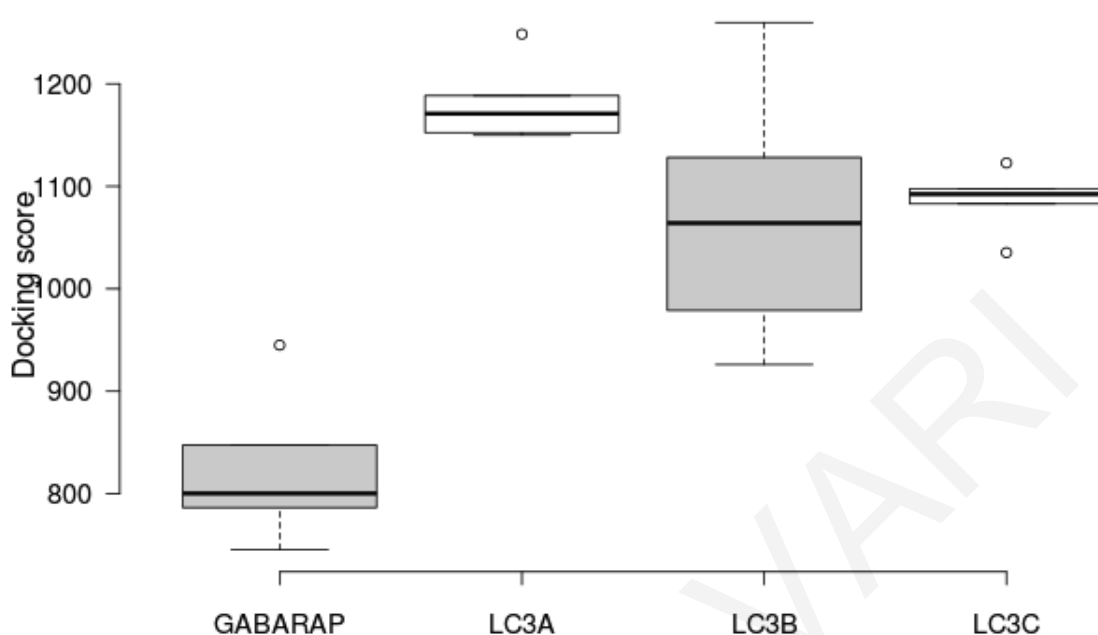


Figure 25. Boxplot representation of the distributions of scores of FYCO1 peptides docked against the Atg8 family.

FYCO1 peptides were docked against proteins of the Atg8 family, specifically GABARAP (N=5), LC3A (N=5), LC3B (N=30) and LC3C (N=5). Center lines show the medians; box limits indicate the 25th and 75th percentiles as determined by R software; whiskers extend 1.5 times the interquartile range from the 25th and 75th percentiles, outliers are represented by dots. Plot created using BoxPlotR (<http://shiny.chemgrid.org/boxplotr/>).

4.3.3 A comprehensive set of experiments

Following the docking of the template structures, we designed and conducted a comprehensive series of protein-protein experiments using the datasets mentioned in the data section.

The experiments involved the docking of all the ligands in the following list with each and every one of the receptors isolated from the template structures. The reason why we use those in the experiments is because we can compute the exact coordinates of the binding site from the interface of the interaction with the native 3DLIR. In addition to the calculation of the grid, those receptors are also in the correct conformation to and ready to receive a binding partner.

The experiments we conducted are the following:

1. Non-native verified LIR motifs: In this set of experiments we extracted as many verified LIR peptides available from the structures we downloaded and we docked those with the template receptors at the designated binding site
2. Unverified LIR motifs from native and non-native structures: This set of experiment aimed at producing a set of scores of non-functional LIRs to get a new range of docking scores of peptides that should not interact with the Atg8 family
3. Randomly selected peptides (decoy set): This will allow us to compare the results of docking experiments performed with peptides extracted from proteins known to be involved in autophagy, with the scores resulting from docking random peptides extracted from a collection of human proteins from diverse biological processes
4. 3DLIRs from non-autophagy proteins: With these experiments we wanted to test whether LIR peptides matching the sequences in **Table 1**, would produce the same results as the ones in the templates

These docking experiments resulted in **>100,000** samples that took weeks of computations to complete. In order to make these results useful to the scientific community we developed a relational database which described in detail in the following section.

4.3.4 Availability

The iLIR3D database is currently available as a MySQL dump that can be provided upon user request. We aim to make it more openly available to the scientific community after publishing initial analyses on this dataset. An instance of the database can be created within seconds following a creation of a new MySQL schema using the MySQL client CLI. Provision of the MySQL statements to build up an empty schema can also be provided, for users who want to use the schema to store their own data.

4.4 Conclusion

In this last chapter we attempted to “give form” to our collection of LIR-motifs by exploring the 3D world. We first discussed about the steps we took towards creating a structural dataset that would allow us to obtain a better understanding of protein-protein interactions between

selective autophagy receptor and adaptor proteins and the proteins of the Atg8 family. A process that revealed many issues afflicting this area, one of which is relatively limited availability of structural data compared to a great abundance of sequence data, which nevertheless resulted in two powerful outcomes:

1. A re-usable collection of Atg8 “receptors” for “virtual screening” experiments in autophagy
2. iLIR3D: A MySQL database that scientists can use to answer autophagy related biological questions

A straightforward application of the iLIR3D database would be to devise rules for LIR-motif specificities towards different Atg8/LC3/GAPARAP homologs. In fact, recent data (Rogov et al. 2017) can provide a nice ground truth dataset for predicting GABARAP versus LC3 specificity. The iLIR3D database (or its possible expansions) could be exploited to generalize to different types of specificity or even to predict interactions in heterologous systems. Preliminary results in line of with these expectations, is the FYCO1 preference towards the MAP1LC3 homologs the Atg8 family we showed using pre-ran docking experiments from the iLIR3D database. Our outcome was in also agreement with the experimental work of Olsvik and colleagues (Olsvik et al. 2015) showing preference of FYCO1 on the LC3 type, which makes the power iLIR3D even more apparent.

One important limitation of our methods that is very crucial to mention is that FRODOCK is a tool that performs rigid-body docking. The problem with this approach is that torsion angles, bond angles and bond lengths of the participating molecules do not change during the formation of the complex. This means that it is highly probable for a genuine 3DLIR to be misclassified if it is not initially in a 3D conformation that favours interaction with the binding site of the Atg8 proteins. Although a first taste of our results with the preceding examples shows that this may work, a more thorough investigation is required in order to confidently say whether rigid-body docking is sufficient or not. Perhaps a better solution to this would be the transitioning to flexible protein-protein docking (usually coming at a higher computational cost) as future work, that would also reflect the disorder to order nature of the proteins involved in selective macroautophagy.

Additional future activities to expand this line of research could also include:

- Model peptides with post-translational modifications, especially phosphorylation which is suspected to be important when within or in the proximity of LIR-motifs ((Birgisdottir et al. 2013); also important in other SLIM-mediated interactions)
- Build models of characterized Atg8/LC3/GABARAP proteins from other (model) species using comparative modeling or threading techniques. Execute the pipeline and populate the database.
- Enable incorporation of data stemming from molecular dynamics simulations.

Diverting from protein-protein docking, a recently published resource called Autophagic Compound Database (Deng et al. 2018), which makes autophagy effective compounds publicly available, along with useful data such as functionality, pathways, binding partners etc, hints another future direction of this research area. Although this resource does not seem very user friendly at its current state, long standing titans like ChEMBL (Gaulton et al. 2012; Gaulton et al. 2017) - a manually curated chemical database of bioactive molecules with drug-like properties - could constitute a potential resource in search of good chemical compound candidates. In such case we could re-use our current set of Atg8 “receptors” in virtual screening experiments, to looking for possible targets that could treat autophagy associated diseases. A broadly used and very efficient software in molecular docking that we also used in previous projects is Autodock Vina (Trott & Olson 2010).

Finally, as in this chapter we are discussing about molecular interactions, another aspect that would be very interesting to explore is the interaction of the proteins of the autophagic machinery with regulatory elements such as non-coding RNAs. In a pre-print released by Horos and colleagues showed evidence of such interaction of the the Vault RNA with the Zinc finger of p62 (Horos et al. 2017), suggesting regulation of autophagy by non-coding RNAs. A very intriguing finding and another potential target for the development of treatments to autophagy related diseases (Amort et al. 2015).

5 Discussion and Future Goals

Even though this research field has been around for a couple of decades, our understanding of the mechanics and the dynamics of this biological process is still at its infancy with many different paths to be explored. Speaking of which, Richard S. Marshall and Richard D. Vierstra in their very recent review in *Plant Biology* catalogued at least 7 different types of selective autophagy including mitophagy, chlorophagy, xenophagy, pexophagy and many others (Marshall & Vierstra 2018).

The pioneering works of Pankiv (Pankiv et al. 2007), Ichimura (Ichimura et al. 2008), Noda (Noda et al. 2008; Noda et al. 2010), Alemu and colleagues (Alemu et al. 2012) were pivotal for the definitions and characterizations of the AIM/LIR-motifs, which paved the way for the development and establishment of computational approaches for the in-silico identification of novel key players of the autophagic machinery. A tool made available to the scientific community is iLIR (Kalvari et al. 2014), which was also the first of its kind. Since its release back in 2013, iLIR seems to have served more than 70 thousand user queries, but also appears to have influenced and driven the development of analogous resources such as the hfAIM (Xie et al. 2016). Although the two resources have significant differences - xLIR composed majorly from human LIR-motifs, hfAIMs composed from proteins involved in pexophagy - they both rely on regular expressions for the identification of putative LIR-motifs, which are somewhat limiting. iLIR however, by incorporating ANCHOR predictions and PSSMs manages to balance out the gap between sensitivity and specificity, resulting in high balanced accuracy. Importantly, the iLIR server is built in such a way that it does not provide 'yes'-'no' type of predictions, but rather reports all possibly relevant biological information: apart from the highly sensitive (but also inspecific xLIR-motifs), WxxL motifs are also reported, along with predicted ANCHORs and PSSM scores. Additional contextual information (e.g. the presence of specific PFAM domains or low complexity regions) becomes also available to its end-users for making informative decisions with regards to downstream experimental validation of specific LIR-motif candidates.

Based on the methods developed for iLIR, batch processing of the complete proteomes of 8 model organisms lead to the development of a freely available database resource for the provision of a collection of LIRCPs (Jacomin et al. 2016). A follow-up work which identified putative LIRCPs in viral species, resulted in the development of a similar database specific

to viruses, namely iLIR@viral (Jacomin et al. 2017). It can be envisaged that, eventually, all sequenced genomes or protein sequences available in sequence databases can be scanned with iLIR (or its successors) and made available to the scientific community.

Driven by the fact that proteins of the autophagic apparatus are abundant in intrinsic disorder regions (Mei et al. 2014) and based on our previous positive results with ANCHOR, we consequently turned to intrinsic disorder data to see whether such information could further enhance the power of our multi-scheme predictors. With data retrieved from MobiDB, we employed a variety of algorithms in search for one that would best fit our data. Compared to our previous results this work yielded even higher balanced accuracies, an outcome that was further improved with the parametrization of the PSSM scores, revealing a PSSM sweetspot at scores > 17 .

Additional predicted features, such as secondary structure, surface accessibility and amyloidogenicity have been tested as independent parameters for filtering LIR motif prediction but without significant results (data not shown). However, we foresee that several of these (and possibly other physicochemical) features could be incorporated into more sophisticated techniques for discriminating functional LIR motifs. For example, as more experimentally verified LIRCPs become available it can be envisaged that powerful machine learning methods (e.g. deep learning artificial neural networks) could be recruited to boost prediction performance.

To allow a different dynamic to our data, we diverted from sequence analysis and transitioned to structural data. As we showcased with the example of FYCO1 preference of MAP1LC3 proteins of the Atg8 family, which is backed up by the experimental work of Olsvik and colleagues (Olsvik et al. 2015), such data could potentially enable the scientific community to give answers to biological questions that sequence data fail to capture. For instance, in their recent work Rogov et al (Rogov et al. 2017) pinpointed the preference of autophagy receptor and adaptor proteins toward the GABARAP Atg8 homologs and in addition to the previously defined AIM and LIR-motifs, introduced for the first time the GABARAP Interacting Motifs (GIMs). It would be very intriguing to examine whether our data could be used to discriminate autophagy proteins based on their Atg8 preference.

Moving on to further structural aspects of the selective autophagy receptor and adaptor proteins and their interactions with Atg8 homologs, the recently determined structure of TRIM5 α in complex with LC3B (Keown et al. 2018) provided fresh insights to features that

we previously eluded. In contrast to the conventional intermolecular parallel β -strand interaction between the LIR-motifs and the Atg8 homologs demonstrated so far, these proteins bind to the Atg8 proteins via an α -helix of their coiled coil domain (Mandell et al. 2014). It is anticipated that such proteins will give rise to new research. For example, let's assume that we have a novel protein candidate that binds to an Atg8 homolog via an α -helix with a presumably good docking score too. Before the existence of experimental evidence to support this unorthodox interaction, a TRIM protein could be falsely discarded from a list of positive samples. This suggests that we can expect more surprises in the years to come and leaves the field open for many more discoveries.

Moreover, post-translational modifications (in particular phosphorylation) can be important around (or within) LIR motifs. Thus, post-translational modification predictions could be combined in the sequence-based prediction and/or incorporated in structural modeling/peptide docking experiments.

Before closing we would like to make apparent that we are aiming to use the knowledge we acquired from this project towards the development of novel and improved tools to better serve the scientific community. For instance, our analysis on disorder data suggested that multi-scheme predictor **xLIR+A2|D|P17** to be the optimal for the most accurate determination of functional LIR-motifs. A possible future improvement would be to fine tune the iLIR web server to take into account intrinsic disorder and predict novel LIRCP instances based on this new multi-scheme predictor. Another potential feature would be to allow batch searches and even upgrade the iLIR web server with more modernized web technologies e.g. AngularJS (<https://angularjs.org/>), Django framework (<https://www.djangoproject.com/>).

We are currently working on expanding our existing LIRCP datasets with manual literature curation (Kalvari and Chadjichristofi, currently underway), a work that may result in better benchmarks, but also in the compilation of suitable datasets. We anticipate that the analysis of a comprehensive dataset will demand utilization of more sophisticated methods e.g. machine learning algorithms.

Finally, solely from personal interest it would be very intriguing to explore the world of RNA-binding proteins (RBPs). In particular the interactions between the proteins of the autophagic machinery and regulatory RNAs, non-coding RNAs. There is evidence showing ncRNAs being recruited to phagophores and ending up to the lysosomes where they get

degraded (Frankel et al. 2017). Preliminary results from Horos and colleagues shed light on the regulation of autophagy by non-coding RNAs, with the Vault RNA interacting with the Zinc finger of p62 (Horos et al. 2017).

One interesting RNA-meets-autophagy topic to explore would be to take all known/predicted LIRCPs from human (or other model species), catalog experimental and/or predicted miRNA sites on the respective genes and see when/whether/which of these miRNA sites get spliced out in alternatively spliced transcripts. Such data are sitting in existing resources and waiting to be analysed. RNAcentral (The RNAcentral Consortium 2018) - the comprehensive database of non-coding RNAs - currently combines non-coding RNA data from 28 expert ncRNA databases constituting a very strong candidate from where we could collect miRNAs, whereas an extensive set of human LIRCPs can be obtained from iLIR database (Jacomini et al. 2016). Exciting times lie ahead.

