



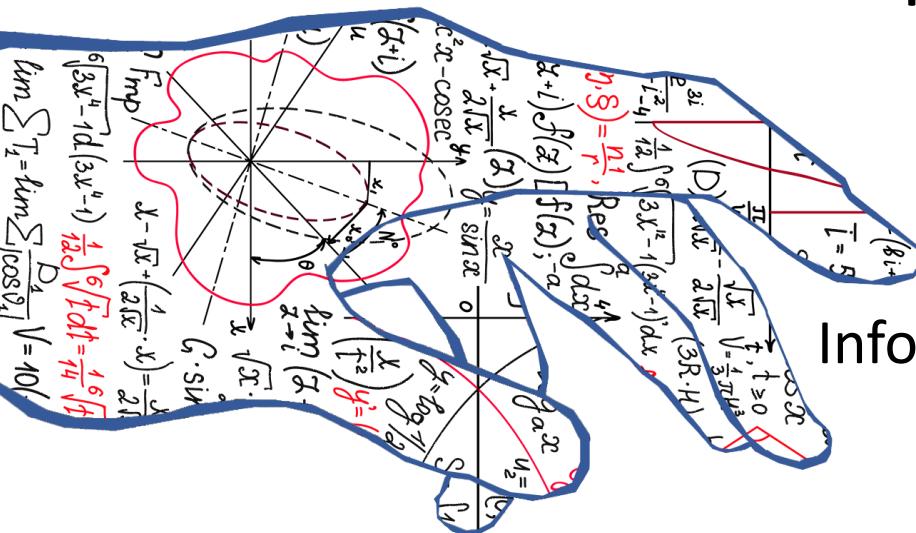
ITMO UNIVERSITY



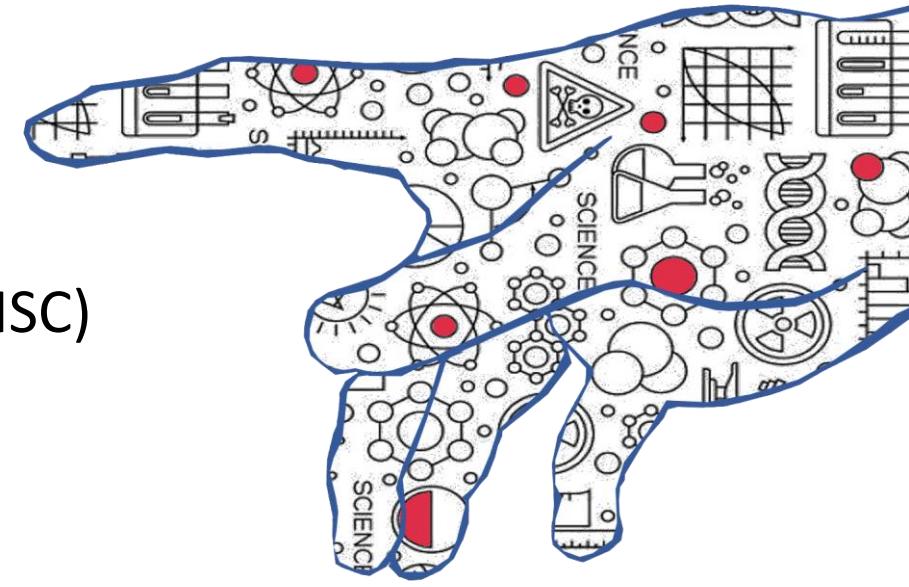
INFOCHEMISTRY SCIENTIFIC CENTER

# Cheminformatics and synthetic biology: computational methods and projects

Prof. Sergey Shityakov

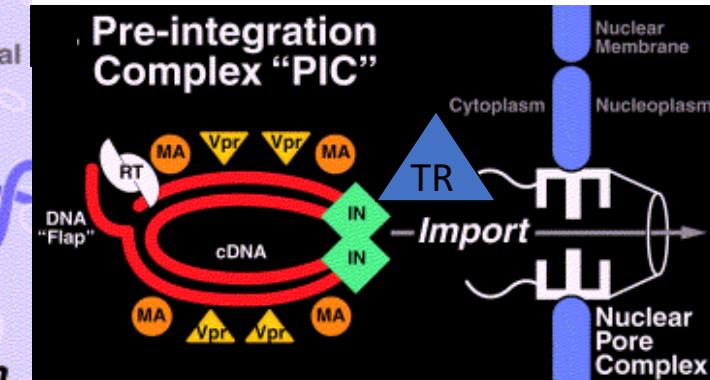
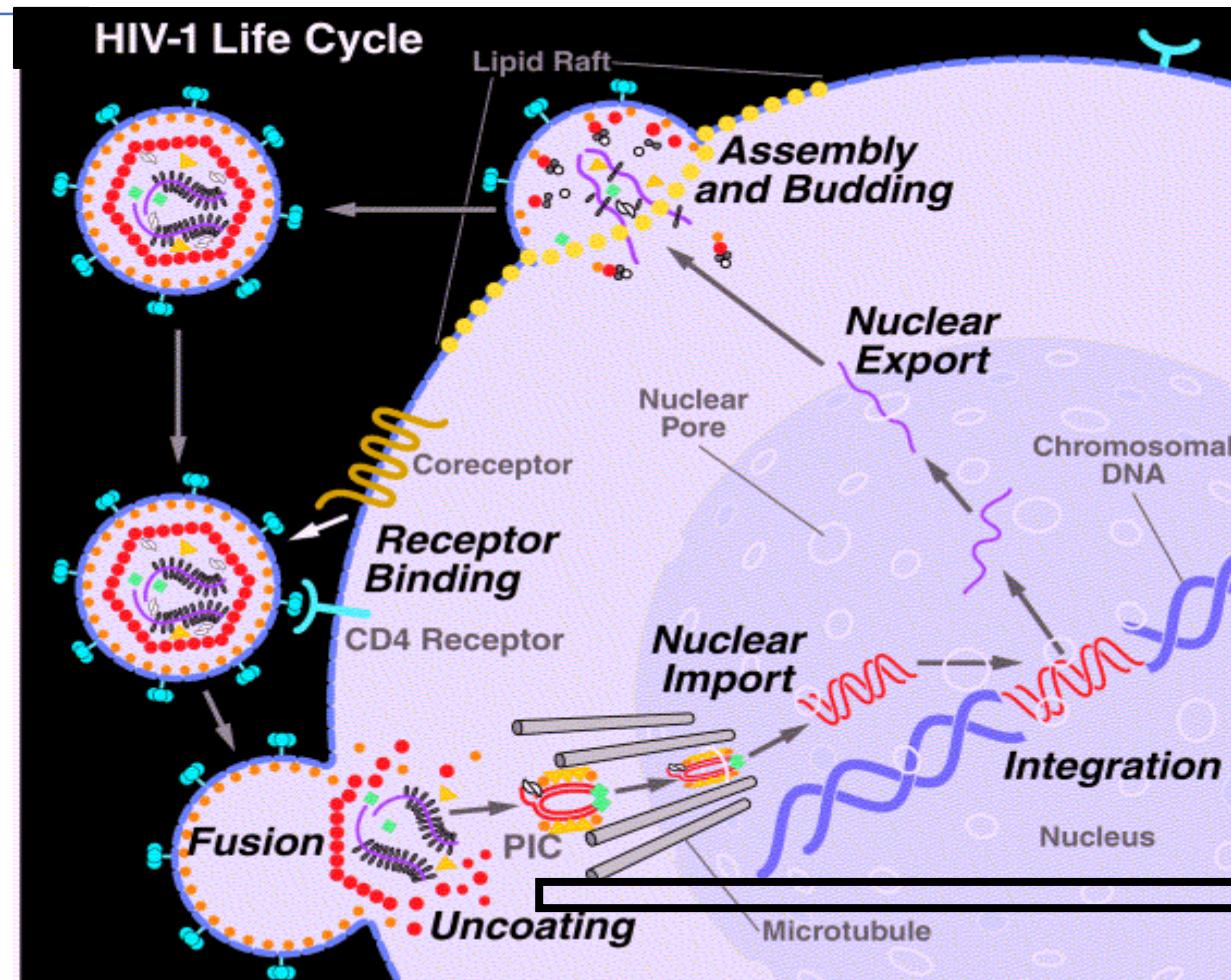


Infochemistry Scientific Center (ISC)  
ITMO University  
Saint-Petersburg, 2024





# HIV-1 PIC



Sherman et al., Micro. Inf, 2002

IT'S MORE than a  
UNIVERSITY



# Synthetic biology of HIV

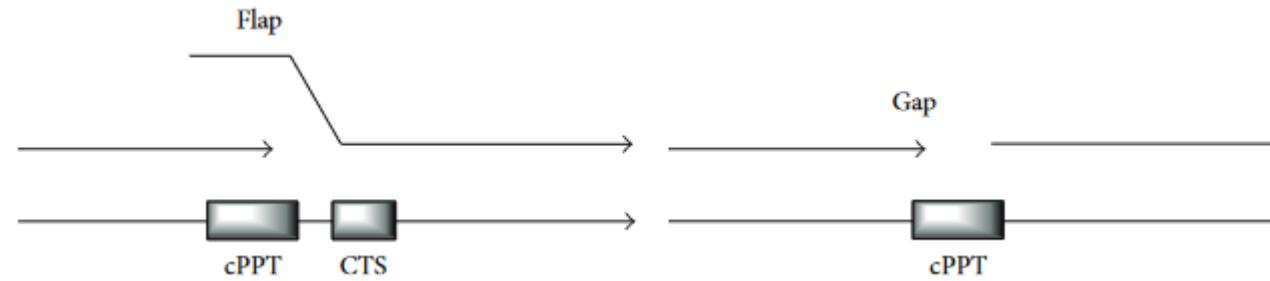
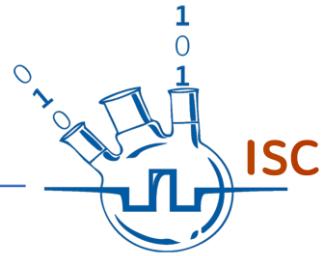


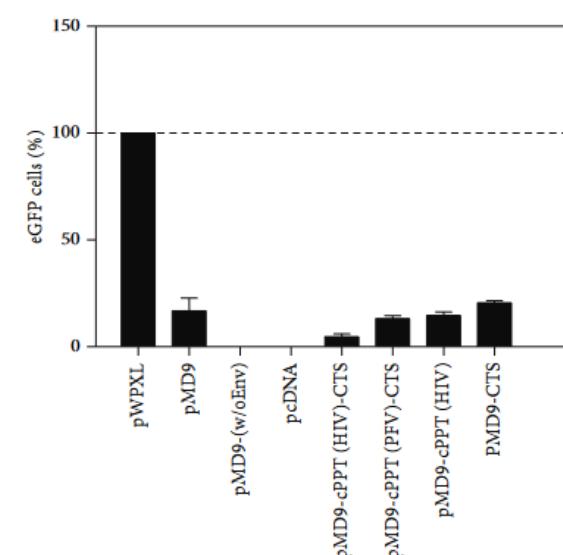
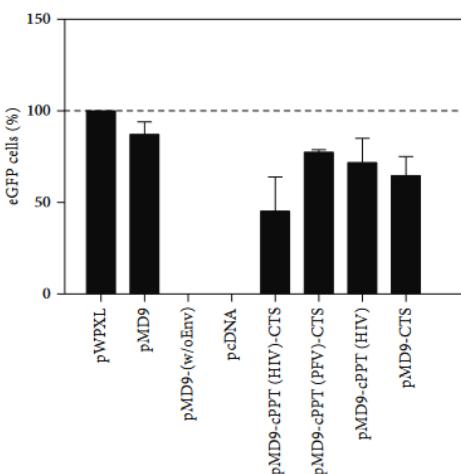
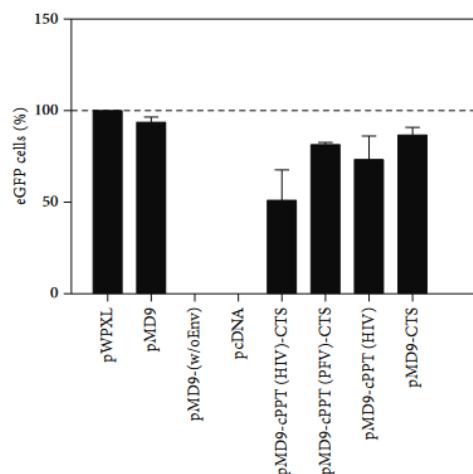
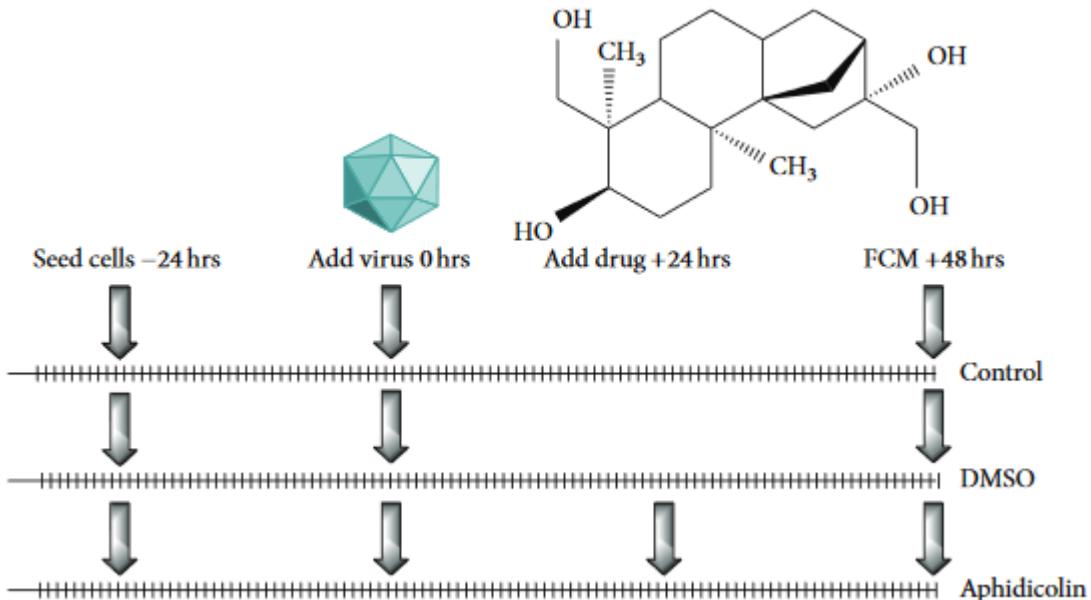
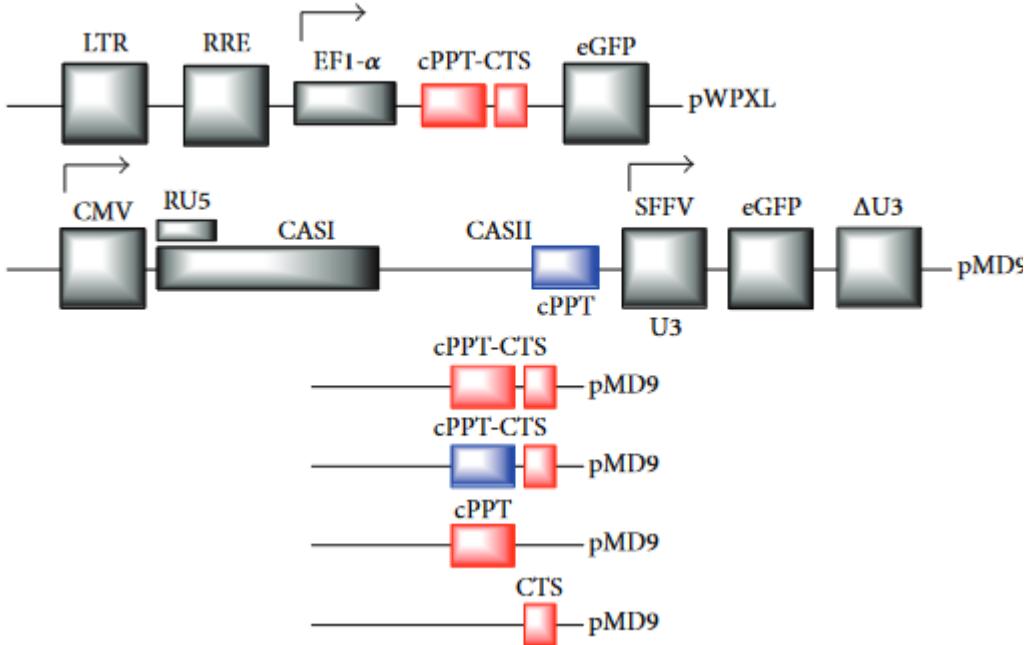
TABLE 1: Sequences of the viral structural elements used in the construction of modified expression vectors.

