COMPUTATIONAL MODELING OF THE PANCREAS: LIFELONG SIMULATIONS OF PANCREATITIS

by

Yerkebulan A Talzhanov

MD, Karaganda State Medical Academy, Kazakhstan, 2007

Submitted to the Graduate Faculty of

Department of Human Genetics

Graduate School of Public Health in partial fulfillment

of the requirements for the degree of

Master of Science

University of Pittsburgh

2012

UNIVERSITY OF PITTSBURGH

Graduate School of Public Health

This thesis was presented

by

Yerkebulan Talzhanov

It was defended on

December 7, 2011

and approved by

Thesis Advisor:
Mahmud M. Barmada, PhD
Associate Professor
Department of Human Genetics
Graduate School of Public Health
University of Pittsburgh

Committee Members:
Daniel E. Weeks, PhD
Professor
Department of Human Genetics
Graduate School of Public Health
University of Pittsburgh

David C. Whitcomb, MD. PhD
Professor
Departments of Medicine, Cell Biology
and Physiology, and Human Genetics
University of Pittsburgh

COMPUTATIONAL MODELING OF THE PANCREAS:
LIFELONG SIMULATIONS OF PANCREATITIS

Yerkebulan Talzhanov, MS

University of Pittsburgh, 2012

The aim of this study is to build a mathematical model of the pancreas and run this model in lifelong simulations in order to understand the mechanisms that predict an increased risk of pancreatitis. The dilemma is that pancreatitis is a complex process with multiple variables, which currently make the onset, severity and outcomes unpredictable in individual patients. Genetic, environmental and metabolic factors are likely to be important for disease severity and progression, with somewhat stochastic events initiating theprocess when stress leads to injury signals. Modeling should begin in the acinar cell, with ability to incorporate duct cells, inflammatory cells, other cells and their interactions into large model. In this work we attempted to build a foundational model that incorporates the main features and interactors. We built a framework that focuses on trypsinogen activation as a major cause of auto-digestion and injury. Additional variables such as production ofpancreatic secretory trypsin inhibitor(PSTI) molecules as a defense line against active trypsin as well as bicarbonate secretion were included in the model. The effects of mutation were modeled as modified rates of trypsinogen production/activation, trypsin inactivation, PSTI production, and bicarbonate secretion. Our framework contains three compartments that represent domains where trypsin could be activated. The domains are acinar cell, lumen of the acinus, and main duct of the pancreas. We used a stochastic approach to test the general flow of the simulation.

The public health and translational significance of this model is thatit would help us to understand the physiology of the pancreas more deeply. It wouldalso allow us to consider many

factors that lead to pancreatitis and predict their behavior under different conditions (e.g., therapy).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 INTODUCTION

The pancreas is a large gland situated in the upper abdominal area that plays an important role in our body by performing both endocrine and exocrine functions. The endocrine function is achieved by the ability of the pancreas to release hormones like insulin and glucagon into the blood stream. These hormones help our body to keep glucose levels within a physiological range. The exocrine function is achieved by production and secretion of alkaline juice with digestive enzymes that helps to digest food. These enzymes flow from the pancreas to the upper part of the small intestines(called the duodenum) via the pancreatic duct. Normally, these digestive enzymes become active when they reach the small intestine and in this form they break down the food we consume. But when these enzymes become activated within the pancreatic gland they begin to digest whatever they meet causinginjury, inflammation, and may produce necrosis and/or fibrosis of the pancreas itself. Autodigestion of the pancreas leads to an inflammatory process that results in pancreatitis. Clinically we define two types of pancreatitis: acute and chronic forms with fast and slow durations respectively. Both formsare serious and in severe cases can lead to complications such as bleeding, infection, and permanent tissue damage[1-4].

Under normal physiological conditions, pancreatic juice contains inactive forms of several digestive enzymes known as proenzymes or zymogens. Full activation of proenzymes occurs only in the small intestine. Enterokinase first converts trypsinogen into active trypsin, and the active trypsin goes on to activate more of itself and the rest of the digestive enzymes[1-4]. There

are several defensive mechanisms that prevent early activation of digestive enzymes inside the pancreas: in the intracellular stage, proenzymes are stored and transported in vesicles. This compartmentalization keeps proenzymes isolated from lysosomal enzymes and preventsthe proenzymes from being activated; along with digestive enzymes,the pancreatic acinar cells also secrete secretory trypsin inhibitor (PSTI), that has a high affinity for trypsin and blocks its proteolytic activity; control of intracellular level of calcium (Ca^{2+}) also plays an important role in trypsin stability by reducing trypsin's degradation rate; and the secretion of sodium-bicarbonate rich fluid by duct cells flushes proenzymes from the pancreatic ductal system into the intestine. Disruption of these protective mechanisms will increase the risk of early activation of digestive enzymes in the pancreas resulting in auto-digestion and pancreatitis[1].

Epidemiologic studies of pancreatitis clearly define a list of associated risk factors. For example, the most common risk factor is the presence of gallstones. When the gallstones pass through the common bile duct they may become lodged at the junction of the bile and pancreatic duct, causing pancreatic juice to be retained in the pancreas. If the proenzymes activate, they will cause injury and inflammation in the pancreas that results in acute pancreatitis[5-7]. Another common risk factor is chronic, heavy alcohol consumption that greatly increases the risk of developing chronic pancreatitis[8-10]. Interestingly, alcohol by itself increases the risk only marginally, but in population terms it h as a high attributable risk because of the worldwide exposure to alcohol. Genetic abnormalities of the pancreas can also lead to increased predisposition to develop pancreatitis and explain a subset of patients with hereditary pancreatitis[11].

Genetic studies have led to the identification of a number of potential genetic variants that contribute to the development of pancreatitis. These haveidentified major risk genes in

which mutations can affect early activation of digestive enzymes in the pancreas: the cationic trypsinogen gene (PRSS1), the pancreatic secretory trypsin inhibitor gene (PSTI) also known as Serine Protease Inhibitor Kazal Type 1 (SPINK1), the cystic fibrosis transmembrane conductance regulator gene (CFTR), calcium-sensing receptor gene (CASR), and chymotrypsin C gene (CTRC) (Figure 1)[8, 12-14]. We know that an active form of trypsinogen, trypsin, is a major protease that converts the rest of digestive enzymes into their active form. Mutations that increase the stability of trypsin will increase the chances of breaking the defense mechanisms and over-activating of trypsinogen and other digestive proenzymes.

PRSS1 mutations that lead to *gain of function* are risk factors for pancreatitis. PRSS1 mutations are inherited in autosomal dominant manner and cause hereditary pancreatitis with a penetrance of 80%. Among the most common PRSS1 mutations we recognize R122H and N29I mutations. The R122H mutation increases trypsin's resistance to autolysis and permanent inactivation. While the N29I mutation alters the structure conformation of trypsinogen, it makes trypsinogen tolerant to trypsin inhibitor[15-18].

PSTI molecules are the first defense line against the autodigestion caused by early activation of trypsinogen into trypsin. It was believed that PSTI can inhibit up to 20% of trypsin activity[19]. However, normally it is not present in the pancreas except after injury and inflammation, which means that PSTI is an acute phase protein that is produced in response to inflammation in the pancreas, thus the fraction of trypsin that PSTI can inhibit is dynamically regulated. It also makes sense that the mutations in PSTI by themselves donot increase the risk of developing pancreatitis, but they accelerate the risk in combination with other mutations that lead to recurrent premature trypsin activation[12, 19, 20].

CFTR is the gene that is responsible for cystic fibrosis when severe mutations are present on both alleles. About 2,000 mutations have been described in the human CFTR gene. But only those mutations that alter bicarbonate secretion by the pancreatic duct cell are actually essential for increased risk of pancreatitis due to the fact that the pancreatic duct cell secretes bicarbonate that attracts water into the duct and eases the transportation of digestive enzymes into the intestine. In the case of reduced bicarbonate secretion, the flow of pancreatic juice though the ducts diminishes, increasing the risk of early trypsin activation. Interestingly a subset of CFTR mutations that cause pancreatitis, but that have no effect on lung, have been described - they selectively impede bicarbonate secretion, but not chloride secretion[21-23].

Finally, mutations in both CASR and CTRC genes may increase the risk of chronic pancreatitis[24, 25]. CASR mutations are responsible for elevating serum calcium levels, while CTRC mutations are responsible for functional defects in the digestive enzyme chymotrypsin C.

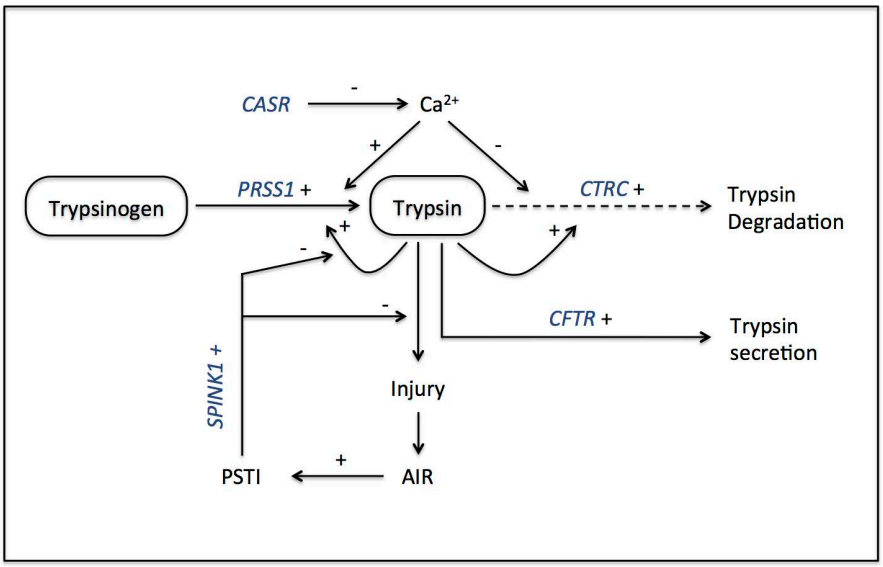

Figure 1: Diagram of trypsin control mechanisms in the pancreas

PRSS1 – protease serine 1; PSTI - pancreatic secretory trypsin inhibitor; SPINK1 – serine protease inhibitor, Kazal type 1; CFTR – cystic fibrosis transmembrane conductance regulator; CASR – calcium-sensing receptor gene, CTRC – chymotrypsin C gene; AIR – Acute injury response;

Adapted fromFeldman, M. Friedman, L.S. and Brandt, L.J.Gastrointestinal and Liver Disease (2010). [26]

Both environmental and genetic risk factors are responsible for developing of pancreatitis. But is pancreatitis important? The statistics for pancreatitis in Table 1 represents the US population[27]. As you can see, the incidence rate of acute pancreatitis is higher compared to incidence rate of chronic pancreatitis. In 2004 almost half of all ambulatory care visits ended with hospitalizations and roughly 9% of hospitalizations resulted in death. In 2003 the acute pancreatitis admissions cost was approximately $2.2 billion, at mean cost of $9,870 per hospitalization and mean cost of $1,670 per hospitalization day[28],based on data on 226,000 acute pancreatitis admissions from 37 participating states. Considering the cost of hospitalization admissions and the fact that majority of patients are chronic, pancreatitis is a serious public health issue. How can we address our work to this matter? We could build a model to predict

therisk of pancreatitis. In order to do that we first need to build a mathematical model of pancreas itself and run lifelong simulations of this model. Both acute pancreatitis and chronic pancreatitis are syndromes, which means that they are signs and symptoms of underlying problem or problems, usually of unknown etiology. Different etiologies, in different amounts and various combinations, will have different effects on the pancreas. Additional modifying factors further increase variability of the response to injury (e.g., inflammation). The reason to do modeling is that we have more than one variable and the overall risk and outcomes are currently unpredictable. If we are able to build a model of the pancreas that successfully calculates the risk of pancreatitis, itmeans that we finally clearly understand and distinguish different risk factors of pancreatitis and can accurately evaluate each of them. It may also help us to predict the resulting effects of curtain public health interventions that are directed at reducing pancreatitis.

Table 1.Statistics for pancreatitis.