6 References

- Alemu, E.A. et al., 2012. ATG8 family proteins act as scaffolds for assembly of the ULK complex: sequence requirements for LC3-interacting region (LIR) motifs. *The Journal of biological chemistry*, 287(47), pp.39275–39290.
- Amaravadi, R., Kimmelman, A.C. & White, E., 2016. Recent insights into the function of autophagy in cancer. *Genes & development*, 30(17), pp.1913–1930.
- Amort, M. et al., 2015. Expression of the vault RNA protects cells from undergoing apoptosis. *Nature communications*, 6, p.7030.
- Avin-Wittenberg, T. & Fernie, A.R., 2014. At long last: evidence for pexophagy in plants. *Molecular plant*, 7(8), pp.1257–1260.
- Babicki, Sasha, David Arndt, Ana Marcu, Yongjie Liang, Jason R. Grant, Adam Maciejewski, and David S. Wishart. 2016. “Heatmapper: Web-Enabled Heat Mapping for All.” *Nucleic Acids Research* 44 (W1): W147–53
- Baldi, P. et al., 2000. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics* , 16(5), pp.412–424.
- Behrends, C. et al., 2010. Network organization of the human autophagy system. *Nature*, 466(7302), pp.68–76.
- Berman, H.M. et al., 2000. The Protein Data Bank. *Nucleic acids research*, 28(1), pp.235–242.
- Birgisdottir, Å.B., Lamark, T. & Johansen, T., 2013. The LIR motif - crucial for selective autophagy. *Journal of cell science*, 126(Pt 15), pp.3237–3247.
- Blast, G., 1997. PSI-BLAST: a new generation of protein database search programs Altschul. *Stephen F*, pp.3389–3402.
- Boyle, K.B. & Randow, F., 2013. The role of “eat-me” signals and autophagy cargo receptors in innate immunity. *Current opinion in microbiology*, 16(3), pp.339–348.
- Darling, A.L. & Uversky, V.N., 2018. Intrinsic Disorder and Posttranslational Modifications: The Darker Side of the Biological Dark Matter. *Frontiers in genetics*, 9, p.158.
- Davey, N.E. et al., 2012. Attributes of short linear motifs. *Molecular bioSystems*, 8(1),

pp.268–281.

Degenhardt, K. et al., 2006. Autophagy promotes tumor cell survival and restricts necrosis, inflammation, and tumorigenesis. *Cancer cell*, 10(1), pp.51–64.

DELANO & L, W., 2002. The PyMOL Molecular Graphics System. <http://www.pymol.org>. Available at: <https://ci.nii.ac.jp/naid/10020095229/> [Accessed November 12, 2018].

Deng, Y. et al., 2018. Autophagic compound database: A resource connecting autophagy-modulating compounds, their potential targets and relevant diseases. *Cell proliferation*, 51(3), p.e12403.

Derbyshire, Dean J., Balaku P. Basu, Louise C. Serpell, Woo S. Joo, Takayasu Date, Kuniyoshi Iwabuchi, and Aidan J. Doherty. 2002. “Crystal Structure of Human 53BP1 BRCT Domains Bound to p53 Tumour Suppressor.” *The EMBO Journal* 21 (14): 3863–72.

Dyson, H.J. & Wright, P.E., 2005. Intrinsically unstructured proteins and their functions. *Nature reviews. Molecular cell biology*, 6(3), pp.197–208.

Fotin, A. et al., 2004. Structure of an auxilin-bound clathrin coat and its implications for the mechanism of uncoating. *Nature*, 432(7017), pp.649–653.

Frankel, L.B., Lubas, M. & Lund, A.H., 2017. Emerging connections between RNA and autophagy. *Autophagy*, 13(1), pp.3–23.

Gao, C. et al., 2010. Autophagy negatively regulates Wnt signalling by promoting Dishevelled degradation. *Nature cell biology*, 12(8), pp.781–790.

Garzon, J.I. et al., 2009. FRODOCK: a new approach for fast rotational protein-protein docking. *Bioinformatics*, 25(19), pp.2544–2551.

Gaulton, A. et al., 2012. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(Database issue), pp.D1100–7.

Gaulton, A. et al., 2017. The ChEMBL database in 2017. *Nucleic acids research*, 45(D1), pp.D945–D954.

Hanson, J. et al., 2017. Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks. *Bioinformatics*, 33(5), pp.685–692.

Horos, R. et al., 2017. The small non-coding vault RNA1-1 acts as a riboregulator of

- autophagy. *bioRxiv*, p.177949. Available at: <https://www.biorxiv.org/content/early/2017/08/18/177949> [Accessed November 4, 2018].
- Ichimura, Y. et al., 2008. Structural basis for sorting mechanism of p62 in selective autophagy. *The Journal of biological chemistry*, 283(33), pp.22847–22857.
- Itoh, T. et al., 2011. OATL1, a novel autophagosome-resident Rab33B-GAP, regulates autophagosomal maturation. *The Journal of cell biology*, 192(5), pp.839–853.
- Jacomin, A.-C. et al., 2016. iLIR database: A web resource for LIR motif-containing proteins in eukaryotes. *Autophagy*, 12(10), pp.1945–1953.
- Jacomin, A.-C. et al., 2017. iLIR@viral: A web resource for LIR motif-containing proteins in viruses. *Autophagy*, 13(10), pp.1782–1789.
- Jiang, S., Wells, C.D. & Roach, P.J., 2011. Starch-binding domain-containing protein 1 (Stbd1) and glycogen metabolism: Identification of the Atg8 family interacting motif (AIM) in Stbd1 required for interaction with GABARAP1. *Biochemical and biophysical research communications*, 413(3), pp.420–425.
- Johansen, T. & Lamark, T., 2011. Selective autophagy mediated by autophagic adapter proteins. *Autophagy*, 7(3), pp.279–296.
- Kalvari, I. et al., 2014. iLIR: A web resource for prediction of Atg8-family interacting proteins. *Autophagy*, 10(5), pp.913–925.
- Karlin, S., and S. F. Altschul. 1990. “Methods for Assessing the Statistical Significance of Molecular Sequence Features by Using General Scoring Schemes.” *Proceedings of the National Academy of Sciences of the United States of America* 87 (6): 2264–68.
- Kelly, J.W., 1996. Alternative conformations of amyloidogenic proteins govern their behavior. *Current opinion in structural biology*, 6(1), pp.11–17.
- Keown, J.R. et al., 2018. A helical LC3-interacting region mediates the interaction between the retroviral restriction factor Trim5 α and mammalian autophagy-related ATG8 proteins. *The Journal of biological chemistry*, 293(47), pp.18378–18386.
- Kirkin, V. et al., 2009. A role for NBR1 in autophagosomal degradation of ubiquitinated substrates. *Molecular cell*, 33(4), pp.505–516.
- Knodler, L.A. & Celli, J., 2011. Eating the strangers within: host control of intracellular

- bacteria via xenophagy. *Cellular microbiology*, 13(9), pp.1319–1327.
- Kraft, C. et al., 2012. Binding of the Atg1/ULK1 kinase to the ubiquitin-like protein Atg8 regulates autophagy. *The EMBO journal*, 31(18), pp.3691–3703.
- Krissinel, E. & Henrick, K., 2007. Inference of macromolecular assemblies from crystalline state. *Journal of molecular biology*, 372(3), pp.774–797.
- Krystkowiak, I., Manguy, J. & Davey, N.E., 2018. PSSMSearch: a server for modeling, visualization, proteome-wide discovery and annotation of protein motif specificity determinants. *Nucleic acids research*, 46(W1), pp.W235–W241.
- Lamark, T. & Johansen, T., 2012. Aggrephagy: selective disposal of protein aggregates by macroautophagy. *International journal of cell biology*, 2012, p.736905.
- Lamark, T., Svenning, S. & Johansen, T., 2017. Regulation of selective autophagy: the p62/SQSTM1 paradigm. *Essays in biochemistry*, 61(6), pp.609–624.
- Letunic, I., Doerks, T. & Bork, P., 2012. SMART 7: recent updates to the protein domain annotation resource. *Nucleic acids research*, 40(Database issue), pp.D302–5.
- Linding, R., Jensen, L.J., et al., 2003. Protein disorder prediction: implications for structural proteomics. *Structure*, 11(11), pp.1453–1459.
- Linding, R., Russell, R.B., et al., 2003. GlobPlot: Exploring protein sequences for globularity and disorder. *Nucleic acids research*, 31(13), pp.3701–3708.
- Lipton, Z.C., Elkan, C. & Naryanaswamy, B., 2014. Optimal Thresholding of Classifiers to Maximize F1 Measure. *Machine learning and knowledge discovery in databases : European Conference, ECML PKDD ... : proceedings. ECML PKDD (Conference)*, 8725, pp.225–239.
- Liu, L. et al., 2012. Mitochondrial outer-membrane protein FUNDC1 mediates hypoxia-induced mitophagy in mammalian cells. *Nature cell biology*, 14(2), pp.177–185.
- Lynch-Day, M.A. et al., 2012. The role of autophagy in Parkinson's disease. *Cold Spring Harbor perspectives in medicine*, 2(4), p.a009357.
- Mandell, M.A. et al., 2014. TRIM proteins regulate autophagy: TRIM5 is a selective autophagy receptor mediating HIV-1 restriction. *Autophagy*, 10(12), pp.2387–2388.
- Markstedt, O., 2017. Kubernetes as an approach for solving bioinformatic problems.

Available at: <https://uu.diva-portal.org/smash/get/diva2:1145028/FULLTEXT01.pdf>.

- Marshall, R.S. & Vierstra, R.D., 2018. Autophagy: The Master of Bulk and Selective Recycling. *Annual review of plant biology*, 69, pp.173–208.
- Martinet, W. et al., 2007. Autophagy in cardiovascular disease. *Trends in molecular medicine*, 13(11), pp.482–491.
- McEwan, D.G. et al., 2015. PLEKHM1 regulates autophagosome-lysosome fusion through HOPS complex and LC3/GABARAP proteins. *Molecular cell*, 57(1), pp.39–54.
- Mei, Y. et al., 2014. Intrinsically disordered regions in autophagy proteins. *Proteins*, 82(4), pp.565–578.
- Mei, Y. et al., 2015. Autophagy and oxidative stress in cardiovascular diseases. *Biochimica et biophysica acta*, 1852(2), pp.243–251.
- Meireles, L.M.C., Dömling, A.S. & Camacho, C.J., 2010. ANCHOR: a web server and database for analysis of protein-protein interaction binding pockets for drug discovery. *Nucleic acids research*, 38(Web Server issue), pp.W407–11.
- Mészáros, B., Erdos, G. & Dosztányi, Z., 2018. IUPred2A: context-dependent prediction of protein disorder as a function of redox state and protein binding. *Nucleic acids research*, 46(W1), pp.W329–W337.
- Mezei, M., 1998. Chameleon sequences in the PDB. *Protein engineering*, 11(6), pp.411–414.
- Miskei, M., Antal, C. & Fuxreiter, M., 2017. FuzDB: database of fuzzy complexes, a tool to develop stochastic structure-function relationships for protein complexes and higher-order assemblies. *Nucleic acids research*, 45(D1), pp.D228–D235.
- Mizushima, N. & Levine, B., 2010. Autophagy in mammalian development and differentiation. *Nature cell biology*, 12(9), pp.823–830.
- Mizushima, N., 2007. Autophagy: process and function. *Genes & development*, 21(22), pp.2861–2873.
- Mohrlüder, J., Hoffmann, Y., et al., 2007. Identification of clathrin heavy chain as a direct interaction partner for the gamma-aminobutyric acid type A receptor associated protein. *Biochemistry*, 46(50), pp.14537–14543.

- Mohrlüder, J., Stangler, T., et al., 2007. Identification of calreticulin as a ligand of GABARAP by phage display screening of a peptide library. *The FEBS journal*, 274(21), pp.5543–5555.
- Moreno, M.-L. et al., 2018. Autophagy Dysfunction and Oxidative Stress, Two Related Mechanisms Implicated in Retinitis Pigmentosa. *Frontiers in physiology*, 9, p.1008.
- Mukhopadhyay, S. et al., 2014. Autophagy and apoptosis: where do they meet? *Apoptosis: an international journal on programmed cell death*, 19(4), pp.555–566.
- Nakatogawa, H. et al., 2012. The autophagy-related protein kinase Atg1 interacts with the ubiquitin-like protein Atg8 via the Atg8 family interacting motif to facilitate autophagosome formation. *The Journal of biological chemistry*, 287(34), pp.28503–28507.
- Necci, M. et al., 2017. MobiDB-lite: fast and highly specific consensus prediction of intrinsic disorder in proteins. *Bioinformatics*, 33(9), pp.1402–1404.
- Nevill-Manning, C.G., Wu, T.D. & Brutlag, D.L., 1998. Highly specific protein sequence motifs for genome analysis. *Proceedings of the National Academy of Sciences of the United States of America*, 95(11), pp.5865–5871.
- Newman, A.C. et al., 2012. TBK1 kinase addiction in lung cancer cells is mediated via autophagy of Tax1bp1/Ndp52 and non-canonical NF- κ B signalling. *PloS one*, 7(11), p.e50672.
- Noda, N.N. et al., 2008. Structural basis of target recognition by Atg8/LC3 during selective autophagy. *Genes to cells: devoted to molecular & cellular mechanisms*, 13(12), pp.1211–1218.
- Noda, N.N., Ohsumi, Y. & Inagaki, F., 2010. Atg8-family interacting motif crucial for selective autophagy. *FEBS letters*, 584(7), pp.1379–1385.
- Novak, I. et al., 2010. Nix is a selective autophagy receptor for mitochondrial clearance. *EMBO reports*, 11(1), pp.45–51.
- Novella, J.A. et al., 2018. Container-based bioinformatics with Pachyderm. *Bioinformatics*. Available at: <http://dx.doi.org/10.1093/bioinformatics/bty699>.
- O’Boyle, N.M. et al., 2011. Open Babel: An open chemical toolbox. *Journal of cheminformatics*, 3, p.33.