Element	Length (bp)	Sequence
cPPT (PFV)	9	AGGAGAGGG
cPPT (HIV)	33	ATCCACAATTTAAAAGAAAAGGGGGATTGGG
CTS	16	AAAAAATTCAAAATT

TABLE 2: Forward and reverse primers for a PCR amplification of the fragments used in the construction of modified expression vectors.

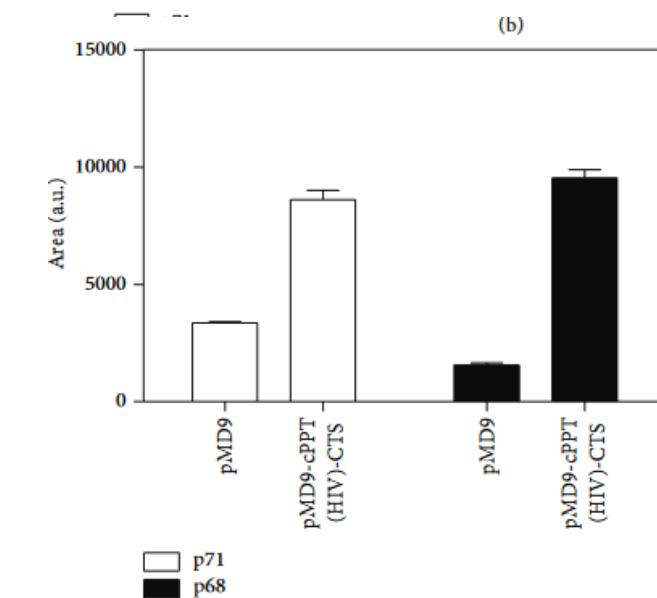
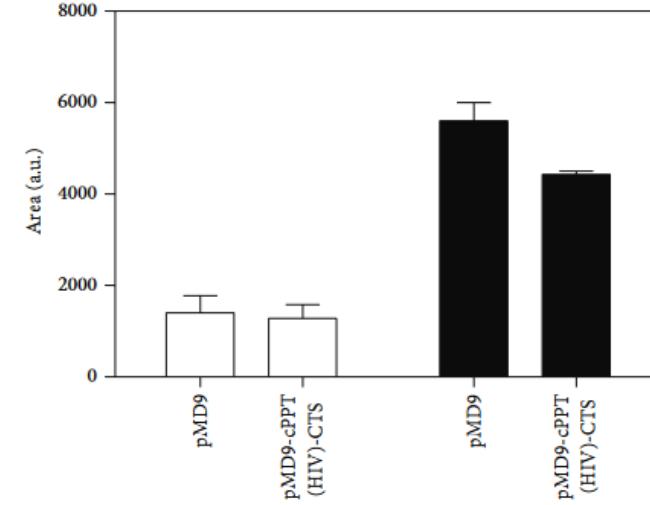
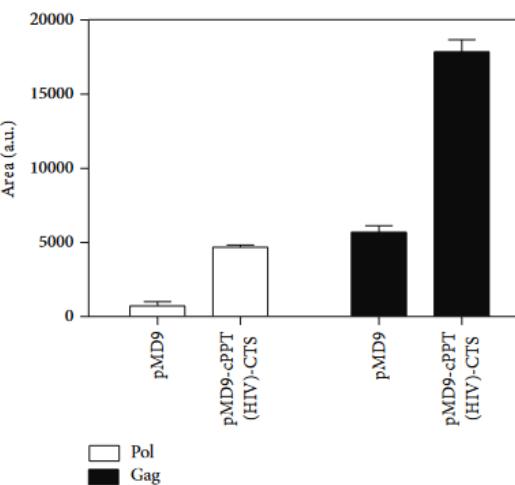
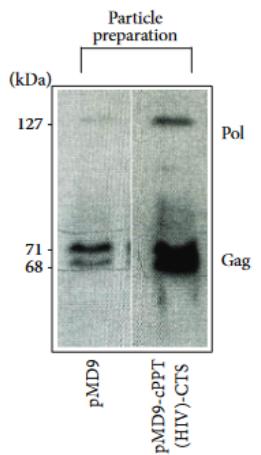
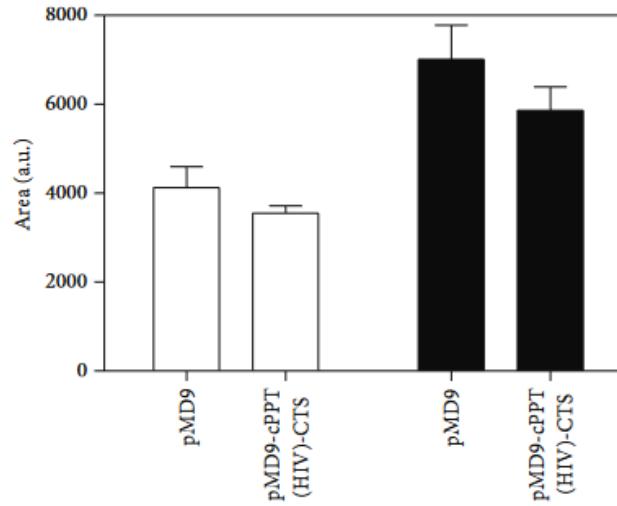
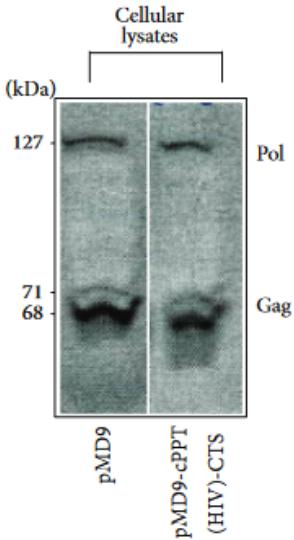
Fragments	Primers
cPPT (PFV)-CTS	5'-TATACAATTGCAGGAGAGGGATTGGGGGTACAGTGCAG-3' 5'-TATAAGGCCTCTGTCCCTGTAATAAACCC-3'
cPPT (HIV)-CTS	5'-TATACAATTGATGGCAGTATCCAC-3' 5'-TATAAGGCCTCTGTCCCTGTAATAAACCC-3'
cPPT (HIV)	5'-TATACAATTGATGGCAGTATCCAC-3' 5'-TATAAGGCCTGTAATTGTTTTGTAATTCT-3'
CTS	5'-TATATAAGGCCTCCCTGTAACCCGAAAATTTG-3' 5'-TATATAAGGCCTCCCTGTAACCCGAAAATTTG-3'

# Synthetic biology of HIV

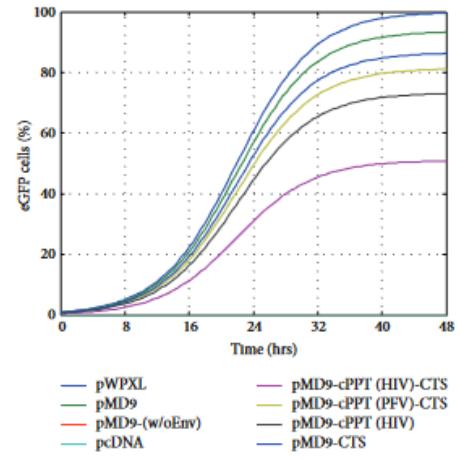


IT'S MORE than a  
UNIVERSITY

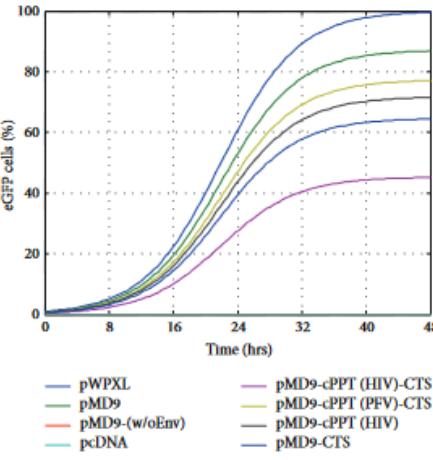
# Synthetic biology of HIV



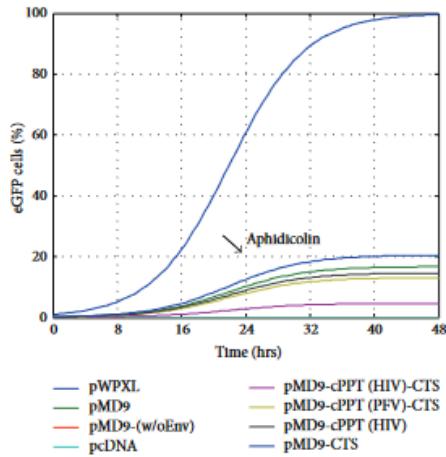
# Synthetic biology of HIV



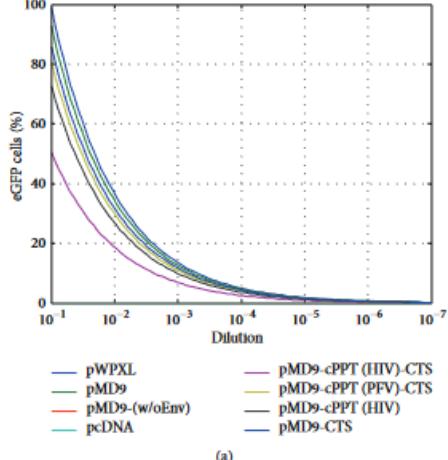
(a)



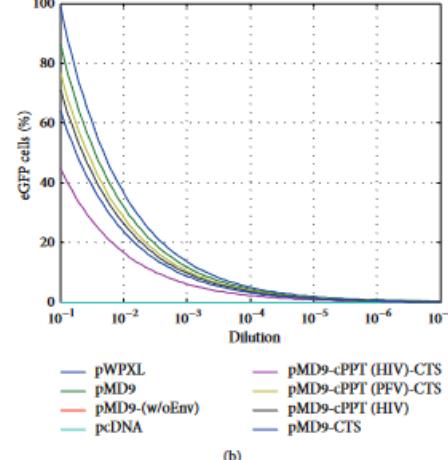
(b)



(a)



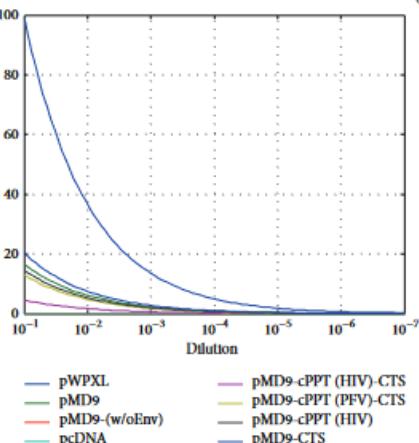
(a)



(b)

$$f(x)_{\text{td}} = \frac{0.01a}{0.01 + e^{(-bx)^b}}$$

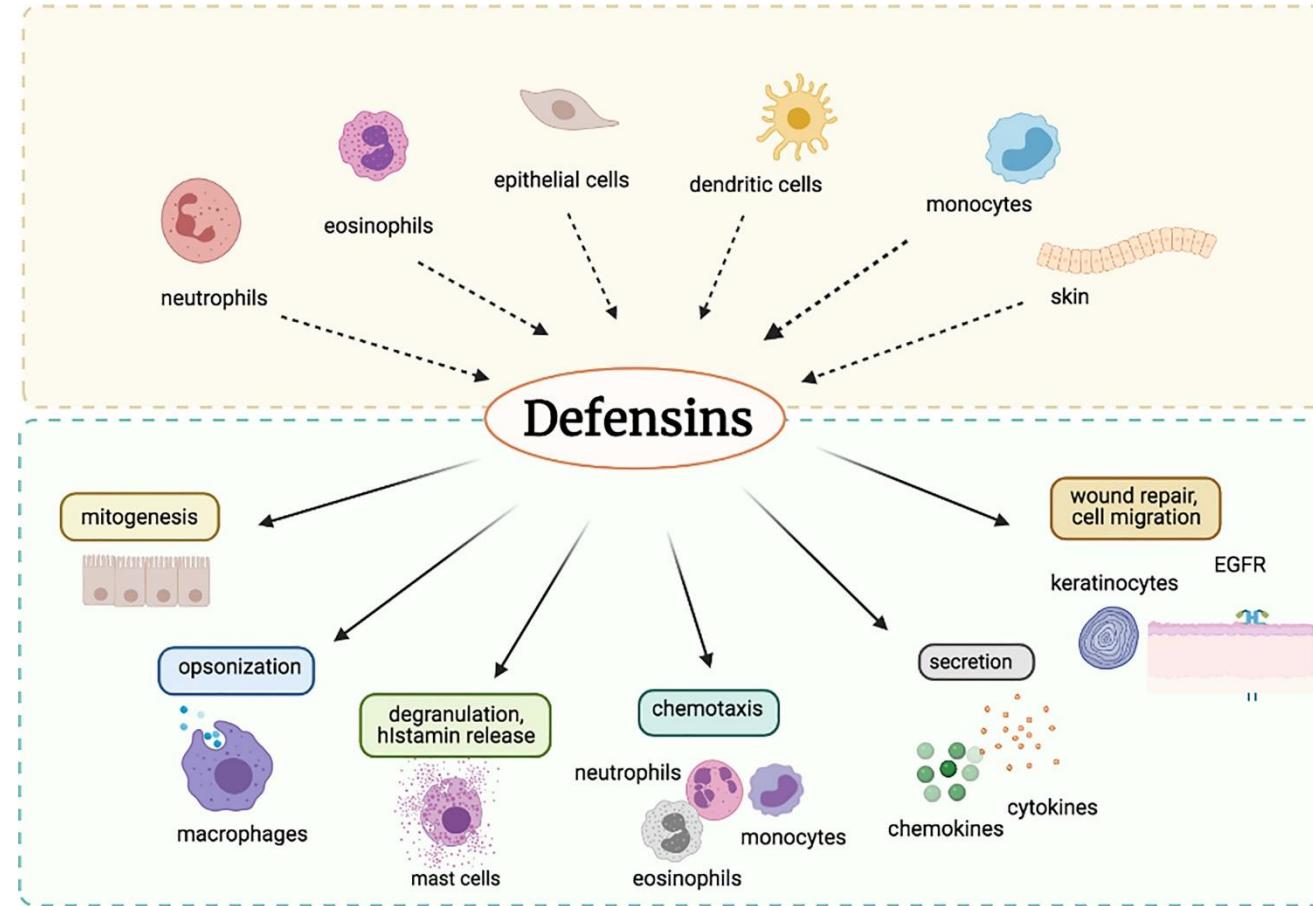
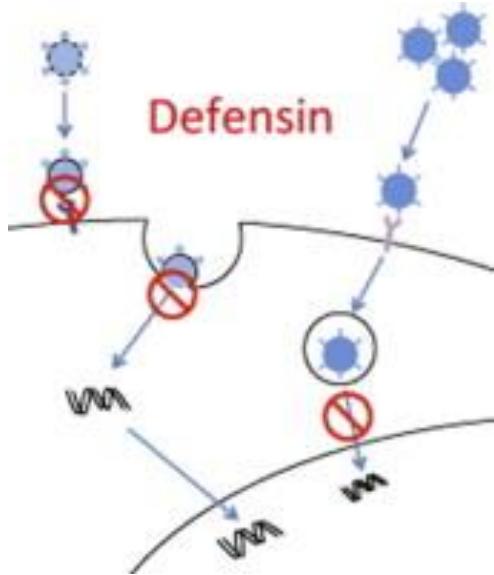
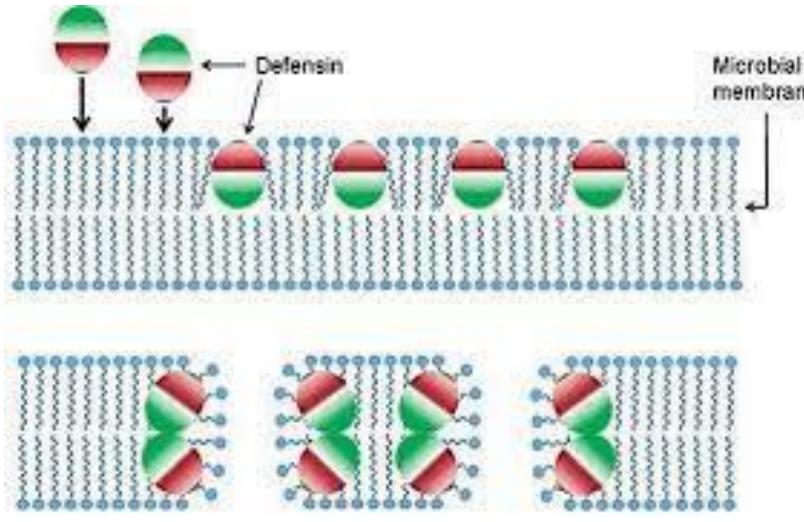
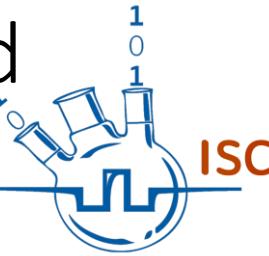
$$f(x)_{\text{vt}} = \frac{a}{0.01 + e^x^b}$$