Prevalence:	1.1 million people (1998)
Incidence:	
Acute:	17 cases per 100,000 people (2003)
Chronic:	8.2 cases per 100,000 people (1981)
Ambulatory care visits:	881,000 (2004)
Hospitalizations:	454,000 (2004)
Mortality:	3,480 deaths (2004)
Prescriptions:	766,000 (2004)

Adapted from National Digestive Diseases Information Clearinghouse (NDDIC)[27]

In this work, we construct a mathematical model of the pancreas. The model should be as simple as possible, but not too simple. For this purpose we focus only on the exocrine function of the pancreas (secretion of digestive enzymes). Because of the exclusive role of trypsin in the activation of other digestive enzymes, we limit our model to considering trypsin and its proenzyme - trypsinogen. Our main task is to observe trypsinogen from the time it is synthesized to the time it is flushed to the intestine. This means that we need to model the work of the

pancreas from the very beginning, its stimulation, to the very end, secretion of pancreatic juice into the intestine. For this purpose we build a general framework of the model in the first place and then assign different parts of the model to different compartments of the pancreas and synchronize their work.

2.0 METHODS

2.1 DEFINITIONS USED

2.1.1 Cell definition

Islet cells and acini are the primary functional units of pancreas[1-3]. They carry out the endocrine and exocrine functions of the pancreas. The acinus has three types of cells, the acinar cell, centroacinar cell, and duct cell (Figure 2A). The acinar cell is the dominant cell type in the pancreas and is responsible for the synthesis of digestive enzymes in response to food intake. Within this protein mixture we are only interested in trypsinogen as a precursor for active trypsin and PSTI as a defense mechanism against active trypsin. We are also interested in calcium because it has a direct influence on the rate of trypsin activation. Othermolecules (such as those that make the acinar cell work) are less important for our modeling.We are also interested in the acinar cell's "status", defined as either alive (active) or dead. When alive,the acinar cell can be stimulated to produce and secrete trypsinogen andcalcium to the lumen (Figure 2B).The acinar cell also secretes PSTI molecules, but only in response to inflammation[12, 16, 17]. As the PSTI is an acute phase inflammatory response molecule and as the assessment of inflammation is being made only once a "day", at this stage it is only possible to proceed with modeling PSTI as being constitutively produced. This allows us in the model to check whether secreted protective

molecules (PSTI) are sufficient to inhibit the damage in the case of early trypsin activation. If trypsin is over activated, we switch the status to the "dead" to simulate autolysis. When the acinar cell is dead, it does not produce any enzymes and does not contribute to the overall secretion. The construction of acinar cell as being alive or dead allow us freely switch between these two statuses. We can mimic regeneration processes and recover a pool of lost acinar cells by simply turning dead acinar cells intoliving cells once.

The centroacinar and duct cells are two cell types with similar functions: they produce bicarbonate (BC) that attracts water and promotes easy movement ("flushing") of the digestive juice[1-4, 29]. In our model there is no difference between these two cells in terms of bicarbonate production. So we simplified the two cell types into one type (our duct cell) (Figure 2B). This duct cell also has two conditions: alive and dead. When alive, it can be stimulated to produce bicarbonate and to secrete it into the lumen.

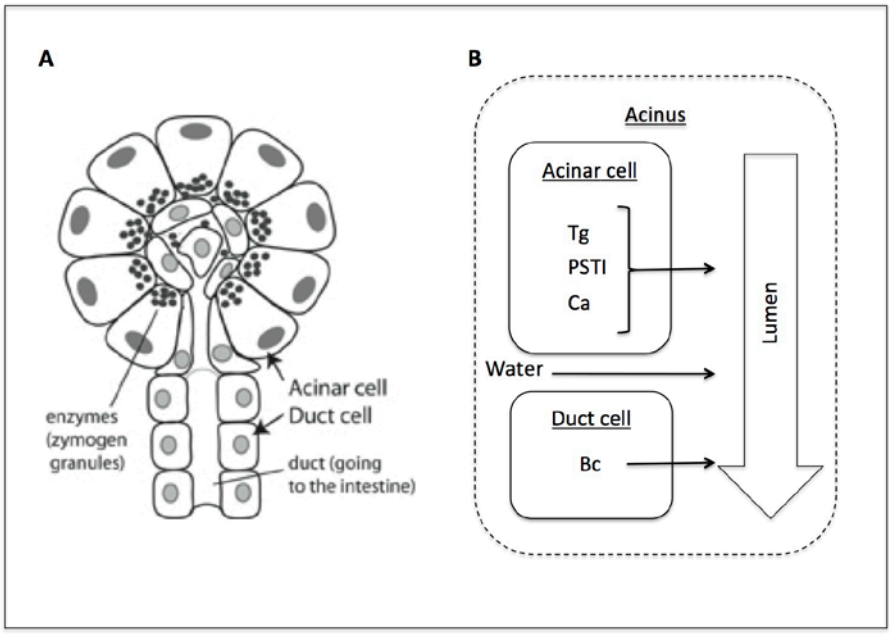

Figure 2.Diagram of the acinus.

A) Diagram of the acinus. Tg – trypsinogen; PSTI - pancreatic secretory trypsin inhibitor ; Ca – calcium; Bc – bicarbonate;*Copiedfrom D.C. Whitcomb and M.M. Barmada. Cell. and Mol. Life Sciences, 2007;*
B) Scheme used in the model to describe acinus.

2.1.2 Framework definition

Acinar cells, by themselves, are not sufficient enough to define a functional unit of the pancreas. Hundreds of acinar cells along with centroacinar cells are organized into an acinus, a round shaped cluster of cells with an open lumen (cavity) inside[1-4] (Figure 2A). This lumen opens to the duct, so enzymes secreted to the lumen can flow to the intestine through the duct. Because the functional unit of the pancreas, the acinus,needs to secrete both digestive juice and bicarbonate (to promote movement of the digestive juice out of the lumen of the acinus), we decided to include duct cellsin the acinus and treat the rest of the duct as a tube that moves digestive juice. In our model therefore, the acinus is represented by two vectors of numbers. The

first vector, named "status", carries numbers that describe the acinus status. The very first component of the “status” vector corresponds to the status of the acinus itself. It has two values 0 or 1, meaning dead or alive respectively. The second component of the “status” vector is responsible for describing whether the acinus has been regenerated or not, coded as 1 or 0 respectively. The third, final, component represents the number of acinar cells in a particular acinus. For example, a newly created acinus would have the following three numbers in “status” vector: {1,0,100}, meaning that the acinus is alive, was never regenerated, and has 100 acinar cells. The second vector, named "lumen", contains a set of variables that correspond to levels of trypsinogen, PSTI, calcium, and bicarbonate in the lumen. In order to simulate the secretion work of acinar cells and duct cell we designed two separate functions. In general it works like this: in the beginning we read the “status” vector. If the acinus is alive, we call the acinar cell function once for each acinar cell in the particular acinus. Then we finish our series by calling the duct cellfunction. Every time we call the acinar cell function we collect the produced molecules to the second vector, which at the end gives us the sum of each of the secreted molecules.

The way we define theacinus allows us to represent the whole pancreas in a matrix (Figure 3B), where each cell of the matrix corresponds to a single acinus.We can also mimic branching system of the pancreas. We treat the left column as a main duct andindividual rows as branches of the main duct. So we collect secreted juice in a particular direction: from right to left in every row and up in the left column. If we want to be more precise, we could even distinguish different parts of the pancreas. For example,we could assign the first few rows to the head of the pancreas, the next few rows to the neck, and the rest of the rows to the body and tail of the pancreas. Furthermore, we could treat the first row as the Santorini duct that opens to the

duodenum separately from the main duct. This representation of the pancreas even allows us to simulate duct obstruction that would immediately eliminate a portion of the pancreatic acini from delivering enzymes to the intestine. For this purpose it would be enough to define a single cell of the matrix as obstruction point and stop collecting pancreatic juice from the acini that are located after the obstruction point.

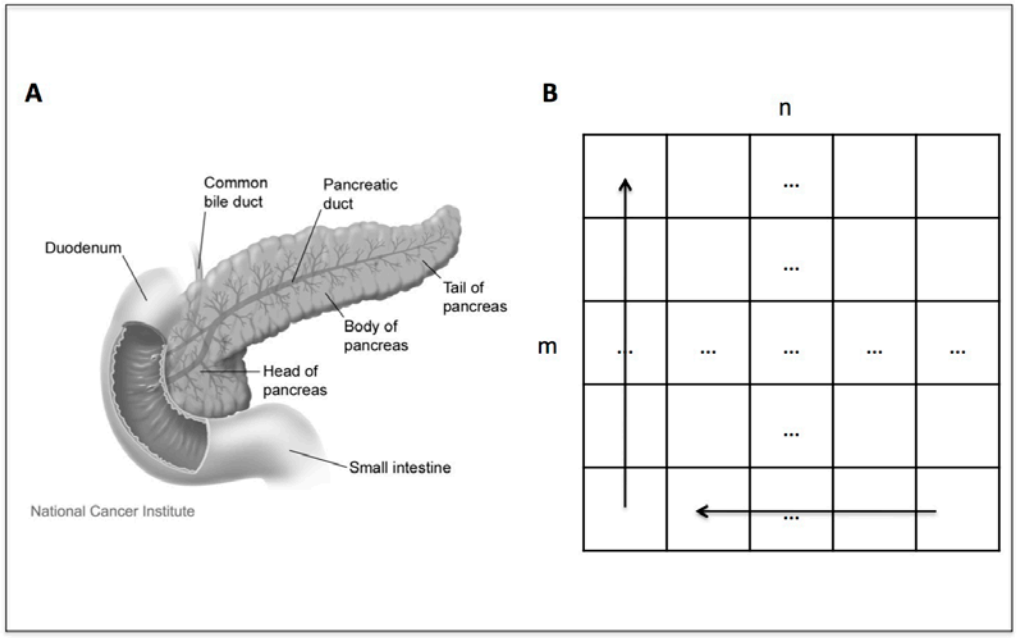

Figure 3.Diagram of the pancreas.

A) Diagram of the pancreas.*Copied from http://en.wikipedia.org/wiki/File:Duodenumandpancreas.jpg*[30];
B) Scheme used in the model to describe pancreas;

2.1.3 Mutations

We are interested in three potentially mutated genes: the cationic trypsinogen gene (PRSS1), the pancreatic trypsin inhibitor gene (PSTI), andthe cystic fibrosis transmembrane conductance regulator gene (CFTR). In order to include mutations in the model, we create a set of variables that will be included at key points of the model. For example, PRSS1 gene mutations can alter

production of trypsinogen, its activation, and degradation of active trypsin. The most common PRSS1 mutations are R122H and N29I. The R122H mutation increases trypsin resistance to autolysis and permanent inactivation,while the N29I mutation alters the structural conformation of trypsinogen that makes it tolerant to trypsin inhibitor[15-17]. For PSTI gene mutations we distinguish between the production rate of PSTI and its capability to bind active trypsin. For simplicity we can make an assumption that all PSTI binds to active trypsin (which is fairly well justified since the production of PSTI is typically much less than that of trypsinogen). This allows us to calibrate the binding rate of PSTI to trypsin with the production rate of PSTI (the less produced, the less bound). Lastly, for CFTR mutations, we are interested only in its effect that reduces bicarbonate production. In order to include these rates in the model we create five variables, such as trypsinogen production (Tg_p), trypsinogen activation (Tg_a), trypsin inactivation (T_i), PSTI production (PSTI_p), and CFTR mutation (CF). We assign them a value between 0 and 1 and include them in our equations. The default value for these rates is typically 1, and we consider only reduced rates, e.g., values less than 1. When these parameters are equal to 1, they have no effect. But once we decrease them they gain influence. For example, Tg_p is included in equation that describes trypsinogen production. Lets say that normally 100 molecules of trypsinogen are produced. When Tg_p is 1, we get all 100 molecules of trypsinogen, but if we reduce Tg_p to 0.90, we will get only 90 molecules of trypsinogen instead. The same rule applies for the rest of parameters in this set. If this approach is proved to be useful we could treat any mutation as a series of these rates and build a library of known mutations.