- Okamoto, K., Kondo-Okamoto, N. & Ohsumi, Y., 2009. Mitochondria-anchored receptor Atg32 mediates degradation of mitochondria via selective autophagy. *Developmental cell*, 17(1), pp.87–97.
- Oldfield, C.J. & Dunker, A.K., 2014. Intrinsically disordered proteins and intrinsically disordered protein regions. *Annual review of biochemistry*, 83, pp.553–584.
- Olsvik, H.L. et al., 2015. FYCO1 Contains a C-terminally Extended, LC3A/B-preferring LC3-interacting Region (LIR) Motif Required for Efficient Maturation of Autophagosomes during Basal Autophagy. *The Journal of biological chemistry*, 290(49), pp.29361–29374.
- Onodera, J. & Ohsumi, Y., 2005. Autophagy is required for maintenance of amino acid levels and protein synthesis under nitrogen starvation. *The Journal of biological chemistry*, 280(36), pp.31582–31586.
- Palikaras, K., Lionaki, E. & Tavernarakis, N., 2018. Mechanisms of mitophagy in cellular homeostasis, physiology and pathology. *Nature cell biology*, 20(9), pp.1013–1022.
- Pankiv, S. et al., 2007. p62/SQSTM1 binds directly to Atg8/LC3 to facilitate degradation of ubiquitinated protein aggregates by autophagy. *The Journal of biological chemistry*, 282(33), pp.24131–24145.
- Pankiv, S. et al., 2010. FYCO1 is a Rab7 effector that binds to LC3 and PI3P to mediate microtubule plus end-directed vesicle transport. *The Journal of cell biology*, 188(2), pp.253–269.
- Parzych, K.R. & Klionsky, D.J., 2014. An overview of autophagy: morphology, mechanism, and regulation. *Antioxidants & redox signaling*, 20(3), pp.460–473.
- Peng, K. et al., 2006. Length-dependent prediction of protein intrinsic disorder. *BMC bioinformatics*, 7, p.208.
- Pettersen, E.F. et al., 2004. UCSF Chimera--a visualization system for exploratory research and analysis. *Journal of computational chemistry*, 25(13), pp.1605–1612.
- Piovesan, D. et al., 2017. DisProt 7.0: a major update of the database of disordered proteins. *Nucleic acids research*, 45(D1), pp.D219–D227.
- Piovesan, D. et al., 2018. MobiDB 3.0: more annotations for intrinsic disorder, conformational diversity and interactions in proteins. *Nucleic acids research*, 46(D1),

pp.D471–D476.

- Popovic, D. et al., 2012. Rab GTPase-activating proteins in autophagy: regulation of endocytic and autophagy pathways by direct binding to human ATG8 modifiers. *Molecular and cellular biology*, 32(9), pp.1733–1744.
- Promponas, V.J. et al., 2000. CAST: an iterative algorithm for the complexity analysis of sequence tracts. *Bioinformatics*, 16(10), pp.915–922.
- Punta, M. et al., 2012. The Pfam protein families database. *Nucleic acids research*, 40(Database issue), pp.D290–301.
- Qu, X. et al., 2007. Autophagy gene-dependent clearance of apoptotic cells during embryonic development. *Cell*, 128(5), pp.931–946.
- Racanelli, A.C. et al., 2018. Autophagy and inflammation in chronic respiratory disease. *Autophagy*, 14(2), pp.221–232.
- Ramírez-Aportela, E., López-Blanco, J.R. & Chacón, P., 2016. FRODOCK 2.0: fast protein-protein docking server. *Bioinformatics*, 32(15), pp.2386–2388.
- Rocchi, A. & He, C., 2015. Emerging roles of autophagy in metabolism and metabolic disorders. *Frontiers of biology*, 10(2), pp.154–164.
- Rogov, V. et al., 2014. Interactions between autophagy receptors and ubiquitin-like proteins form the molecular basis for selective autophagy. *Molecular cell*, 53(2), pp.167–178.
- Rogov, V.V. et al., 2017. Structural and functional analysis of the GABARAP interaction motif (GIM). *EMBO reports*, p.e201643587.
- Rose, P.W. et al., 2015. The RCSB Protein Data Bank: views of structural biology for basic and applied research and education. *Nucleic acids research*, 43(Database issue), pp.D345–56.
- Rowell, John P., Kathryn L. Simpson, Katherine Stott, Matthew Watson, and Jean O. Thomas. 2012. “HMGB1-Facilitated p53 DNA Binding Occurs via HMG-Box/p53 Transactivation Domain Interaction, Regulated by the Acidic Tail.” *Structure* 20 (12): 2014–24.
- Rubinsztein, D.C., Mariño, G. & Kroemer, G., 2011. Autophagy and aging. *Cell*, 146(5), pp.682–695.

- Ryter, S.W. & Choi, A.M.K., 2015. Autophagy in lung disease pathogenesis and therapeutics. *Redox biology*, 4, pp.215–225.
- Sancho, A. et al., 2012. DOR/Tp53inp2 and Tp53inp1 constitute a metazoan gene family encoding dual regulators of autophagy and transcription. *PloS one*, 7(3), p.e34034.
- Sandilands, E. et al., 2011. Autophagic targeting of Src promotes cancer cell survival following reduced FAK signalling. *Nature cell biology*, 14(1), pp.51–60.
- Santana-Codina, N., Mancias, J. & Kimmelman, A.C., 2017. The Role of Autophagy in Cancer. *Annual Review of Cancer Biology*, 1(1), pp.19–39.
- Satoo, K. et al., 2009. The structure of Atg4B-LC3 complex reveals the mechanism of LC3 processing and delipidation during autophagy. *The EMBO journal*, 28(9), pp.1341–1350.
- Schad, E. et al., 2018. DIBS: a repository of disordered binding sites mediating interactions with ordered proteins. *Bioinformatics*, 34(3), pp.535–537.
- Schreiber, A. & Peter, M., 2014. Substrate recognition in selective autophagy and the ubiquitin-proteasome system. *Biochimica et biophysica acta*, 1843(1), pp.163–181.
- Scott, R.C., Schuldiner, O. & Neufeld, T.P., 2004. Role and regulation of starvation-induced autophagy in the Drosophila fat body. *Developmental cell*, 7(2), pp.167–178.
- Shpilka, T. et al., 2011. Atg8: an autophagy-related ubiquitin-like protein family. *Genome biology*, 12(7), p.226.
- Singh, A.P., Mishra, S. & Jabin, S., 2018. Sequence based prediction of enhancer regions from DNA random walk. *Scientific reports*, 8(1), p.15912.
- Skytte Rasmussen, M. et al., 2017. ATG4B contains a C-terminal LIR motif important for binding and efficient cleavage of mammalian orthologs of yeast Atg8. *Autophagy*, 13(5), pp.834–853.
- Stadel, D. et al., 2015. TECPR2 Cooperates with LC3C to Regulate COPII-Dependent ER Export. *Molecular cell*, 60(1), pp.89–104.
- Stolz, A., Ernst, A. & Dikic, I., 2014. Cargo recognition and trafficking in selective autophagy. *Nature cell biology*, 16(6), pp.495–501.
- Suad, Oded, Haim Rozenberg, Ran Brosh, Yael Diskin-Posner, Naama Kessler, Linda J. W.

- Shimon, Felix Frolov, Atar Liran, Varda Rotter, and Zippora Shakked. 2009. “Structural Basis of Restoring Sequence-Specific DNA Binding and Transactivation to Mutant p53 by Suppressor Mutations.” *Journal of Molecular Biology* 385 (1): 249–65.
- Svenning, S. et al., 2011. Plant NBR1 is a selective autophagy substrate and a functional hybrid of the mammalian autophagic adapters NBR1 and p62/SQSTM1. *Autophagy*, 7(9), pp.993–1010.
- The RNAcentral Consortium, 2018. RNAcentral: a hub of information for non-coding RNA sequences. *Nucleic acids research*. Available at: <http://dx.doi.org/10.1093/nar/gky1034>.
- The UniProt Consortium, 2018. UniProt: the universal protein knowledgebase. *Nucleic acids research*. Available at: <http://dx.doi.org/10.1093/nar/gky092>.
- Trott, O. & Olson, A.J., 2010. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2), pp.455–461.
- Tsolis, A.C. et al., 2013. A consensus method for the prediction of “aggregation-prone” peptides in globular proteins. *PloS one*, 8(1), p.e54175.
- Uddin, M.S. et al., 2018. Autophagy and Alzheimer’s Disease: From Molecular Mechanisms to Therapeutic Implications. *Frontiers in aging neuroscience*, 10, p.04.
- Ueno, T. & Komatsu, M., 2017. Autophagy in the liver: functions in health and disease. *Nature reviews. Gastroenterology & hepatology*, 14(3), pp.170–184.
- von Muhlinen, N. et al., 2012. LC3C, bound selectively by a noncanonical LIR motif in NDP52, is required for antibacterial autophagy. *Molecular cell*, 48(3), pp.329–342.
- Walsh, I. et al., 2012. ESpritz: accurate and fast prediction of protein disorder. *Bioinformatics*, 28(4), pp.503–509.
- Wang, B. et al., 2016. Dysregulation of autophagy and mitochondrial function in Parkinson’s disease. *Translational neurodegeneration*, 5, p.19.
- Wei, L. et al., 2018. Comparative analysis and prediction of quorum-sensing peptides using feature representation learning and machine learning algorithms. *Briefings in bioinformatics*. Available at: <http://dx.doi.org/10.1093/bib/bby107>.
- White, Eileen, Janice M. Mehnert, and Chang S. Chan. 2015. “Autophagy, Metabolism, and

Cancer.” *Clinical Cancer Research: An Official Journal of the American Association for Cancer Research* 21 (22): 5037–46.

Wild, P. et al., 2011. Phosphorylation of the autophagy receptor optineurin restricts Salmonella growth. *Science*, 333(6039), pp.228–233.

Wirawan, Ellen, Tom Vanden Berghe, Saskia Lippens, Patrizia Agostinis, and Peter Vandenabeele. 2012. “Autophagy: For Better or for Worse.” *Cell Research* 22 (1): 43–61.

Wright, P.E. & Dyson, H.J., 1999. Intrinsically unstructured proteins: re-assessing the protein structure-function paradigm. *Journal of molecular biology*, 293(2), pp.321–331.

Wright, P.E. & Dyson, H.J., 2015. Intrinsically disordered proteins in cellular signalling and regulation. *Nature reviews. Molecular cell biology*, 16(1), pp.18–29.

Xie, Q. et al., 2016. hfAIM: A reliable bioinformatics approach for in silico genome-wide identification of autophagy-associated Atg8-interacting motifs in various organisms. *Autophagy*, 12(5), pp.876–887.

Yamaguchi, M. et al., 2010. Autophagy-related protein 8 (Atg8) family interacting motif in Atg3 mediates the Atg3-Atg8 interaction and is crucial for the cytoplasm-to-vacuole targeting pathway. *The Journal of biological chemistry*, 285(38), pp.29599–29607.

Yang, Z.R. et al., 2005. RONN: the bio-basis function neural network technique applied to the detection of natively disordered regions in proteins. *Bioinformatics* , 21(16), pp.3369–3376.

Yang, Zhifen, and Daniel J. Klionsky. 2009. “An Overview of the Molecular Mechanism of Autophagy.” *Current Topics in Microbiology and Immunology* 335: 1–32.

Zaffagnini, G. & Martens, S., 2016. Mechanisms of Selective Autophagy. *Journal of molecular biology*, 428(9 Pt A), pp.1714–1724.

Zaffagnini, G. et al., 2018. Phasing out the bad-How SQSTM1/p62 sequesters ubiquitinated proteins for degradation by autophagy. *Autophagy*, 14(7), pp.1280–1282.

7 Supplement

UniProt Accession	UniProt ID	PDB ID	#Chains	Resolution (Å)	Method	Function	Template
C0H519	C0H519_PLAF7	4EOY	6	2.22	X-RAY DIFFRACTION	TRANSPORT PROTEIN	Yes
O14641	DVL2_HUMAN	5SUZ	2	1.84	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		3CBX	2	1.7	X-RAY DIFFRACTION	PROTEIN BINDING	No
		2REY	1	1.55	X-RAY DIFFRACTION	GENE REGULATION	No
		3CC0	3	1.75	X-RAY DIFFRACTION	PROTEIN BINDING	No
		3CBY	2	1.5	X-RAY DIFFRACTION	PROTEIN BINDING	No
		4WIP	3	2.69	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		5LNP	4	1.99	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		5SUY	4	1.88	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		3CBZ	1	1.38	X-RAY DIFFRACTION	PROTEIN BINDING	No
O75143	ATG13_HUMAN	3WAO	4	2.6	X-RAY DIFFRACTION	APOPTOSIS	No
		5C50	2	1.63	X-RAY DIFFRACTION	PROTEIN BINDING	No
		3WAP	1	3.1	X-RAY DIFFRACTION	PROTEIN TRANSPORT	No
		3WAN	2	1.77	X-RAY DIFFRACTION	PROTEIN BINDING	No
O75385	ULK1_HUMAN	4WNP	4	1.88	X-RAY DIFFRACTION	TRANSFERASE/TRANSFERASE INHIBITOR	No
		5CI7	1	1.74	X-RAY DIFFRACTION	TRANSFERASE/TRANSFERASE Inhibitor	No
		4WNO	1	1.56	X-RAY DIFFRACTION	TRANSFERASE/TRANSFERASE Inhibitor	No
O95352	ATG7_HUMAN	3VH2	1	3.3	X-RAY DIFFRACTION	METAL BINDING PROTEIN	No
P22681	CBL_HUMAN	2K4D	1	N/A	SOLUTION NMR	Ligase	No
		2Y1M	6	2.67	X-RAY DIFFRACTION	LIGASE	No
		1B47	3	2.2	X-RAY DIFFRACTION	SIGNAL TRANSDUCTION	No
		1YVH	2	2.05	X-RAY DIFFRACTION	LIGASE,SIGNALING PROTEIN,IMMUNE SYSTEM	No
		3BUM	2	2	X-RAY DIFFRACTION	LIGASE	No
		3BUW	4	1.45	X-RAY DIFFRACTION	LIGASE/SIGNALING PROTEIN	No
		3OB2	2	2.1	X-RAY DIFFRACTION	Ligase/signaling Protein	No
		4A4C	3	2.7	X-RAY DIFFRACTION	LIGASE/TRANSFERASE	No

		20O9	3	2.1	X-RAY DIFFRACTION	LIGASE	No
		2Y1N	4	2	X-RAY DIFFRACTION	LIGASE/TRANSFERASE	No
		2JUJ	1	N/A	SOLUTION NMR	LIGASE	No
		3BUN	2	2	X-RAY DIFFRACTION	LIGASE/SIGNALING PROTEIN	No
		4A49	2	2.21	X-RAY DIFFRACTION	LIGASE	No
		2CBL	2	2.1	X-RAY DIFFRACTION	COMPLEX (PROTO-ONCOGENE/PEPTIDE)	No
		3BUX	4	1.35	X-RAY DIFFRACTION	LIGASE/SIGNALING PROTEIN	No
		3PLF	4	1.92	X-RAY DIFFRACTION	PROTEIN BINDING/LIGASE	No
		4GPL	1	3	X-RAY DIFFRACTION	Ligase/ligase inhibitor	No
		5J3X	6	2.82	X-RAY DIFFRACTION	LIGASE	No
		1FBV	3	2.9	X-RAY DIFFRACTION	LIGASE	No
		3BUO	4	2.6	X-RAY DIFFRACTION	LIGASE/SIGNALING PROTEIN	No
		3OB1	2	2.2	X-RAY DIFFRACTION	Ligase/signaling Protein	No
		4A4B	3	2.79	X-RAY DIFFRACTION	LIGASE/TRANSFERASE	No
P27797	CALR_HUMAN	3POS	3	1.65	X-RAY DIFFRACTION	CHAPERONE	No
		5LK5	10	2.3	X-RAY DIFFRACTION	calcium-binding protein	No
		3DOW	2	2.3	X-RAY DIFFRACTION	PROTEIN TRANSPORT	No
		3POW	1	1.55	X-RAY DIFFRACTION	CHAPERONE	No
		3RG0	1	2.57	X-RAY DIFFRACTION	CHAPERONE	No
P35193	Atg19_YEAST	5JGE	6	1.91	X-RAY DIFFRACTION	PROTEIN TRANSPORT	No
		2ZPN	8	2.7	X-RAY DIFFRACTION	PROTEIN TRANSPORT	Yes
		2KZB	1	N/A	SOLUTION NMR	PROTEIN TRANSPORT	No
P35222	CTNB1_HUMAN	1JPW	6	2.5	X-RAY DIFFRACTION	CELL ADHESION	No
		2G57	1	N/A	SOLUTION NMR	ONCOPROTEIN	No
		3FQR	3	1.7	X-RAY DIFFRACTION	IMMUNE SYSTEM	No
		3SL9	8	2.2	X-RAY DIFFRACTION	SIGNALING PROTEIN, PROTEIN BINDING	No
		1G3J	4	2.1	X-RAY DIFFRACTION	TRANSCRIPTION	No
		1TH1	4	2.5	X-RAY DIFFRACTION	CELL ADHESION/ANTITUMOR PROTEIN	No
		3TX7	2	2.76	X-RAY DIFFRACTION	PROTEIN BINDING	No
		1LUJ	2	2.5	X-RAY DIFFRACTION	STRUCTURAL PROTEIN	No

