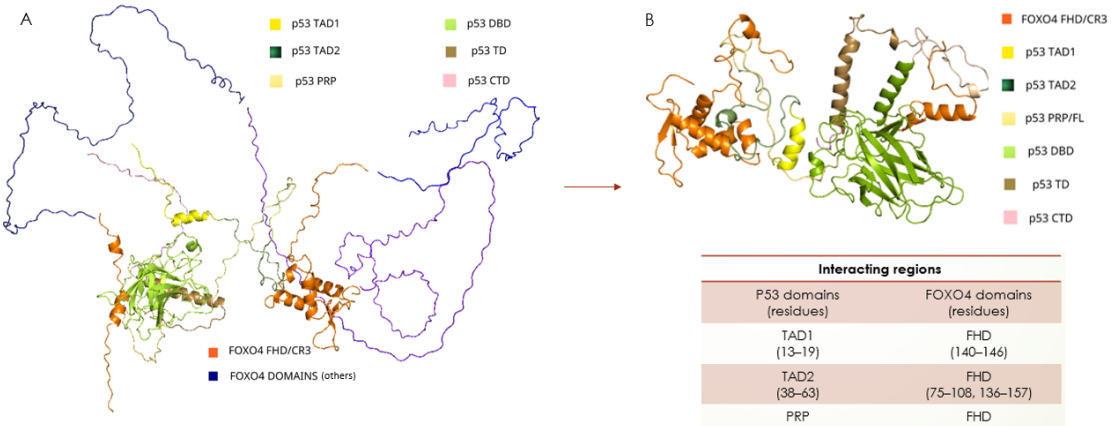
CONTENTS

1 STRUCTURAL BIOINFORMATICS PROJECTS

A. MSc Thesis: Study of structural interaction and molecular dynamics of FOXO transcription factors with molecules of antioncogenic gene p53, 2023 Athens (Manuscript in preparation)

Abstract: The transcription factors of FOXO family and the tumor suppressing proteins of p53 are involved in the regulation of several signaling pathways related to DNA repair, cell cycle arrest and active action of apoptosis. There are indications that FOXO4 and p53 bind each other within the nucleus of senescent cells promoting the transcription of p21 (cell cycle inhibitor). Recently, it has been shown that inhibition of FOXO4-p53 interaction promotes apoptosis in senescent cells. Inevitably, this interaction has been characterized as biological target in terms of its anti-aging (senolytic) effect. The exact structure of this complex has not been resolved experimentally yet, owing to the highly dynamic regions of these two proteins. However, knowledge of the interaction interface of this complex is of paramount importance for the development of small molecules that could inhibit formation of the complex.

The scope of this investigation is to provide rational models of the p53/FOXO4 complex based on experimental evidence. First, we collected biochemical, biophysical and structural information, including the available X-ray, NMR and Cryo-EM structures of isolated FOXO4 and p53 domains, as well as prediction models of their structures (AlphaFold, RoseTTAFold). These data were analyzed to select appropriate structural templates for the modeling of putative p53/FOXO4 complexes with different orientations and initial conformations. HADDOCK was used in a unique way for molecular docking. Based on the calculated free energy of binding, we selected the most favored models that satisfied experimental constraints. These models were evaluated by means of stability in solvent using classical molecular dynamics simulations (MD) with AMBER. Conformational and protein-protein interactions analysis (using PyMOL) was conducted. The favored models of p53/FOXO4 complex were also investigated in the presence of double-stranded DNA fragments that correspond to response elements of p21 promoter, with the aim to identify the most appropriate p53/FOXO4 models. As a result, we propose a rational p53/FOXO4 complex that is compatible with the pre-initiation complex of p21 gene. Based on that complex and other published information, a hypothesis was made about the role of the complex on p21 transcription.



Interacting regions	
P53 domains (residues)	FOXO4 domains (residues)
TAD1 (13–19)	FHD (140–146)
TAD2 (38–63)	FHD (75–108, 136–157)
PRP (73–82)	FHD (75–108)
DBD (113–122 , 126–129, 282–295)	CR3 (469–505)
FL (307–325)	CR3 (469–505)

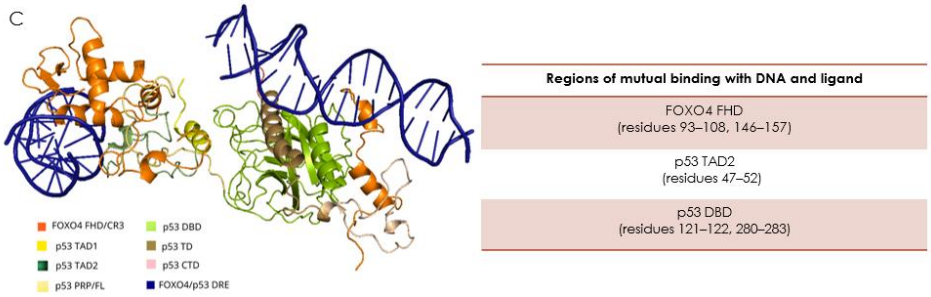


Figure. Investigation of conformational space and intermolecular interactions of the p53/FOXO4 complex. **A)** Docking pose with the lowest docking score satisfying all the experimentally identified residue-residue interactions. **B)** Most favored equilibrium structure of complex's interacting domains and key interacting regions, after 5 independent MD runs of 250ns at 300K. **C)** Docking pose of the complex in the presence of double-stranded DNA fragments (p21 promoter) satisfying experimental constraints and highlighting regions of mutual binding with DNA and ligand.