2.1.4 Inflammation

Inflammation (I) is a protective mechanism against defection, but may also cause injury (cell death in our case). It is recognized as a type of nonspecific innate immune response to injury. The main purpose of inflammation is to remove infectious agents and initiate healing. Technically, inflammation is the complex process that directs immune system components to migrate to the site of injury and do their work[31, 32]. In our model we are interested only in a few aspects of inflammation like causation, duration time and its effects. We describe inflammation, as a variable that takes values from 1 to 2, where 1 means no inflammation and 2 means the highest level of inflammation.We assume that inflammation can increase suddenly, in our case during a day, but reduces gradually, which means it takes few days for inflammation to disappear. We wrote two functions that either increase or decrease inflammation. Every time we call the increasing function we add 0.5 units to inflammation and every time we call the decreasing function we subtract 0.05 unitsfrom inflammation (Figure 4). In the model we distinguish two levels of inflammation, a local level that corresponds to inflammation of a single acinus and consequently influencesthe acinar cells of the acinus, and a general level that corresponds to inflammation of the whole pancreas. For general inflammation we also have a threshold equal to 1.55. Whenever general inflammation is increased twice in a short period of time, it reaches or pushes over the threshold and we call this an attack of the pancreatitis. In the model cell death and trypsin overactivation arethe only stimuli that can increase inflammation. Once inflammation is increased it will affect key points in the model like increasing PSTI production and increasing trypsinogen activation both in the acinar cell and in the lumen[16, 19, 20]. Without further injury or cell death, inflammation is expected to go down slowly. We designed this model so that inflammation isreduced only if there is no cell death at all during the

simulatedday, meaning no damage has occurred. Currently we do not distinguish between acute and chronic inflammation. However, we could make inflammation variable more flexible in order to capture the differences between acute and chronic pancreatitis, if our approach is proven to be useful.

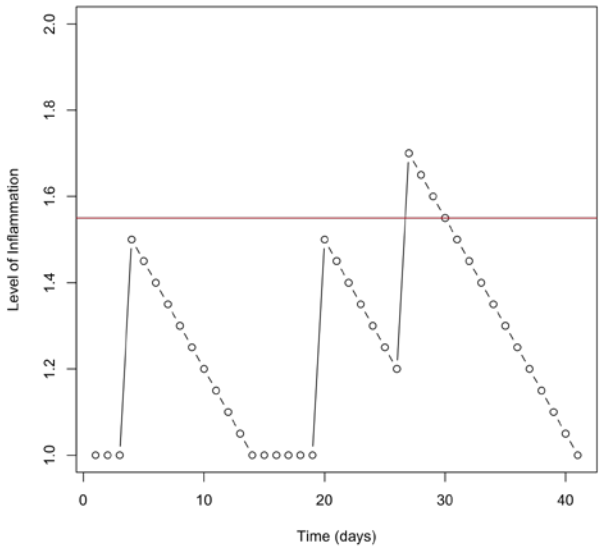

Figure 4. Inflammation.

We describe inflammation, as a variable that takes values from 1 to 2, where 1 means no inflammation and 2 means the highest level of inflammation. We can either increase inflammation by0.5 units in the case of damage or decrease it by0.05 units in the case of no damage. Whenever general inflammation reaches or pushes over the threshold (1.55) we call it a pancreatitis attack.

2.2 PROGRAMMING

2.2.1 Parameters used

We store parameters in a separate text file in order to keep the model flexible. If we want to run the simulation with a different set of parameters, we simply need to modify the text file that

contains these parameters, but not the program itself. The parameters and their shortcuts used in the model are listed in theTable 2. First we specify the number of patients and the number of days that we want to follow each patient. Then we define the size of the pancreas by providing numbers of branches of the main and secondary ducts along with the number of acinar cells that each acinus contains.

The next set of parameters describes the molecules that we are interested in, they are as follows: trypsinogen, PSTI (secretory trypsin inhibitor), calcium, bicarbonate, and water. It is important to understand that numbers assigned to trypsinogen and PSTI represent their total amount of produced molecules by the acinar cell. The parameter for calcium isa rate that calcium level increases when digestive enzymes go from acinar cell to lumen, and from lumen to main duct. Water has similar meaning calcium in this set of parameters. It describes increased water level in the main duct compared to the lumen of the acinus. The final parameter in this set corresponds to the bicarbonate secreted by the duct cells.

The next set of parameters is designed to modify all the equations we used in the model. They are: trypsinogen production, trypsinogen activation, trypsin inactivation, PSTI production, and CFTR mutation. Thisparameters of this set are multipliers in different parts of the equations we wanted to modify. When these parameters are equal to 1, they have no effect. But once we increase or decrease them, they gain influence. For example, Tg_p is included in equation that describes trypsinogen production. Let us say that normally 100 molecules of trypsinogen are produced. When Tg_p is 1, we get all 100 molecules of trypsinogen, but if we reduce Tg_p to 0.90 we will get only 90 molecules of trypsinogen instead. The same rule applies for the rest of the parameters in this set.

The next set of parameters defines the variability used in the model. We have three compartments of the model where trypsin can be activated. They are the acinar cell, the lumen and the duct respectively. For each compartment we specify the percentage of existing trypsinogen to be activated. We named these numbers as CA, LA, and DA. It is important to mention that C++ is case sensitive[33], so CA and Ca are distinguished from each other. We also specify a molecule "range", the range within which the production of any molecule can vary. We use this parameter to introduce random noise to molecule production. For example, if we specify trypsinogen production as 100 molecules,after we add random noise, trypsinogen can take any value between 95 to 105 molecules.

Finally the last parameter (Thr) definesthe tolerance of the acinar cell to the damage caused by trypsin. Whenever the level of trypsin molecules that were not blocked by PSTI exceeds this threshold, we assume that the acinar cell has been so damagedthat it is no longer able to recover and so is destroyed.

Since this is a preliminary model of pancreatitis we agreed to take one set of parameters that gives a good-looking survival curve and use it as a reference. We didn't expect our reference survival curve to be an exact match to any existing data, but to have curves for the three attacks located in the middle of the graph and we wanted the three curves to be spread out enough from each other so that variations between them can be observed (Figure 5).

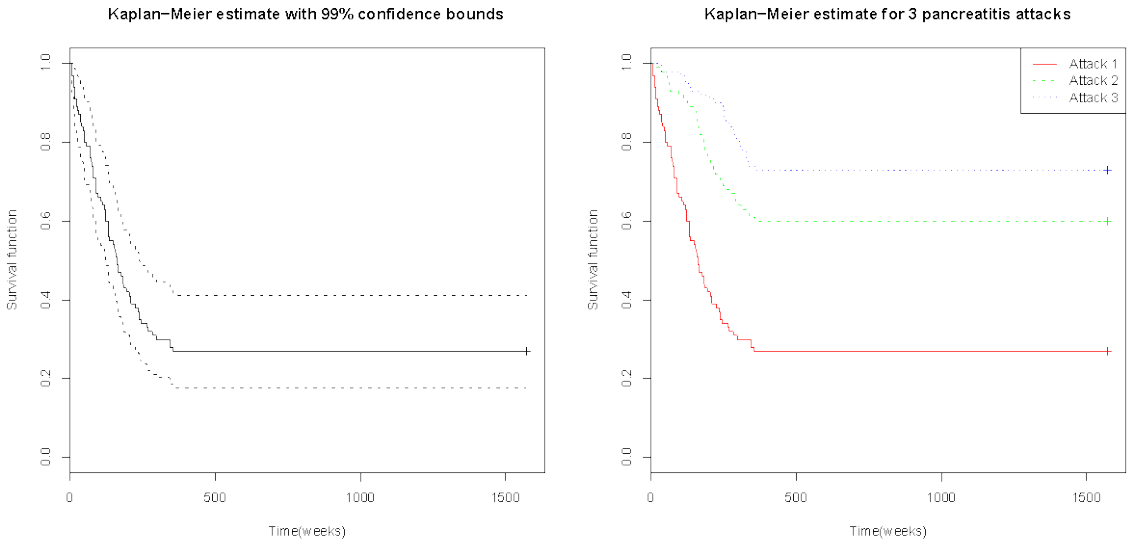

Figure 5.Reference survival curve of the pancreatitis.

Survival curve of the first pancreatitis attack with 99% confidence intervals (left graph);

Survival curves for first, second, and third attacks of pancreatitis respectively (right graph)

Table 2. Parameters used in the model.

Parameter	Symbol	Value
Number of patients	NP	100
Number of days	ND	11000
Number of main ducts	D1	30
Number of secondary ducts	D2	30
Number of acinar cells	D3	100
Trypsinogen	TG	100
PSTI	P	20
Calcium	Ca	5
Bicarbonate	BC	1
Water	W	5.5
Trypsinogen production	Tg_p	1
Trypsinogen activation	Tg_a	1
Trypsin inactivation	T_i	1
PSTI production	P_p	1
CFTR mutation	CF	1
Trypsin activation in the cell	CA	0.053
Trypsin activation in the lumen	LA	0.03
Trypsin activation in the duct	DA	0.03
Molecule range	MR	0.1
Threshold for inflammation	Thr	4.832

2.2.2 Building the framework

The first step in the program is to create a 30x30 matrix. Each cell of the matrix corresponds to a single acinus that contains 100 acinar cells and a duct cell. The size of the pancreas, and the number of branches that ducts have, varies from person to person. Textbooks provide us only with approximate ranges of those numbers, from 20 to 40 branches for main duct and the same range for secondary ducts[1-3]. So we decided to create a matrix that corresponds to the averageexpected size of the pancreas.

2.2.3 Pancreas stimulation

Secretion of trypsinogen and its flushing into the intestine is a continuous process.Conversion from trypsinogen into active trypsin can occur at any time during this process. In order to simplify our model we decided to separate this process into three major domains (stages): inside the cell,in the lumen, and in the duct. This division was guided by the fact that activation of trypsinogen can occur either when it is still inside the cell or outside the cell when it was secreted to the lumen or during transportation though the duct[16-20].

2.2.3.1 Acinar cell stimulations

In response to food intake first we stimulate acinar cells to produce trypsinogen, PSTI, andcalcium. The real-world production rates of these molecules are unknown, thatis why we decided to create them ourselves, but keep them proportional to each other. The levels of trypsinogen and PSTI molecules are assigned according to parameters predefined in the external control file. Suppose we synthesize100molecules of trypsinogen while synthesizing only 20

molecules of PSTI. We are not interested in the exact amount of secreted calcium, but in the rate it increases as it goes from the acinar cell to the lumen and from the lumen to the main duct. So on this level of the acinar cell we assign 1 to the calcium. We also introduce a random variability by applying the function that alters the molecule levels. Furthermore, we include parts that modify trypsinogen and PSTI production rates in order to give means to model effects of mutations.

Tg = (100 + random noise) * (trypsinogen production);

PSTI = (20 + random noise) * Inflammation^2 * (PSTI production);

Ca = (1 + random noise) *(1+(Inflammation-1)/2);

We modeled PSTI and calcium as inflammation-dependent variables. However, if we apply the same influence of inflammation on both PSTI and calcium, this effect will be canceled during further simulation, when we convert some trypsinogen into active trypsin. Since we believe that the main role of inflammation is to reduce damage, we modeled the PSTI responses to the increased inflammation exponentially, whilecalcium responseslinearly.

2.2.3.2 Duct cell stimulation

The next step is to stimulate duct cells to produce bicarbonate. We are not interested in exact amount of bicarbonate secreted, but in the influence it has on t he water. For the sake of simplicity we assign 1 to the bicarbonate level and add some random noise. We included bicarbonate in the water secretion in linear manner. So it starts lowering water secretion when bicarbonate value becomes less than 1:

BC = (1+random noise)

2.2.3.3 Activation of Trypsinogen

We modeled three levels of trypsinogen activation (inside the acinar cell, in the lumen, and in the main duct). For these three cases we created three separate functions that activate some portion of trypsinogen into trypsin and check whether the existing PSTI molecules are numerous enough to stop damage. The percentage (p) of trypsinogen activated in the acinar cell, the lumen, and the duct are specified in the external file with the rest of model parameters and they named as CA, LA, and DA respectively. We also applied a function that brings randomness to these percentages as well. For example, if CA=5.3%, after we introduce randomness it takes any value from 0 to 5.3% with a uniform distribution (In the equation below, we notate this as '(0<p<CA)'). We also have the function that makes sure that calcium influences trypsin activation only in the presence of inflammation[34, 35]. This function is called "cum" and takes two values inflammation and calcium respectively. It returns value 1 when there is no inflammation and inflammation times calcium when there is inflammation. Finally we included parts that modify trypsin activation in order to give the means to model effects of mutations. They are trypsinogen activation (Tg_a) and trypsin inactivation (T_i) respectively. The equations used to activate trypsin in the lumen and duct are identical, however they give different results. The reason of this difference is that before trypsin activation in the duct, we divided the trypsinogen, PSTI and calcium levels by the water level in order to mimic dilution.