IT'S MORE than a  
UNIVERSITY

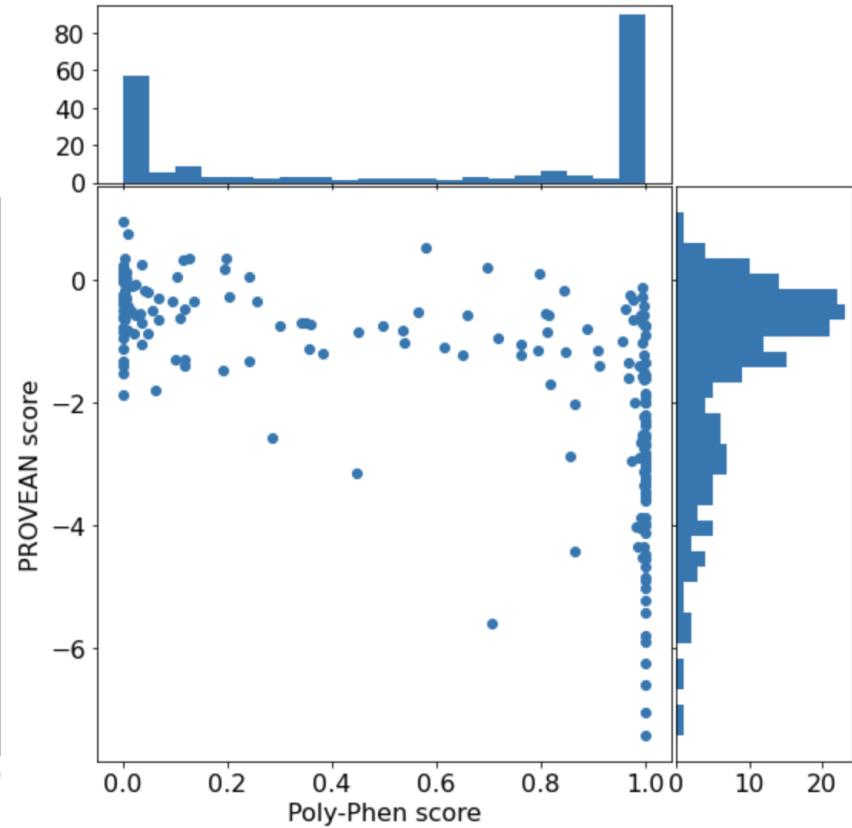
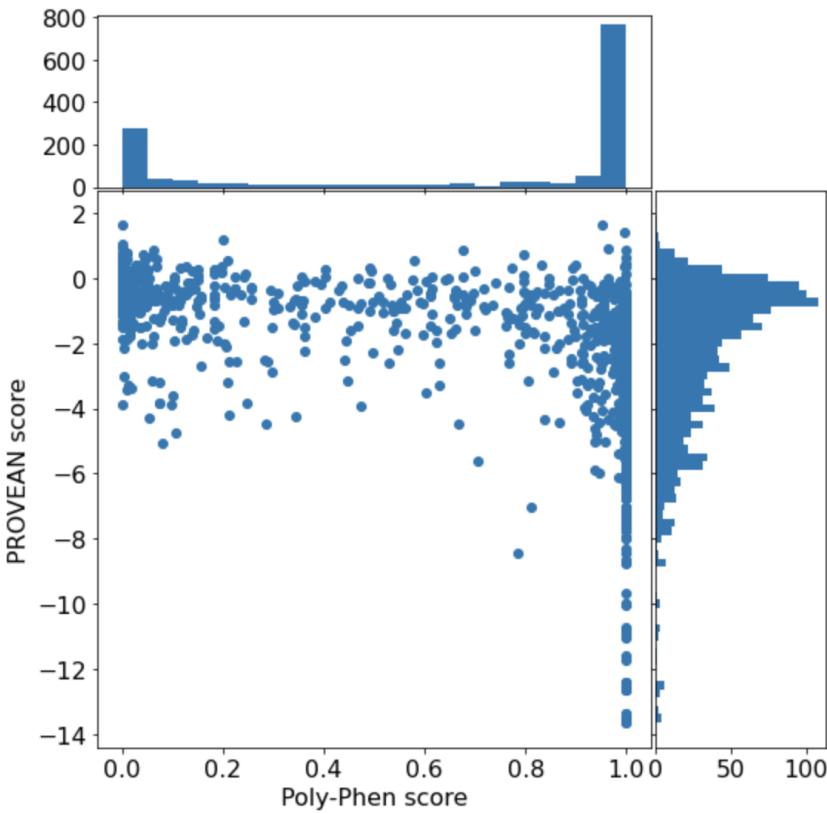
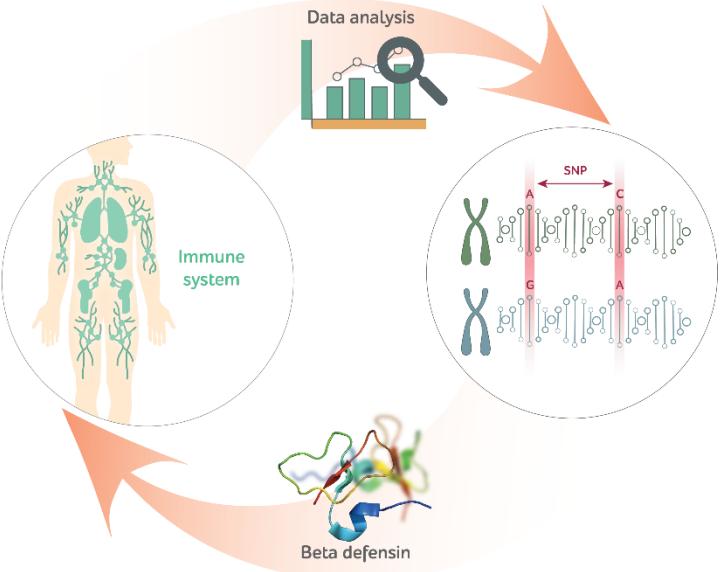
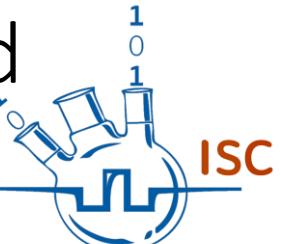


# nsSNPs in DEFB1 gene reveal impact on protein-ligand binding sites



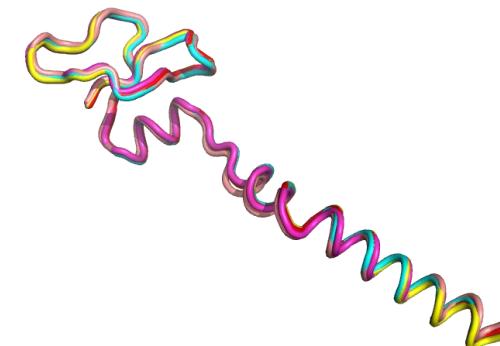
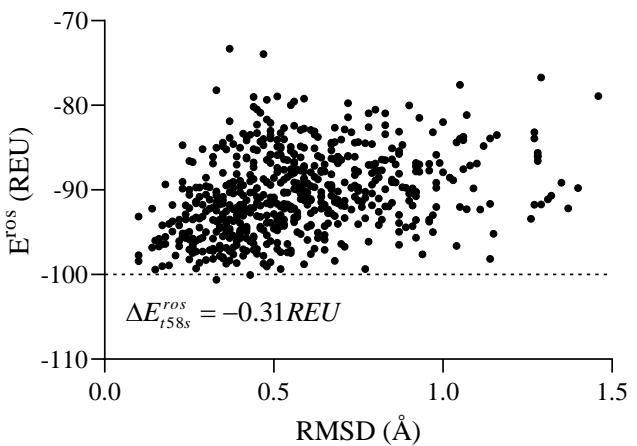
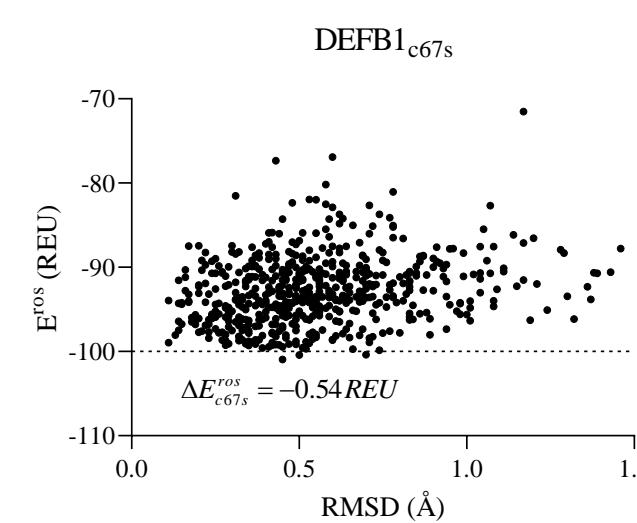
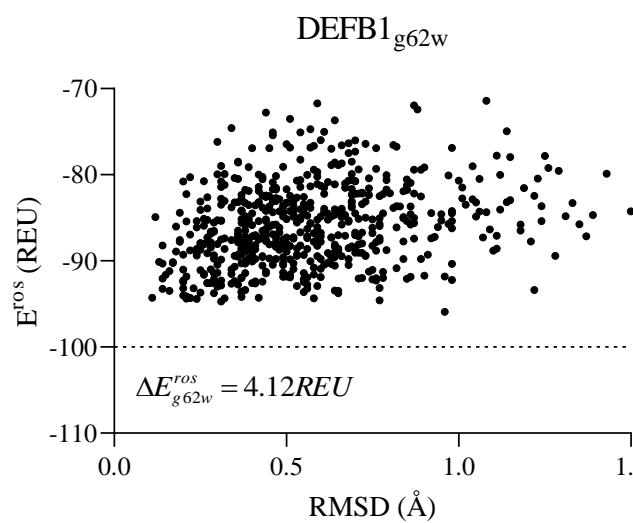
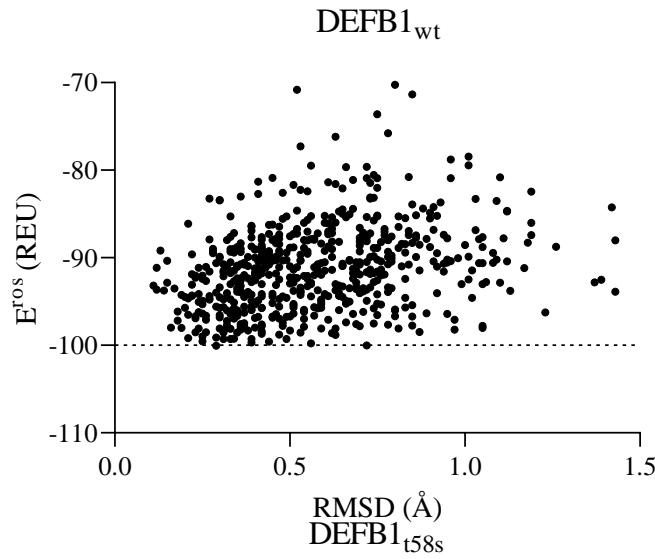
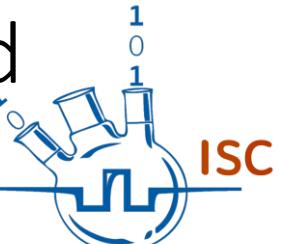


# nsSNPs in DEFB1 gene reveal impact on protein-ligand binding sites

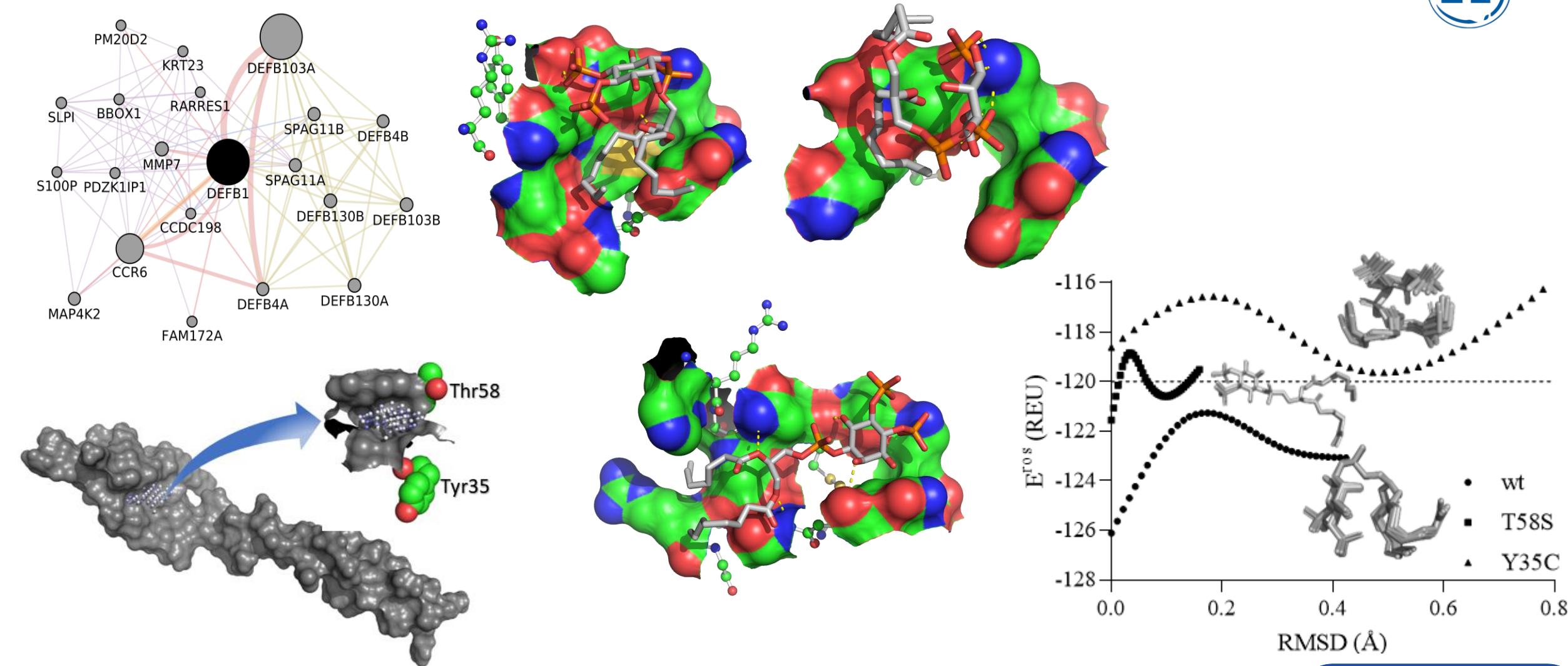
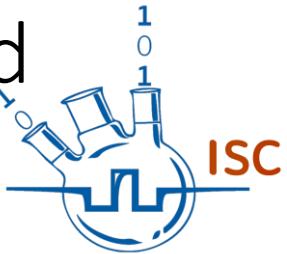