		2GL7	6	2.6	X-RAY DIFFRACTION	TRANSCRIPTION	No
		1QZ7	2	2.2	X-RAY DIFFRACTION	CELL ADHESION	No
		2Z6H	1	2.2	X-RAY DIFFRACTION	CELL ADHESION	No
		3SLA	5	2.5	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		1JDH	2	1.9	X-RAY DIFFRACTION	TRANSCRIPTION	No
		3FQN	3	1.65	X-RAY DIFFRACTION	IMMUNE SYSTEM	No
		1P22	3	2.95	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		4DJS	1	3.03	X-RAY DIFFRACTION	SIGNALING PROTEIN/INHIBITOR	No
		1T08	3	2.1	X-RAY DIFFRACTION	cell adhesion/cell cycle	No
		3DIW	4	2.1	X-RAY DIFFRACTION	SIGNALING PROTEIN/CELL ADHESION	No
P40344	Atg3_YEAST	4GSL	4	2.7	X-RAY DIFFRACTION	PROTEIN TRANSPORT	No
		2DYT	1	2.5	X-RAY DIFFRACTION	LIGASE	No
		3T7G	4	2.08	X-RAY DIFFRACTION	LIGASE	No
P40458	Atg32_YEAST	3VXW	2	3	X-RAY DIFFRACTION	PROTEIN TRANSPORT	Yes
P41743	KPCI_HUMAN	5LI1	2	2	X-RAY DIFFRACTION	TRANSFERASE	No
		3A8W	2	2.1	X-RAY DIFFRACTION	TRANSFERASE	No
		5LI9	1	1.79	X-RAY DIFFRACTION	TRANSFERASE	No
		1WMH	2	1.5	X-RAY DIFFRACTION	Transferase/CELL CYCLE	No
		3A8X	2	2	X-RAY DIFFRACTION	TRANSFERASE	No
		3ZH8	3	2.74	X-RAY DIFFRACTION	TRANSFERASE	No
		1VD2	1	N/A	SOLUTION NMR	transferase	No
		5LIH	4	3.25	X-RAY DIFFRACTION	TRANSFERASE	No
		1ZRZ	1	3	X-RAY DIFFRACTION	TRANSFERASE	No
P46934	NEDD4_HUMAN	4BE8	1	3	X-RAY DIFFRACTION	LIGASE	No
		4N7F	2	1.1	X-RAY DIFFRACTION	PROTEIN BINDING	No
		5C9I	1	2.44	X-RAY DIFFRACTION	ligase/ligase inhibitor	No
		5AHT	1	N/A	SOLUTION NMR	ISOMERASE	No
		2KQ0	2	N/A	SOLUTION NMR	LIGASE	No
		2XBF	1	2.5	X-RAY DIFFRACTION	LIGASE	No
		4N7H	2	1.7	X-RAY DIFFRACTION	PROTEIN BINDING	No
		2M3O	2	N/A	SOLUTION NMR	PEPTIDE BINDING PROTEIN/PROTEIN BINDING	No

		4BBN	3	2.51	X-RAY DIFFRACTION	LIGASE/SIGNALING PROTEIN	No
		5C7J	4	3	X-RAY DIFFRACTION	LIGASE/SIGNALING PROTEIN	No
		3B7Y	2	1.8	X-RAY DIFFRACTION	LIGASE	No
		2KPZ	2	N/A	SOLUTION NMR	LIGASE	No
		2XBB	4	2.68	X-RAY DIFFRACTION	LIGASE/PROTEIN BINDING	No
Q00610	CLH1_HUMAN	3LVG	6	7.94	X-RAY DIFFRACTION	STRUCTURAL PROTEIN	No
		1BPO	3	2.6	X-RAY DIFFRACTION	MEMBRANE PROTEIN	No
		3LVH	6	9	X-RAY DIFFRACTION	STRUCTURAL PROTEIN	No
		4G55	1	1.69	X-RAY DIFFRACTION	ENDOCYTOSIS	No
		2XZG	1	1.7	X-RAY DIFFRACTION	ENDOCYTOSIS	No
Q12292	ATG34_YEAST	2KZK	1	N/A	SOLUTION NMR	PROTEIN TRANSPORT	No
Q12983	BNIP3_HUMAN	2KA2	2	N/A	SOLUTION NMR	MEMBRANE PROTEIN	No
		2J5D	2	N/A	SOLUTION NMR	MEMBRANE PROTEIN	No
		2KA1	2	N/A	SOLUTION NMR	MEMBRANE PROTEIN	No
Q13043	STK4_HUMAN	2J08	2	N/A	SOLUTION NMR	TRANSFERASE	No
		3COM	2	2.2	X-RAY DIFFRACTION	TRANSFERASE	No
		4NR2	8	2	X-RAY DIFFRACTION	Transferase	No
Q13137	CACO2_HUMAN	4XKL	4	2.1	X-RAY DIFFRACTION	PROTEIN BINDING/METAL BINDING PROTEIN	No
		2MXP	1	N/A	SOLUTION NMR	METAL BINDING PROTEIN	No
		5AAQ	1	N/A	SOLUTION NMR	CALCIUM-BINDING PROTEIN	No
		3V VW	2	2.5	X-RAY DIFFRACTION	PROTEIN TRANSPORT	Yes
		4GXL	2	2.02	X-RAY DIFFRACTION	PROTEIN BINDING	No
		3V VV	1	1.35	X-RAY DIFFRACTION	PROTEIN TRANSPORT	No
		4HAN	4	2.55	X-RAY DIFFRACTION	SUGAR BINDING PROTEIN	No
Q13188	STK3_HUMAN	5BRM	15	2.65	X-RAY DIFFRACTION	Transferase/Signaling Protein	No
		5DH3	2	2.47	X-RAY DIFFRACTION	TRANSFERASE/TRANSFERASE INHIBITOR	No
		3WWS	4	2.01	X-RAY DIFFRACTION	TRANSFERASE	No
		4HKD	4	1.5	X-RAY DIFFRACTION	TRANSFERASE	No
		4OH9	2	1.7	X-RAY DIFFRACTION	TRANSFERASE	No
		4LON	10	1.4	X-RAY DIFFRACTION	TRANSFERASE	No

		4LG4	6	2.42	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
Q13501	SQSTM_HUMAN	4UF9	3	10.3	ELECTRON MICROSCOPY	SIGNALING PROTEIN	No
		1Q02	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
		2JY8	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
		2K6Q	2	N/A	SOLUTION NMR	APOPTOSIS INHIBITOR/APOPTOSIS	Yes
		2KNV	2	N/A	SOLUTION NMR	PROTEIN BINDING	No
		2K0B	1	N/A	SOLUTION NMR	SIGNALING PROTEIN	No
		4MJS	24	2.5	X-RAY DIFFRACTION	TRANSFERASE/PROTEIN BINDING	No
		4UF8	4	10.9	ELECTRON MICROSCOPY	SIGNALING PROTEIN	No
		2JY7	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
		2ZJD	4	1.56	X-RAY DIFFRACTION	Apoptosis inhibitor/Apoptosis	Yes
Q14596	NBR1_HUMAN	1WJ6	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
		2CP8	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
		2BKF	1	1.56	X-RAY DIFFRACTION	ZINC-FINGER PROTEIN	No
		2MGW	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
		2L8J	2	N/A	SOLUTION NMR	SIGNALING PROTEIN/PROTEIN BINDING	Yes
		4OLE	4	2.52	X-RAY DIFFRACTION	STRUCTURAL GENOMICS, UNKNOWN FUNCTION	No
		2G4S	1	2.15	X-RAY DIFFRACTION	METAL BINDING PROTEIN	No
		2MJ5	2	N/A	SOLUTION NMR	PROTEIN BINDING	No
Q14677	EPN4_HUMAN	2V8S	2	2.22	X-RAY DIFFRACTION	PROTEIN TRANSPORT	No
		1XGW	1	1.9	X-RAY DIFFRACTION	Endocytosis	No
		2QY7	3	2	X-RAY DIFFRACTION	PROTEIN BINDING	No
Q15459	SF3A1_HUMAN	2DT6	1	N/A	SOLUTION NMR	RNA BINDING PROTEIN	No
		1ZKH	1	N/A	SOLUTION NMR	GENE REGULATION	No
		2DT7	2	N/A	SOLUTION NMR	RNA BINDING PROTEIN	No
Q86V97	KBTB6_HUMAN	4XC2	8	1.9	X-RAY DIFFRACTION	IMMUNE SYSTEM	No
Q86VP1	TAXB1_HUMAN	4NLH	2	1.9	X-RAY DIFFRACTION	PROTEIN BINDING	No
		4Z4K	2	2.8	X-RAY DIFFRACTION	Fluorescent Protein, Metal Binding Protein	No
		4BMJ	11	2.75	X-RAY DIFFRACTION	APOPTOSIS	No
		5AAS	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
		4Z4M	2	2.15	X-RAY DIFFRACTION	Fluorescent Protein, Metal Binding Protein	No
		2M7Q	1	N/A	SOLUTION NMR	Metal Binding Protein	No

Q8TD19	NEK9_HUMAN	3ZKF	12	2.6	X-RAY DIFFRACTION	CONTRACTILE PROTEIN/PEPTIDE	No
		3ZKE	12	2.2	X-RAY DIFFRACTION	CONTRACTILE PROTEIN/PEPTIDE	No
Q8WWW0	RAS5_HUMAN	4LGD	8	3.05	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		4OH8	2	2.28	X-RAY DIFFRACTION	transferase/Apoptosis	No
Q96CV9	OPTN_HUMAN	3VTV	1	1.7	X-RAY DIFFRACTION	PROTEIN BINDING	No
		2LUE	2	N/A	SOLUTION NMR	PROTEIN BINDING	No
		5EOA	4	2.5	X-RAY DIFFRACTION	PROTEIN BINDING/TRANSFERASE	No
		3VTW	3	2.52	X-RAY DIFFRACTION	PROTEIN BINDING	No
		5B83	6	2.69	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		5EOF	4	2.05	X-RAY DIFFRACTION	PROTEIN BINDING/TRANSFERASE	No
		2LO4	1	N/A	SOLUTION NMR	PROTEIN TRANSPORT	No
		5AAZ	1	N/A	SOLUTION NMR	PROTEIN BINDING	No
Q96RU3	FNB1_HUMAN	2EFL	1	2.61	X-RAY DIFFRACTION	ENDOCYTOSIS/EXOCYTOSIS	No
Q9BQS8	FYCO1_HUMAN	5LXI	4	1.44	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		5D94	2	1.53	X-RAY DIFFRACTION	PROTEIN BINDING/PEPTIDE	No
		5CX3	8	2.3	X-RAY DIFFRACTION	PROTEIN BINDING	Yes
		5LXH	6	1.58	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
Q9GZZ9	UBA5_HUMAN	5HKH	3	2.55	X-RAY DIFFRACTION	SIGNALING PROTEIN	No
		3GUC	2	2.25	X-RAY DIFFRACTION	TRANSFERASE	No
		5IAA	4	1.85	X-RAY DIFFRACTION	CELL CYCLE	No
		5L95	4	2.1	X-RAY DIFFRACTION	CELL CYCLE	No
		3H8V	2	2	X-RAY DIFFRACTION	TRANSFERASE	No
Q9H1Y0	ATG5_HUMAN	4TQ0	6	2.7	X-RAY DIFFRACTION	PROTEIN BINDING	No
		5D7G	8	3	X-RAY DIFFRACTION	APOPTOSIS	No
		4GDL	3	2.88	X-RAY DIFFRACTION	protein binding	No
		4TQ1	2	1.8	X-RAY DIFFRACTION	PROTEIN BINDING	No
		4GDK	6	2.7	X-RAY DIFFRACTION	PROTEIN BINDING	No
Q9NT62	ATG3_HUMAN	4NAW	16	2.19	X-RAY DIFFRACTION	PROTEIN TRANSPORT/LIGASE	No
Q9Y4P1	ATG4B_HUMAN	2Z0E	2	1.9	X-RAY DIFFRACTION	HYDROLASE/STRUCTURAL PROTEIN	Yes
		2CY7	1	1.9	X-RAY DIFFRACTION	hydrolase	No

		2ZZP	2	2.05	X-RAY DIFFRACTION	HYDROLASE/STRUCTURAL PROTEIN	Yes
		2D1I	2	2	X-RAY DIFFRACTION	HYDROLASE	No
		2Z0D	2	1.9	X-RAY DIFFRACTION	HYDROLASE/STRUCTURAL PROTEIN	Yes

Table 21. Selective autophagy receptor and adaptor protein structures.

PDB IDs were extracted from UniProt using the UniProt accession. Metadata such as structure resolution, function, number of chains and method of structure determination were obtained RSCB PDB using PDB IDs.