B. Pharmacophore virtual screening on the estrogen receptor using Pharmit, 2021 Patra

- Visualization of crustal structure of estrogen receptor in complex with afimoxifene (PDB ID: 3ert) and protein-drug interaction analysis using PyMOL.
- Loading of 3ert in Pharmit web page and visualization of pharmacophore features.
- Definition of exclusive three-dimensional space of 3ert.
- Virtual screening of Molport database based on the selected space with defined pharmacophore features.
- Selection of best hit compounds (with lower RMSD) and protein-compound interaction analysis in PyMOL. Fitting of chemical groups in specific pharmacophoric features and understanding of the relevant chemical space that needed to be filled.

2 BIOINFORMATICS PROJECTS

A. Effect of vitamin C on melanoma cells and extraction of biomarkers, 2021 Patra

Bioinformatic analysis of Next Generation RNA Sequencing data from GEO database. The goal of this study is to determine the differentiation in gene expression between 3 "Control" and 3 "Treatment" (with vitamin C) samples of A2058 cell line and the extraction of biomarkers.

a) Preprocessing and analysis of FASTQ files using Galaxy

- Loading FATSQ FTP files by the database ENA of EBI to Galaxy software and building list of dataset pairs (paired-end sequencing by Illumina HiSeq 2000). The sequence length of each read is 99 bases.
- Quality Control of dataset collections and use of "MultiQC" tool. The "Per base sequence quality" diagram implied that sequencing of the first 13 bases and the last 15 bases of each read has lower quality than the average. In addition, the existence of Illumina Universal Adapter is confirmed at the positions 60-85 in a small percentage of reads.

- Use of “Trimmomatic” tool to process the FASTQ files per collection. We removed the Illumina Universal Adapter and the bases with lower quality (HEADCROP & CROP). AVGQUAL helped us reject whole reads with lower quality than the average.
- Use of alignment program HISAT2 to create BAM files out of the processed FASTQ files. We used hg38 as reference genome and the output of UCSC Main Table as reference sequence of known human gene transcripts for being aligned with the processed reads.
- Use of “featureCounts” tool to quantify the expression of genes from the BAM file of each collection. The output of UCSC Main Table was used as gene annotation file.
- Use of “DESeq2” tool to determine the differentially expressed genes between the count tables of “Treatment” and “Control” collection produced by “featureCounts”.
- Use of “Filter data” tool to detect the statistically significant differentially expressed genes. We used P-value < 00.2 as threshold. The final tabular file consisted of 387 transcripts.
- Familiarity with IGV (Integrative Genomics Viewer).

b) Bioinformatic analysis and interpretation of biomarkers

- Enrichment analysis using David web tool. ID transcript of Ensemble was used as identifier and the differentially expressed transcripts as background. 337 transcripts were identified by David.
- With GO terms, the majority of differentially expressed genes (14) referred to immune response, cell adhesion and proliferation. 111 genes referred to integral components of membranes and 12 genes are involved in GTP binding (indication of cell proliferation).
- With KEGG terms, 15 genes are associated with pathways in cancer and 13 genes with focal adhesion, TNF and PI3K-Akt signaling pathway.
- Creation of protein-protein interaction network using STRING-DB. As import, we used the gene IDs by David. The threshold of confidence was set at 0.700.

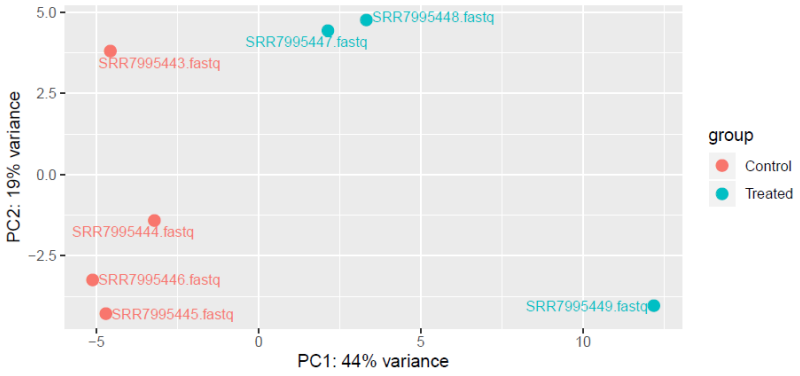


Figure. PCA plot of FASTQ files. The control and treated samples cluster separately. That suggests that vitamin C treatment had a significant effect on gene expression.