We think of acinar cell as a system that has other means of resistance to damage from active trypsin besides PSTI. In order to model this resistance we first subtract PSTI from trypsinogen, then apply modifying factors for trypsin activation and check whether activated trypsin exceeds a certain threshold. If it exceeds the threshold we destroy the acinar cell, otherwise the acinar cell secretes the produced molecules to the lumen. In the lumen and the duct

we again activate some portion of trypsinogen and check whether existing PSTI molecules are enough to stop damage. So the equations we used in the simulation are listed below.

In the acinar cell:

T = (TG-PSTI) * (0<p<CA) * cum(Inflammation, Ca) * Tg_a / T_i

In the lumen:

T = TG * (0<p<LA) * cum(Inflammation, Ca) * Tg_a / T_i

In the duct

T = TG * (0<p<DA) * cum(Inflammation, Ca) * Tg_a / T_i

2.2.3.4 Pancreas growth and recovery

Experiments with mice and rates have shown that the pancreas has incredible power to regenerate. It was shown that pancreatic cells could go through temporary dedifferentiation, undergo proliferation and then again differentiate into the cells of their own,meaning that the acinar cells give rise to the pool of acinar cells and the duct cells give rise to the pool of duct cells[36]. In severe cases like partial pancreatectomy, it is the duct cells that are responsible for recovery of the whole pancreas[37]. They undergo through temporary dedifferentiation, proliferation, and differentiation, but this time they are not limited to a single cell type, but are able to supply all types of pancreatic cells. However, certain conditions like chronic inflammation or chronic and heavy alcohol exposure can severely limit the recovery capacity[9, 14]. In order to reserve the means to model these conditions and prevent immortality of the pancreas we limit the number of recoveries of the acinus, but not the recovery of the acinar cells. In the model it lo oks like this:every day of simulation we run a special function that checks two checkpoints: the number of the acinar cells in every acinus and the total number of living acini.

1. When the number of cells has reduced to less than half of the initial number, we increase the number of cells to normal (acinar cell recovery).
2. The second checkpoint checks whether the total number of acini is severely reduced. If it is reduced too much from its original value, we regenerate the pancreas by converting dead acini into living, but only once: an acinus is only revivedif it was never regenerated before. In the case of successful recovery we assign the inflammation status of the acinus equal to the general inflammation status and thesecond number of "status" vector takes value of 1, indicatingthat the recovery of the acinus has occurred. This indicator is made in order to prevent the"immortality" of the pancreas. It also leaves room to add activation of the sartoliny cells in the future to simulate fibrosis.

2.2.4 Stimulation flow chart

The stimulation of the model has three levels. First we stimulate the matrix, then the acinus, and finally the acinar cell. However collection of the digestive enzymes goes in the opposite direction. The acinar cell produces digestive enzymes to the acinus and then they are collected to the duct (matrix) (Figure 6). Once acinar cell is stimulated it produces three types of molecules, such as trypsinogen, PSTI, and calcium. In our model we associate the acinar cell with the function that creates a vector of six numbers. The first four numbers correspond to molecules that both acinar and duct cells produce, the last two numbers correspond to water and the inflammation variables respectively. In the diagram below we keep track of only five of the numbers, because they correspond to molecules that pass through the levels. The sixth number is kept in mind, because it corresponds to inflammation, which is a local variable. When we call the

function that stimulates acinar cells we use parameters that were defined from the beginning of the simulation. For all five molecules we also apply the function that add some random noise to each of the numbers. For example, let's assign the following number of molecules {TG, P, Ca, BC, W} ={100, 20, 1, 0, 0} for trypsinogen, PSTI, calcium, bicarbonate, and water respectively. The inflammation variable takes on a value equal to the inflammation of the acinus. In this example for simplicity we will skip the function that adds random noise. After we assign values for the molecules, we convert some of the trypsinogen into active trypsin and check whether there are enough existing PSTI molecules to block active trypsin and stop damage. If there are not enough PSTI molecules, then we assume that the acinar cell experienced severe damage and so destroy the cell. This will lead to an increase in local inflammation. In the case when number of PSTI molecules exceeds activated trypsin, we reduce the number of trypsinogen and PSTI molecules by thenumber of activated trypsin. Then this vector is added to another vector that corresponds to the lumen of the acinus. To continue our illustrative example (Figure 7), suppose that 5 trypsinogen molecules were converted into trypsin. Since we have more than enough PSTI molecules (20), we reduce both trypsinogen and PSTI molecules by 5. So after trypsinogen activation the vector will contain the following numbers {95,15,1,0,0}, which then goes to the lumen. Suppose that we are able to successfully stimulate the acinar cell n times. That would mean that the acinus will have n times more molecules than a single acinar cell can produce and the lumen vector will look like {95n,15n,n,0,0}. At this point we let the duct cell produce bicarbonate and temporarily assign 1 to the water, so the vector would look like {95n,15n,n,1,1}. Then we increase the level of water in order to mimic increased volume ofpancreatic juice due to the secretion of bicarbonate by the duct cells. Suppose water was increased 5.5 times. It would also mean that all other molecules were diluted 5.5 times and the final vector would look like

{17.27n, 2.73n, 0.18n, 0.18, 1}. T hen we change the calcium level in order to mimic difference in the calcium level between the acinar cell and the lumen. Suppose the calcium level increased 5 times, so the lumen vector would be as {17.27n, 2.73n, 0.9n, 0.18, 1}. A fter increasing both water and calcium levels we again convert some amount of trypsinogen into trypsin and check whether active trypsin was successfully inhibited. If the acinus is destroyed, we increase general inflammation that corresponds to inflammation of the whole pancreas. In the case of successful acinus stimulation we collect pancreatic juice into the duct. If the number of the successful acinus stimulations were"m", then the duct would have "m" times more molecules compared to the amount that a single acinus can produce and the duct vector would be as {17.27nm, 2.73nm, 0.9nm, 0.18m, m}. At the level of duct we again increase the level of calcium in order to mimic the difference in the calcium levels between the lumen and the main duct. Suppose we again increased calcium level 5 times, so duct vector would look like {17.27nm, 2.73nm, 4.5nm, 0.18m, m}. At this point we convert some amount of trypsinogen for the last time. Then we check whether there are enough remaining PSTI molecules to avoid damage. In the case of damage we increase the general inflammation. In case of successful stimulation of the matrix we simply clear the duct vector, because it has no further use. Note that inflammation is increased only once in each level of stimulation per day of simulation. This means that even if more than one acinar cell was destroyed, the local (acinus) inflammation would be increased only once at a single day of the simulation. A similar rule applies for general inflammation - it is increased only once for any number of destroyed acini. We made this simplification because at the current stage we don't know exactly how inflammation depends on the number of dead acinar cells or acini. That is why we trigger inflammation when at least one acinar cell or acinus dies and increase inflammation again the next day of simulation, if one or more acinar cells or acini die.

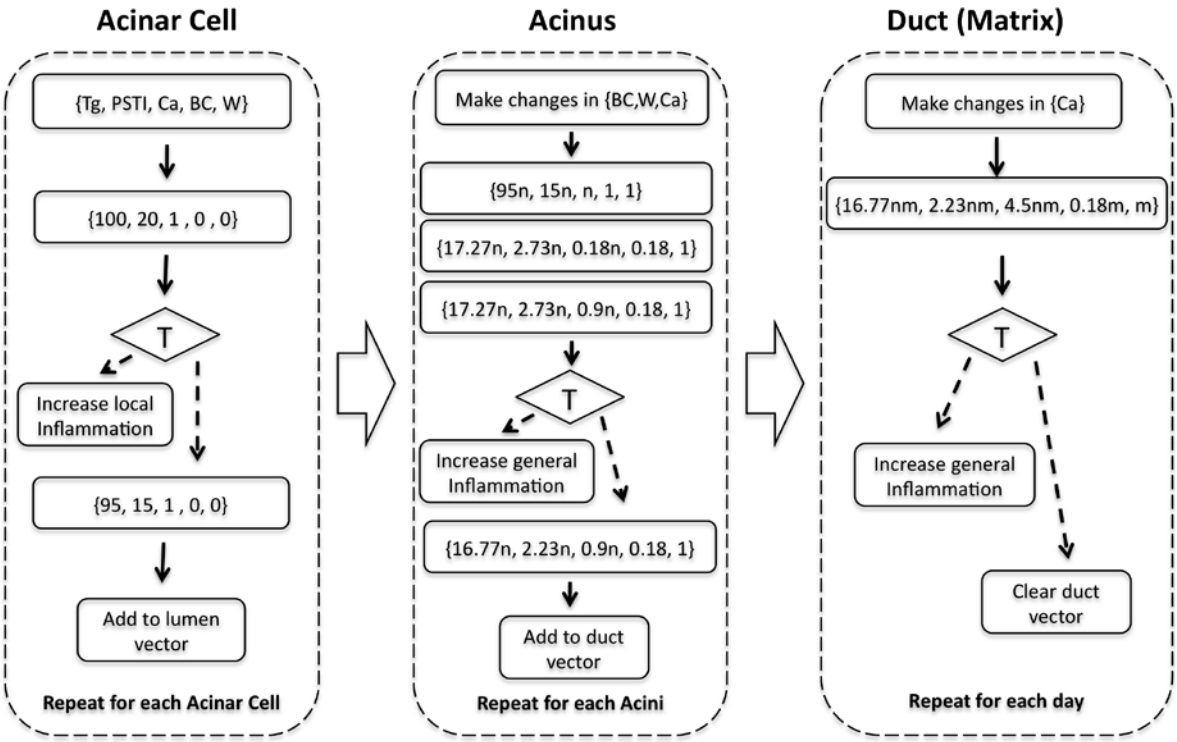

Figure 6.Simulation flowchart.

Tg – trypsinogen; PSTI - pancreatic secretory trypsin inhibitor; Ca – calcium; BC – bicarbonate; W – water; T – trypsin activation; n – number of successfully activated acinar cells; m – number of successfully activated acini;

2.2.5 Overall simulation flow chart

The overall simulation starts with specifying the parameters of the model (Figure 7). Parameters that are read from external file are listed in Table 2. After the program successfully readsthe parameters, it creates a matrix, where the number of rows corresponds to the number of main duct branches and the number of columns corresponds to the number of secondary duct branches. Each cell of the matrix is turned into an acinus and set for further stimulations. Then the whole matrix is synchronized for inflammation. In this step we make sure that none of acinus

(local) inflammations is less than pancreatic (general) inflammation. We thought that whenever local inflammation increases it may or may not affect general inflammation. But once general inflammation increases it always affects every local inflammation. We also check the size of the pancreas and apply recovery processes when either of the numbers of the acini or the acinar cells dramatically drops down. For each patient we create the pancreas that is full size and has no inflammation, that is why we skipped the synchronization and recovery for the first day of the simulation. After synchronization and recovery step, we actually stimulate the pancreas. The stimulation is passed from the pancreas to every living acinus and then to every living acinar cell. In each step of the stimulation damage can occur. Depending on a scale of the damage we increase either local or general inflammation. If general inflammation increases above certain threshold we call it as an acute pancreatitisattack. We record the day and number of the pancreatitis attack for future output. We follow patients for a maximum of three pancreatitis attacks. So after the third pancreatitis attack or patient's death we stop the simulation for the current patient. If the conditions to stop the simulation are not met, we continue the simulation by collecting pancreatic juice. In the case of no damage during the simulation we reduce both local and general inflammations. At this point we loop the simulation to the point when we synchronize pancreas for inflammation and performrecovery and continue repeating stimulations as many times as the number of days defined in the parameters. After we have run the simulation for the specified number of days or have disruptedthe simulation due to any other reason, we stop the simulation and repeat it for the next patient starting from creating a new pancreas (matrix).