# nsSNPs in DEFB1 gene reveal impact on protein-ligand binding sites

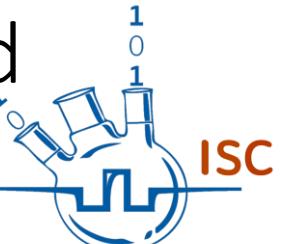
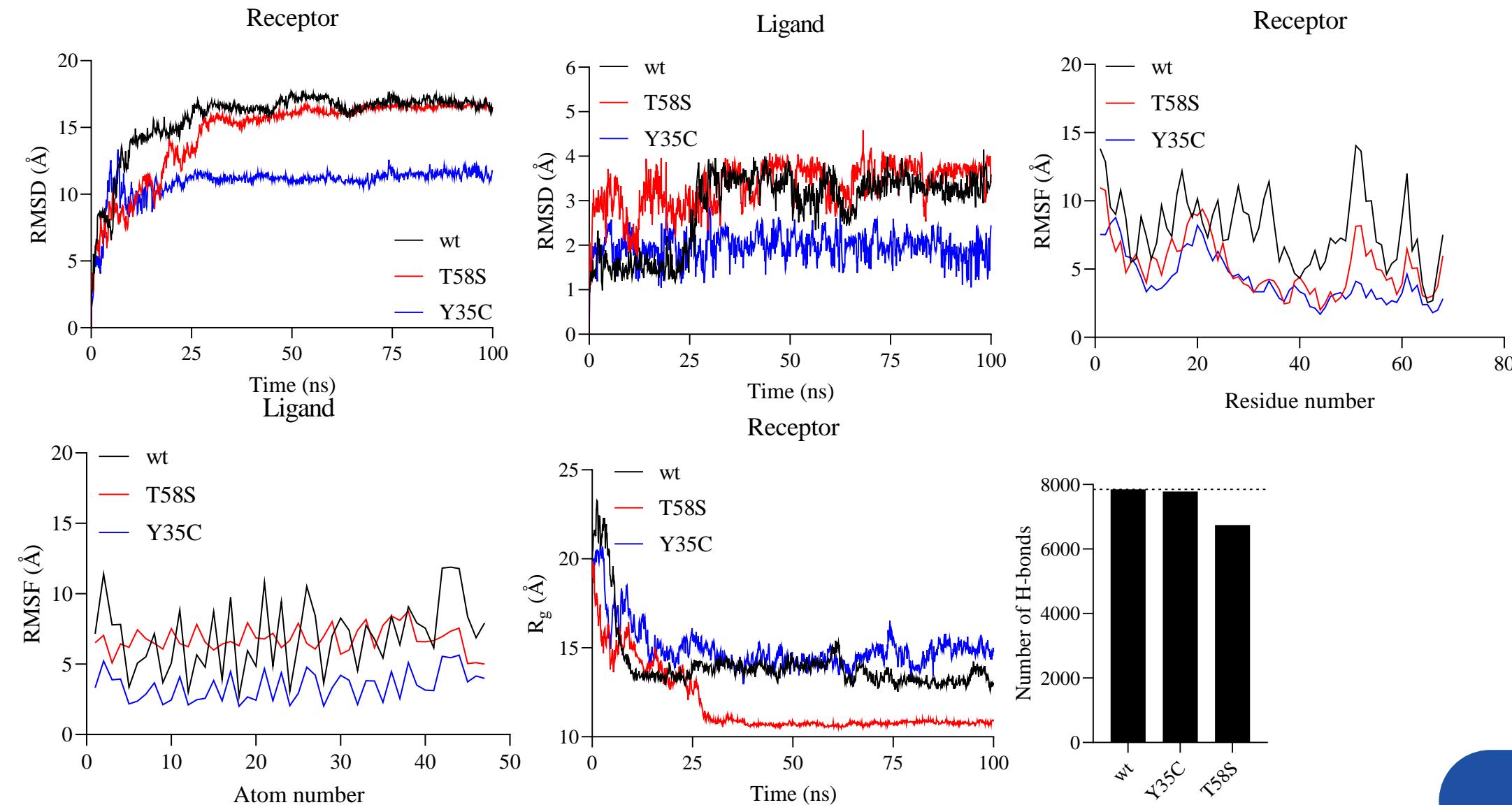


# nsSNPs in DEFB1 gene reveal impact on protein-ligand binding sites

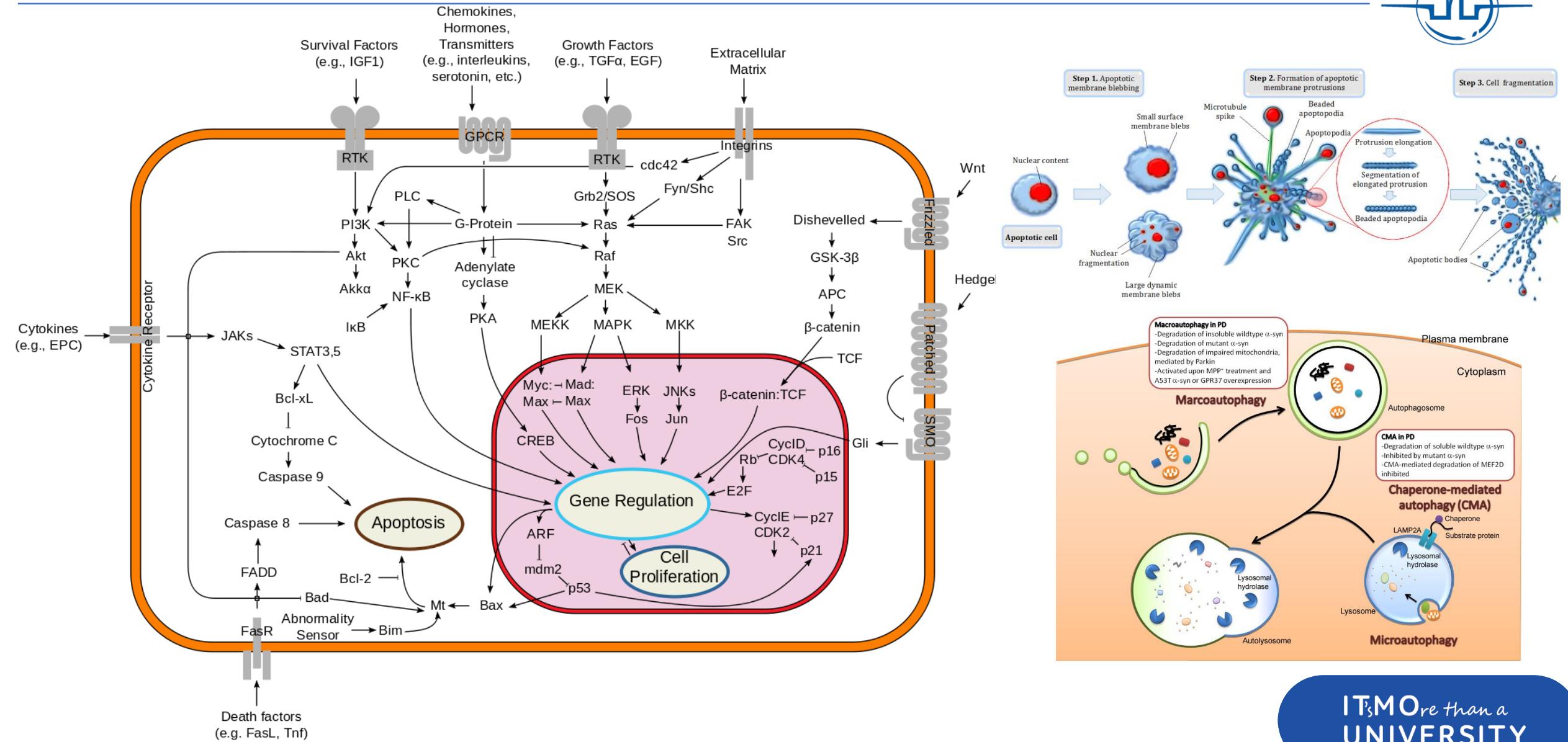
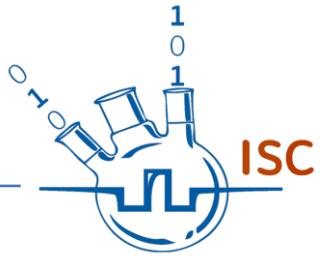




# nsSNPs in DEFB1 gene reveal impact on protein-ligand binding sites

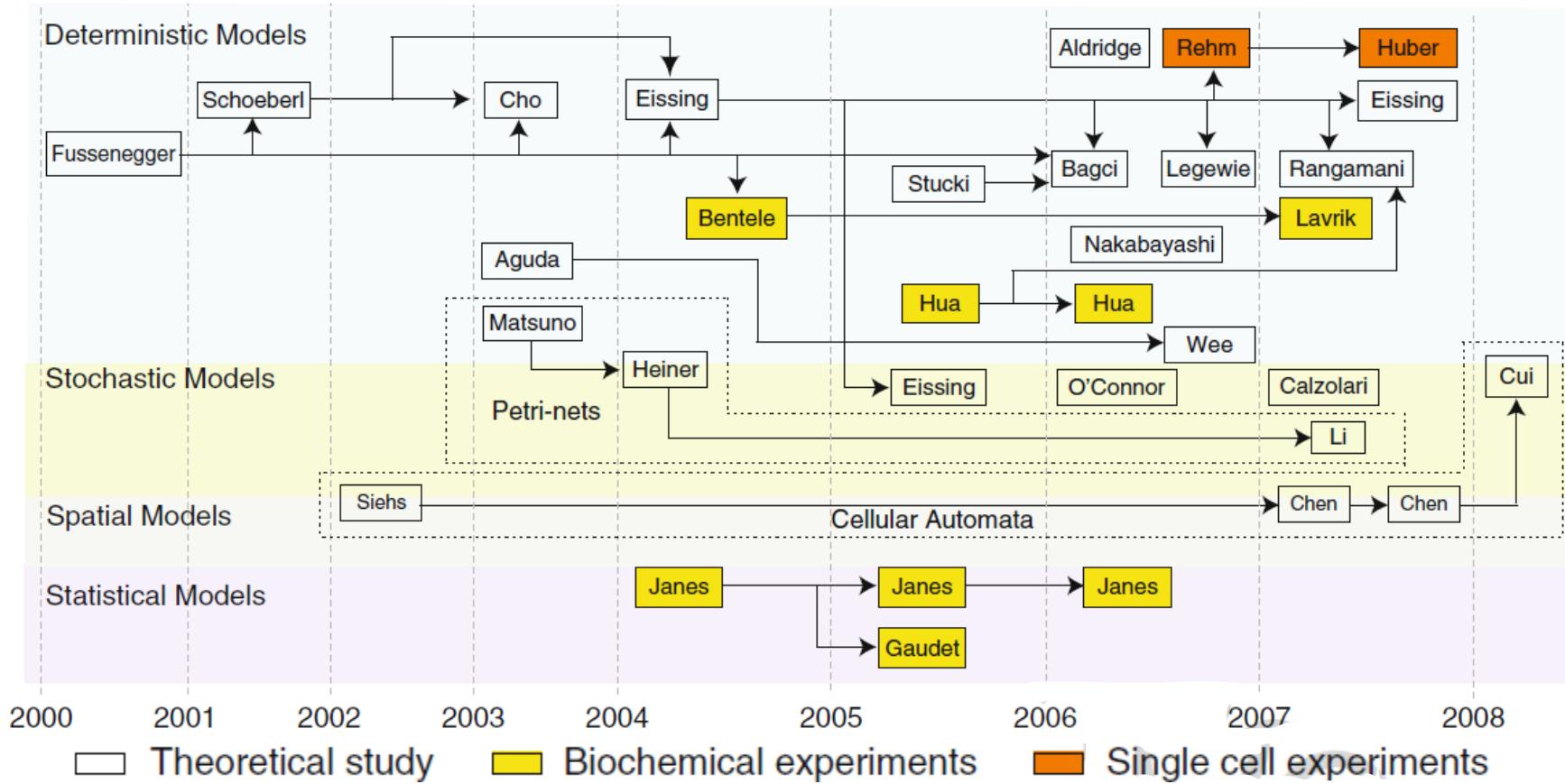
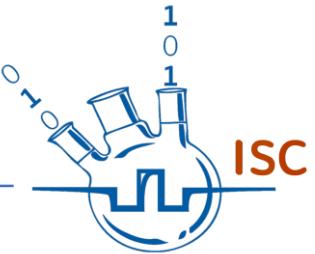



# Cell death





# Cell death modelling



(From Huber, Bullinger and Rehm, Systems Biology Approaches to the Study of Apoptosis 2009)

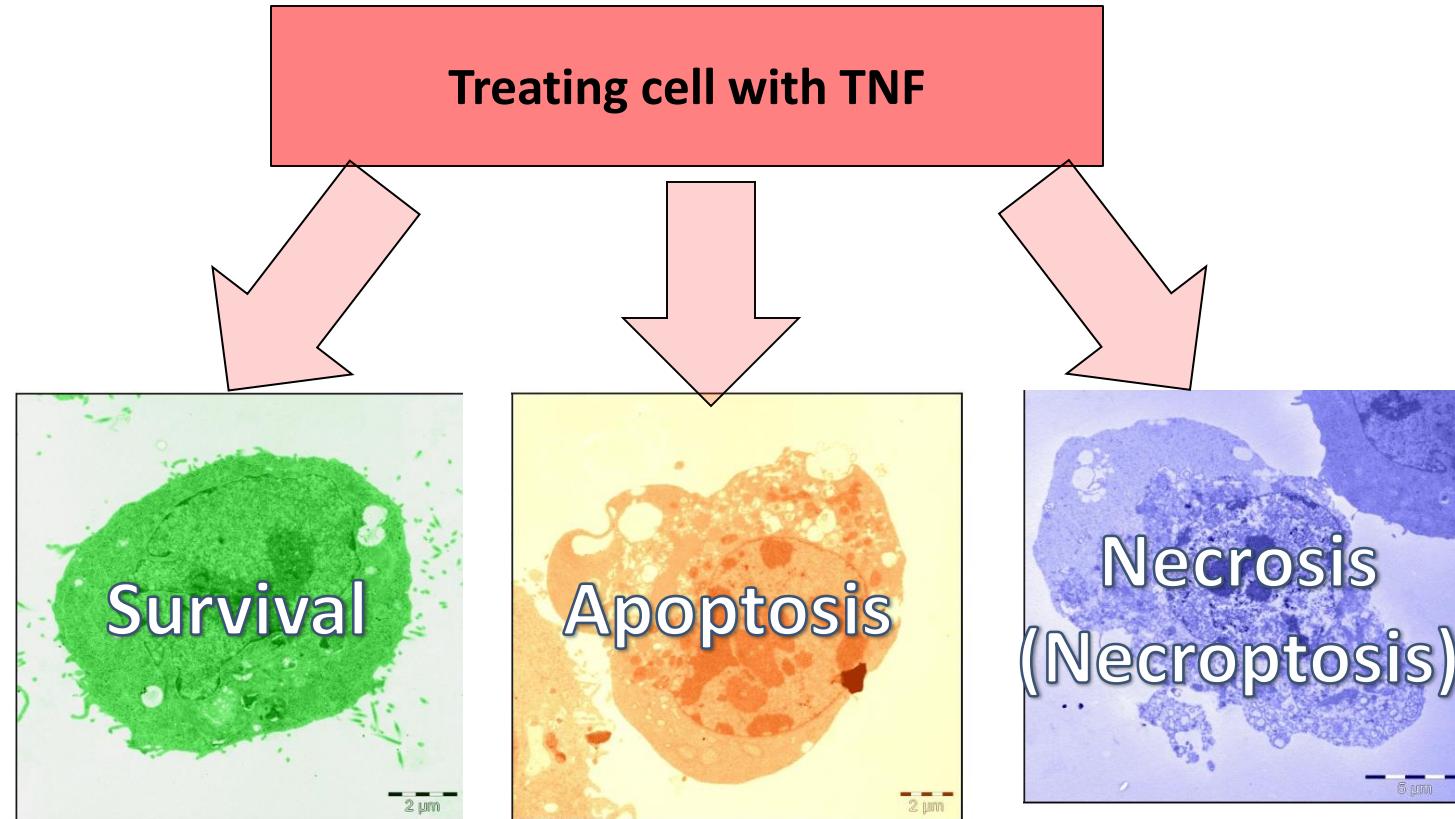
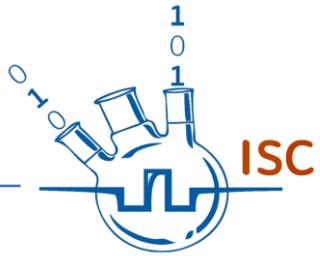
## Cell Death Modalities