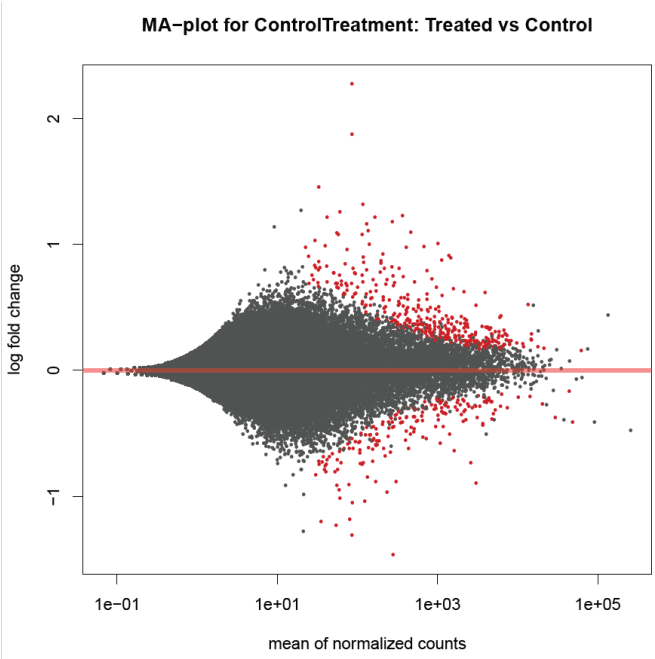


Figure. MA plot for expressed genes in treated and control samples. Several genes are up- and downregulated in the treated samples.

B. Study of public databases of transcriptomic profile generated by microarrays and RNA sequencing as well as study of normalization algorithms of omics data, 2021 Patra

Thorough study of Gene Expression Omnibus (GEO), ArrayExpress (AE) and Gene Expression Archive (GEA), in terms of their history, content, magnitude, curation, search and upload. Familiarity with MIAME (Minimum Information About a Microarray Experiment) and MINSEQE (Minimum Information about a high-throughput SEQuencing Experiment) format and with databases of raw RNA-seq data (SRA and ENA).

Underlining the significance of normalization techniques and study of relevant algorithms applied in untargeted proteomics with experimental bias. Some examples are linear and local (LoessF) regression normalization, variance stabilization normalization (Vsn), quantile and EigenMS normalization. Most of them are mostly applied via R packages.

C. Study of CFTR (Cystic Fibrosis Transmembrane Conductance Regulator) and ACE2 (Angiotensin-converting enzyme 2*)* gene and phylogenetic analysis, 2020 Patra

- Use of databases such as NCBI (PUBMED, GENE, ENTREZ), OMIM, EXPASY (PROT-PARAM, SMART) and SOFTBERRY (PSSFINDER) to collect information about the gene (cDNA, chromosome, size, exons-introns) and its protein (sequence, functional domains, structure, biological function, relevant diseases, isoelectric point, molecular weight, cellular location).
- Use of UCSC and Ensembl genome Browsers (tissue expression profile, genetic variants, SNPs etc.).
- Use of EXPASY (NETNGLYC, NETPHOS) to find the sites of post translational modifications (phosphorylation and N-glycosylation sites).
- Impact of SNPs on secondary structure.
- Phylogenetic analysis using MEGA X.

3 RELATIONAL DATABASES PROJECTS

A. Design, implementation of Relational Database associating drugs with diseases using SQL and Python, 2021 Patra (*Part of code at Appendix A)

Scope of this project is the design and implementation of a relational database, in which active substances of drugs, diseases and their potential association through biological information can be saved, as well as the respective information retrieval. The biological information includes relevant genes to the disease, translated proteins by the mRNA of the genes and the association of these proteins with the active substances of drugs.
Created entities along with their primary keys:

a) Disease: Cui_UMLS: a code name according to UMLS (UnifiedMedicalLanguageSystem®)
b) Gene_NDP: Approved_symbol: gene symbol
c) Proteins: Uniprot_id
d) Drugs: NAME

Relationship tables with their primary and foreign keys:

a) GeneDisease
- Association_id (AUTO_INCREMENT) (Primary Key)
- CUI_UMLS (Foreign Key in reference to table “Disease”)
- Approved_symbol (Foreign Key in reference to table “Gene_NDP”)

b) GeneProtein
- Association_id (AUTO_INCREMENT) (Primary Key)
- Approved_symbol (Foreign Key in reference to table “Gene_NDP”)
- Uniprot_id (Foreign Key in reference to table “Proteins”)

c) DrugProtein
- Association_id (AUTO_INCREMENT) (Primary Key)
- NAME (Foreign Key in reference to table “Drugs”)
- Uniprot_id (Foreign Key in reference to table “Proteins”)

Data was collected from DisGeNet, genenames.org, Uniprot and DrugBank. The database was created at phpMyAdmin web tool. Various select queries were realized. An example follows in which the name, symbol and genome coordinates are asked regarding genes associated with coronary heart disease (CUI: C0010068).

```
SELECT Genes_NDP.Approved_name, Genes_NDP.Approved_symbol, Genes_NDP.Locus

FROM Genes_NDP

JOIN GeneDisease ON Genes_NDP.Approved_symbol = GeneDisease.Approved_symbol

JOIN Disease ON GeneDisease.CUI_UMLS = Disease.CUI_UMLS

WHERE Disease.CUI_UMLS = 'C0010068'
```

The same process was conducted creating a python script using Notepad++ as editor and some Notepad++ plugins for the connection with a remote Linux Ubuntu server. Specifically, NppFTP (sharing python script between remote server and Notepad++) and Python Indent (managing spaces in Python code). Putty was the used SSH client to execute the python script at the remote server. An example of python script is written at Appendix A referring to the SQL database connection, to the parsing of files, to the creation of tables, the insertion of data and the realization of a query.