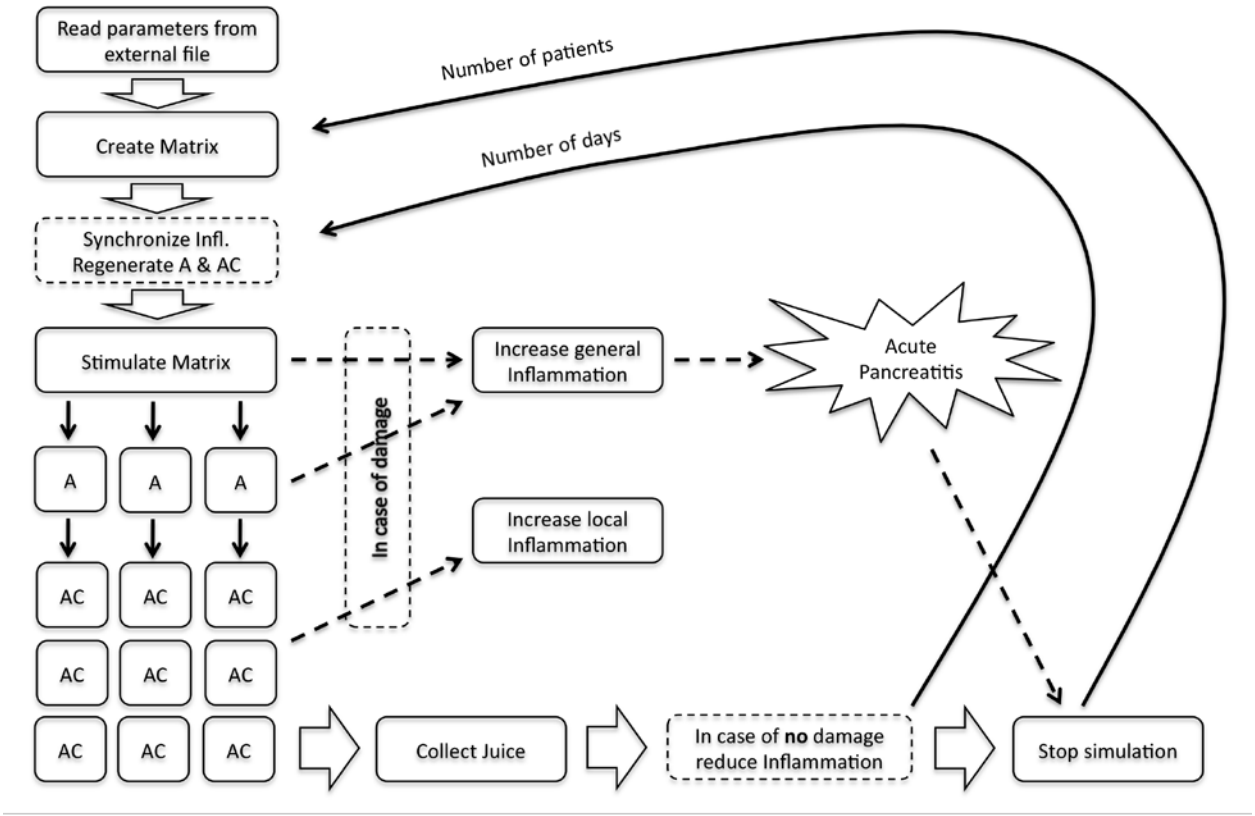

Figure 7.Overall simulation flowchart.

A – acini; AC – acinar cell;

2.2.6 Statistical analysis

In our model we simulated 100 patients. For each patient we recorded three time points that corresponded to the times of the first, the second and the third attacks of pancreatitis. For those patients who never had any attacks of pancreatitis, these time points were right censored. We used Kaplan-Meier estimator to estimate survival function of not developing the first, the second and the third attacks of pancreatitis respectively.Survival curves from different sets of the parameters were compared against the reference survival curve using the G-rho family of tests.

2.2.7 Software

The simulation model itself was written in C++ using Xcode Version 3.2.2. The simulations themselves were run on our Gattaca cluster. Output data were then processed using R 2.11.1.

3.0 RESULTS

3.1 WORKING PARAMETERS

In our simulations, we tested working parameters in a set of 100 patients with 11,000 days follow up. There was no difference in terms of the structure and size of the simulated pancreas itself. Each time the created matrix was 30x30, which represents a pancreas with 30 branches of the main duct and 30 branches of the secondary ducts. Every cell of the matrix represented a single acinus with 100 acinar cells.

Ideally when the model is built, the most appropriate way to test it is to turn on and off different parts of the model and see how such actions influence on the general behavior of the model. In the current work we useda stochastic approach that brings some degree of expected variability to the behavior of the model. Without knowing this variability it would be hard to explain changes in the model's behavior, whether they are due to modifications in the model or due to random chance. For this purpose we tried to find out the range within which each of the parameters can vary (Table 3). In order to test variability due to random chance we run our simulations with the same working parameters 33 times. The survival curve for the first pancreatitis attack from the first simulation was set as a reference survival curve. The rest of the survival curves from 32 additional simulations were then compared to the reference one. None of

the additional simulations resulted in survival curves that significantly differed from the reference one with an alpha level of 0.01.

The results in Table 3were obtained using the original code of the program (Appendix A). Later, the code of the program was revised according to the comments of the members of the defense committee. The modified program code can be found in the Appendix B; it was this revised code that was described in the previous sections. The changes that were made in the program code are explained in the following discussion section (4.3.1).

Using the original code in Appendix A, we started our simulation with the total amount of produced trypsinogen molecules as 100. Then to assess sensitivity to the initial parameter values, we ran a series of simulations with a grid of alternate values of produced trypsinogen molecules while keeping the rest of the parameters constant. We found that trypsinogen can change from 99.94 to 100.01 without causing a statistically significant difference in thesurvival curve of pancreatitis compared to that obtained with the initial set of the parameters. The comparisons of survival curves were made using an alpha level of 0.01. The same procedure was applied to the rest of the parameters. As suggested before we took a total amount of PSTI equal to 20% of total trypsinogen, which gives us 20 molecules of PSTI. Further simulations showed that this parameter could vary within a small range from 19.99 to 20.06 without causing statistically significant differences in the survival curve of the pancreatitis. The next parameter we tested is the rate at which calcium level increases while enzymes move from acinar cells to the lumen and then from the lumen to the main duct. We took the initial rate of this variable to be equal to 5. The initial value of calcium inside the acinar cell was taken as 1 molecule, then it increases 5 times in the lumen and again 5 times in the main duct. We have discovered that this rate was able to vary within a small range from 4.98 to 5.01 without causing any statistically

significant changes. We started with an initial bicarbonate level as 1 molecule, but found that the range within which the level of bicarbonate can range is rather large. We found that bicarbonate can be as low as 0.7 and as high as 1.25 and still produce survival curves almost identical to those obtained with the initial set of parameters, suggesting that the level of bicarbonate may go even lower and higher than we tested. The next parameter we tested was water production rate. Initially we took it to be equal to 5.5 and found that its lowest value (equal to 4.4), but not the highest value. Water level can go as high as 800 and yet doesnot cause any significant changes in the survival curve of the pancreatitis.

The set of parameters responsible for mutations that alter production rates of trypsinogen, PSTI, and water (through bicarbonate production) were initially set equal to 1. A series of simulations showed that trypsinogen production rate is very sensitive to any changes, even the slightest change as ± 0.001 resulted in significant difference from initial survival curve of the pancreatitis. Both trypsinogen activation and trypsin inactivation rates are also very sensitive; they range from .9995 to 1.0005. PSTI production rate can be as low as 0.9995, but higher than 1.003. The last parameter of mutations that corresponds to the CFTR mutation didnot show any strict borders - it could take values lower than 0.35 and higher than 1.06.

The next three parameters have been taken to represent the activation rate of the trypsinogen in the acinar cell (CA), the lumen (LA) and the duct (DA) with initial values 0.053, 0.03, and 0.03 (or 5.3%, 3%, and 3%) respectively. It seems that CA is the most sensitive among all three parameters. It can take value as low as 0.05294, but not greater than 0.05301, which means that even a small increase of this value by 1e-5 results in significant difference in the survival curve of the pancreatitis. LA is less sensitive to changes and can vary by ± 2e-4. DA is the least sensitive to changes. It can be lower than 0.024 and higher than 0.036. The next

parameter is MR, the range within which such parameters like trypsinogen, PSTI, calcium, and bicarbonate can vary. Initially it was taken to be equal to 0.1 (or 10%) and it could take values as low as 0.0996, but not greater than 0.1001. The final parameter that was tested is Thr, the threshold for active trypsin that can result in the acinar cell death. Initially this value was set to 4.832, and it can vary from 4.8308 to 4.8359.

Table 3. Working parameters.

Parameter	Symbol	Value	Range
Number of patients	NP	100	-
Number of days	ND	11000	-
Number of main ducts	D1	30	-
Number of secondary ducts	D2	30	-
Number of acinar cells	D3	100	-
Trypsinogen	TG	100	99.94-100.01
PSTI	P	20	19.99-20.06
Calcium	Ca	5	4.98-5.01
Bicarbonate	BC	1	<0.7->1.25
Water	W	5.5	4.44->800
Trypsinogen production	Tg_p	1	>0.999-<1.001
Trypsinogen activation	Tg_a	1	0.9995-1.0005
Trypsin inactivation	T_i	1	>0.9995-1.0005
PSTI production	P_p	1	0.9995->1.003
CFTR mutation	CF	1	<0.35->1.06
Trypsin activation in the cell	CA	0.053	0.05294-<0.05301
Trypsin activation in the lumen	LA	0.03	0.0298-0.0302
Trypsin activation in the duct	DA	0.03	<0.024->0.036
Molecule range	MR	0.1	0.0996-<0.1001
Threshold for inflammation	Thr	4.832	4.8308-4.8359

4.0 DISCUSSION

4.1 WORKING PARAMETERS

After we run our simulations and tested working parameters, we found several points that we would like to change. We made few modifications in the program code in order to eliminate these questions. These modifications didn't change overall results of the simulations and are explained more detailed in the discussion part (4.3.1). In this work we present working parameters and their discussion using original program code of the model (Appendix A).

The first step of testing our simulation model was to figure out working parameters (i.e.,are theparameters that give us a stable survival curve of the pancreatitis). Initially the idea was to adjust these parameters to existing statistical data of pancreatitis in the general population. But this task seemed to be very tricky. Two issues that we faced in our simulation werethe time required to run the simulation and lack of statistical data about the pancreatitis in the general population.

Our mathematical model was designed such that you need to specify the number of patients and the total amount of days you want to follow each patient. The model simulates one patient at a time for the specified numbers of days or until three pancreatitis attacks occur and then repeats the same simulation for the next patient with the same parameters. The size of the matrix we created for the pancreas simulation was also important. Depending on the number of

the acini and the acinar cells that a single acinus contains, the time required for simulation changed dramatically. The "bigger" the pancreas (i.e., the larger the matrix), the more time the simulation took. In order to save time and still be able to keep the modeling parameters large enough to represent an entire population we decided to simulate 100 patients for 11,000 days (~30 years) follow up. We also agreed to use a matrix size equal to 30x30, meaning 30 main duct branches and 30 secondary duct branches or 900 acini in total. Each acinus had 100 acinar cells, for a total of 90,000 acinar cells in the pancreas. Even with these modest restrictions,a powerful computer resource, called "The Gattaca Cluster", took about 10 hours to make one simulation. Due to the architecture of The Gattaca Cluster we were able to submit the model multiple times simultaneously using one processor per a job.We submitted the model 13 times simultaneously with different sets of parameters. A single run of the model takes about 10 hours, by submitting the job in parallel we were able to run the model 13 timesand spendonly time required for one run.

The availability of existing descriptive data was our second main concern. Since our primary outcome was a measure of the survival function of the pancreatitis in the population, our best choice was to apply statistical methods for survival analysis to our output data from the simulation. In order to fit our model to existing data we needed to find statistics for incidence of pancreatitis from a cohort study with as long a follow up as possible, to get a measure of how frequent a second or third attack is in the same patient. When we were first designing this model we couldnot find any data that matched our criteria, though we continue to search for suitable data sets, and hope to get usable data from Dr. Whitcomb's North American Pancreatitis Study (NAPS2). Since this is a preliminary model of pancreatitis we agreed to take one set of parameters and use the resulting survival curve from this set as a reference survival curve. We

didn't expect our reference survival curve to be an exact match to any existing data, but to have curves for the three attacks of pancreatitis located in the middle of the graph and spread out enough from each other so that variations between them can be observed (Figure 6). If we take too small an incidence rate for pancreatitis, we would probably have had to simulate hundreds of thousands of patients in order to collect enough patients with even a first attack of pancreatitis, since the incidence rate for acute pancreatitis is 17 per 100,000 people. There also will be limited space if we try modeling protective effects of some of the parameters.

4.2 SENSITIVITY OF THE PARAMETERS

Once we achieved a desired survival curve, we used the resulting set of parameters as a reference for further simulations. All further simulations were obtained using the original program code. In order to test variability due to random chance we run our simulations with the same working parameters 32 additional times. Obtained survival curves were then compared to the reference one. None of the additional simulations resulted in survival curves that significantly differed from the reference one with an alpha level of 0.01.