(From Galuzzi et al, Cell Death and Diff, 2007)





# Apoptosis vs. Necrosis vs. Survival



OPEN ACCESS Freely available online

PLOS COMPUTATIONAL BIOLOGY

## Mathematical Modelling of Cell-Fate Decision in Response to Death Receptor Engagement

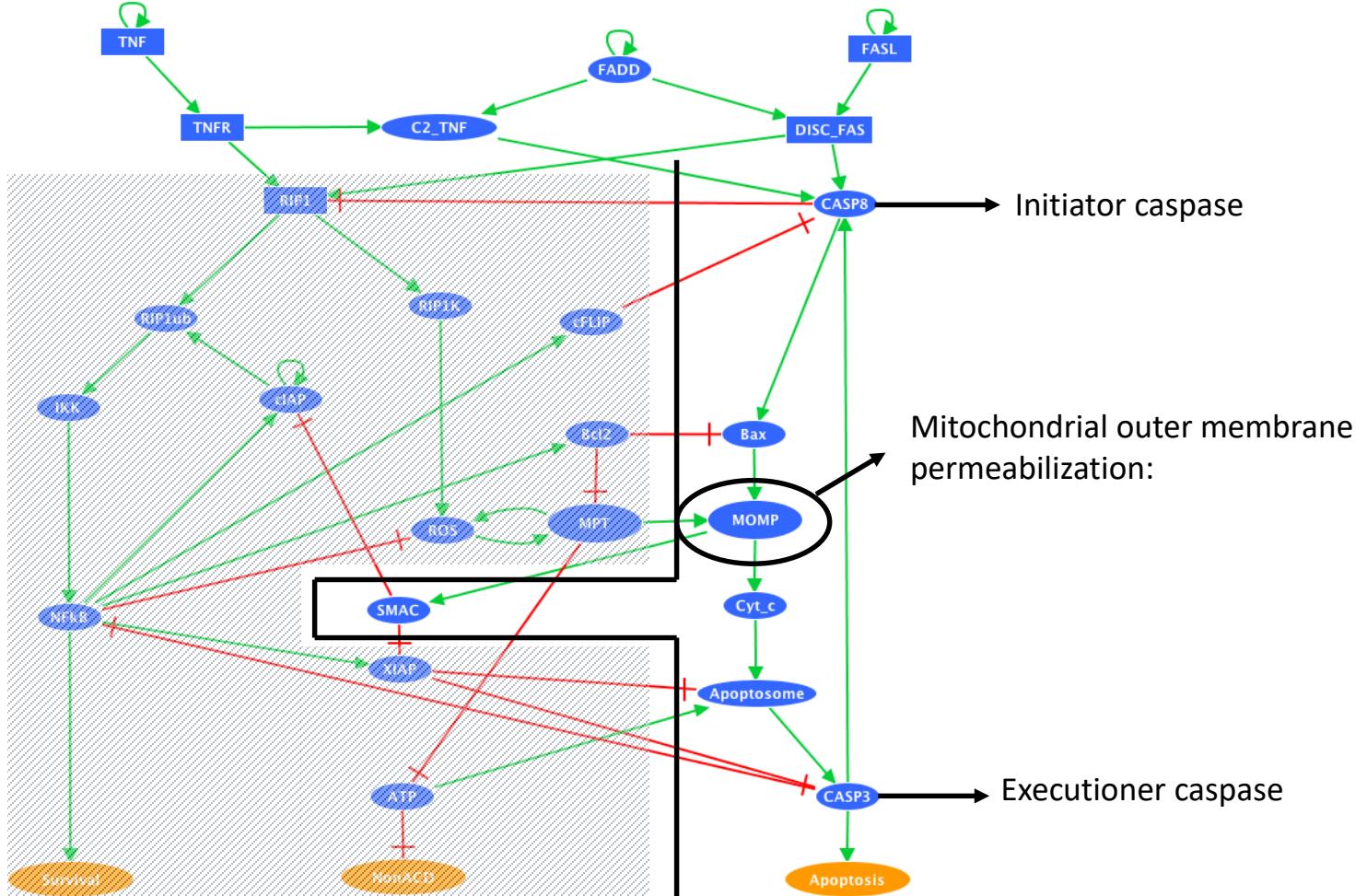
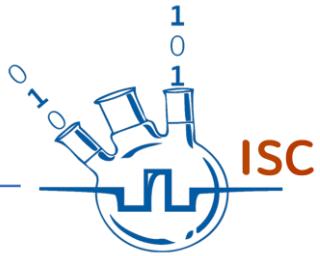
Laurence Calzone<sup>1,2,3\*</sup>, Laurent Tournier<sup>1,2,3</sup>, Simon Fourquet<sup>1,2,3</sup>, Denis Thieffry<sup>4,5</sup>, Boris Zhivotovsky<sup>6</sup>, Emmanuel Barillot<sup>1,2,3†</sup>, Andrei Zinov'yev<sup>1,2,3‡</sup>

<sup>1</sup> Institut Curie, Paris, France, <sup>2</sup> Ecole des Mines ParisTech, Paris, France, <sup>3</sup> INSERM U900, Paris, France, <sup>4</sup> TAGC – INSERM U928 & Université de la Méditerranée, Marseille, France, <sup>5</sup> CONTRAINTES Project, INRIA Paris-Rocquencourt, France, <sup>6</sup> Karolinska Institutet, Stockholm, Sweden

EMOre than a  
NIVERSITY

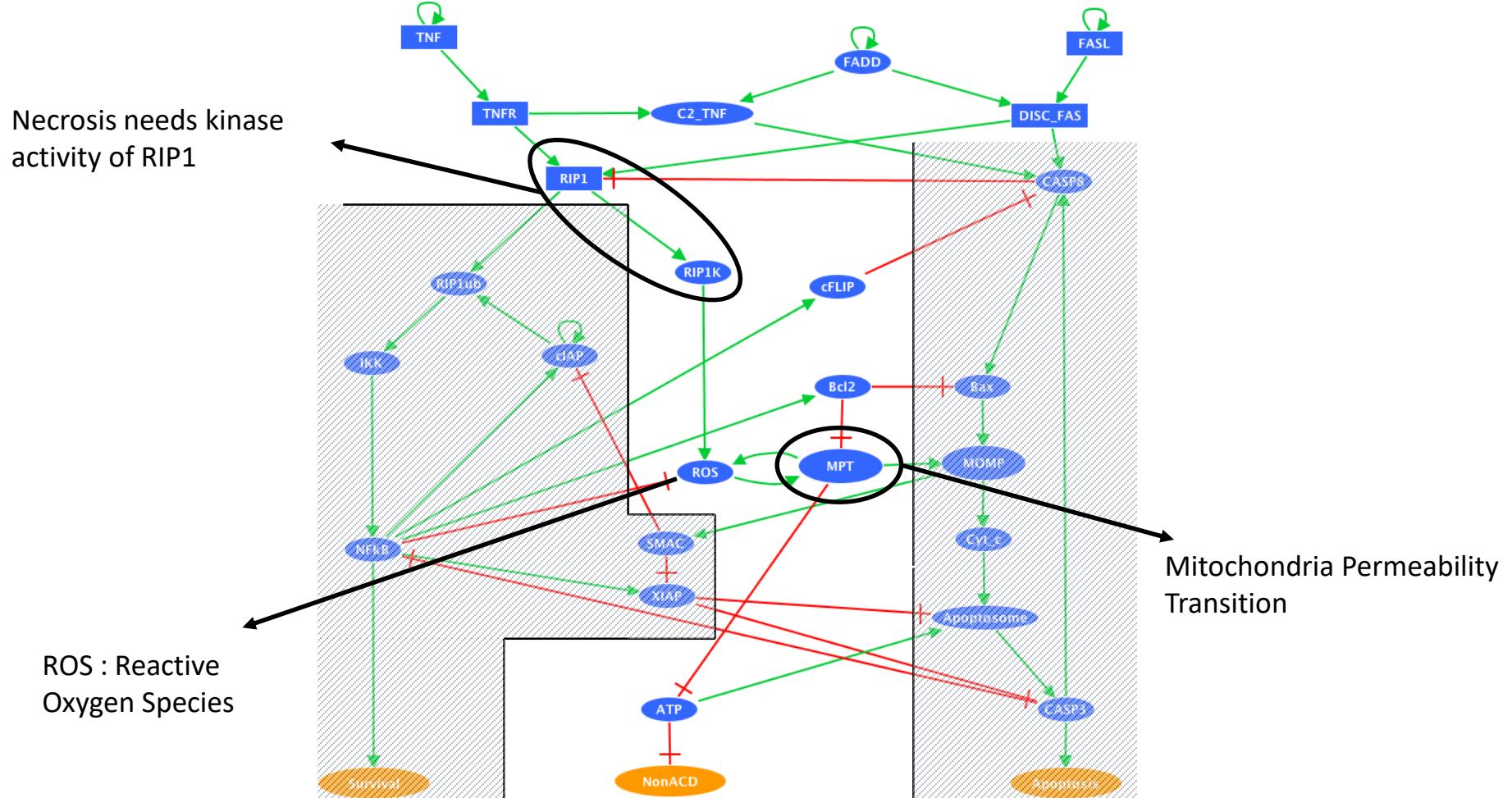


# Apoptosis



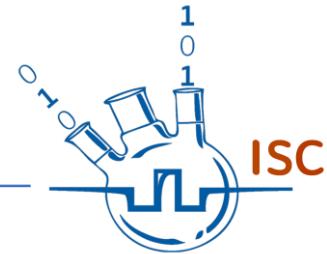


# Necrosis





# Boolean modeling



Назначить булеву функцію на узел

Example of CASP8

**CASP8 = 0** when

DISC-Fas=0 and DISC-TNF=0 and CASP3=0

(equivalent to no external signals from death receptors  
and no intracellular problems)

cFLIP=1

(equivalent to inhibition by the NFkB pathway)

**CASP8 = 1** when

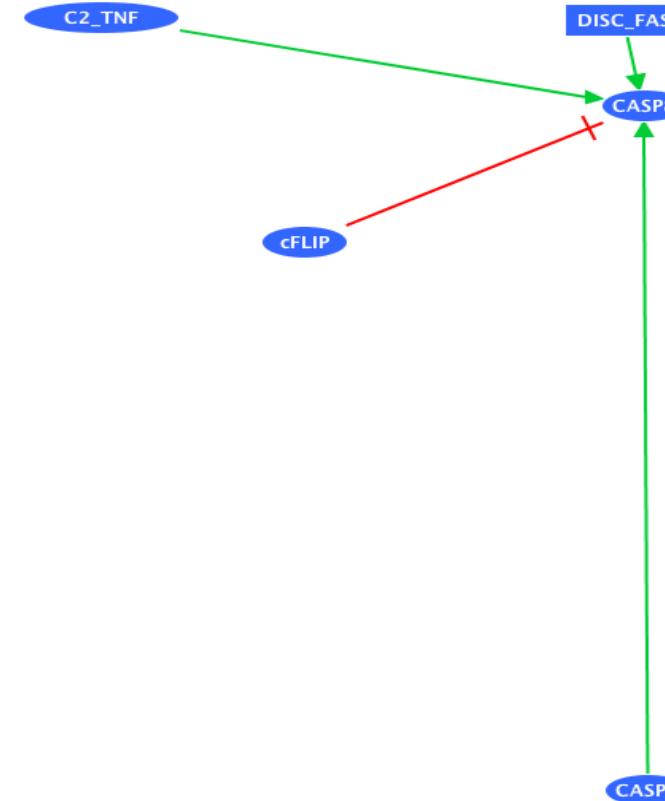
DISC-Fas=1 or/and DISC-TNF=1

(equivalent to signal from death receptors)

CASP3=1

(amplification signal, feedback activation)