4 MACHINE LEARNING AND DATA MINING PROJECTS

A. Preprocessing of Frail-or-Safe clinical dataset and classification analysis to predict the "fried" parameter, 2021 Patra
(*Part of code at Appendix B)

The dataset contains information which was collected during the clinical evaluation of older people from medical experts. This information represents the clinical status of the older person across different domains (physical, psychological, cognitive, etc). The "fried" parameter categorizes the older population into: Frail, Pre-frail, Non-frail. This categorization is generated by 5 of the measurements, namely the weight_loss, exhaustion_score, gait_speed_slower, grip_strength_abnormal, and low_physical_activity, which were not included in the classification analysis.

The analysis was conducted in R using R studio as editor. Firstly, the erroneous, missing and extreme values were detected through the function *factor()* and subsequent graphs of independent variables in association with "fried" parameter. They were all replaced with "*NA*". The patients/entries (rows) and their features (columns) with more than 20% of their values as "*NA*" were removed. The remaining "*NA*" of numerical variables were replaced with the medium of each column. The categorical variables with many "*NA*" were eventually removed, as they were proved insignificant by the decision tree, as well as the entries with extra "NA".

The regression tree was selected as one of the classification algorithms for the prediction of "*fried*" parameter. 70% of dataset was used as train set and 30% as test set respectively. A confusion matrix was created for both sets and accuracy score was the quality metric of the model. Efforts for the improvement of the model included the merge of "*frail*" and "*pre-frail*" classes as one, indicated by the confusion matrix, the exclusion of insignificant variables and the enforcement of cost matrix.

predictions	Frail	Non Frail	Pre Frail
Frail	39	1	11
Non Frail	5	121	33
Pre Frail	23	20	103

Figure. Confusion matrix of train set (2/3 of entries). The first row of the table gives us the true classes of dataset, whereas the first column of the table depicts the predicted classes generated by the regression tree model. This table shows that 11 entries of class "Pre-Frail" were falsely categorized at "Frail" class by the model. That is an argument for the merge of these two classes.

Random forest (ensemble learning) was the second selected classification algorithm with less chance of overfitting. The initial (default) parameters of the model were 500 trees and 6 variables tried per split (feature space partition). A confusion matrix was created concerning the "out of bag" data (test sets). Furthermore, using the error rate table produced by the model, I created a graph displaying the progression of the out of bag error (oob) as the number of trees rises. The parameter of variables tried per split was also put into investigation implementing a relative "for" loop. The loop meant to output the out of bag error for models testing 1 to 10 variables per split. It was proved that the default parameters produced the best model.

```
randomForest(formula = fried ~ ., data = rio_csv, proximity = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 6

        OOB estimate of  error rate: 33.84%
Confusion matrix:
           Frail Non Frail Pre Frail class.error
Frail         40         6        49   0.5789474
Non Frail      1       153        54   0.2644231
Pre Frail     12        56       155   0.3049327
```

Figure. Parameters of random forest model. The model is made up of 500 trees and 6 variables are tried per split. Additionally, the relevant confusion matrix is depicted. The class "Frail" has large out of bag error, as it is confused with "Pre-Frail" class.

Figure. This graph shows the progression of out of bag error (oob: error in the classification of test sets) for each class and the overall estimated oob as the number of trees rises. We observe that, after 200 trees, the oob (blue line) is stabilized.

Finally, a Multidimensional scaling diagram was also created. The proximity matrix of the best random forest model was converted to distance matrix. The biggest percentage of variance of classes is distributed between two axes, X (3.4%) and Y (1.9%), and we observe how patients are distributed between these two axes.