Once we settled on working parameters, we tried to find the range within which the parameters can change and yet not cause statistical deviations from the reference curve. We made a series of simulations where we changed a single parameter at a time, while keeping the rest of the parameters constant. Changes in most of the parameters had a large effect on the resulting survival curves, suggesting the simulation was highly dependent on the values we

picked for those variables. Some of the ranges in which changes did not have an effect were not symmetrical when compared against the reference values. That is, the lower and upper boundaries were not equidistance from the reference (initial) value. I presume this kind of asymmetry can be explained by the variability of the model itself. For the same reason we couldnot find exact boundaries of some parameters. For example, trypsinogen production had an initial value equal to 1 a nd even slightest changes (such as ± 0.001) resulted in significant departures from the reference survival curves.

The opposite situation was seen with three parameters: bicarbonate, water, and CFTR mutations. They could vary significantly without causing any significant difference in the survival curve. We couldnot even find boundary values for these parameters. It didnot matter how much we changed them, they seemed to have no effect on the outcome. The explanation of tolerance to such variability lies in the way we used water in the model.It happened because activated trypsin was reverse proportional to water. The more water we had, the less trypsin had been activated.

4.3 FUTURE DEVELOPMENT

4.3.1 Modification applied to the program code

In the initial version of the program code (Appendix A) we had divided the molecules in the lumen to the number of successfully worked acinar cells. This division was made in order to average produced molecules in every acinus. In the revised version of the program code (Appendix B) we did not apply this averaging. The revised version of the program gave the same

results as original version and the reason why removing of the averaging did notalter results because it did not change proportion between trypsinogen and PSTI in the lumen, but added one extra operation in the calculations.

Initially we modeled that the acinar cell could produce chloride that had influencedthe water production through bicarbonate in the lumen. We simplified bicarbonate production and made the water production directly dependent on chloride. However after revising this concept we decided to rename chloride as bicarbonate and move bicarbonate production out of the acinar cell function to the duct cell function. This change in the program code also didnot alter the outcome of the simulation. When chloride was produced by the acinar cell, it was then averaged in the lumen, which means that the variability of chloride was narrowed down. In the revised version of the program code we produce bicarbonate only once in the lumen, so the variability of bicarbonate remained greater than if it was produced by the acinar cell and then averaged in the lumen. If we remember that water could vary within a very big range without causing any departures in the outcome of the simulation, we think that the water variability is enough to compensate increased bicarbonate variability.

A final modification was made in the function that activates some portion of trypsinogen in the main duct. We used following formula to activate trypsinogen:

T = TG * (0<p<DA) * cum(Inflammation, Ca) * Tg_a / (T_i*Water)

and then compared activated trypsin (T) with existing PSTI molecules. In this formula we divided trypsin to water to mimic dilution of the pancreatic juice in the main duct. We assumed that water variable should appear in the above formula of trypsin activation twice. It should come with trypsinogen and calcium variables because both of then supposed to be divided to water in the main duct. In this case we also shoulddivide PSTI to water before comparing it to activated

trypsin. But when trypsin was compared to PSTI, one division to water could be canceled leaving only one division to water in the formula of trypsin activation. However this assumption turned out to be wrong because we forgot that the "cum" function returns 1 in the case of no inflammation and inflammation times calcium in the case of any inflammation. This means that the water variableis not canceled out as originally assumed. So in the revised version of the program code we dilute the pancreatic juice before activating trypsin in the main duct. As we expected this change in the program code did not alter working parameters, except those parametersthat describe trypsin activation, they are CA, LA, and DA. After testing a r evised version of the model, we noticed that the amount of trypsinogen activated in the acinar cell (CA) appears to be the most important parameter among these three parameters mentioned above. With CA equal to 0.052566(± 2e-6) we could obtain a survival curve similar to the reference survival curve. The value of other two parameters (LA and DA) remained at the same level as in the initial version of the model (0.03 and 0.03 respectively).

4.3.2 The curve becomes stable after 500th week

We have never observed a change in the survival curve after the 500th week (3500 days) of simulation. This is also true for the rate of second and third attacks of pancreatitis. This behavior of the resulting survival curves remained the same even after we modified the programming code. One could say that there is no need to run the simulation for 11,000 days and wait 10 hours when you can run the simulation only 3,500 days and get the same results. More importantly that could reduce the time required to run simulation. This might be true at the moment, but what is the real reason behind this behavior? It is definitely in the way the model was designed. Debugging this question would require following changes of the model's parameters day by day

and analyzing their behavior. For this purpose we modified the program, such that in the new version we get an output of a set of parameters (general inflammation, total number of acini and acinar cells) each day so we could follow their changes in real time. This has not yet revealed the cause of the lack of changes after 500 weeks, but we continue to investigate the issue.

4.3.3 Parallelize simulation

A complete run of the model that includes a single set of parameters and 100 patients takes roughly 10 hours. The modifications we made to the program code were not directed to solve this time issue, so remains in the revised program code. Due to the fact that all cases of pancreatitis occurred in the first third of the simulated time, we could save time by running the model only 3,500 days instead of 11,000 days. However when we fix this behavior, we will again face a timing problem and it will become even more important if we want to follow our patients for an entire lifetime (for example 50-60 years or longer). The main problem is that the program does relatively small calculations and repeats them an enormous number of times. This leads to overloading of the CPU, while using very little RAM. One possible solution would be to rewrite portions of the code to implement parallelization. Currently we used a powerful computer resource, called the Gattaca cluster, to run our simulation. We were able to submit our simulation multiple times, where each simulation was run independently from others. Thus, we could submit 13 simulations and have them all finish in the time required for one simulation, which is about 10 hours. In order to save computational time and allow the simulation to be more effective at using resources within a single computer (such as modern multi-core processors), we need to break the computations within a single simulation itself into individual, independent components. OpenMP[38] or pthreads[39] libraries could be used for this purpose. The basic idea is

to define a point where the main flow of operation could be divided into several sub operations, which then could be run simultaneously. After a successful completion of all sub operations the results from each of them are synchronized into one and then the main flow of operation continues. In my opinion the best choice would be to divide simulation among patients. We designed the model such that the whole simulation is repeated for each patient. Basically we looped the main core of the model for all patients and this loop is located at the very beginning of the program. In order to quickly parallelize this calculation, we could define 10 groups of 10 patients each and run all groups simultaneously. This would require some changes to the loop itself, but not within it. An alternative solution would be to deal with the matrix that corresponds to the pancreas. We could separate tasks by columns or by rows. The idea remains the same, but it would require making many more changes within the program.

4.3.4 Switch to deterministic model

We built the general framework of the model such that each part of the model operates separately. Each step used a stochastic approach to produce some results that then were used in the next step as an input. This approach is good for a pilot study where one wants to test the general flow of the model and test whether it works in a desired sequence. In order to make this model more realistic so that it could be able to model causes of pancreatitis and then be used for more accurate predictions we would like reduce the usage of the stochastic approach to an acceptable degree and apply more of a deterministic approach to the model. Due to the design of our model we can work on every part of it individually. We can even replace some parts with already existing mathematical models. For example, there is an already existing mathematical model of the pancreatic duct cells suggested by Whitcomb and Ermentrout (2004)[40] that would

be suitable for this purpose. The Whitcomb/Ermentrout model explains the basis of generating high bicarbonate concentrations in pancreatic juice. Most importantly it gives insight into the workings of CFTR and some possible outcomes of CFTR mutations. We believe that with few adjustments we could successfully incorporate this mathematical model into our pancreatic model. By adopting this mathematical model we would be able to focus our efforts more on modeling influences of bicarbonate rich fluid on trypsinogen activation.

It is worth mentioning that the accuracy of any mathematical model is directly dependent on our current understanding of the biology of the processes we try to model. I doubt that we fully understand the whole complexity of these processes, which means that the equations we use to model biological processes are not as good as we would like them to be. We still have much room for improvement.

4.3.5 Include more factors

As I mentioned before our model is considered to be a framework, each part of which needs to be improved. We would like to concentrate our main effort on modeling the acinar cell. Currently the acinar cell, in our model, is modeled using random number generator with only one stimulus and it only produces molecules such as trypsinogen and PSTI. I think that we need to add another stimulus such as food intake. Surely the amount and quality of food we eat influences pancreas function, but indirectly through hormones and other signals. Our model is not mature enough to capture these transitional signaling pathways. One way that we would like to improve our model is to allow it to capture all the complexity of interaction between signals such as acetylcholine (ACh) of vagal nerve stimulation and cholecystokinin (CCK) and secretin of duodenum.

One of the main changes we need to do in order to make the model more comparable to the real data is to assign more appropriate units to all substrates of the model. Currently we assigned number of molecules to trypsinogen, trypsin, PSTI, calcium and bicarbonate; however real data deal with concentrations rather than molecules. In this work we tried to model the pancreas in general, which is why we thought that as long we keep amount of substrates proportional to each other it would suffice to count number of the molecules in the model. However, I think that in order to make results of this model more compatible to the real data it would be reasonable to use concentrations of above substrates instead of number of the molecules whenwe further developthe model.

In our model, inappropriate calcium levels directly influences trypsinogen activation, but the biological role of calcium is not limited to this one function. In intracellular environments, calcium plays the role of mediator for both acetylcholine and cholecystokinin signaling. It would be very important to us to include this function of calcium as well.

In the current model we apply one signal for stimulation and responses from different acini are the same. However it might be interesting in the future to make stimulation of different parts of pancreas proportional to the level of stimulation. This will become important to model feedback regulation and hyperstimulation.

Probably it would be a good idea to model other functions of duct cell. The duct not only produces fluid and reduces the calcium concentration in the lumen, but it maintains a high pH, which makes trypsin inactive until the pH normalizes. Currently we have variables that describe water and calcium, but not pH. It would be a very good improvement to the model if we could model influence of pH as well.

We need to increase the number of possible statuses of the acinus. These statusesshould be living or dead, with living divided into resting or secreting, and the state being either naive or stress-response (e.g., making PSTI).Please recall thatPSTI is the product of the SPINK1 gene and is normally not synthesized. They are only made by the acinar cells during a stress response.

In order to model disease we need to include other cell types such as islet cells and inflammatory cells including stellate cells, macrophages and other leukocytes.

We need to modify inflammation to be able to distinguish between acute and chronic inflammations. It would be very useful for modeling either acute or chronic pancreatitis. And we also need to modify the causes of injury. We should consider the possibility that injury can occur through mechanisms different than trypsinogen activation. Also, injury and inflammation is not the sameand loss of the acinar cell mass can occur through many pathways.

We may need to include other environmental effects like alcohol or smoking. Environmental factors play a major role in clinical pancreatitis. By including them in the model we would have means to accurately evaluate their effects on survival function of the pancreatitis.

Finally our model lacks the memory that passes from day to day in the simulation. Currently we record the total number of acini, the number of the acinar cells in each acinus and both local and general inflammations. However in the case of successful implementation of the improvements listed above we would have records of much more variables that pass from day to day in the simulation. We think that increased number of variables that can be followed during the simulation will dramatically improve chances of better understanding the pancreas and related risk of the pancreatitis.

4.4 WHAT IS THE APPLICATION?

Based on our plans we will reduce randomness of the model by switching to deterministic modeling and include more factors like different stimuli. This will give us a good opportunity to provide insight into pancreatitis in terms of production and activation of trypsinogen. By altering different aspects of the model we will be able to mimic the effects of mutations that disrupt the normal work of the pancreas. Currently we divide mutations into 5 groups, those that have an effect on trypsinogen production, trypsinogen activation, PSTI production, trypsin inactivation, and bicarbonate secretion (CFTR mutations). We treat mutations as changes in these rate constants, and included them to the model by simply multiplying the respective rate by a constant. This is a very straightforward way of adding mutational effects to the model. Once our model becomes mature we will be able to interpret the effects of these mutations more accurately, and perhaps even predict expected mutations in individuals based on their clinical course.

The prediction of risk of developing pancreatitis due to these mutations is main goal of our work. In our current work, we tested the sensitivity of the model due to changes of the working parameters including variables that represent mutations. We continue the work on developing the model and we believe that this model, once fully developed, will have the power to make predictions of the risk of developing pancreatitis for individuals from the general population and for those who inherited disease-related mutations.