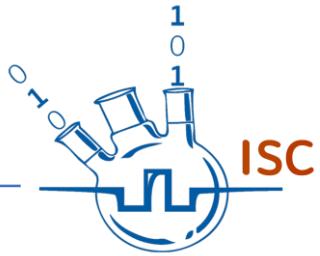
AND no cFLIP



One node = one species

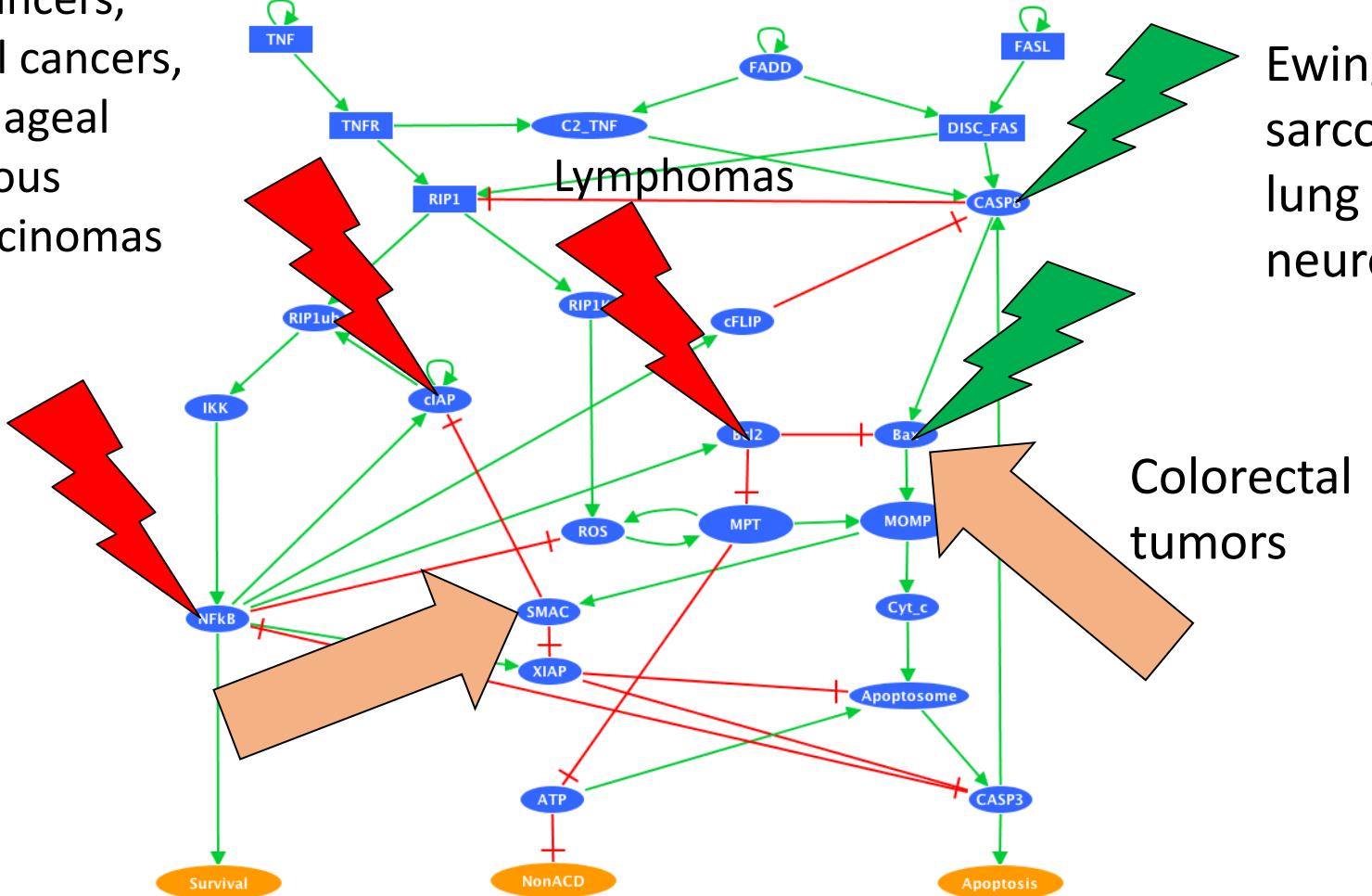


# Target molecules in various tumors



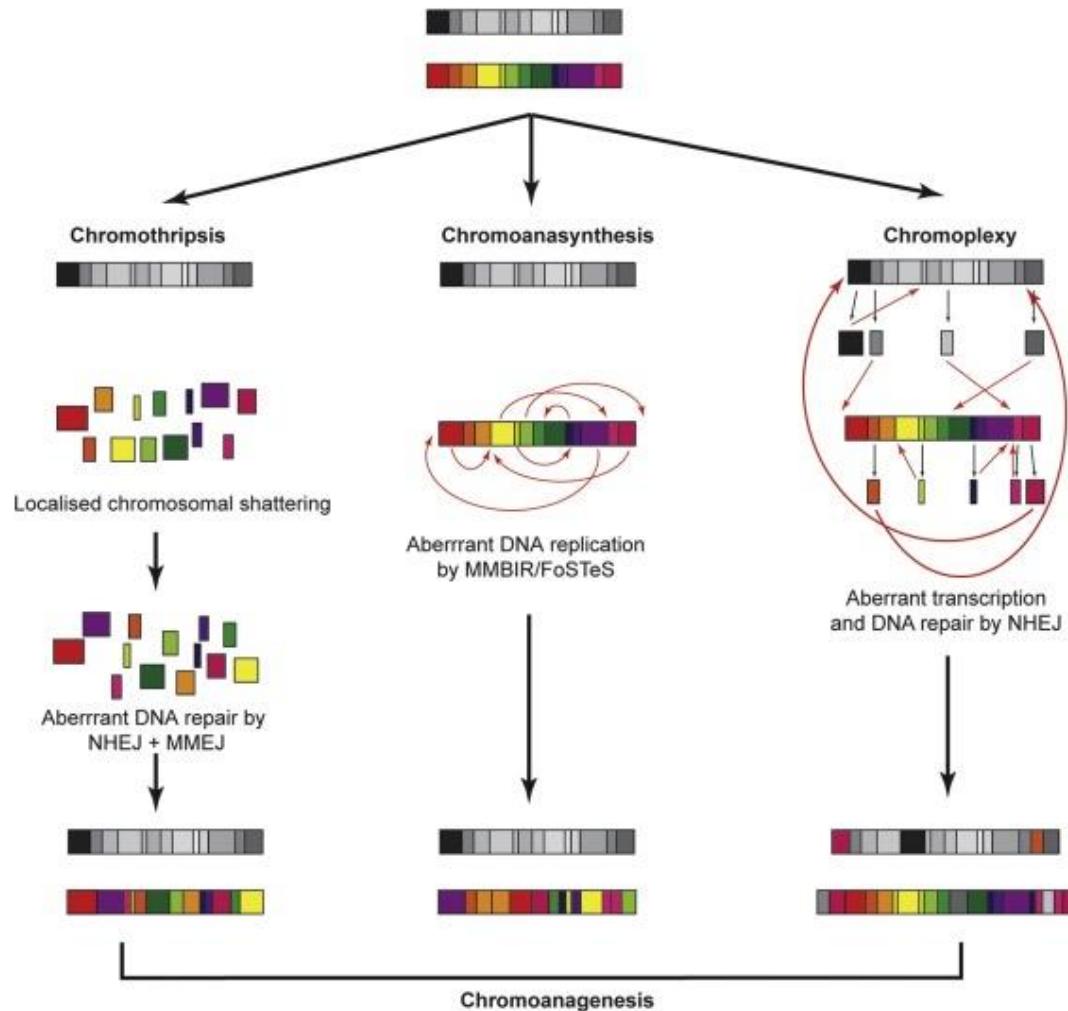
Lung cancers,  
cervical cancers,  
oesophageal  
squamous  
cell carcinomas

Lymphomas,  
breast cancer





# Genome chaos



## Stress pathway

Cytokine deprivation  
Intracellular damage

Oncogenes

BH3

Bcl-2

Bax

cyt C

Apaf-1

caspase-9

tBid

FADD

caspase-8

caspases 3, 6, 7

Apoptosis

## Death receptor pathway

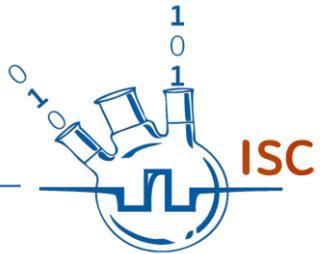
FasL, TNF $\alpha$ , TRAIL

Death receptors

ITSMORE than a  
UNIVERSITY



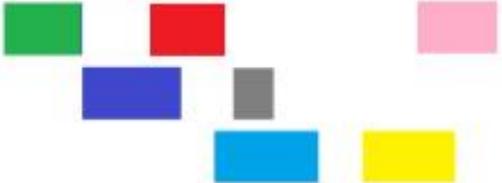
# Genome chaos



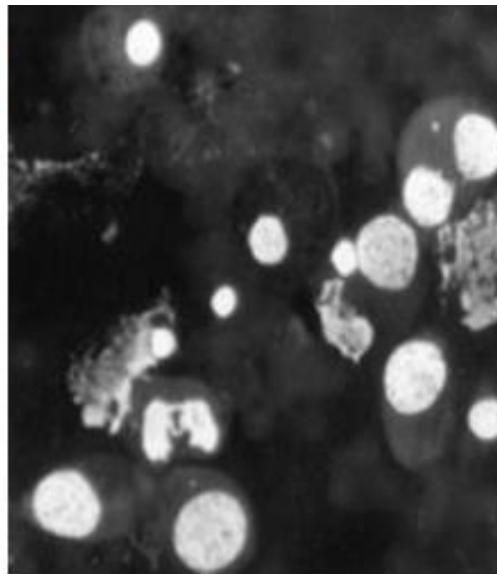
Normal chromosome



Chromothripsis

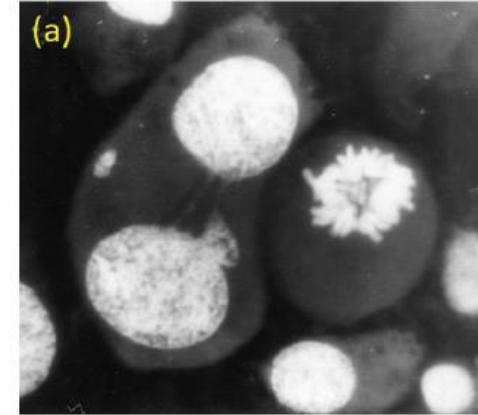


Altered chromosome



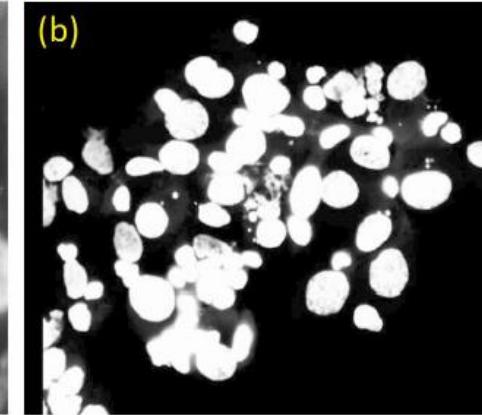
(a)

(c)



(b)

(d)



(d)



# Genome chaos algorithm



```
gene_sequence = list("ATGGCGCACGCTGGGAGAACAGGGTACGATAA  
# Calculate the number of nucleotides  
num_nucleotides = len(gene_sequence)  
  
print("Number of Nucleotides:", num_nucleotides)  
  
import random  
  
# Mutation rate  
mutation_rate = 1.0e-4  
  
# Number of cell divisions  
num_divisions = 60  
  
# Initialize the gene sequence (as a list of characters, for example)  
gene_sequence = list("ATGGCGCACGCTGGGAGAACAGGGTACGATAACCAGGAGATAGTGATGAAGTACA  
  
# Function to apply mutations  
def apply_mutation(gene_sequence, mutation_rate):  
    mutated_sequence = []  
    for base in gene_sequence:  
        if random.random() < mutation_rate:  
            # Mutate the base (for simplicity, just change to a random base)  
            mutated_base = random.choice("ATCG")  
            mutated_sequence.append(mutated_base)  
        else:  
            mutated_sequence.append(base)  
    return mutated_sequence  
  
# Simulate cell divisions and mutations  
for _ in range(num_divisions):  
    gene_sequence = apply_mutation(gene_sequence, mutation_rate)  
  
# Print the mutated gene sequence  
mutated_gene = ''.join(gene_sequence)  
print("Mutated Gene Sequence:", mutated_gene)
```

```
# Genetic code dictionary  
genetic_code = {  
    "TTT": "F", "TTC": "F", "TTA": "L", "TTG": "L",  
    "CTT": "L", "CTC": "L", "CTA": "L", "CTG": "L",  
    "ATT": "I", "ATC": "I", "ATA": "I", "ATG": "M",  
    "GTT": "V", "GTC": "V", "GTA": "V", "GTG": "V",  
    "TCT": "S", "TCC": "S", "TCA": "S", "TCG": "S",  
    "CCT": "P", "CCC": "P", "CCA": "P", "CCG": "P",  
    "ACT": "T", "ACC": "T", "ACA": "T", "ACG": "T",  
    "GCT": "A", "GCC": "A", "GCA": "A", "GCG": "A",  
    "TAT": "Y", "TAC": "Y", "TAA": "*", "TAG": "*",  
    "CAT": "H", "CAC": "H", "CAA": "Q", "CAG": "Q",  
    "AAT": "N", "AAC": "N", "AAA": "K", "AAG": "K",  
    "GAT": "D", "GAC": "D", "GAA": "E", "GAG": "E",  
    "TGT": "C", "TGC": "C", "TGA": "*", "TGG": "W",  
    "CGT": "R", "CGC": "R", "CGA": "R", "CGG": "R",  
    "AGT": "S", "AGC": "S", "AGA": "R", "AGG": "R",  
    "GGT": "G", "GGC": "G", "GGA": "G", "GGG": "G",  
}  
  
# Function to translate a DNA sequence to a protein sequence  
def translate_dna_to_protein(dna_sequence):  
    protein_sequence = []  
    for i in range(0, len(dna_sequence), 3):  
        codon = dna_sequence[i:i + 3]  
        amino_acid = genetic_code.get(codon, 'X') # 'X' for unknown codons  
        protein_sequence.append(amino_acid)  
    return ''.join(protein_sequence)  
  
# Translate the reference and mutated sequences  
translated_reference_sequence = translate_dna_to_protein(reference_sequence)  
translated_mutated_sequence = translate_dna_to_protein(mutated_gene)  
  
print("Translated Ref Gene Sequence:", translated_reference_sequence)  
print("Translated Mut Gene Sequence:", translated_mutated_sequence)
```

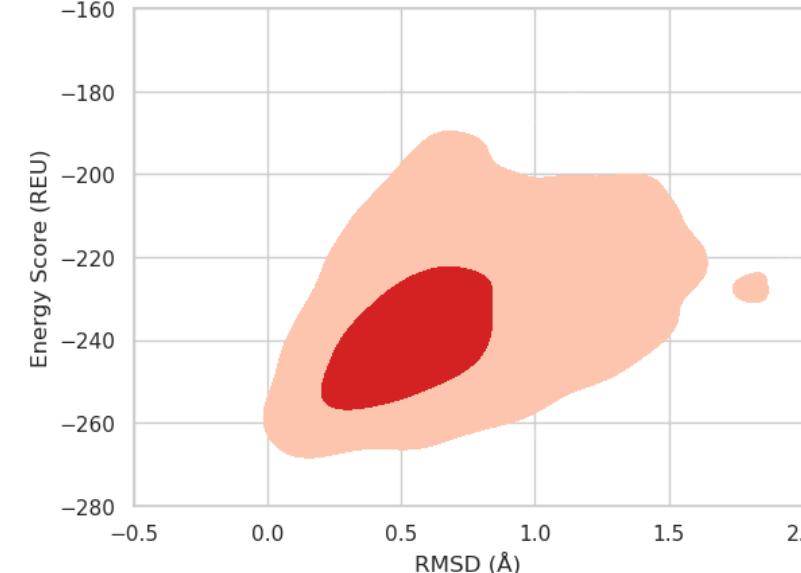
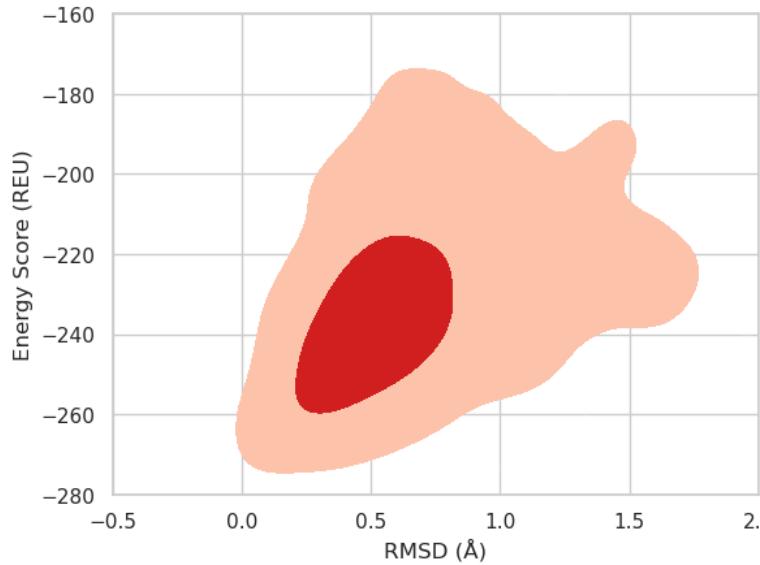
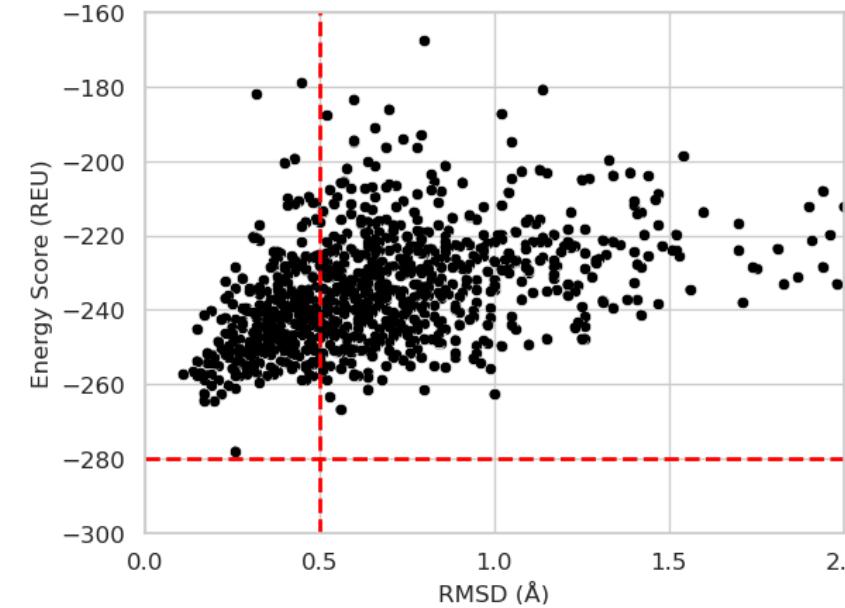
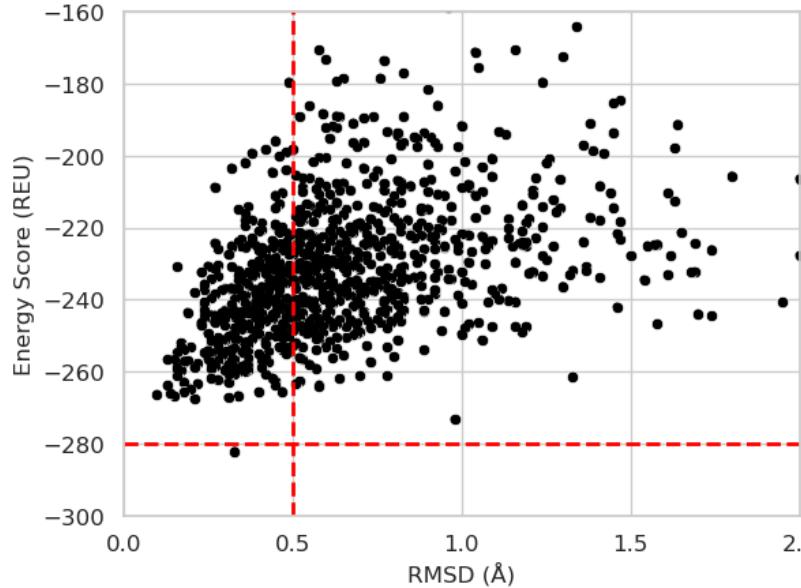
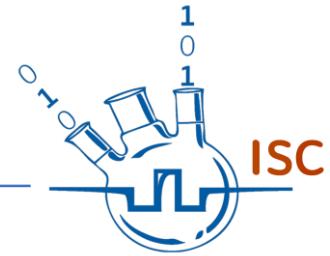
Protein Sequence 1: MAHAGRTGYDNREIVMKYIHYI  
LTLRQAGDDFSRRYRRDFAEMSQLHLTPFTARGRFATVVEI  
LFDFSWLSLKTLSSLALVGACITLGAYLGHK\*