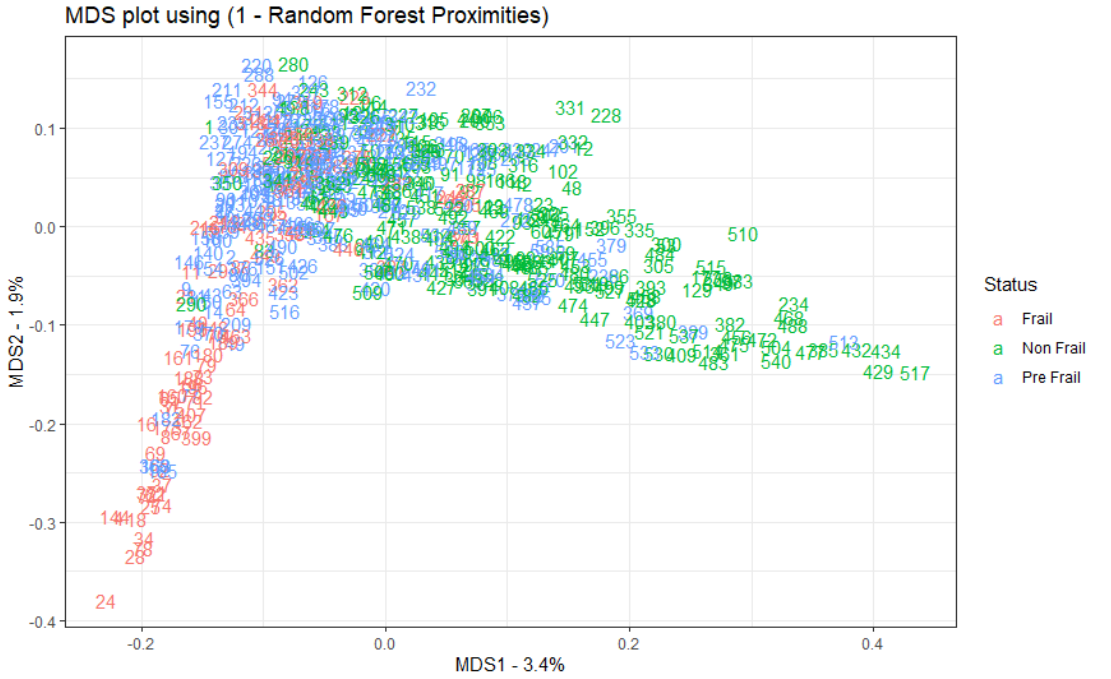


Figure. Multidimensional scaling plot of the given classes as they have been mapped in feature space by the random forest model. Two are the main axes with the biggest percentage of variance. On these two dimensions, we observe the distribution of classes. X axis mainly determines the "distance" between the classes. We observe that the "Non-Frail" class (green numbers) is clearly distinguished from the "Frail" (red numbers) and the "Pre-Frail" (blue numbers) class.

B. Preprocessing of weather dataset and regression analysis to interpret the "temperature" variable, 2021 Patra

(*Part of code at Appendix C)

The analysis was conducted in R using R studio as editor. The dataset refers to measurements (temperature, humidity etc.) per 10 minutes regarding the weather of Athens in 2008. We detected remote and extreme values through diagrams of each variable (atmospheric pressure, humidity, wind direction etc.) in association with the outcome (temperature). These entries are removed. Moreover, we analyzed the given time into month, day and hour, and we created the new variable "day" which includes two categorical values, "day" and "night". That was created based on the diagram of hour with solar radiation. Another variable was created ("season") based on the number of the month. Through diagrams, we observed the association of humidity with temperature (decreasing function) and the association of solar radiation with temperature for each season and hour of the day. Some variables (id, wind.direction, dt) are removed, since either they have no cause-effect relationship with the variations of temperature or their information has been passed to other variables (dt). Firstly, we implemented regression trees through the "rpart" and "rpart.plot" library with ANOVA as method.

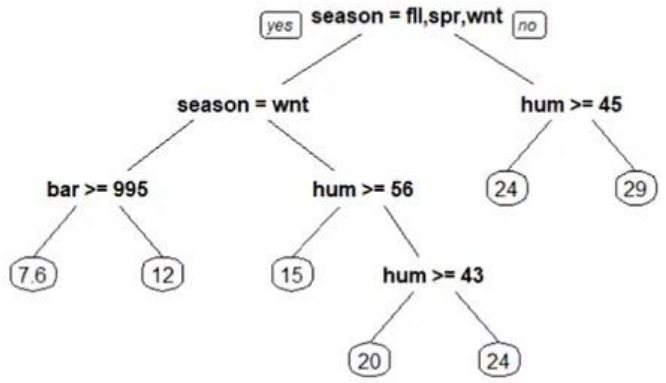


Figure. Regression tree of the dataset "weather" with "temperature" as dependent variable. "Season" is the most important variable that determines "temperature". "Humidity" and "atmospheric pressure" are also quite important. At the leaves of the tree, the mean temperature is depicted. Based on the tree, association rules can be extracted.

We observe that the variance of "temperature" depends mostly on "season" and "humidity". The depicted interactions were used to a multiple regression model. The residuals of this model had their median near at 0 and its p-value was very low. Its correlation with the entries and its ability to interpret the variation of "temperature" were very high (R-squared = 0.82). Additionally, we attempted the prediction of "temperature" through the interpretation of variation of residuals of "temperature" in association with the residuals of each variable. We used the forecasting tool ARIMA, we selected the residuals per variable, and we built a regression tree.

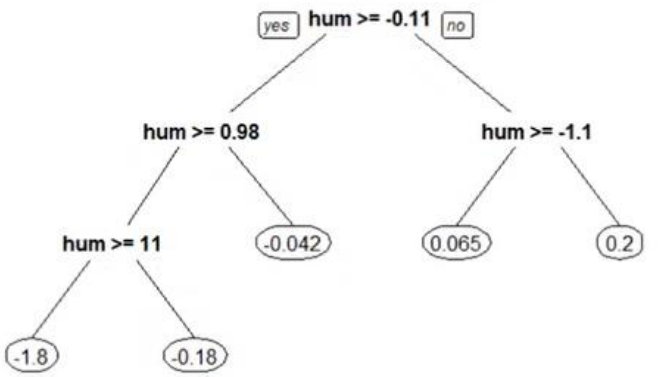


Figure. Regression tree of residuals generated by ARIMA. ARIMA was implemented on every variable of dataset "weather". Residuals of "temperature" was the dependent variable.

Through the interactions of tree, we built a multiple regression model. That model gave us an additional 28% to 70% of ARIMA, concerning the ability to predict "temperature".

C. Analysis of single-cell RNA-sequencing data in MATLAB, 2020 Patra

Classification of data using boosting (Adaptive Boosting for Multiclass Classification, Random undersampling boosting, Totally corrective boosting) and bagging (Bootstrap Aggregation) algorithms. Sampling was realized through 10-fold cross validation and the used evaluation metrics were accuracy, sensitivity and F1-score.