IOANNA KALVARI

7.1 dizscan.py code

```
1  """
2  dizscan.py
3
4  This is a script to identify disorder regions
5  overlaps by incorporating data from MobiDB
6
7  Developer: Ioanna Kalvari
8  """
9
10
11 import os
12 import sys
13 import copy
14 import urllib2
15 import json
16
17
18 # -----
19
20
21 def fetch_consensus_disorder_curated_data(accession):
22     """
23     Fetches consensus curated data from MobiDB based on
24     UniProt accession and re-organises them in a simpler
25     way in a dictionary
26
27     accession: A valid UniProt accession
28
29     return: A reconstructed dictionary with MobiDB data
30     """
31
32     disorder_url = "http://mobidb.bio.unipd.it/ws/%s/consensus"
33     acceptHeader = 'application/json' # text/csv and text/plain supported
34     request = urllib2.Request(disorder_url % accession, headers={"Accept" : acceptHeader})
35
36     # Send request
```



```

37 response = urllib2.urlopen(request)
38
39 # Parse JSON response di Python dict
40 data = json.load(response)
41
42 curated_data = {}
43
44 # [u'solvent_exposure', u'lips', u'ss_populations', u'disorder', u'interactions']
45 # print data["mobidb_consensus"].keys()
46 if "db" in data["mobidb_consensus"]["disorder"]:
47     for item in data["mobidb_consensus"]["disorder"]["db"]:
48         if item["method"] not in curated_data:
49             curated_data[item["method"]] = item["regions"]
50
51 return curated_data
52
53 # -----
54
55
56 def fetch_consensus_disorder_indirect_data_by_method(accession):
57     """
58     Fetces consensus indirect (derived) data from MobiDB based on
59     UniProt accession and re-organises them in a simpler
60     way in a dictionary
61
62     accession: A valid UniProt accession
63
64     return: A reconstructed dictionary with MobiDB data
65     """
66
67     disorder_url = "http://mobidb.bio.unipd.it/ws/%s/consensus"
68     acceptHeader = 'application/json' # text/csv and text/plain supported
69     request = urllib2.Request(disorder_url % accession, headers={"Accept" : acceptHeader})
70
71     # Send request
72     response = urllib2.urlopen(request)
73
74     # Parse JSON response di Python dict
75     data = json.load(response)
76

```

```

77 indirect_data = {}
78
79 # [u'solvent_exposure', u'lips', u'ss_populations', u'disorder', u'interactions']
80 # print data["mobidb_consensus"].keys()
81 if "derived" in data["mobidb_consensus"]["disorder"]:
82     for item in data["mobidb_consensus"]["disorder"]["derived"]:
83         # organise disorder regions by method
84         if item["method"] not in indirect_data:
85             indirect_data[item["method"]]=item["regions"]
86
87 return indirect_data
88
89
90 # -----
91
92
93 def fetch_consensus_disorder_predicted_data_by_method(accession):
94     """
95     Fetces consensus predicted data from MobiDB based on
96     UniProt accession and re-organises them in a simpler
97     way in a dictionary
98
99     accession: A valid UniProt accession
100
101     return: A reconstructed dictionary with MobiDB data
102     """
103
104     disorder_url = "http://mobidb.bio.unipd.it/ws/%s/consensus"
105     acceptHeader = 'application/json' # text/csv and text/plain supported
106     request = urllib2.Request(disorder_url % accession, headers={"Accept" : acceptHeader})
107
108     # Send request
109     response = urllib2.urlopen(request)
110
111     # Parse JSON response di Python dict
112     data = json.load(response)
113
114
115     predicted_data = {}
116

```

```

117 # [u'solvent_exposure', u'lips', u'ss_populations', u'disorder', u'interactions']
118 # print data["mobidb_consensus"].keys()
119 if "predictors" in data["mobidb_consensus"]["disorder"]:
120     for item in data["mobidb_consensus"]["disorder"]["predictors"]:
121         # organise disorder regions by method
122         if item["method"] not in predicted_data:
123             predicted_data[item["method"]]=item["regions"]
124
125     return predicted_data
126
127 # -----
128
129 def fetch_disorder_data(accession, type="curated"):
130     """
131     Returns all indirect disorder data associated with the accession
132     provided. The type of data retrieved from MobiDB needs to be
133     specified and the output is a dictionary with raw MobiDB
134     data.
135
136     accession: A valid UniProt accession
137     type: The type of data to fetch (predicted, indirect, curated)
138
139     return: A dictionary with raw data of a particular type
140     """
141     disorder_url = "http://mobidb.bio.unipd.it/ws/%s/disorder"
142     acceptHeader = 'application/json' # text/csv and text/plain supported
143     request = urllib2.Request(disorder_url % accession, headers={"Accept" : acceptHeader})
144
145     # Send request
146     response = urllib2.urlopen(request)
147
148     # Parse JSON response di Python dict
149     data = json.load(response)
150
151     # handle data
152     if type == "curated":
153         type = "db"
154
155     elif type == "indirect":
156         type = "derived"

```

```

157
158     elif type == "pedicted":
159         type = "predictors"
160
161     if type not in data["mobidb_data"]["disorder"]:
162         return None
163
164     return data["mobidb_data"]["disorder"][type]
165
166
167 # -----
168
169
170 def scan_db_data_for_disorder_regions(accession, start, end, lir, type):
171     """
172     This function will scan the provided lir for any possible disorder
173     regions that match the data retrieved from mobiDB
174
175     accession: A valid Uniprot protein id (e.g. Q13501)
176     start: Start coordinate of the LIR region
177     end: End coordinate of the LIR region
178     lir: A string representing the amino acid sequence of the LIR peptide
179     type: The type of the disordered data to fetch (e.g. curated, indirect, predicted)
180
181     return: A dictionary of all LIR/mobiDB overlaps found per method (e.g. )
182     """
183
184     disorder_strings = {}
185
186     disorder_data = fetch_disorder_data(accession, type)
187
188     if disorder_data is not None:
189
190         dislir_dict = None
191         for database in disorder_data:
192             for region in database["regions"]:
193                 dislir_dict = search_for_overlaps(start, end, region[0], region[1],
194                                                  str(region[2]), len(lir), pos_dict = dislir_dict)
195
196         # construct disordered lir string

```

```

197         disorder_string = construct_disorder_lir_string(dislir_dict)
198
199         # update dictionary
200         disorder_strings[str(database["method"])] = disorder_string
201
202     return disorder_strings
203
204 return None
205
206
207 # -----
208
209
210 def scan_indirect_data_for_disorder_regions(accession, start, end, lir, concensus=False):
211     """
212     """
213
214     disorder_strings = {}
215     disorder_data = fetch_disorder_data(accession, type="derived")
216
217     if disorder_data is not None:
218         if concensus is False:
219             struct_dislir_dict = None
220             for structure_case in disorder_data:
221                 if "pdb_id" in structure_case:
222                     pdb_label = structure_case["pdb_id"] + '_' + structure_case["chain_id"]
223                     method = structure_case["method"]
224                     regions = structure_case["regions"]
225                     for region in regions:
226                         struct_dislir_dict = search_for_overlaps(start, end, region[0], region[1],
227                                                                     str(region[2]), len(lir), pos_dict = struct_dislir_dict)
228
229                     disorder_string = ''
230                     disorder_string = construct_disorder_lir_string(struct_dislir_dict)
231
232                     if pdb_label not in disorder_strings:
233                         disorder_strings[pdb_label] = {"distring": disorder_string, "method": method}
234     else:
235         struct_dislir_dict = {}
236         for structure_case in disorder_data:

```

```

237         regions = structure_case["regions"]
238     for region in regions:
239         struct_dislir_dict = search_for_overlaps(start, end, region[0], region[1],
240                                                str(region[2]), len(lir), pos_dict = struct_dislir_dict)
241
242         disorder_strings = construct_disorder_lir_string(struct_dislir_dict)
243
244     return disorder_strings
245
246 # -----
247
248 # deprecated
249 def _search_for_overlaps(lir_start, lir_end, mobi_start, mobi_end, mobi_label, seq_length, pos_dict = None):
250     """
251     Deprecated function
252
253     lir_start: LIR peptide start position
254     lir_end: LIR peptide end position
255     mobi_start: MobiDB start position
256     mobi_end:  MobiDB end position
257     mobi_label: A character indicating if the position is disordered or structured D=disorder, S=structured,
258     seq_length: The length of the peptide
259
260     return: A dictionary where keys are the start-end range numbers and values are the D/S labels from
261     MobiDB or ? depending on whether there's an overlap or data available
262     """
263
264     # variable declaration and initialization
265     positions = {}
266     index = -1
267     boundary = -1
268
269     # initialising position matrix
270     if pos_dict is None:
271         index = lir_start
272         while index <= lir_end:
273             positions[index] = '?'
274             index +=1
275     else:
276         positions = copy.deepcopy(pos_dict)

```

```

277
278 # case A - partial overlap
279 if lir_start > mobi_start and lir_start < mobi_end and lir_end > mobi_end:
280     boundary = mobi_end
281     index = lir_start
282
283 # case B - partial overlap
284 elif lir_start < mobi_start and lir_end > mobi_start and lir_end < mobi_end:
285     boundary = lir_end
286     index = mobi_start
287
288 # case C - partial overlap
289 elif mobi_start < lir_start and lir_end < mobi_end and lir_start < mobi_end:
290     boundary = lir_end
291     index = lir_start
292
293 # case D - partial overlap
294 elif lir_start < mobi_start and lir_start < mobi_end and lir_end > mobi_end:
295     boundary = mobi_end
296     index = mobi_start
297
298 # need to check length here - full overlap (I)
299 elif lir_start == mobi_start and lir_end == mobi_end:
300     boundary = lir_end
301     index = lir_start
302
303 # case G - partial overlap starting from the same position
304 elif lir_start == mobi_start and lir_end > mobi_start and lir_end < mobi_end:
305     boundary = lir_end
306     index = lir_start
307
308 # case H - partial overlap, same ending coords
309 elif mobi_start > lir_start and lir_start < mobi_end and lir_end == mobi_end:
310     boundary = lir_end
311     index = mobi_start
312
313 # case J
314 elif lir_start == mobi_start and lir_end > mobi_start and lir_end > mobi_end:
315     boundary = mobi_end
316     index = lir_start

```

```

317
318 # case K
319 elif mobi_start < lir_start and lir_start < mobi_end and lir_end == mobi_end:
320     boundary = mobi_end
321     index = lir_start
322
323 # case E - no overlap left end
324 elif lir_start < mobi_start and lir_end < mobi_start and lir_end < mobi_end:
325     return positions
326
327 # case F - no overlap right end
328 elif mobi_start < lir_start and mobi_end < lir_start and lir_start > mobi_start:
329     return positions
330
331 else:
332     return positions
333
334 while index<=boundary:
335     positions[index] = mobi_label
336     index+=1
337
338 return positions
339
340 # -----
341
342
343 def search_for_overlaps(lir_start, lir_end, mobi_start, mobi_end, mobi_label, pos_dict = None):
344     """
345     Searches for LIR-motif/disorder overlaps based on the data retrieved from MobiDB. Identification of
346     overlaps is done using iLIR and MobiDB coordinates and returns a dictionary which encapsulates
347     disordered positions of the LIR-motif
348
349     lir_start: LIR-motif start position
350     lir_end: LIR-motif end position
351     mobi_start: MobiDB start position
352     mobi_end: MobiDB end position
353     mobi_label: MobiDB residue label D/S
354     pos_dict: The position dictionary to be modified. Either an initialised ?????? or an intermediate
355     one when processing many different predictions
356

```



```

357     return: A dictionary with all disorder positions identified
358     """
359     # variable declaration and initialization
360     positions = {}
361     index = -1
362     boundary = -1
363
364     # initialising position matrix
365     if pos_dict is None:
366         index = lir_start
367         while index <= lir_end:
368             positions[index] = '?'
369             index +=1
370     else:
371         positions = copy.deepcopy(pos_dict)
372
373     # A
374     if lir_start < mobi_start and mobi_start < lir_end and lir_end < mobi_end:
375         boundary = lir_end
376         index = mobi_start
377     # B
378     elif mobi_start < lir_start and lir_start < mobi_end and mobi_end < lir_end:
379         boundary = mobi_end
380         index = lir_start
381     # C
382     elif lir_start == mobi_start and lir_end < mobi_end:
383         boundary = lir_end
384         index = lir_start
385     # D
386     elif lir_start == mobi_start and mobi_end < lir_end:
387         boundary = mobi_end
388         index = mobi_start
389     # E
390     elif mobi_start < lir_start and lir_end == mobi_end:
391         boundary = lir_end
392         index = lir_start
393     # F
394     elif lir_start < mobi_start and lir_end == mobi_end:
395         boundary = lir_end
396         index = mobi_start

```

```

397 # G
398 elif lir_start < mobi_start and mobi_start < lir_end and mobi_end < lir_end:
399     boundary = mobi_end
400     index = mobi_start
401 # H
402 elif mobi_start < lir_start and lir_end > mobi_start and lir_end < mobi_end:
403     boundary = lir_end
404     index = lir_start
405 # I
406 elif lir_start == mobi_start and lir_end == mobi_end:
407     boundary = lir_end
408     index = lir_start
409 # J
410 elif lir_start < mobi_start and lir_end <= mobi_start and lir_end < mobi_end:
411     return positions
412 # K
413 elif mobi_start < lir_start and mobi_end <= lir_start and lir_end > mobi_end:
414     return positions
415
416
417 while index<=boundary:
418     positions[index] = mobi_label
419     index+=1
420
421 return positions
422
423 # -----
424
425
426 def calculate_disorder_fraction(disorder_str):
427     """
428     Calculates the fraction of disorder residues
429     found in a LIR-motif
430
431     disorder_str: The disorder string in the form of 'DDDDSD'
432
433     return: disorder fraction in float
434     """
435
436     length = len(disorder_str)

```

```

437
438     count_ds = disorder_str.count('D')
439
440     dis_fraction = count_ds/length
441
442
443     return dis_fraction
444
445
446 # -----
447
448
449 def calculate_disorder_percentage(disorder_str):
450     """
451     Calculate and return the proportion of the disordered region in a peptide
452     given a disordered string as generated by a disorder scanner
453
454     disorder_str: A disorder string representing the disordered residues in
455     a given string
456
457     return: returns the disorder percentage
458     """
459
460     # D_PPE, D_NPE, D_PA, D_WC D_WCD_WCD_WCD_WCD_WCD_WC
461     # replace predicted D types with Ds
462     dtypes = ["D_PPE", "D_NPE", "D_PA", "D_WC"]
463
464     for case in dtypes:
465         disorder_str = disorder_str.replace(case, 'D')
466
467     count_ds = disorder_str.count('D')
468
469     # counting length after replacement for accuracy purposes
470     length = len(disorder_str)
471     disorder_proportion = 0
472
473     if length > 0:
474         disorder_proportion = count_ds*100/length
475
476     return disorder_proportion

```

```

477
478 # -----
479
480
481 def construct_disorder_lir_string(dispos):
482     """
483     Takes the dictionary of calculated disordered LIR-motif
484     positions and constructs the disorder string dSTR
485
486     dispos: Calculated disorder positions
487
488     return: Disorder string
489     """
490
491     disordered_string = ''
492     if dispos is not None:
493         pos_list = sorted(dispos.keys())
494
495
496         for pos in pos_list:
497             disordered_string = disordered_string + dispos[pos]
498
499     return disordered_string
500
501 # -----
502
503
504
505 def print_lir_disorder_data(uniprot_acc, lir, start, end, scanner_results_dict):
506     """
507     Prints the disorder region scanning results in a "pretty" format
508
509     uniprot_acc: A valid UniProt accession
510     lir: The LIR-motif sequence (as in iLIR3D)
511     start: The start position of the LIR-motif
512     end: The end position of the LIR-motif
513     scanner_results_dict:
514
515     returns: void
516     """

```

```

517
518 for db in scanner_results_dict.keys():
519     disorder_string = scanner_results_dict[db]
520
521     disorder_percentage = calculate_disorder_percentage(disorder_string)
522
523     print "%s\t%s-%s\t%s\t%s\t%s\t%s" % (uniprot_acc, str(start), str(end), lir,
524                                         disorder_string, str(disorder_percentage),
525                                         chr(37), db)
526
527
528 # -----
529
530
531
532 def print_disorder_report(peptide_file, type):
533     """
534     Prints on the screen the disorder results that were computed, in a
535     veyr simple tab delimited format
536
537     peptide_file: This is a tab delimited file containing the lir string,
538     start and end coordinates per
539     candidate uniprot accession
540
541     return: void
542     """
543
544     peptide_file_handle = open(peptide_file, 'r')
545
546     for peptide_line in peptide_file_handle:
547         components = peptide_line.strip().split('\t')
548
549         uniprot_acc = components[0]
550         start = int(components[2])
551         end = int(components[3])
552         lir = components[1]
553
554         disorder_overlaps = scan_db_data_for_disorder_regions(uniprot_acc, start, end, lir, type)
555
556         if disorder_overlaps is not None:

```

```

557     for db in disorder_overlaps.keys():
558
559         dstring = disorder_overlaps[db]
560         disorder_percentage = calculate_disorder_percentage(dstring)
561
562         print "%s\t%s-%s\t%s\t%s\t%s\t%s" % (uniprot_acc, str(start), str(end), lir,
563                                             dstring, str(disorder_percentage),
564                                             chr(37), db)
565     else:
566         print "%s NA" % uniprot_acc
567
568
569 # -----
570
571
572 def mobidb_indirect_list_data_to_pdb_dict(uniprot_accession):
573     """
574     Coverts the MobiDB indirect data into a dictionary where keys are in the form of
575     pdb_id followed by chain_id and separated by '_' e.g. 2K6Q_B, to simplify the
576     scanning process with structures. Values are lists of tuples (start, end, method)
577
578     uniprot_accession: A valid uniprot accession
579
580     return: The new reformatted dictionary
581     """
582
583     pdb_formatted_data = {}
584
585     accession_data = fetch_disorder_data(uniprot_accession, type="indirect")
586     indirect_data = accession_data["mobidb_data"]["disorder"]["derived"]
587
588     for structure in indirect_data:
589         new_key = structure["pdb_id"] + '_' + structure["chain_id"]
590
591         if new_key not in pdb_formatted_data:
592             pdb_formatted_data[new_key] = [{"regions": indirect_data["regions"],
593                                           "method": indirect_data["method"]}]}
594             #pdb_formatted_data[new_key]["regions"] = indirect_data["regions"]
595             #pdb_formatted_data[new_key]["method"] = indirect_data["method"]
596     else:

```

```

597         pdb_formated_data[new_key].append({"regions": indirect_data["regions"]
598                                             ,"method": indirect_data["method"]})
599
600     return pdb_formated_data
601
602
603 # -----
604
605
606 def scan_peptide_for_consensus_disorder_regions(accession, lir_start, lir_end, lir, type = "curated"):
607     """
608     Scans a LIR-motif and looks for disorder overlaps with MobiDB based on the
609     start-end coordinates (lir_start, lir_end) and the corresponding UniProt
610     accession, which is used to retrieve the data.
611
612     accession: A valid UniProt accession
613     lir_start: The start position of a LIR-motif
614     lir_end: The end position of a LIR-motif
615     lir: The LIR-motif sequence
616     type: Type of data to retrieve from MobiDB (curated, predicted, indirect)
617
618     return: A dictionary with constructed disorder strings
619     """
620
621     data = {}
622     disorder_strings = {}
623
624     if type == "curated":
625         data = fetch_consensus_disorder_curated_data(accession)
626
627     elif type == "indirect":
628         data = fetch_consensus_disorder_indirect_data_by_method(accession)
629
630     elif type == "predicted":
631         data = fetch_consensus_disorder_predicted_data_by_method(accession)
632
633     if bool(data) is not False:
634         #dis_dict = None
635         for method in data.keys():
636             disorder_string = ''

```

```

637     dis_dict = None
638     regions = data[method]
639     if len(regions)>0:
640         for region in regions:
641             mobi_start = region[0]
642             mobi_end = region[1]
643             mobi_label = region[2]
644             seq_length = len(lir)
645
646             dis_dict = search_for_overlaps(lir_start, lir_end, mobi_start, mobi_end,
647                                           mobi_label, seq_length, pos_dict = dis_dict)
648
649
650             disorder_string = construct_disorder_lir_string(dis_dict)
651
652     else:
653         lir_len = len(lir)
654         i=0
655
656         while i<lir_len:
657             disorder_string += '?'
658             i+=1
659
660     disorder_strings[method] = disorder_string
661
662     return disorder_strings
663
664 # -----
665
666
667 def disorder_report_generator(protein_file, type = "all"):
668     """
669     Generates a human readable report showing the disorder
670     residues identified for each LIR-motif
671
672     protein_file: The input file with uniprot protein id and protein accession,
673     start and end coordinates of the lir motif, the lir string and verified digit
674     1 if so, 0 if unverified
675     type: The type of data we want to use in the calculation of the disorder data
676

```



```

717 disorder_strings[method],
718 str(calculate_disorder_percentage(disorder_strings[method])),
719 chr(37), verified, method, "curated")
720
721 elif type == "predicted":
722     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
723                                                                    lir, type = "predicted")
724
725     if bool(disorder_strings) is not False:
726         for method in disorder_strings.keys():
727             print "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s" %(uniprot_id, str(start), str(end), lir,
728                                                             disorder_strings[method],
729                                                             str(calculate_disorder_percentage(disorder_strings[method])),
730                                                             chr(37), verified, method, "predicted")
731
732 # print all
733 else:
734     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
735                                                                    lir, type = "curated")
736
737     if bool(disorder_strings) is not False:
738         for method in disorder_strings.keys():
739             print "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s" %(uniprot_id, str(start), str(end), lir,
740                                                             disorder_strings[method],
741                                                             str(calculate_disorder_percentage(disorder_strings[method])),
742                                                             chr(37), verified, method, "curated")
743
744     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
745                                                                    lir, type = "indirect")
746
747     if bool(disorder_strings) is not False:
748         for method in disorder_strings.keys():
749             print "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s" %(uniprot_id, str(start), str(end), lir,
750                                                             disorder_strings[method],
751                                                             str(calculate_disorder_percentage(disorder_strings[method])),
752                                                             chr(37), verified, method, "indirect")
753
754     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
755                                                                    lir, type = "predicted")
756
757     if bool(disorder_strings) is not False:

```



```

797
798 # Send request
799 response = urllib2.urlopen(request)
800
801 # Parse JSON response di Python dict
802 data = json.load(response)
803
804 predicted_data = {}
805
806 if accession not in acc_data:
807     acc_data[accession] = {}
808
809     if "derived" in data["mobidb_consensus"]["disorder"]:
810         acc_data[accession]["indirect"] = "Yes"
811     else:
812         acc_data[accession]["indirect"] = "No"
813     if "predictors" in data["mobidb_consensus"]["disorder"]:
814         acc_data[accession]["predicted"] = "Yes"
815     else:
816         acc_data[accession]["predicted"] = "No"
817     if "db" in data["mobidb_consensus"]["disorder"]:
818         acc_data[accession]["curated"] = "Yes"
819     else:
820         acc_data[accession]["curated"] = "No"
821
822 return acc_data
823
824 # -----
825
826
827 def check_disorder_data_availability_by_data_type(uniprot_acc_input):
828     """
829     This function loads all data from MobiDB and checks whether
830     there's available data for each accession provided.
831
832     uniprot_acc_input: It can either be a list of valid uniprot
833     accessions or a single uniprot accession
834
835     return: A dictionary with the accession data retrieved from MobiDB
836     """

```

```

837
838 acc_data = {}
839 uniprot_accs = []
840
841 if os.path.isfile(uniprot_acc_input):
842     fp = open(uniprot_acc_input, 'r')
843     uniprot_accs = [x.strip() for x in fp]
844     fp.close()
845
846 else:
847     uniprot_accs.append(uniprot_acc_input)
848
849
850 disorder_url = "http://mobidb.bio.unipd.it/ws/%s/consensus"
851 acceptHeader = 'application/json' # text/csv and text/plain supported
852
853
854 for accession in uniprot_accs:
855     request = urllib2.Request(disorder_url % accession, headers={"Accept" : acceptHeader})
856
857     # Send request
858     response = urllib2.urlopen(request)
859
860     # Parse JSON response di Python dict
861     data = json.load(response)
862
863     predicted_data = {}
864
865     if accession not in acc_data:
866         acc_data[accession] = {}
867
868         if "derived" in data["mobidb_consensus"]["disorder"]:
869             acc_data[accession]["indirect"] = "Yes"
870         else:
871             acc_data[accession]["indirect"] = "No"
872         if "predictors" in data["mobidb_consensus"]["disorder"]:
873             acc_data[accession]["predicted"] = "Yes"
874         else:
875             acc_data[accession]["predicted"] = "No"
876         if "db" in data["mobidb_consensus"]["disorder"]:

```

```

877         acc_data[accession]["curated"] = "Yes"
878     else:
879         acc_data[accession]["curated"] = "No"
880
881     return acc_data
882
883 # -----
884
885 def disorder_to_iLIR3Ddb(protein_file, type = "all"):
886     """
887     Based on the uniprot accessions listed in the protein_file input and according to the specified type,
888     it fetches all relevant data from MobiDB and searches for disorder overlaps. The overlaps are computed
889     for all the LIR regions supplied in the input file. The output is in tabular format arranged specifically
890     to be loaded into iLIR3D database using mysqlimport.
891
892     protein_file: A file in tabular format which contains the uniprot id and accession, the start and end
893     coordinates of the LIR, the LIR sequence and LIR acc and if the LIR is experimentally verified or not.
894     type: The type of disorder data to fetch from MobiDB. One of ('all', 'predicted', 'curated', 'indirect')
895
896     return: void
897     """
898
899     str_with_ver_string = "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s"
900     str_no_ver = "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s"
901
902     # Move code from main here
903
904     fp = open(protein_file, 'r')
905     verified = ''
906     for line in fp:
907         line = line.strip().split('\t')
908         lir_acc = line[0].strip()
909         uniprot_id = line[1].strip()
910         uniprot_acc = line[2].strip()
911         start = int(line[3].strip())
912         end = int(line[4].strip())
913         lir = line[5].strip()
914         verified = line[6].strip()
915
916         """

```

```

917 if len(line) > 6:
918     if line[6] == '0':
919         verified = "unverified"
920     else:
921         verified = "verified"
922     """
923
924 if type == "indirect":
925     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
926                                                                    lir, type="indirect")
927
928     if bool(disorder_strings) is not False:
929         for method in disorder_strings.keys():
930             if len(line) == 5:
931                 print str_no_ver % (lir_acc, uniprot_id, str(start), str(end), lir,
932                                     disorder_strings[method],
933                                     str(calculate_disorder_percentage(disorder_strings[method])),
934                                     method, "indirect")
935             else:
936                 print str_with_ver_string % (lir_acc, uniprot_id, str(start), str(end), lir,
937                                               disorder_strings[method],
938                                               str(calculate_disorder_percentage(disorder_strings[method])),
939                                               verified, method, "indirect")
940
941 elif type == "curated":
942     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
943                                                                    lir, type = "curated")
944
945     if bool(disorder_strings) is not False:
946         for method in disorder_strings.keys():
947             if len(line) == 5:
948                 print str_no_ver % (lir_acc, uniprot_id, str(start), str(end), lir,
949                                     disorder_strings[method],
950                                     str(calculate_disorder_percentage(disorder_strings[method])),
951                                     method, "curated")
952             else:
953                 print str_with_ver_string % (lir_acc, uniprot_id, str(start), str(end), lir,
954                                               disorder_strings[method],
955                                               str(calculate_disorder_percentage(disorder_strings[method])),
956                                               verified, method, "curated")

```

```

957
958
959 elif type == "predicted":
960     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
961                                                                    lir, type = "predicted")
962
963     if bool(disorder_strings) is not False:
964         for method in disorder_strings.keys():
965             if len(line) == 5:
966                 print str_no_ver % (lir_acc, uniprot_id, str(start), str(end), lir,
967                                     disorder_strings[method],
968                                     str(calculate_disorder_percentage(disorder_strings[method])),
969                                     method, "predicted")
970             else:
971                 print str_with_ver_string % (lir_acc, uniprot_id, str(start), str(end), lir,
972                                             disorder_strings[method],
973                                             str(calculate_disorder_percentage(disorder_strings[method])),
974                                             verified, method, "predicted")
975
976 # print all
977 else:
978     disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
979                                                                    lir, type = "curated")
980
981     if bool(disorder_strings) is not False:
982         for method in disorder_strings.keys():
983             if len(line) == 5:
984                 print str_no_ver % (lir_acc, uniprot_id, str(start), str(end), lir,
985                                     disorder_strings[method],
986                                     str(calculate_disorder_percentage(disorder_strings[method])),
987                                     method, "curated")
988             else:
989                 print str_with_ver_string % (lir_acc, uniprot_id, str(start), str(end), lir,
990                                             disorder_strings[method],
991                                             str(calculate_disorder_percentage(disorder_strings[method])),
992                                             verified, method, "curated")
993
994 disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
995                                                                lir, type="indirect")
996

```



```

997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036

if bool(disorder_strings) is not False:
    for method in disorder_strings.keys():
        if len(line) == 5:
            print str_no_ver % (lir_acc, uniprot_id, str(start), str(end), lir,
                                disorder_strings[method],
                                str(calculate_disorder_percentage(disorder_strings[method])),
                                method, "indirect")
        else:
            print str_with_ver_string % (lir_acc, uniprot_id, str(start), str(end), lir,
                                         disorder_strings[method],
                                         str(calculate_disorder_percentage(disorder_strings[method])),
                                         verified, method, "indirect")

disorder_strings = scan_peptide_for_consensus_disorder_regions(uniprot_acc, start, end,
                                                             lir, type = "predicted")

if bool(disorder_strings) is not False:
    for method in disorder_strings.keys():
        if len(line) == 5:
            print str_no_ver % (lir_acc, uniprot_id, str(start), str(end), lir,
                                disorder_strings[method],
                                str(calculate_disorder_percentage(disorder_strings[method])),
                                method, "predicted")
        else:
            print str_with_ver_string % (lir_acc, uniprot_id, str(start), str(end), lir,
                                         disorder_strings[method],
                                         str(calculate_disorder_percentage(disorder_strings[method])),
                                         verified, method, "predicted")

fp.close()

# -----

def complex_distring_to_simple(complex_str):
    """
    Convert from complex disorder string to a simpler form only containing
    D,S and ? characters

```

```

1037
1038     complex_str: The complex disorder string to modify
1039
1040     return: A simpler form of the complex query
1041     """
1042
1043     # D_PPE, D_NPE, D_PA, D_WC D_WCD_WCD_WCD_WCD_WCD_WC
1044     # replace predicted D types with Ds
1045     dtypes = ["D_PPE", "D_NPE", "D_PA", "D_WC"]
1046
1047     for case in dtypes:
1048         simple_disorder_str = complex_str.replace(case, 'D')
1049
1050
1051     return simple_disorder_str
1052
1053 # -----
1054
1055 if __name__ == '__main__':
1056
1057     # input here is:
1058     # lir_acc\tuniprot_id\tuniprot_acc\tlir_start\tlir_end\tlir_sequence\tverified
1059
1060     protein_file = sys.argv[1]
1061     # predicted, indirect, curated, all
1062     data_type = sys.argv[2]
1063
1064     # This function prints out iLIR3D ready disorder data
1065     disorder_to_iLIR3Ddb(protein_file, type = data_type)
1066
1067
1068
1069
1070
1071
1072
1073
1074