Protein Sequence 2: MAHAWRTGYDNREIVMKYIHYI  
LTLRQAGDDFSRRYRRDFAEMSQLHLTPFTARGRFATVVEI  
LFDFSWLSLKTLSSLALVGACITLGAYLGHK\*

Sequence Identity: 99.58%

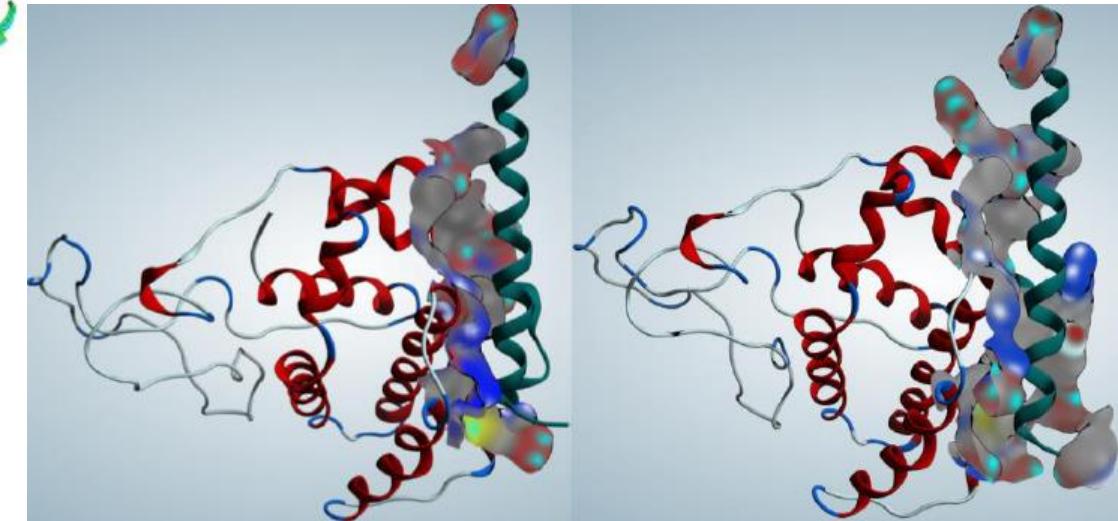
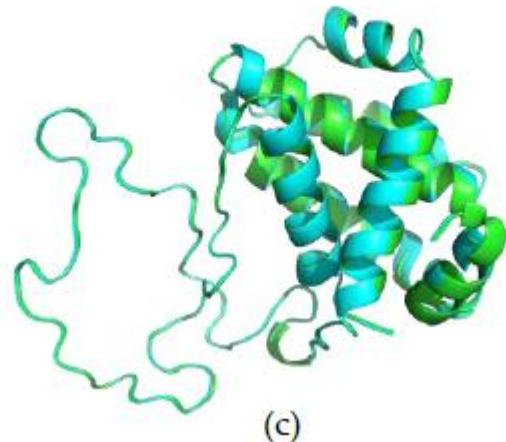
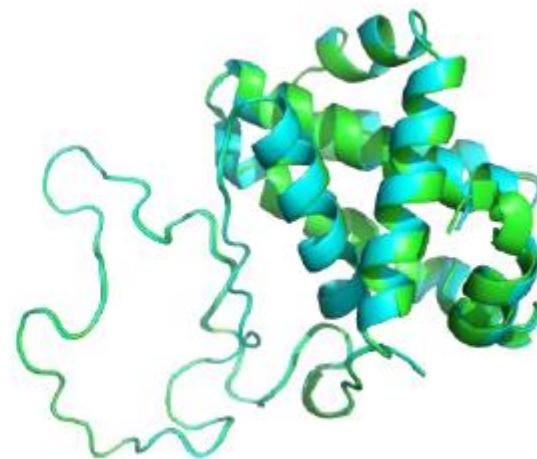
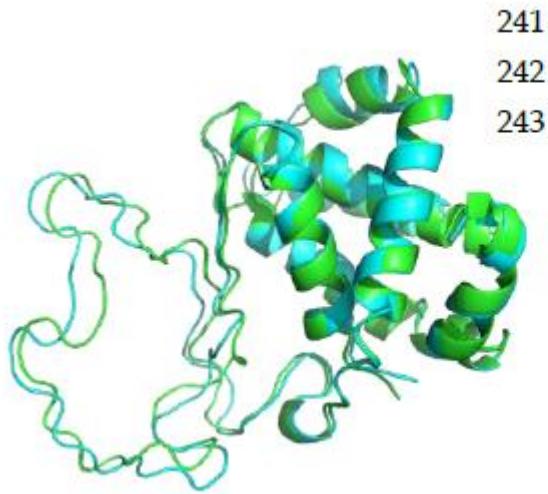
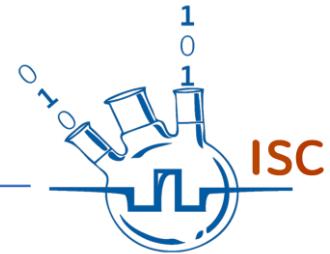


# Genome chaos





# Genome chaos (BCL-2/BH3 interaction)





# Genome chaos

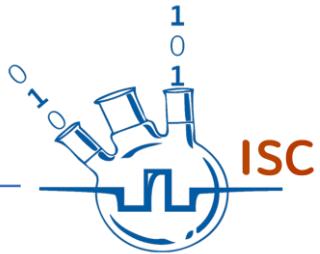
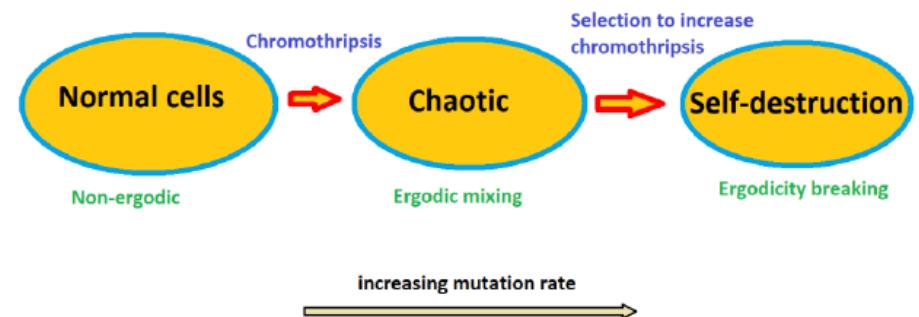
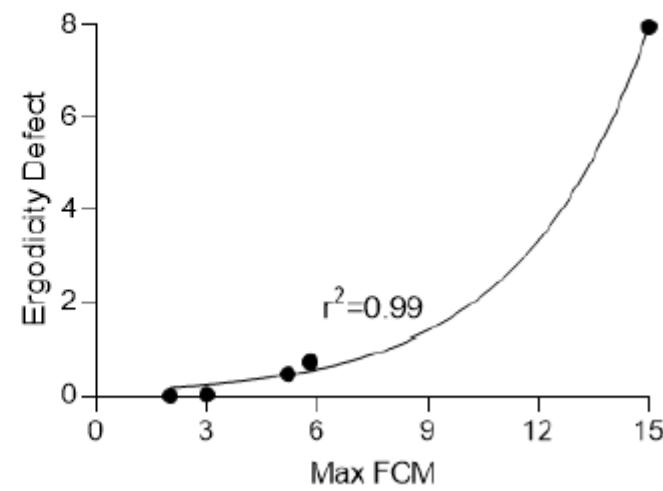
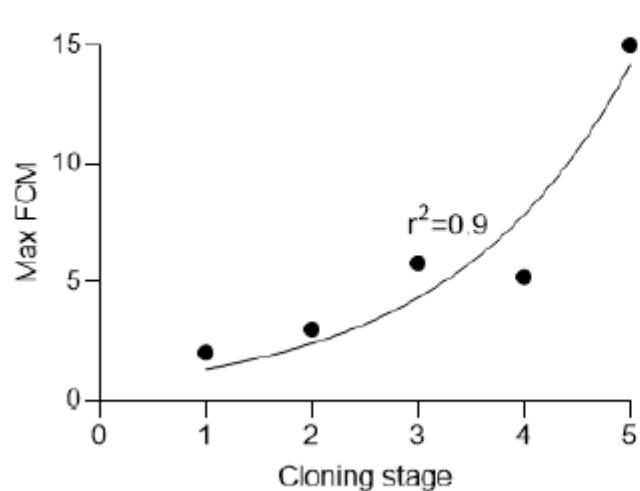


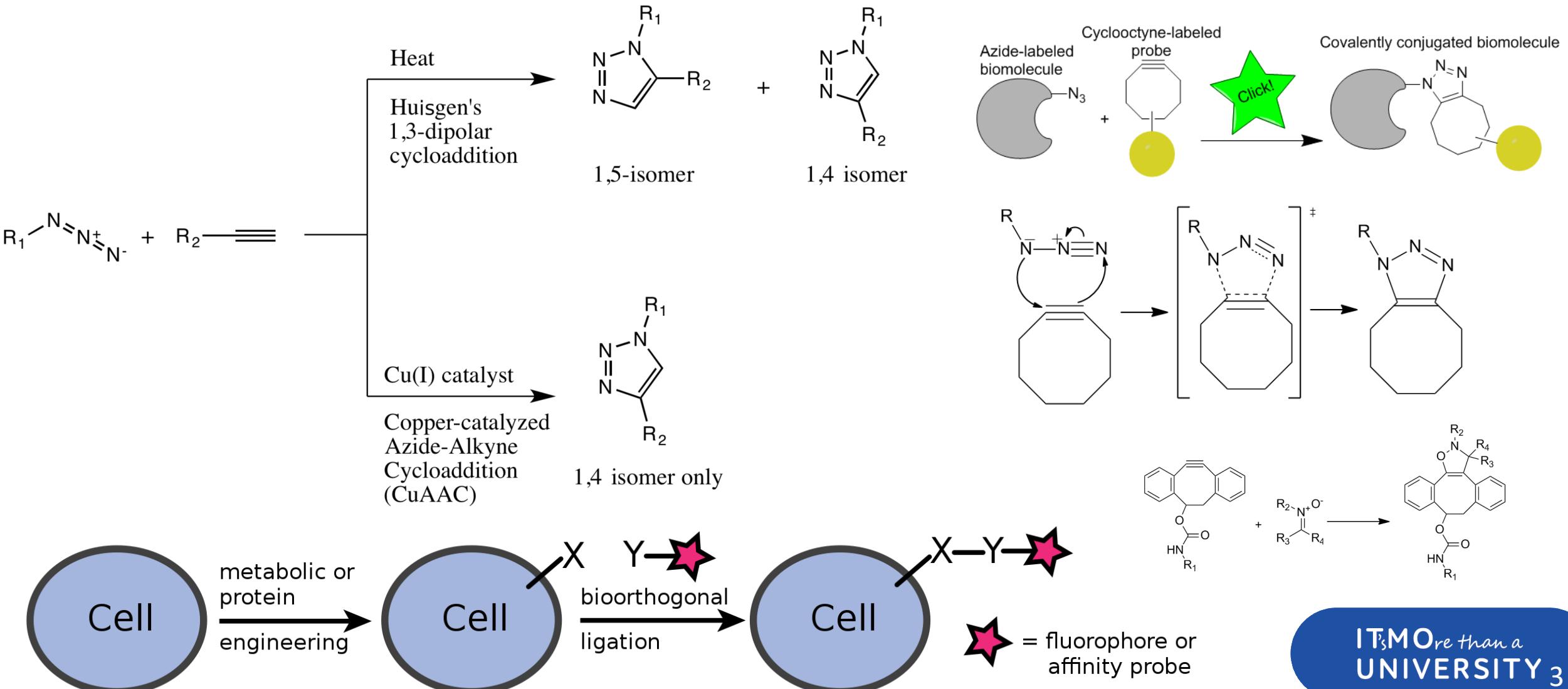
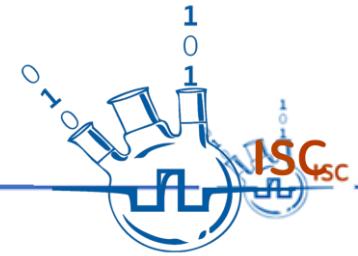
Table 1. Changes of observed minimum, average, and maximum FCM and calculated effective mutation rate and ergodicity defect at different stages of the selection for increasing FCM.

Stage	Clones	Min FCM	Average FCM	Max FCM	Effective mutation rate, $\times 10^{-4}$	Ergodicity defect
0	48	0.0	0.6	2.0	6.99	0
1	50	0.0	0.8	3.0	8.99	0.03
2	52	0.0	1.7	5.8	17.98	0.74
3	48	0.0	1.6	5.2	16.98	0.48
4	47	1.0	4.7	15.0	47.95	7.96



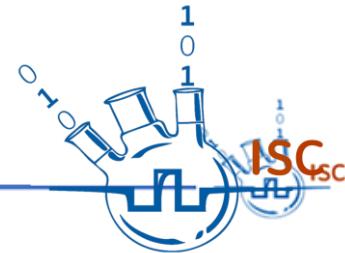


# Click reaction and orthogonal chemistry





# NuroClick software



Home / Browse / Science & Engineering / Chemistry / AutoClickChem



## AutoClickChem

Brought to you by: jdurrant

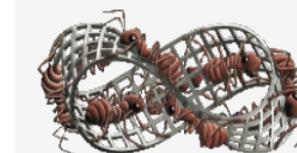
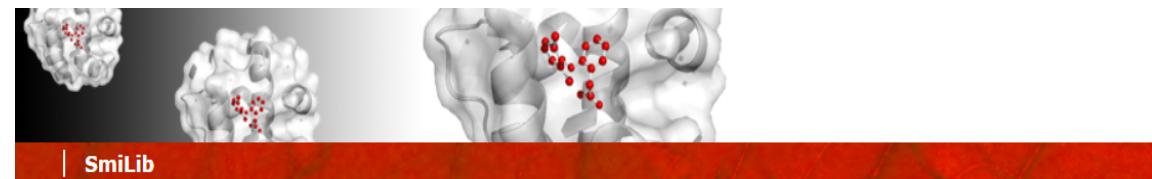
★★★★★ 1 Review

Downloads: 1 This Week

Last Update: 2014-06-03

[Download](#)  [Get Updates](#) [Share This](#)

Windows | Mac | Linux



## SmiLib v2.0



## SmiLib v2.0

SmiLib is a free, platform independent software tool for rapid combinatorial library enumeration in the flexible and portable SMILES notation. SmiLib enumerates combinatorial libraries at rates of approximately 9,000,000 molecules per minute on fast computers.