Clustering of data and comparison of the algorithms Kmeans, Hierarchical Clustering (single linkage) and DBSCAN. Internal evaluation schemes were Dunn index and Silhouette coefficient. External criteria were Purity and Rand measure.

D. Analysis of an image via SVD (Singular Value Decomposition) in MATLAB, 2020 Patra (*Part of code at Appendix D)

Scope of this analysis was the generation of singular vectors and singular values matrices, with the aim to approach the initial image with the most efficient way, in terms of computer memory cost. The error of approach was calculated in two distance metrics, 2-norm and Frobenius norm. The stored bytes were calculated based on the elements of matrices of each approach (8 bytes cost by each element). A relative function was also generated.

5 MOLECULAR BIOLOGY PROJECTS

A. BS thesis: Study of cytotoxicity of [Re (CO)3 (enrofloxasine) (imidazole)] in K-562 and in U87-MG cells and uptake and cell distribution of [99mTc (CO)3 (enrofloxasine) (imidazole)] in K-562 cells, 2016 Thessaloniki

Organometallic rhenium complexes have recently been considered in the development of novel antitumor agents due to their suitable properties. The laboratories of Medicinal Chemistry and Inorganic Chemistry synthesized rhenium tricarbonyl complexes with quinolone antimicrobial agents enrofloxacin and levofloxacin. I was called to test the cytotoxicity and subcellular distribution of the first.

Regarding cytotoxicity, [Re (CO)3 (enrofloxasine) (imidazole)] and free enrofloxasine were compared to their death effect on K-562 and U87-MG cells, with intend to evaluate the possible antitumor action of the complex. That was assured by the dominant death rate of K562 cells with the presence of the complex and the respective reduced IC50 on both types of cells. This result indicates this rhenium complex as a potential antitumor agent.

The study of absorption and subcellular distribution was conducted on [99mTc(CO)3(erx)(im)] with the goal of revealing the subcellular target of the compound. That would be additional information for its mechanism of action. The complex was evaluated as capable of entering K562 cells and its highest concentration was shown at mitochondria and cytoplasm. Consequently, mitochondria were considered as the main subcellular location and the mitochondrial DNA as its main target.

6 APPENDIX (CODE)

APPENDIX A

"Design, implementation of Relational Database associating drugs with diseases using SQL and Python" project

```
import MySQLdb
```

```
from MySQLdb.cursors import SSCursor
import os
import codecs

def connectDB():
	return MySQLdb.connect("pez.insybio.com", "tsoukas", "******", "tsoukas_diseasedb")

def getTest(conn):
	conn.begin()
	cursor = conn.cursor()
	query = ("SELECT * FROM test")
	cursor.execute(query)
	results = cursor.fetchall()
	for row in results:
		print (row)
	cursor.close()

if __name__ == "__main__":

	filename="test_file.txt";
	genes=[[] for i in range(2)];
	diseases=[[] for i in range(2)];
	gene2diseases=[[] for i in range(3)];

	line_counter=0;
	print("Processing file:"+filename);
	file_input_open = open(filename,"r");
	for line1 in file_input_open:
		if ((line_counter%float(50000))==0):
			print(str(line_counter)+" lines have been parsed. Please wait!")
		if line_counter>0:
			words=line1.split("\t");
			if (words[1] not in genes[0]) and (words[2] not in genes[1]):
				genes[0].append(words[1]);
				genes[1].append(words[2]);
			if (words[3] not in diseases[0]) and (words[4] not in diseases[1]):
				diseases[0].append(words[3]);
```

```
                    diseases[1].append(words[4]);
                #if words[1] not in gene2diseases[0]:
                gene2diseases[0].append(words[1]);
                #if words[3] not in gene2diseases[1]:
                gene2diseases[1].append(words[3]);
                #if words[7] not in gene2diseases[2]:
                gene2diseases[2].append(words[7]);
        line_counter=line_counter+1;
    file_input_open.close();

    # enter data to database
    conn = connectDB()
    if (conn != -1):
        print("BD Connection Established")
        print("Tables are being created");
        #creating tables
        conn.begin()
        cursor = conn.cursor()
        create_genes_table_query= """CREATE TABLE IF NOT EXISTS genes(geneSymbol varchar(255) NOT NULL, geneName varchar(255) NOT NULL, PRIMARY KEY (geneSymbol))"""
        cursor.execute(create_genes_table_query)
        conn.commit()
        cursor.close()

        print("Data are being inserted to database");
        #inserting values
        for i in range(0, len(genes[0])):
            conn.begin()
            cursor = conn.cursor()
            insert_genes_table_query= """INSERT INTO genes VALUES('"""+genes[0][i]+"""','"""+genes[1][i].replace("\'","")+"""')""";
            #print(insert_genes_table_query)
            cursor.execute(insert_genes_table_query)
            conn.commit()
            cursor.close()
        print("Genes inserted succesfully");
        print("Select queries are now realized");
```

```
                        conn.begin()
                        cursor = conn.cursor()
                        insert_select_alzheimer_genes_table_query= """SELECT * FROM gene2disease
WHERE diseaseId=%s""";
                        try:

cursor.execute(insert_select_alzheimer_genes_table_query,["umls:C0002395"])
                                results = cursor.fetchall()
                                print("Genes related to alzheimer:")
                                for row in results:
                                        # Now print fetched result
                                        print (str(row[0])+","+str(row[1])+','+str(row[2])+','+str(row[3]));
                        except:
                                print("Error: unable to fecth data")
                        conn.commit()
                        cursor.close()
                        conn.close()
                else:
                print("problem")
```