APPENDIX A: ORIGINAL CODE

```
#include <string>
#include <iostream>
#include <fstream>
#include <vector>
#include <math.h>
#include <time.h>
#include </Developer/Headers/Matrix.h>
#include </Developer/Headers/MatrixIO.h>

usingnamespacestd;

//We store all parameters in a single vector. To make things easier we gave unique names to digits from 0 to 23.
//Then we use the names as indexes whenever we want to refer to the parameter vector.
//Here is interpretation of names we used.
//0     Tg – Trypsinogen
//1     P – PSTI
//2     Ca - Calcium
//3     BC - Bicarbonate
//4     W – Water
//5     I - Inflammation
//6     D1 – Number of branches in main duct
//7     D2 – Number of branches in secondary duct
//8     D3 – Number of acinar cells in the acinus
//9     M – Food intake, meal variable (reserved variable for future use)
//10    Tg_p – Trypsinogen production
//11    Tg_a – Trypsynigen activation
//12    T_i – Trypsin inactivation
//13    P_p – PSTI production
//14    CF – CFTR mutation
//15    CA - Trypsin activation in the cell (it is different from Ca, because C++ is case sensitive)
//16    LA - Trypsin activation in the lumen
//17    DA - Trypsin activation in the duct
//18    MR – Molecule range
//19    Thr – Threshold for inflammation
//20    NP – Number of patients
//21    ND – Munber of days
//22    NA – Total number of acini
//23    NC – Total number of acinar cells

enummol_code1 {Tg=0, P=1, Ca=2, Cl=3, W=4, I=5, D1=6, D2=7, D3=8, M=9, Tg_p=10, Tg_a=11, T_i=12, P_p=13, CF=14, CA=15, LA=16,
DA=17, MR=18, Thr=19, NP=20, ND=21, NA=22, NC=23};



//This is a function that adds some random error.
//x1 - is the number we want to change.
//x2 - is a range from which error is drawn.
//used when TG, PSTI, Ca, and BC are assigned in the cells.

double err(double x1, double x2){
        int n1, n2;
        double n3;
        bool b=1;
```

```
        while (b){
            n1=rand()%101-50;
            n2=rand()%101;
            if (101-abs(2*n1)>n2) b=0;
        }
        n3=x1+x1*x2*n1/50;
        return n3;
}


//This function returns randomly chosen numbers from 0 to 1 with uniform distribution. The numbers are rounded to second digit
//after point. Used in functions that activate trypsinogen in cell, lumen and duct.

double perc(double x){
        double n=x*(rand()%101)/100;
        return n;
}


//This function is used in trypsinogen activation. It makes sure that calcium has no effect
//on the trypsinogen activation without inflammation.

double cum(double infl, double calcium){
        if (infl==1) return1;
        elsereturn infl*calcium;
}


//This function creates a vector filled with numbers that correspond to molecules produced by
//acinar cell. It uses two vectors that store model parameters and lumen parameters.

vector<double> stimulate_A(vector<double>& vl, vector<double>& vp){
        vector<double> temp (6);
        temp[Tg] = err(vp[Tg],vp[MR])*vp[Tg_p];
        temp[P]      = err(vp[P],vp[MR])*pow(vl[I],2)*vp[P_p];
        temp[Ca] = err(1,vp[MR])*(1+(vl[I]-1)/2);
        temp[Cl] = err(1,vp[MR]);
        temp[W] = 1;
        temp[I] = vl[I];
        return temp;
}


//This function activates some portion of trypsinogen in the acinar cell. Then it checks
//whether existing PSTI molecules are enough to stop active trypsin from causing damage.
//If existing PSTI molecules are not enough, acinar cell is destroyed.

bool check_T_cell(vector<double>& vt, vector<double>& vp){
        bool b=0;
        double p=perc(vp[CA]);
        double temp=(vt[Tg]-vt[P])*p*cum(vt[I],vt[Ca])*vp[Tg_a]/vp[T_i];
        if (temp<vp[Thr]) {vt[Tg]*=p; vt[P]*=p; b=1;}
        return b;
}


//the same idea, but for whole acinus

bool check_T_lumen(vector<double>& vl, vector<double>& vp){
        bool b=0;
        double p=perc(vp[LA]);
        double temp=vl[Tg]*p*cum(vl[I],vl[Ca])*vp[Tg_a]/vp[T_i];
        if (temp<=vl[P]) {vl[Tg]-=temp; vl[P]-=temp; b=1;}
        return b;
```

```
}


//the same idea, but for whole pancreas

bool check_T_duct(vector<double>& vd, vector<double>& vp){
        bool b=0;
        double p=perc(vp[DA]);
        double temp=vd[Tg]*p*cum(vd[I],vd[Ca])*vp[Tg_a]/(vp[T_i]*vp[W]);
        if (temp<=vd[P]) {vd[Tg]-=temp; vd[P]-=temp; b=1;}
        return b;
}


//Once this function is called it increases inflammation to 0.5, but don't allow inflammation exceed 2 (its maximum level)

void increase_I(double& infl){
        if (1<=infl and infl<=1.5) infl+=0.5;
        elseif (1.5<infl and infl<=2) infl=2;
        elsethrow1;
}



//This function designed to reduce inflammation each time it called, but don't allow inflammation be lower than 1 (its minimum level)

void reduce_I(double& infl){
        if (1<=infl and infl<1.05) infl=1;
        elseif (1.05<=infl and infl<=2) infl-=0.05;
        elsethrow2;
}



//*********************
//***                           ***
//***    Acinus                 ***
//***                           ***
//*********************

class A{
public:

        //This function returns first or second number stored in the "status" vector.
        //Because these two numbers take only 0 or 1, we can also interpreted them as Boolean.

        bool get_status(int index){
           bool b;
           if (index==0or index==1) b=status[index];
           elsethrow3;
           return b;
        }



        //This function returns 3rd number of the "status" vector,
        //that corresponds to number of active acinar cells in the acinus.

        int get_cells(){returnstatus[2];}



        //This function designed to reduce inflammation each time it called, but for acinus.

        void reduce_acinus_I(){
           if (1<=lum[I] andlum[I]<1.05) lum[I]=1;
           elseif (1.05<=lum[I] andlum[I]<=2) lum[I]-=0.05;
           elsethrow4;
```

```
}


//This function generates acinus or simply fills up both vectors of class A (acinus).
void gen_acinus(vector<double>& vp){
   status.push_back(1);          //Means that acinus is working
   status.push_back(0);          //Means that acinus was never destroyed
   status.push_back(vp[D3]);     //Number of active cells in the acinus

   //Fills first four numbers of vector, that correspont Tg, P, Ca, Cl respectively.
   for(int i=0; i<4; i++) lum.push_back(0);
   lum.push_back(1);                        //water
   lum.push_back(vp[I]);                    //inflammation that takes value from general inflammation.
}



//The same as previous function, but this time second number of vector "status"
//takes value of 1, meaning that this particular acinus was destroyed once.

void regen_acinus(vector<double>& vp){
   status.clear();
   status.push_back(1);
   status.push_back(1);
   status.push_back(vp[D3]);

   lum.clear();
   for(int i=0; i<4; i++) {lum.push_back(0);}
   lum.push_back(1);
   lum.push_back(vp[I]);
}



//Main function of class A (acinus)
bool stimulate_acinus(vector<double>& vp){
   bool li=0;              //Indicator whether damage happened in the lumen
   vector<double> vtemp;   //A vector to store temporary numbers
   int dead=0;             //Number of dead acing cells


   //This loop designed to call function that stimulate acinar cell the same
   //times as number of active acinar cells in the acinus. It also checks
   //whether something happened in the acinar cell itself.
   for (int i=0; i<status[2]; i++){
            vtemp = stimulate_A(lum, vp);
            if (check_T_cell(vtemp, vp))
                     for (int i=0; i<4; i++)
                              lum[i]+=vtemp[i];
            else dead++;
            vtemp.clear();
   }

   //Increases inflammation if any acinar cell died
   if (dead>0) increase_I(lum[I]);

   //There are two conditions when acinus supposed to be destroyed:
   //when all acinar cells in the acinus die or activated trypsin exceeds existing PSTI.
   //In other cases acinus continues to work, but number of active acinar cells is reduced
   //to number of acinar cells that died during simulation.

   if (dead<status[2]){
            status[2]-=dead;

            for (int i=0; i<4; i++) lum[i]/=status[2]; //average molecules in the pancreatic juice

            //This step was taken to show that Ca level in the lumen are
            //increased compared to level inside acinar cell.
```

```
                lum[Ca]*=vp[Ca];

                //Checks whether existing PSTI molecules enough to stop activation
                //of trypsinogen in the lumen.
                if (check_T_lumen(lum, vp)){
                        lum[W]=vp[W]*lum[Cl]*vp[CF];                    //Increase level of water
                        li=1;
                }
                elsestatus[0]=0;
        }
        elsestatus[0]=0;

        //If nothing happens with acinus or with any of it's acing cells,
        //the inflammation is reduced.
        if (status[0] and dead==0) reduce_acinus_I();

        dead=0;
        return li;
    }



    //This function helps to collect all molecules from lumen to duct.
    //Or simply add corresponding numbers to a vector that represents duct.
    void collect(vector<double>& vd){
        for (int i=0; i<5; i++) {vd[i]+=lum[i];}
        //Reset vector, so we can use it again for next acinus.
        lum[Tg]=lum[P]=lum[Ca]=lum[Cl]=0; lum[W]=1;
    }



    //Synchronize inflammation from entire pancreas to particular acinus.
    //Makes sure that local inflammation not less than general inflammation.
    void sync_acinus(vector<double>& vp){
        if (vp[I]>lum[I]) lum[I]=vp[I];
    }



    //Recover number of acinar cells if it dramatically drops down.
    void regen_cells(vector<double>& vp){
        if (status[2]<=vp[D3]/2){
                status[2]=vp[D3];
                lum[I]=vp[I];
        }
    }

private:
    //This vector stores three numbers: 1st two numbers correspond to status of acinus itself
    //and indicator of whether acinus was destroyed or not respectively. These two numbers
    //takes only two values (0 and 1) and can be interpreted as Boolean. 3rd number corresponds
    //to number of active acinar cells in the acinus.

    vector<int>status;

    //This vector contains six numbers. Their position as following: trypsinogen, PSTI,
    //calcium, chloride, water, and inflammation of the acinus.

    vector<double>lum;
};



//*********************
//***                 ***
//***   Matrix        ***
//***                 ***
//*********************
```

```
typedef Numeric_lib::Matrix<A,2> Matrix;


//This function creates a matrix with defined numbers of rows and columns.
//It also fills each cell of the matrix with empty acinus.

void gen_matrix(Matrix& f, vector<double>& vp){
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                        f(i,j).gen_acinus(vp);
}


//When called this function finds acini and renew them.
void regen_matrix(Matrix& f, vector<double>& vp){
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                        if (!f(i,j).get_status(0) and !f(i,j).get_status(1)) f(i,j).regen_acinus(vp);
}



//This function starts whole simulation of the matrix (pancreas).
vector<double> stimulate_matrix(Matrix& f, vector<double>& vp, bool& di){

        //This vector is for temporary use.
        //It has 5 empty numbers plus 6th one, equal to general inflammation.
        vector<double> vtemp (5);
        vtemp.push_back(vp[I]);
        int count=0;                            //Counts how many acini worked in one simulation
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                        if (f(i,j).get_status(0)){              //Check whether acini is ready to work
                                f(i,j).regen_cells(vp);//Fills up acinus with acinar cells if needed
                                f(i,j).sync_acinus(vp);//Synchronize inflammation along whole pancreas

                        //Stimulate acinus and then collect produced molecules (juice) to main duct
                                if (f(i,j).stimulate_acinus(vp)) {f(i,j).collect(vtemp); count++;}
                                else {di=1;}            //Indicates whether something went wrong.
                        }
        //average each type of collected molecules in the main duct
        //then further increase Ca level
        for(int i=0; i<5; i++) vtemp[i]/=count;
        vtemp[Ca]*=vp[Ca];
        return vtemp;
}



//Report total number of active acini and acinar cells it returns FALSE when everything is OK
//and TRUE when number of acini is too small. In other words this function indicates whether
// recoveryof panaceas is required
bool report(Matrix& f, vector<double>& vp){
        bool r=0;
        vp[NA]=vp[NC]=0;
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                        if (f(i,j).get_status(0)){
                                vp[NA]++;
                                vp[NC]+=f(i,j).get_cells();
                        }
        if (vp[NA]<(vp[D1]*vp[D2]/10)) r=1;
        return r;
}



//*****************************
```

```
//***                                   ***
//***    Main Function                  ***
//***                                   ***
//*****************************

int main (){
try{
        srand(time(NULL));              //Turn on random number generator
        vector<double> param, param2;   //Used to store parameters from external file
        vector<int> days;               //Used to catch days when pancreatitis attacks occurred
        doublevalue;                            //Temporary number


        while (!(cin>>ws).eof()){
           cin>>value;
           param.push_back(value);
        }

        //The vector that store parameters is constantly modified during the simulation
        //to keep initial set of parameters intact we copy it to second vector.

        param2=param;


        cout<<"ID\tTime\tStatus\tGroup\n";
        for(int n=1; n<=param[NP]; n++){//Loop whole simulation for number of patients
           bool AP=0, Di=0;                     //Indicators of pancreatitis and duct inflammation
           days.clear();

           Matrix frame (param[D1], param[D2]);         //Create matrix
           gen_matrix(frame, param);                    //Fill matrix
           vector<double> juice;                                //Temporary vector

           for (int i=0; i<param[ND]; i++){             //Loop simulation for number of days
                   if (report(frame, param))regen_matrix(frame, param);         //Regenerate acini if needed
                   juice=stimulate_matrix(frame, param, Di);//Collect juice (produced molecules from all acini)

                   if (Di) increase_I(param[I]);                        //Increase general inflammation in response to acinus death
                   if (check_T_duct(juice, param)) {                    //This part activates some trypsinogen and checks if PSTI is
enough to avoid dammage
                           if (!Di)
                                   reduce_I(param[I]);          //This part is a little tricky due to complexity of conditions
whether general inflammation is needed to be reduced or increased
                           reduce_I(param[I]);
                   }
                   elseif (!Di) increase_I(param[I]);
                   Di=0;                                                        //Reset Di

                   //These set of conditions are designed to catch time when pancreatitis
                   //attacks were occurred and stop simulation after 3rd attack or total
                   //number of acinar cells is too small (the last condition was never true)
                   if (param[I]>1.5and !AP) {days.push_back(i/30); AP=1;}
                   if (param[I]==1) AP=0;
                   if ((param[NC]<100) || (days.size()==3)) {
                           days.push_back(0);
                           i=param[ND];
                   }
           }


           //These set of conditions help to make output looks nice
           if (days.size()>0) {cout<<n <<"\t"<<days[0] <<"\t1\t1\n";}
           else {cout<<n <<"\t"<<ceil(param[ND]/7) <<"\t0\t1\n";}

           if (days.size()>1) {cout<<n <<"\t"<<days[1] <<"\t1\t2\n";}
           else {cout<<n <<"\t"<<ceil(param[ND]/7) <<"\t0\t2\n";}

           if (days.size()>2) {cout<<n <<"\t"<<days[2] <<"\t1\t3\n";}
           else {cout<<n <<"\t"<<ceil(param[ND]/7) <<"\t0\t3\n";}
```

```
            param=param2;                    //reset parameters and use them again for next patient
        }
        return0;
}
catch (int e) {
        cerr<<"\nError in "<<e <<endl;
}
}
```