If you wish to publish results obtained with SmiLib, please cite:

A. Schüller, V. Hähnke, G. Schneider; SmiLib v2.0: A Java-Based Tool for Rapid Combinatorial Library Enumeration, *QSAR & Combinatorial Science* **2007**, 3, 407-410.

Copyright © 2006-2008, Johann Wolfgang Goethe-Universität, Frankfurt am Main, Germany. All rights reserved. Use is subject to [license terms](#).

**Current version: 2.0 rc4.** A convenience method to run SmiLib as a Java library from within your own Java projects was added. Please see the [change log](#) for a list of program modifications.

- [Run SmiLib](#)
- [User Manual](#)
- [API Documentation](#)
- [Download](#)
- [Contact](#)

### Disclaimer

© 2013 Andreas Schüller |  
Webdesign: Michael Meissner  
[Webmaster](#)

### Run SmiLib

SmiLib is available as a Java Web Start graphical user interface program.

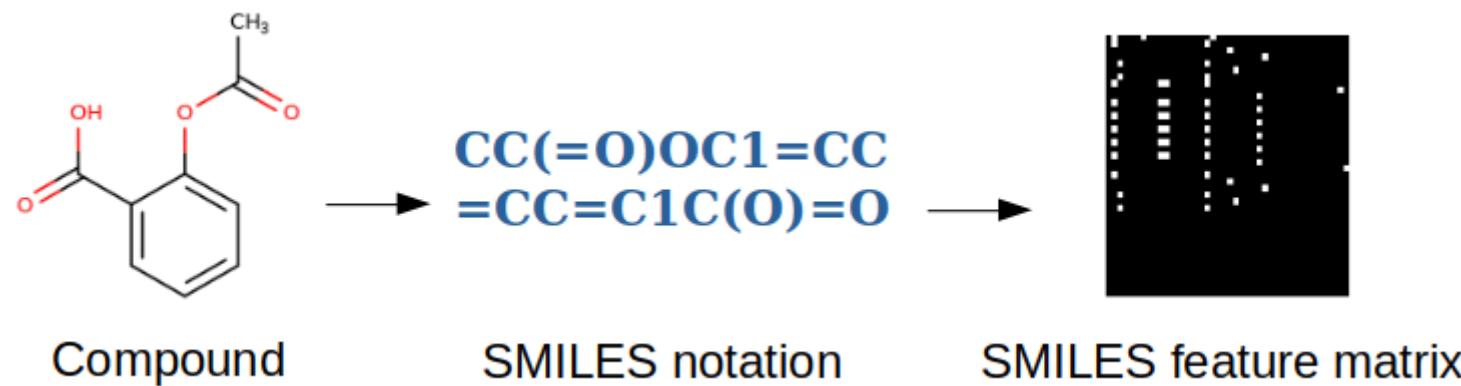
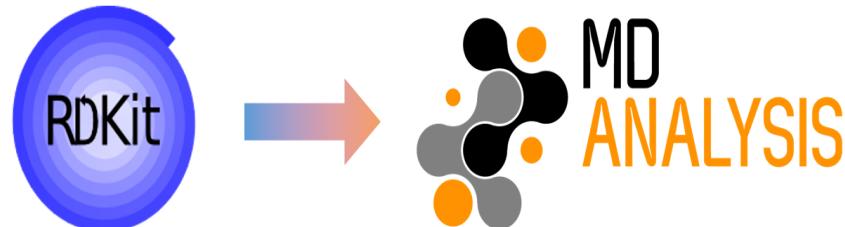
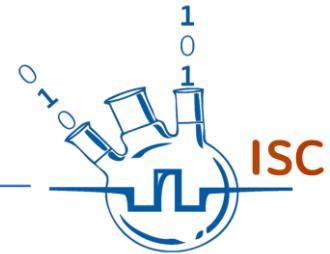
[Click here to start SmiLib](#)

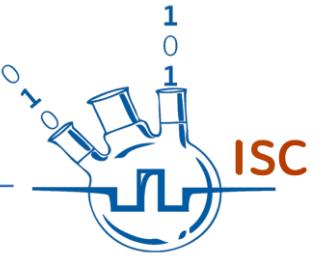
Java Runtime Environment (JRE) is needed to run SmiLib.

IT'sMOre than a  
UNIVERSITY 3



# NuroClick software

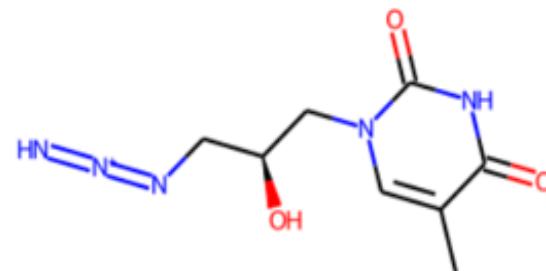
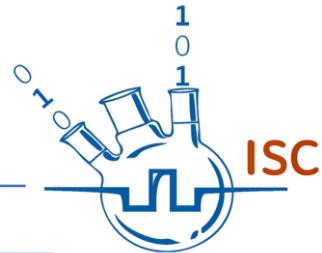




# How it works?

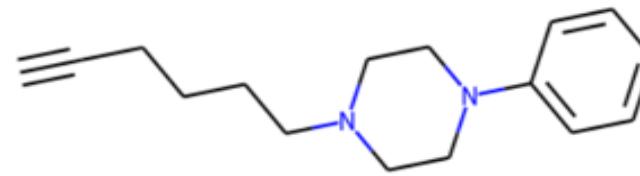


# Step 1: reagents upload



C1=CC=C(C=C1)C(=O)N=[N+]#[N-]  
C1=CC=C2C(=C1)C(=CN2)C(=O)N=[N+]#[N-]  
CC(=O)N[C@H]([C@H]([C@H]([C@H]([C@H](O)N=C([N+]#[N-])C(=O)OC  
CCOC(=O)C(CCCN=[N+]#[N-])CC=C(C)C

**Upload azides**

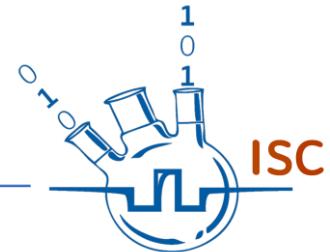


COC(=O)C1=CC=C(C=C1)C#CC2CCNCC2  
C#CC(C(=O)NCCOCCOCC(=O)O)N  
CC(C)(C)OC(=O)CCOCCC#C  
C#CCOC1=CC=C(C=C1)N2C(=O)NNC2=O  
C#CCOCCOCCOCCOCCSSCCOCCOCCOCC#C

**Upload alkynes**



## Step 2: click reaction initiation



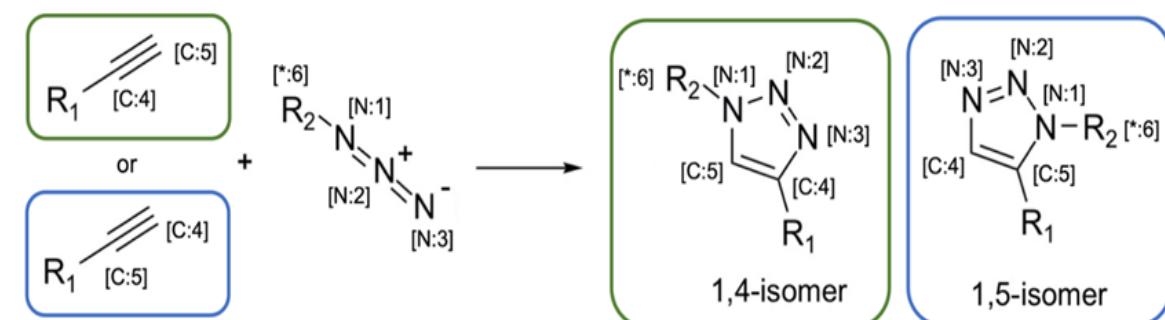
Starting library generation...  
Irrelevant reagents: [N-]=[N+]=NC(=O)c1ccccc1 and C#CCOCCOCOCOCSSCCOCOCOCOCOC#C  
Irrelevant reagents: [N-]=[N+]=NC(=O)c1[nH]c2cccc12 and C#CCOCCOCOCOCOCSSCCOCOCOCOCOC#C  
Irrelevant reagents: COC(=O)[C@H](N=[N+]=[N-])[C@H](NC(C)=O)OC and C#CCOCCOCOCOCSSCCOCOCOCOCOC#C  
Irrelevant reagents: CCOC(=O)C(CC=C(C)C)CCN=[N+]=[N-] and C#CCOCCOCOCOCSSCCOCOCOCOCOC#C

=====

Finished library generation!  
Time: 00:00:02.  
32 compounds were generated from 5 alkynes and 4 azides.  
12 1,4-isomers and 12 1,5-isomers.  
Warning: 8 molecules of 32 were generated from internal alkynes and could not be assigned 1,4/1,5 isometry.  
Failed to construct 4 molecules.

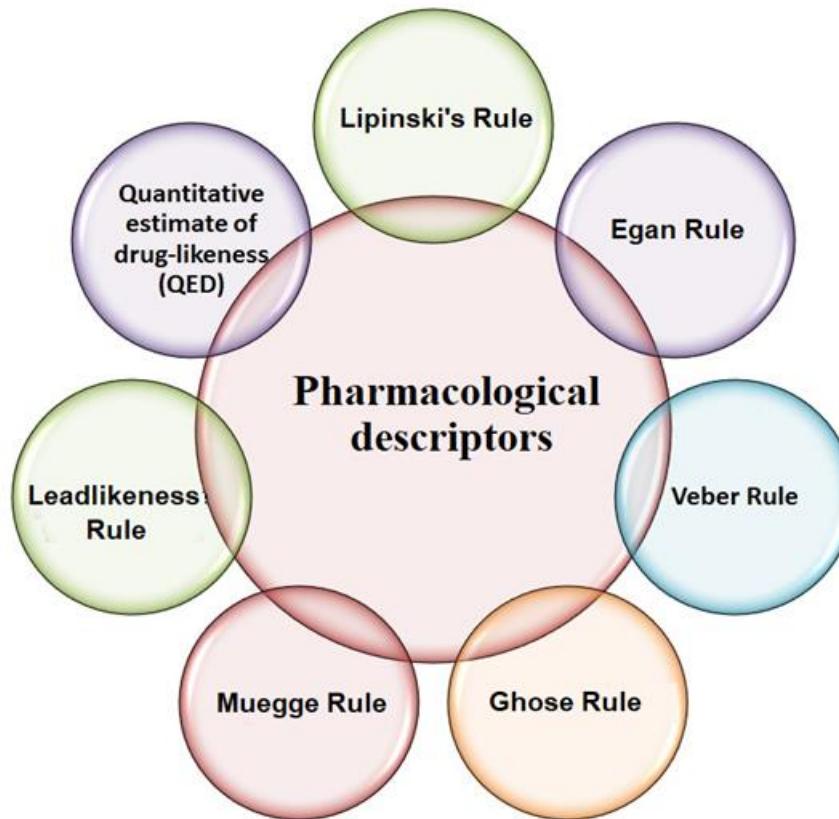
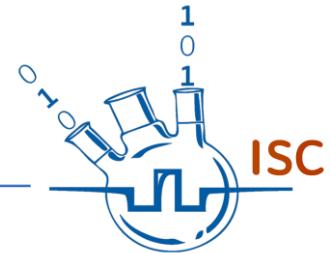
[Get reaction products](#)

[Back](#) [Next](#)





## Step 3: PK/PD descriptor calculation



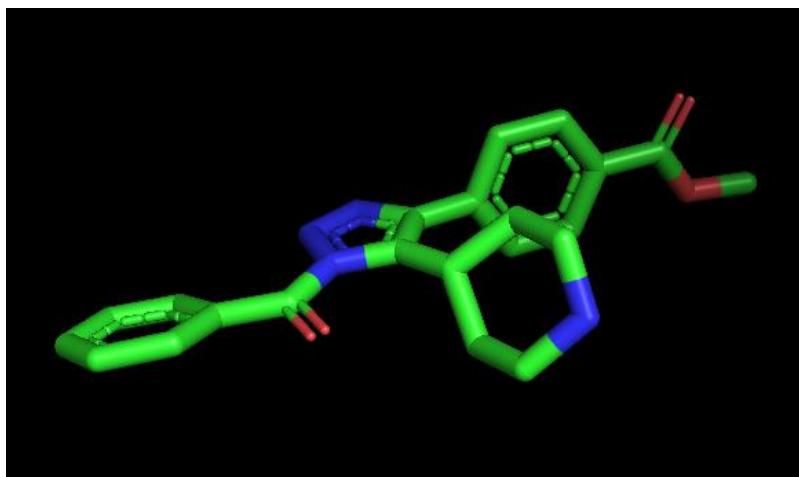
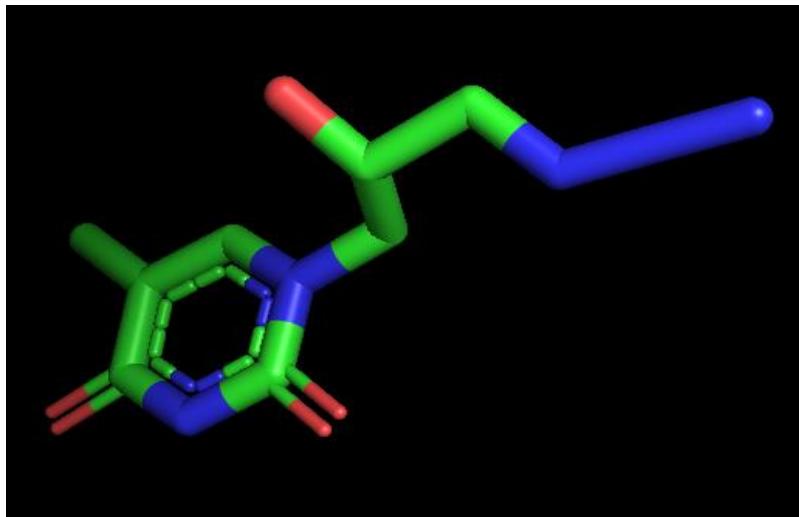
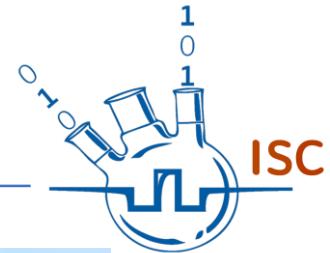
Select physical properties:

- Molecular weight (g/mol) from  to
- Heavy atoms from  to
- Aromatic heavy atoms from  to
- Fraction Csp3 from  to
- Rotatable bonds from  to
- H-bond acceptors from  to
- H-bond donors from  to
- Molar refractivity from  to
- TPSA from  to

Back Next



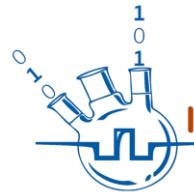
## Step 4: save the results



Convert products to other formats:

- cml - Chemical Markup Language
- d2s - Text/Office Document
- dna - DNA Sequence
- fasta - FASTA (Automatic recognition)**
- fasta:dna - FASTA (DNA sequence)
- fasta:peptide - FASTA (peptide sequence)
- fasta:rna - FASTA (RNA sequence)
- gout - Gaussian Output Format
- inchi - InChI
- mol - MDL Molfile
- mol2 - Tripos Mol2

Back      Next



Thank you for your attention