APPENDIX B

"Preprocessing of Frail-or-Safe clinical dataset and classification analysis to predict the "fried" parameter" project

```
#example of histogram for detecting excessive values
rio_csv$fried<-as.numeric(factor(rio_csv$fried))
library(ggplot2)

ggplot(data=rio_csv, aes(x=social_phone, y=fried)) +
 geom_point(shape=1) +
#  geom_smooth() +
 theme(axis.title.x = element_text(face="bold", colour="#990000", size=20),axis.text.x  = element_text(vjust=0.5,
size=16),axis.title.y = element_text(face="bold", colour="#990000", size=20),axis.text.y  = element_text(angle=90,
vjust=0.5, size=16)) +
 scale_x_continuous(name="social_phone") +
 scale_y_continuous(name="fried")+
 theme(legend.position="none") +
 geom_rug(col=rgb(0.9,0,0,alpha=.1), sides="lrtb", size=1)
```

```
which(rio_csv$social_phone>700)

rio_csv$social_phone[rio_csv$social_phone>1000] <- NA
rio_csv$social_phone[is.na(rio_csv$social_phone)]<-mean(rio_csv$social_phone,na.rm= TRUE)

#regression tree
library(rpart)
library(rpart.plot)
install.packages("plyr")
library(plyr)

rio_csv$fried <- as.factor(rio_csv$fried)
rio_csv$fried <- revalue(rio_csv$fried, c("1"="Frail", "2"="Non Frail", "3"="Pre Frail"))

set.seed(21)

samples <-sample(2, nrow(rio_csv), replace = TRUE, prob= c(0.7,0.3))
train<- rio_csv[samples ==1,]
test <- rio_csv[samples ==2,]

fit = rpart(fried ~., method="class", data=train)
prp(fit)
summary(fit)
prop.table(fit$variable.importance)
round(100*prop.table(fit$variable.importance),2)

#confusion matrix
predictions<-predict(fit,train,type = "class")
head(predictions)
table(predictions,train$fried)
confmat<-table(predictions,train$fried)

#accuracy rate
sum(diag(confmat))/sum(confmat)

#random forest
set.seed(42)
```

```
library(ggplot2)
install.packages("cowplot")
library(cowplot)
install.packages("randomForest")
library(randomForest)

rio_csv$fried <- as.factor(rio_csv$fried)
model <- randomForest(fried ~ ., data=rio_csv, proximity=TRUE)

# investigation of improved model as for number of trees and trained valuables per split
oob.error.data <- data.frame(
 Trees=rep(1:nrow(model$err.rate), times=4),
 Type=rep(c("OOB", "Frail", "Non Frail", "Pre Frail"), each=nrow(model$err.rate)),
 Error=c(model$err.rate[,"OOB"],
       model$err.rate[,"Frail"],
       model$err.rate[,"Non Frail"],
       model$err.rate[,"Pre Frail"]))
model

ggplot(data=oob.error.data, aes(x=Trees, y=Error)) +
 geom_line(aes(color=Type))

oob.values <- vector(length=10)
for(i in 1:10) {
 temp.model <- randomForest(fried ~ ., data=rio_csv, mtry=i, ntree=500)
 oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate),1]
}

oob.values
min(oob.values)
which(oob.values == min(oob.values))

#best model
model <- randomForest(fried ~ .,
                data=rio_csv,
                ntree=500,
                proximity=TRUE,
```

```
                mtry=6)

#mds diagram
model <- randomForest(fried ~ ., data=rio_csv, proximity=TRUE)

distance.matrix <- as.dist(1-model$proximity)

mds.stuff <- cmdscale(distance.matrix, eig=TRUE, x.ret=TRUE)

mds.var.per <- round(mds.stuff$eig/sum(mds.stuff$eig)*100, 1)

mds.values <- mds.stuff$points
mds.data <- data.frame(Sample=rownames(mds.values),
                X=mds.values[,1],
                Y=mds.values[,2],
                Status=rio_csv$fried)

ggplot(data=mds.data, aes(x=X, y=Y, label=Sample)) +
 geom_text(aes(color=Status)) +
 theme_bw() +
 xlab(paste("MDS1 - ", mds.var.per[1], "%", sep="")) +
 ylab(paste("MDS2 - ", mds.var.per[2], "%", sep="")) +
 ggtitle("MDS plot using (1 - Random Forest Proximities)")
```