```

7.2 consensus_disorder_calculator.py code

```
1 import os
2 import sys
3 import dizscan as dislib
4 from ilir3d.lib import lir3d_db_connector as db
5
6
7 """
8 MySQL query to generate inputs used by this query
9 select lir_acc, disorder_string from sars_lir_disorder
10 """
11
12 # -----
13
14
15 def calculate_per_residue_consensus_disorder_score(disorder_string_list, dis_percentage = 50, todb=False):
16     """
17     Computes the consensus disorder string and loads the data into the database
18
19     disorder_string_list: A file containing all disorder strings per LIRCP protein
20     percentage_per_position: disorder percentage of each LIR-motif residue position
21
22     return: void
23     """
24
25     disorder_dict = {}
26     fp_in = open(disorder_string_list, 'r')
27
28     consensus_disorder = {}
29     db_data = []
30
31     for line in fp_in:
32         line = line.strip().split('\t')
33         lir_acc = line[0]
34         dis_lir = line[1]
35
```

```

36     if lir_acc not in disorder_dict:
37         disorder_dict[lir_acc] = [dis_lir]
38     else:
39         disorder_dict[lir_acc].append(dis_lir)
40
41 fp_in.close()
42
43 dis_string_list = []
44 diz_string_score_dict = {}
45 db_disorder_data = []
46 for lir in disorder_dict.keys():
47     no_dis_strings = len(disorder_dict[lir])
48     # convert complex chars to Ds&Ss
49     dis_string_list = [dislib.complex_diststring_to_simple(x) for x in disorder_dict[lir]]
50
51     # FIX LIR LENGTH HERE ...
52
53     # calculate the length of the lir simply by using the first element in the list
54     lir_len = len(dis_string_list[0])
55
56     # initialize diz string score dictionary
57     temp_diz_string = {}
58     index = 0
59     while index < lir_len:
60         diz_string_score_dict[index] = 0
61         index+=1
62
63     index = 0
64     # loop over residues
65     while index < lir_len:
66         # loop over strings
67         for diz_str in dis_string_list:
68             # need to revise this and see how to handle these cases
69             if len(diz_str)==lir_len:
70                 if diz_str[index] == 'D':
71                     diz_string_score_dict[index]+=1
72         index+=1
73
74     # now generate consensus
75     consensus_dis_str = generate_percentage_consensus_per_residue_dizstring(diz_string_score_dict,

```

```

76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

```

```

no_dis_strings,
dis_percentage = dis_percentage)

# calculate consensus disorder percentage
calc_dis_percentage = dislib.calculate_disorder_percentage(consensus_dis_str)

if todb is True:
    db_disorder_data.append((consensus_dis_str, calc_dis_percentage, lir))
else:
    print "%s\t%s\t%s" % (lir, consensus_dis_str, calc_dis_percentage)

if todb is True:
    load_consensus_disorder_todb(db_disorder_data)

# -----

def generate_percentage_consensus_per_residue_dizstring(diz_string_score_dict, no_samples, dis_percentage = 50):
    """
    Computes the percentage per residue position of the disorder string dSTR

    diz_string_score_dict: A dictionary with all disorder scores
    percentage: The percentage per residue position of the LIR-motif

    return:
    """

    lir_len = len(diz_string_score_dict.keys())

    index = 0
    consensus_diz_string = ""
    while index < lir_len:
        # calculated disorder percentage
        calc_dis_percentage = (diz_string_score_dict[index]/no_samples)*100
        if calc_dis_percentage >= int(dis_percentage):
            consensus_diz_string = consensus_diz_string + 'D'
        # if percentage does not meet requirement we consider the residue as structured
        else:
            consensus_diz_string = consensus_diz_string + 'S'
        index+=1

```

```

116     return consensus_diz_string
117
118
119 # -----
120
121
122 def load_consensus_disorder_todb(disorder_data):
123     """
124     A list of tuples with the new disorder data to load to db
125
126     disorder_data: Disorder data
127
128     return: void
129     """
130
131     cnx = db.connect()
132     cursor = cnx.cursor(buffered=True)
133
134     query = "update sars_lir set cdSTR=%s,disorder_percentage=%s where lir_acc=%s"
135
136     cursor.executemany(query, disorder_data)
137     cnx.commit()
138
139     cursor.close()
140     cnx.close()
141
142     print "Done loading disorder data in the database"
143
144 # -----
145
146 def calculate_per_residue_consensus_disorder_score_advanced(disorder_string_list, dis_percentage = 50, todb=False):
147
148     disorder_dict = {}
149     fp_in = open(disorder_string_list, 'r')
150
151     consensus_disorder = {}
152     db_data = []
153
154     for line in fp_in:
155         line = line.strip().split('\t')

```

```

156     lir_acc = line[0]
157     dis_lir = line[1]
158     source = line[2] # predicted, indirect, curated
159
160     if lir_acc not in disorder_dict:
161         disorder_dict[lir_acc] = {source: [dis_lir]}
162     else:
163         if source in disorder_dict[lir_acc]:
164             disorder_dict[lir_acc][source].append(dis_lir)
165         else:
166             disorder_dict[lir_acc][source] = [dis_lir]
167
168     fp_in.close()
169
170     dis_string_list = []
171     diz_string_score_dict = {}
172     db_disorder_data = []
173
174     for lir in disorder_dict.keys():
175         # assuming there's only one curated dSTR, but might need to work with a list
176         if "curated" in disorder_dict[lir]:
177             consensus_dis_str = disorder_dict[lir]["curated"][0]
178             # calculate consensus disorder percentage
179             calc_dis_percentage = dislib.calculate_disorder_percentage(consensus_dis_str)
180
181             if todb is True:
182                 db_disorder_data.append((consensus_dis_str, calc_dis_percentage, lir))
183             else:
184                 print "%s\t%s\t%s" % (lir, consensus_dis_str, calc_dis_percentage)
185
186             continue
187
188     else:
189         # construct a unified dSTR list
190         unified_dSTR_list = []
191         for source in disorder_dict[lir]:
192             unified_dSTR_list.extend(disorder_dict[lir][source])
193
194         no_dis_strings = len(unified_dSTR_list)
195         # convert complex chars to Ds&Ss

```

```

196 dis_string_list = [dislib.complex_diststring_to_simple(x) for x in unified_dSTR_list]
197
198 # calculate the length of the lir simply by using the first element in the list
199 lir_len = len(dis_string_list[0])
200
201 # initialize diz string score dictionary
202 temp_diz_string = {}
203
204 index = 0
205 while index < lir_len:
206     diz_string_score_dict[index] = {'D': 0, 'S': 0}
207     index+=1
208
209 index = 0
210 # loop over residues
211 while index < lir_len:
212     # loop over strings
213     for diz_str in dis_string_list:
214         # see how to handle the cases where len(diz_str) != lir_len
215         if len(diz_str) == lir_len:
216             if diz_str[index] == 'D':
217                 diz_string_score_dict[index]['D']+=1
218             elif diz_str[index] == 'S':
219                 diz_string_score_dict[index]['S']+=1
220     index+=1
221
222 # now generate consensus
223
224 consensus_dis_str = generate_consensus_dSTR(diz_string_score_dict)
225 # calculate consensus disorder percentage
226 calc_dis_percentage = dislib.calculate_disorder_percentage(consensus_dis_str)
227
228 if todb is True:
229     db_disorder_data.append((consensus_dis_str, calc_dis_percentage, lir))
230 else:
231     print "%s\t%s\t%s" % (lir, consensus_dis_str, calc_dis_percentage)
232
233
234 if todb is True:
235     load_consensus_disorder_todb(db_disorder_data)

```



```

236
237 # -----
238
239 def generate_consensus_dSTR(diz_string_score_dict):
240
241     dSTR = ""
242     for position in sorted(diz_string_score_dict.keys()):
243         if diz_string_score_dict[position]['D'] > diz_string_score_dict[position]['S']:
244             dSTR+='D'
245         else:
246             dSTR+='S'
247
248     return dSTR
249
250 # -----
251 if __name__ == '__main__':
252
253     """ Query to generate input list
254     select lir_acc, disorder_string, mobidb_data from sars_lir_disorder
255     where lir_acc > 96
256     order by lir_acc
257     """
258
259
260     disorder_string_list = sys.argv[1]
261     disorder_percentage = int(sys.argv[2])
262
263
264     if "--loadDB" in sys.argv:
265         calculate_per_residue_consensus_disorder_score_advanced(disorder_string_list,
266                                                                 dis_percentage = 50, todb=True)
267     else:
268         calculate_per_residue_consensus_disorder_score_advanced(disorder_string_list,
269                                                                 dis_percentage = disorder_percentage)

```

7.3 anchor2_scanner.py code

```
1 import os
2 import sys
3
4 from ilir3d.lib import lir3d_db_connector as db
5
6 # -----
7
8
9 def iupred2_results_to_dict(anchor2_result_file):
10     """
11     Parses iupred2 result output and loads iupred2 and anchor2
12     (if available) scores into a dictionary where keys are the
13     position of each residue
14
15     anchor2_result_file: The output of iupred2A script running with
16     -d option for ANCHOR2
17
18     return: a dictionary with all data in the iupred2A results file
19     """
20
21     fp = open(anchor2_result_file, 'r')
22
23     iupred2_dict = {}
24
25     for iupred2_line in fp:
26         if iupred2_line[0] != '#':
27             iupred2_scores = iupred2_line.strip().split('\t')
28
29             if len(iupred2_scores) == 3: # only iupred2 prediction
30                 iupred2_dict[int(iupred2_scores[0])] = {"IUPred2": float(iupred2_scores[2])}
31             else:
32                 iupred2_dict[int(iupred2_scores[0])] = {"IUPred2": float(iupred2_scores[2]),
33                                                         "ANCHOR2": float(iupred2_scores[3])}
34
35     fp.close()
36
37     return iupred2_dict
```

```

38
39 # -----
40
41
42 def pull_lir_information_from_db(uniprot_id):
43     """
44     Pulls all the necessary LIR information based on uniprot id.
45     This is the LIR accession and start-end positions
46
47     uniprot_id: A valid uniprot id
48
49     return: Data retrieved from the database
50     """
51
52     cnx = db.connect()
53     cursor = cnx.cursor(buffered=True)
54
55     query = "select lir_acc, lir_start, lir_end from sars_lir where uniprot_id=\'%s\'"
56
57     cursor.execute(query % uniprot_id)
58
59
60     data = cursor.fetchall()
61
62     cursor.close()
63     cnx.close()
64
65     return data
66
67 # -----
68
69
70 def calculate_iupred2_prediction_percentage(iupred2_dict, lir_start, lir_end, type = "IUPred2", threshold = 0.5):
71     """
72     Predicts iupred percentage based on the iupred type and the lir overlap
73
74     iupred2_dict: A dictionary with the relevant protein iupred2 scores
75     lir_start: The lir start coordinate
76     lir_end: The lir end coordinate
77     threshold: A threshold according to which the prediction is considered true

```

```

78
79     return: The percentage of the overlap
80     """
81
82     num_involved_residues = 0
83     index = int(lir_start)
84
85     while index <= int(lir_end):
86         if iupred2_dict[index][type] >= threshold:
87             num_involved_residues+=1
88             index+=1
89
90     lir_len = int(lir_end)-int(lir_start)+1
91
92     prediction_percentage = (num_involved_residues*100)/lir_len
93
94     return prediction_percentage
95
96 # -----
97
98
99 def get_iupred2_lir_disorder_string(iupred2_dict, lir_start, lir_end, threshold = 0.5):
100     """
101     Generates a disorder string based on the iupred2 disorder predictions in iupred2_dict
102     and the corresponding lir coordinates
103
104     iupred2_dict: A dictionary with the relevant protein iupred2 scores
105     lir_start: The lir start coordinate
106     lir_end: The lir end coordinate
107     threshold: A threshold according to which the prediction is considered true
108
109     return: A disorder string
110     """
111
112     disorder_string = ""
113     index = int(lir_start)
114
115     while index <= int(lir_end):
116         if iupred2_dict[index]["IUPred2"] >= threshold:
117             disorder_string = disorder_string + 'D'

```

```

118         else:
119             disorder_string = disorder_string + 'S'
120             index+=1
121
122     return disorder_string
123
124     # -----
125
126     def load_anchor2_results_to_db(data):
127         """
128         This function updates anchor2 field in sars_lir table in iLIR3D database
129
130         data: A list of tuples in the format (X,Y) where X: anchor2 prediction and
131         Y: the accession of the corresponding lir (lir_acc)
132         """
133
134         cnx = db.connect()
135         cursor = cnx.cursor(buffered=True)
136
137         query = "update sars_lir set anchor3=%s where lir_acc=%s"
138
139         cursor.executemany(query, data)
140         cnx.commit()
141
142         cursor.close()
143         cnx.close()
144
145         print "Anchor2 results loaded to DB!"
146
147     # -----
148
149
150     def load_iupred2_results_to_db(data):
151         """
152         Creates a new iupred2 entry in the database, disorder table in particular
153
154         data: sars_lir_disorder attributes to load into the database
155
156         return: void
157         """

```

158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197

```
cnx = db.connect()
cursor = cnx.cursor(buffered=True)

query = "insert into sars_lir_disorder(lir_acc2,uniprot_id,lir_start,lir_end,lir,disorder_string,percentage,verified,mobidb_method,mobidb_method_order)
values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"

cursor.executemany(query, data)
cnx.commit()

cursor.close()
cnx.close()

print "Iupred2 results loaded to DB!"

# -----

def pull_input_for_iupred2_predictions(uniprot_id=None):
    """
    Pulls necessary information from the DB for
    uniprot_id: If uniprot_id is None, it will pull information of all lirs
    return a dictionary of values to be used to produce iupred2 entries for
    sars_lir_disorder
    """
    cnx = db.connect()
    cursor = cnx.cursor(buffered=True)

    query = ""
    data = {}
    if uniprot_id is None:
        query = "select lir_acc, uniprot_id, lir_start, lir_end, lir, verified from sars_lir order by uniprot_id"
        cursor.execute(query)
    else:
        query = "select lir_acc, uniprot_id, lir_start, lir_end, lir, verified from sars_lir where uniprot_id=%s"
        cursor.execute(query % uniprot_id)
```

```

198
199
200     for row in cursor:
201         if row[1] not in data:
202             data[row[1]] = {row[0]: {"start": row[2], "end": row[3], "lir": row[4], "verified": row[5]}}
203         else:
204             data[row[1]][row[0]] = {"start": row[2], "end": row[3], "lir": row[4], "verified": row[5]}
205
206     cursor.close()
207     cnx.close()
208
209     return data
210
211 # -----
212
213 def produce_anchor2_data_binary(iupred_results, threshold = 0.5, cutoff = 50):
214     """
215     Produces new data based on the iupred2 predictions provided and a threshold for the
216     anchor2 overlap
217
218     iupred_results: An iupred2 result file or
219     threshold = 50
220     """
221
222     anchor2_data = []
223     if os.path.isdir(iupred_results):
224
225         iupred2_result_files = os.listdir(iupred2_result_directory)
226
227         for iupred2_result_file in iupred2_result_files:
228             # parse iupred2
229             iupred2_result_file_path = os.path.join(iupred2_result_directory, iupred2_result_file)
230             iupred2_dict = iupred2_results_to_dict(iupred2_result_file_path)
231
232             uniprot_id = iupred2_result_file.partition('.')[0]
233             protein_lirs = pull_lir_information_from_db(uniprot_id)
234
235             for lir in protein_lirs:
236                 lir_start = lir[1]
237                 lir_end = lir[2]