APPENDIX C

"Preprocessing of weather dataset and regression analysis to interpret the "temperature" variable" project

```
install.packages("ggplot2")
install.packages("rpart")
install.packages("rpart.plot")
################################
weather = read.table("weather.all.2008.csv", header=T, sep=",")
names(weather)
w.month = as.numeric(format(as.Date(weather$dt), "%m"))
w.day = as.numeric(format(as.Date(weather$dt), "%d"))
w.date = as.POSIXlt(strptime(weather$dt, "%Y-%m-%d %H:%M"))
```

```
w.hour = w.date$hour

weather = data.frame(weather, month=w.month, day=w.day, hour=w.hour)


names(weather)

[1] "id"    "temp"  "bar"   "hum"   "rad"   "w.d"   "w.s"   "dt"    "month"

[10] "day"   "hour"


#### example of variables-temperature graph

library(ggplot2)

ggplot(data=weather, aes(x=sol, y=temp)) +

  geom_point(shape=1) +

  geom_smooth() +

  theme(axis.title.x = element_text(face="bold", colour="#990000", size=20),axis.text.x  = element_text(vjust=0.5, size=16),axis.title.y = element_text(face="bold", colour="#990000", size=20),axis.text.y  = element_text(angle=90, vjust=0.5, size=16)) +

  scale_x_continuous(name="Radiation (rad)") +

  scale_y_continuous(name="Temperature (temp)")+

  theme(legend.position="none") +

  geom_rug(col=rgb(0.9,0,0,alpha=.1), sides="lrtb", size=1)


##### create pseudovariables

ggplot(data=weather.f, aes(x=hour, y=sol)) +

  geom_point(shape=1) +

  geom_smooth() +

  theme(axis.title.x = element_text(face="bold", colour="#990000", size=20),axis.text.x  = element_text(vjust=0.5, size=16),axis.title.y = element_text(face="bold", colour="#990000", size=20),axis.text.y  = element_text(angle=90, vjust=0.5, size=16)) +

  scale_x_discrete(name="hour") +

  scale_y_continuous(name="radiation (rad)")+

  theme(legend.position="none") +

  geom_rug(col=rgb(0.9,0,0,alpha=.1), sides="lrtb", size=1)


hour.f = rep("day", length(weather.f$hour))

hour.f

hour.f[weather.f$hour==0] = c("night")

hour.f[weather.f$hour==1] = c("night")

hour.f[weather.f$hour==2] = c("night")

hour.f[weather.f$hour==3] = c("night")
```

```
hour.f[weather.f$hour==4] = c("night")
hour.f[weather.f$hour==20] = c("night")
hour.f[weather.f$hour==21] = c("night")
hour.f[weather.f$hour==22] = c("night")
hour.f[weather.f$hour==23] = c("night")

### find maximal model ###
library(rpart)
library(rpart.plot)
fit = rpart(temp ~., method="anova", data=weather.f)
prp(fit)
summary(fit)
prop.table(fit$variable.importance)
round(100*prop.table(fit$variable.importance),2)

### multiple regression ###
fit.lm = lm(temp ~ . + season*bar + season*I(hum^2) , data=weather.f)
summary(fit.lm)
### use AIC to simplify model ###
fit.lm = step(fit.lm, direction = "both")
summary(fit.lm)

### best fit - clean data ###
##arima
library(forecast)
to.clean.temp = auto.arima(weather.f$temp)
temp.c=to.clean.temp$residuals

weather.c = weather.f
weather.c$temp = auto.arima(weather.f$temp)$residuals
weather.c$bar = auto.arima(weather.f$bar)$residuals
weather.c$hum = auto.arima(weather.f$hum)$residuals
weather.c$sol = auto.arima(weather.f$sol)$residuals
weather.c$w.d = auto.arima(weather.f$w.d)$residuals
weather.c$w.s = auto.arima(weather.f$w.s)$residuals

n.fit = rpart(temp ~., method="anova", data=weather.c)
```

```
prp(n.fit)

summary(n.fit)

prop.table(n.fit$variable.importance)

round(100*prop.table(n.fit$variable.importance),2)


n.fit.lm = lm(temp ~ . + I(hum^3) , data=weather.c)

summary(n.fit.lm)

n.fit.lm = step(n.fit.lm, direction="both")

summary(n.fit.lm)
```

APPENDIX D

"Analysis of an image via SVD (Singular Value Decomposition) in MATLAB" project

```
Pic = imread('C:\Users\tsouk\Desktop\mypic.jpg')

Q = im2double(Pic)

whos Q

Q = rgb2gray(im2double(Pic))


%SVD

[U,S,V] = svd(Q)

imshow(Q)

semilogy(diag(S))

ylabel('singular values')


%approaching picture

kmine = 200

Qmine = U(:,1:kmine)*S(1:kmine,1:kmine)*V(:,1:kmine).'

imshow(Qmine)

er_norm2 = norm(Q-Qmine)

er_froben= norm(Q-Qmine,"fro")


%calculation of bytes

bytesmine = kmine*(8 * size(Q,1) + 8 * size(Q,2) + 8)


%function taking the picture "Q" and the first "k" singular values. Output is the approaching picture "B" and the
%error "er" by norm-2

function [B,er] = mono_compress(Q,k)
```

```
C = Q
%C = rgb2gray(im2double(Pic))
C = im2double(C)
C = rgb2gray(C)
C = double(C)
[U,S,V] = svd(C)
Cmat = U*S*V';
k5=k
U5=U(:,1:k5)
S5=S(1:k5,1:k5)
V5=V(:,1:k5)
k5mat=U5*S5*V5'
imshow(k5mat)
er_norm5=norm(Cmat-k5mat)
er=er_norm5
B=k5mat
End
```