```

```

238         anchor2_score = calculate_iupred2_prediction_percentage(iupred2_dict,
239                                                                lir_start, lir_end,
240                                                                type = "ANCHOR2", threshold = threshold)
241         # convert anchor2 prediction to binary based on
242         if anchor2_score >= cutoff:
243             anchor2_data.append((1, lir[0]))
244         else:
245             anchor2_data.append((0, lir[0]))
246
247     elif os.path.isfile(iupred_results):
248         iupred2_dict = iupred2_results_to_dict(iupred_results)
249
250         uniprot_id = iupred_results.partition('.')[0]
251         protein_lirs = pull_lir_information_from_db(uniprot_id)
252
253         for lir in protein_lirs:
254             lir_start = lir[1]
255             lir_end = lir[2]
256             anchor2_score = calculate_iupred2_prediction_percentage(iupred2_dict,
257                                                                    lir_start, lir_end,
258                                                                    type = "ANCHOR2", threshold = threshold)
259             # convert anchor2 prediction to binary based on
260             if anchor2_score >= threshold:
261                 anchor2_data.append((1, lir[0]))
262             else:
263                 anchor2_data.append((0, lir[0]))
264
265         load_anchor2_results_to_db(anchor2_data)
266
267     # -----
268
269
270 def produce_iupred2_data(iupred_results, threshold = 0.5, uniprot_id = None):
271     """
272
273     iupred_results: A dictionary with multiple iupred results or a single file
274     threshold: Threshold defines accepted values for disorder prediction
275     return:
276     """
277     db_iupred2_data = pull_input_for_iupred2_predictions(uniprot_id=uniprot_id)

```



```

278
279 iupred2_data = []
280
281
282 method = ''
283
284 if "--anchor2" in sys.argv:
285     method = "anchor2"
286 elif "--iupred2" in sys.argv:
287     method = "iupred2"
288
289 if os.path.isdir(iupred_results):
290
291     iupred2_result_files = os.listdir(iupred2_result_directory)
292
293     for iupred2_result_file in iupred2_result_files:
294         # parse iupred2
295         iupred2_result_file_path = os.path.join(iupred2_result_directory, iupred2_result_file)
296         iupred2_dict = iupred2_results_to_dict(iupred2_result_file_path)
297
298         uniprot_id = iupred2_result_file.partition('.')[0]
299         protein_lirs = pull_lir_information_from_db(uniprot_id)
300
301         for lir in protein_lirs:
302             lir_start = lir[1]
303             lir_end = lir[2]
304
305             lir_iupred2_score = calculate_iupred2_prediction_percentage(iupred2_dict,
306                                                                           lir_start, lir_end,
307                                                                           type = "IUPred2", threshold = threshold)
308
309             lir_iupred2_disrting = get_iupred2_lir_disorder_string(iupred2_dict,
310                                                                     lir_start,
311                                                                     lir_end, threshold = threshold)
312
313             lir_acc = lir[0]
314             lir_string = db_iupred2_data[uniprot_id][lir_acc]["lir"]
315             verified = db_iupred2_data[uniprot_id][lir_acc]["verified"]
316             #method = "iupred2"
317             data_type = "predicted"

```

```

318
319
320         iupred2_data.append((lir_acc, uniprot_id, lir_start, lir_end, lir_string,
321                             lir_iupred2_disrting, lir_iupred2_score,
322                             verified, method, data_type, threshold))
323
324     elif os.path.isfile(iupred_results):
325
326         iupred2_dict = iupred2_results_to_dict(iupred_results)
327         uniprot_id = iupred_results.partition('.')[0]
328         protein_lirs = pull_lir_information_from_db(uniprot_id)
329
330     for lir in protein_lirs:
331         lir_start = lir[1]
332         lir_end = lir[2]
333
334         lir_iupred2_score = calculate_iupred2_prediction_percentage(iupred2_dict,
335                                                                     lir_start, lir_end,
336                                                                     type = "IUPred2", threshold = threshold)
337
338         lir_iupred2_disrting = get_iupred2_lir_disorder_string(iupred2_dict,
339                                                                lir_start,
340                                                                lir_end, threshold = threshold)
341
342         lir_acc = lir[0]
343         lir_string = db_iupred2_data[uniprot_id][lir_acc]["lir"]
344         verified = db_iupred2_data[uniprot_id][lir_acc]["verified"]
345         #method = "iupred2"
346         data_type = "predicted"
347
348         iupred2_data.append((lir_acc, uniprot_id, lir_start, lir_end,
349                             lir_string, lir_iupred2_disrting, lir_iupred2_score,
350                             verified, method, data_type, threshold))
351
352     # function to update the database
353     load_iupred2_results_to_db(iupred2_data)
354
355 # -----
356 def produce_anchor2_data_disorder(iupred_results, threshold = 0.5, uniprot_id = None):
357     """

```

```

358
359 :param iupred_results:
360 :param threshold:
361 :return:
362 """
363 db_iupred2_data = pull_input_for_iupred2_predictions(uniprot_id=uniprot_id)
364
365 iupred2_data = []
366
367
368 method = 'anchor2'
369
370 """
371 if "--anchor2" in sys.argv:
372     method = "anchor2"
373 elif "--iupred2" in sys.argv:
374     method = "iupred2"
375 """
376
377 if os.path.isdir(iupred_results):
378
379     iupred2_result_files = os.listdir(iupred2_result_directory)
380
381     for iupred2_result_file in iupred2_result_files:
382         # parse iupred2
383         iupred2_result_file_path = os.path.join(iupred2_result_directory, iupred2_result_file)
384         iupred2_dict = iupred2_results_to_dict(iupred2_result_file_path)
385
386         uniprot_id = iupred2_result_file.partition('.')[0]
387         protein_lirs = pull_lir_information_from_db(uniprot_id)
388
389         for lir in protein_lirs:
390             lir_start = lir[1]
391             lir_end = lir[2]
392
393             lir_iupred2_score = calculate_iupred2_prediction_percentage(iupred2_dict,
394                                                                           lir_start, lir_end,
395                                                                           type = "ANCHOR2", threshold = threshold)
396
397             lir_iupred2_disrting = get_iupred2_lir_disorder_string(iupred2_dict,

```

```

398                                     lir_start,
399                                     lir_end, threshold = threshold)
400
401     lir_acc = lir[0]
402     lir_string = db_iupred2_data[uniprot_id][lir_acc]["lir"]
403     verified = db_iupred2_data[uniprot_id][lir_acc]["verified"]
404     #method = "iupred2"
405     data_type = "predicted"
406
407
408     iupred2_data.append((lir_acc, uniprot_id, lir_start, lir_end, lir_string,
409                         lir_iupred2_disrting, lir_iupred2_score,
410                         verified, method, data_type, threshold))
411
412 elif os.path.isfile(iupred_results):
413
414     iupred2_dict = iupred2_results_to_dict(iupred_results)
415     uniprot_id = iupred_results.partition('.')[0]
416     protein_lirs = pull_lir_information_from_db(uniprot_id)
417
418     for lir in protein_lirs:
419         lir_start = lir[1]
420         lir_end = lir[2]
421
422         lir_iupred2_score = calculate_iupred2_prediction_percentage(iupred2_dict,
423                                                                     lir_start, lir_end,
424                                                                     type = "ANCHOR2", threshold = threshold)
425
426         lir_iupred2_disrting = get_iupred2_lir_disorder_string(iupred2_dict,
427                                                                 lir_start,
428                                                                 lir_end, threshold = threshold)
429
430         lir_acc = lir[0]
431         lir_string = db_iupred2_data[uniprot_id][lir_acc]["lir"]
432         verified = db_iupred2_data[uniprot_id][lir_acc]["verified"]
433
434         data_type = "predicted"
435
436         iupred2_data.append((lir_acc, uniprot_id, lir_start, lir_end,
437                             lir_string, lir_iupred2_disrting, lir_iupred2_score,
438                             verified, method, data_type, threshold))

```

```
438
439
440     # function to update the database
441     load_iupred2_results_to_db(iupred2_data)
442
443     # -----
444
445     if __name__ == '__main__':
446
447
448         iupred2_result_directory = sys.argv[1]
449         threshold = float(sys.argv[2])
450         cutoff = int(sys.argv[3])
451
452         if "--anchor2" in sys.argv:
453             produce_anchor2_data_binary(iupred2_result_directory, threshold = threshold, cutoff=cutoff)
454             #produce_anchor2_data_disorder(iupred2_result_directory, threshold = threshold, uniprot_id = None)
455         elif "--iupred2" in sys.argv:
456             produce_iupred2_data(iupred2_result_directory, threshold = threshold, uniprot_id = None)
```

7.4 spot_scanner.py code

```
1 import os
2 import sys
3
4 import ilir3d.lib.lir3d_db_connector as db
5
6
7 # -----
8
9 def parse_spot_list_desc(spot_desc):
10     """
11     Parses the SPOT-disorder description file
12
13     spot_desc: The output SPOT-disorder description file
14     return: A dictionary with uniprot_ids and filename mappings
15     """
16
17     id_mapings = {}
18     fp = open(spot_desc, 'r')
19
20     for line in fp:
21         line = line.strip().split(' ')
22         filename = line[0]
23         if filename not in id_mapings:
24             id_mapings[filename] = {}
25             uniprot = line[1].split('|')
26             uniprot_id = uniprot[2]
27             uniprot_acc = uniprot[1]
28             desc = (' ').join(line[3:])
29
30             id_mapings[filename]["id"] = uniprot_id
31             id_mapings[filename]["acc"] = uniprot_acc
32             id_mapings[filename]["desc"] = desc
33
34     fp.close()
35
36     return id_mapings
37
```

```

38 # -----
39
40 def parse_spot_file(spot_file):
41     """
42     Parses the SPOT-disorder output file and loads all data
43     in a dictionary
44
45     spot_file: The output file of SPOT-disorder
46
47     return: A dictionary with SPOT disorder data
48     """
49
50     spot_dict = {}
51
52     fp = open(spot_file, 'r')
53
54     # drop header
55     header = fp.readline()
56
57     for line in fp:
58         line = line.strip().split('\t')
59         spot_dict[int(line[0])] = line[3]
60
61     fp.close()
62
63     return spot_dict
64
65 # -----
66
67 def load_regions_from_db(uniprot_id):
68     """
69     Loads all necessary regions from iLIR3D database in order
70     to identify disorder overlaps
71
72     return: Query data in a list
73     """
74
75     cnx = db.connect()
76     cursor = cnx.cursor(buffered=True)
77

```

```

78 # this will limit results to only the old ones
79 query = "select lir_acc, lir_start, lir_end from sars_lir where uniprot_id=\'%s\'"
80
81 cursor.execute(query % uniprot_id)
82
83 data = cursor.fetchall()
84
85 cursor.close()
86 cnx.close()
87
88 return data
89
90 # -----
91
92 def dstr_constructor(lir_dstr_dict):
93     """
94     Constructs the disorder string (dstr) based on the given dictionary
95
96     lir_dstr_dict: A dictionary with disordered residues for each position
97     of a LIR-motif
98
99     return: The disorder string dSTR
100     """
101
102     dstr = ""
103
104     for lir_index in sorted(lir_dstr_dict.keys()):
105         dstr+=lir_dstr_dict[lir_index]
106
107     return dstr
108
109 # -----
110
111 def spot_disorder_overlap_scanner(spot_results, lir_regions):
112     """
113     Scans a LIR-motif for disordered residues based on the LIR-motif
114     start and end positions.
115
116     uniprot_id: A valid UniProt id
117     spot_file: The output file of SPOT-disorder with all calculated

```



```

118 disordered regions of a sequence file defined by uniprot_id
119 lir_regions: A list with all lir regions stored in tuples as (lir_acc, start, end)
120
121 return: A dictionary with lir all disorder strings
122 """
123
124 prot_dstrs = {}
125 lir_dstr_dict = {}
126
127 spot_annotations = parse_spot_file(spot_results)
128
129 for region in lir_regions:
130
131     # initialize dictionary
132     start = int(region[1])
133     end = int(region[2])
134     index = start
135
136     while index <= end:
137         lir_dstr_dict[index] = '?'
138         index += 1
139
140     # now look for disorder annotations
141     index = start
142     while index <= end:
143         if index in spot_annotations:
144             lir_dstr_dict[index] = spot_annotations[index]
145             index += 1
146
147     # assign the dstr to its corresponding LIR
148     dstr = '' # sanity initialization
149     dstr = dstr_constructor(lir_dstr_dict)
150
151     prot_dstrs[region[0]] = dstr
152     lir_dstr_dict = {}
153
154 return prot_dstrs
155
156 # -----
157

```

```

158 def calculate_disorder_percentage(dstr):
159     """
160     Calculates percentage of D characters found in dstr
161
162     dstr: Disorder string dSTR (e.g. DDDSSD)
163
164     return: Disorder percentage
165     """
166
167     lir_length = len(dstr)
168     number_of_Ds = dstr.count('D')
169
170     disorder_percentage = (number_of_Ds * 100) / lir_length
171
172     return disorder_percentage
173
174
175 # -----
176
177 def fetch_disorder_fields_from_db(lir_acc):
178     """
179
180     uniprot_id:
181     return:
182     """
183
184     cnx = db.connect()
185     cursor = cnx.cursor(buffered=True)
186
187     # this will limit results to only the old ones
188     query = "select lir, verified from sars_lir where lir_acc=%s"
189
190     cursor.execute(query % lir_acc)
191
192     data = cursor.fetchall()[0]
193
194     cursor.close()
195     cnx.close()
196
197     return data

```

```

198 # -----
199
200
201 if __name__ == '__main__':
202     # main produces iLIR3D ready data for import in tabular format
203
204     spot_desc = sys.argv[1] # /path/to/spotd/list.desc
205     spot_output_dir = sys.argv[2] # /path/to/spotd
206
207     metadata = parse_spot_list_desc(spot_desc)
208
209     for case in metadata:
210         uniprot_id = metadata[case]["id"]
211         uniprot_acc = metadata[case]["acc"]
212
213         lir_regions = []
214         lir_regions = load_regions_from_db(uniprot_id)
215
216         lir_region_dict = {}
217
218         for region in lir_regions:
219             lir_region_dict[region[0]] = {"start": region[1], "end": region[2]}
220
221     spot_file = os.path.join(spot_output_dir, case + ".spotd")
222
223     spot_dstrs = spot_disorder_overlap_scanner(spot_file, lir_regions)
224
225     for lir_acc in sorted(spot_dstrs.keys()):
226         dstr = spot_dstrs[lir_acc]
227         lir_start = lir_region_dict[lir_acc]["start"]
228         lir_end = lir_region_dict[lir_acc]["end"]
229
230         (lir_seq, verified) = fetch_disorder_fields_from_db(int(lir_acc))
231
232         print "%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s" % (lir_acc, uniprot_id,
233                                                         lir_start, lir_end,
234                                                         lir_seq, dstr,
235                                                         str(calculate_disorder_percentage(dstr)),
236                                                         verified, "spot", "predicted", "")
237

```

IOANNA KALVARI