APPENDIX B: REVISED CODE

```
#include <string>
#include <iostream>
#include <fstream>
#include <vector>
#include <math.h>
#include <time.h>
#include </Developer/Headers/Matrix.h>
#include </Developer/Headers/MatrixIO.h>

usingnamespacestd;

//We store all parameters in a single vector. To make things easier we gave unique names to digits from 0 to 23.
//Then we use the names as indexes whenever we want to refer to the parameter vector.
//Here is interpretation of names we used.
//0     Tg – Trypsinogen
//1     P – PSTI
//2     Ca - Calcium
//3     BC - Bicarbonate
//4     W – Water
//5     I - Inflammation
//6     D1 – Number of branches in main duct
//7     D2 – Number of branches in secondary duct
//8     D3 – Number of acinar cells in the acinus
//9     M – Food intake, meal variable (reserved variable for future use)
//10    Tg_p – Trypsinogen production
//11    Tg_a – Trypsynigen activation
//12    T_i – Trypsin inactivation
//13    P_p – PSTI production
//14    CF – CFTR mutation
//15    CA - Trypsin activation in the cell (it is different from Ca, because C++ is case sensitive)
//16    LA - Trypsin activation in the lumen
//17    DA - Trypsin activation in the duct
//18    MR – Molecule range
//19    Thr – Threshold for inflammation
//20    NP – Number of patients
//21    ND – Munber of days
//22    NA – Total number of acini
//23    NC – Total number of acinar cells

enummol_code1 {Tg=0, P=1, Ca=2, BC=3, W=4, I=5, D1=6, D2=7, D3=8, M=9, Tg_p=10, Tg_a=11, T_i=12, P_p=13, CF=14, CA=15, LA=16,
DA=17, MR=18, Thr=19, NP=20, ND=21, NA=22, NC=23};



//This is a function that adds some random error.
//x1 - is the number we want to change.
//x2 - is a range from which error is drawn.
//used when TG, PSTI, Ca, and BC are assigned in the cells.

double err(double x1, double x2){
        int n1, n2;
        double n3;
        bool b=1;
```

```
        while (b){
            n1=rand()%101-50;
            n2=rand()%101;
            if (101-abs(2*n1)>n2) b=0;
        }
        n3=x1+x1*x2*n1/50;
        return n3;
}


//This function returns randomly chosen numbers from 0 to 1 with uniform distribution. The numbers are rounded to second digit
//after point. Used in functions that activate trypsinogen in cell, lumen and duct.

double perc(double x){
        double n=x*(rand()%101)/100;
        return n;
}


//This function is used in trypsinogen activation. It makes sure that calcium has no effect
//on the trypsinogen activation without inflammation.

double cum(double infl, double calcium){
        if (infl==1) return1;
        elsereturn infl*calcium;
}


//This function creates a vector filled with numbers that correspond to molecules produced by
//acinar cell. It uses two vectors that store model parameters and lumen parameters.

vector<double> stimulate_A(vector<double>& vl, vector<double>& vp){
        vector<double> temp (6);
        temp[Tg] = err(vp[Tg],vp[MR])*vp[Tg_p];
        temp[P]      = err(vp[P],vp[MR])*pow(vl[I],2)*vp[P_p];
        temp[Ca] = err(1,vp[MR])*(1+(vl[I]-1)/2);
        temp[BC] =0;
        temp[W] = 0;
        temp[I] = vl[I];
        return temp;
}


//This function activates some portion of trypsinogen in the acinar cell. Then it checks
//whether existing PSTI molecules are enough to stop active trypsin from causing damage.
//If existing PSTI molecules are not enough, acinar cell is destroyed.

bool check_T_cell(vector<double>& vt, vector<double>& vp){
        bool b=0;
        double p=perc(vp[CA]);
        double temp=(vt[Tg]-vt[P])*p*cum(vt[I],vt[Ca])*vp[Tg_a]/vp[T_i];
        if (temp<vp[Thr]) {vt[Tg]*=p; vt[P]*=p; b=1;}
        return b;
}


//the same idea, but for whole acinus

bool check_T_lumen(vector<double>& vl, vector<double>& vp){
        bool b=0;
        double p=perc(vp[LA]);
        double temp=vl[Tg]*p*cum(vl[I],vl[Ca])*vp[Tg_a]/vp[T_i];
        if (temp<=vl[P]) {vl[Tg]-=temp; vl[P]-=temp; b=1;}
        return b;
```

```
}


//the same idea, but for whole pancreas

bool check_T_duct(vector<double>& vd, vector<double>& vp){
        bool b=0;
        double p=perc(vp[DA]);
        double temp=vd[Tg]*p*cum(vd[I],vd[Ca])*vp[Tg_a]/vp[T_i];
        if (temp<=vd[P]) {vd[Tg]-=temp; vd[P]-=temp; b=1;}
        return b;
}


//Once this function is called it increases inflammation to 0.5, but don't allow inflammation exceed 2 (its maximum level)

void increase_I(double& infl){
        if (1<=infl and infl<=1.5) infl+=0.5;
        elseif (1.5<infl and infl<=2) infl=2;
        elsethrow1;
}



//This function designed to reduce inflammation each time it called, but don't allow inflammation be lower than 1 (its minimum level)

void reduce_I(double& infl){
        if (1<=infl and infl<1.05) infl=1;
        elseif (1.05<=infl and infl<=2) infl-=0.05;
        elsethrow2;
}



//**********************
//***                          ***
//***    Acinus                ***
//***                          ***
//**********************

class A{
public:

        //This function returns first or second number stored in the "status" vector.
        //Because these two numbers take only 0 or 1, we can also interpreted them as Boolean.

        bool get_status(int index){
           bool b;
           if (index==0or index==1) b=status[index];
           elsethrow3;
           return b;
        }



        //This function returns 3rd number of the "status" vector,
        //that corresponds to number of active acinar cells in the acinus.

        int get_cells(){returnstatus[2];}



        //This function designed to reduce inflammation each time it called, but for acinus.

        void reduce_acinus_I(){
           if (1<=lum[I] andlum[I]<1.05) lum[I]=1;
           elseif (1.05<=lum[I] andlum[I]<=2) lum[I]-=0.05;
           elsethrow4;
```

```
}


//This function generates acinus or simply fills up both vectors of class A (acinus).
void gen_acinus(vector<double>& vp){
   status.push_back(1);              //Means that acinus is working
   status.push_back(0);              //Means that acinus was never destroyed
   status.push_back(vp[D3]);         //Number of active cells in the acinus

   //Fills first four numbers of vector, that correspont Tg, P, Ca, Cl respectively.
   for(int i=0; i<4; i++) lum.push_back(0);
   lum.push_back(1);                 //water
   lum.push_back(vp[I]);                        //inflammation that takes value from general inflammation.
}



//The same as previous function, but this time second number of vector "status"
//takes value of 1, meaning that this particular acinus was destroyed once.

void regen_acinus(vector<double>& vp){
   status.clear();
   status.push_back(1);
   status.push_back(1);
   status.push_back(vp[D3]);

   lum.clear();
   for(int i=0; i<4; i++) {lum.push_back(0);}
   lum.push_back(1);
   lum.push_back(vp[I]);
}



//Main function of class A (acinus)
bool stimulate_acinus(vector<double>& vp){
   bool li=0;                //Indicator whether damage happened in the lumen
   vector<double> vtemp;     //A vector to store temporary numbers
   int dead=0;               //Number of dead acing cells


   //This loop designed to call function that stimulate acinar cell the same
   //times as number of active acinar cells in the acinus. It also checks
   //whether something happened in the acinar cell itself.
   for (int i=0; i<status[2]; i++){
            vtemp = stimulate_A(lum, vp);
            if (check_T_cell(vtemp, vp))
                     for (int i=0; i<4; i++)
                              lum[i]+=vtemp[i];
            else dead++;
            vtemp.clear();
   }

   //Increases inflammation if any acinar cell died
   if (dead>0) increase_I(lum[I]);

   //There are two conditions when acinus supposed to be destroyed:
   //when all acinar cells in the acinus die or activated trypsin exceeds existing PSTI.
   //In other cases acinus continues to work, but number of active acinar cells is reduced
   //to number of acinar cells that died during simulation.

   if (dead<status[2]){
            status[2]-=dead;

            for (int i=0; i<4; i++) lum[i]/=lum[W]; //dilute molecules in the main duct

            //This step was taken to show that Ca level in the lumen are
            //increased compared to level inside acinar cell.
```

```
                lum[Ca]*=vp[Ca];

                //Checks whether existing PSTI molecules enough to stop activation
                //of trypsinogen in the lumen.
                if (check_T_lumen(lum, vp)){
                        lum[BC]=err(1,vp[MR]);          //Assign bicarbonate
                        lum[W]=vp[W]*lum[BC]*vp[CF];                    //Increase level of water
                        li=1;
                }
                elsestatus[0]=0;
        }
        elsestatus[0]=0;

        //If nothing happens with acinus or with any of it's acing cells,
        //the inflammation is reduced.
        if (status[0] and dead==0) reduce_acinus_I();

        dead=0;
        return li;
    }



    //This function helps to collect all molecules from lumen to duct.
    //Or simply add corresponding numbers to a vector that represents duct.
    void collect(vector<double>& vd){
        for (int i=0; i<5; i++) {vd[i]+=lum[i];}
        //Reset vector, so we can use it again for next acinus.
        lum[Tg]=lum[P]=lum[Ca]=lum[BC]=0; lum[W]=1;
    }



    //Synchronize inflammation from entire pancreas to particular acinus.
    //Makes sure that local inflammation not less than general inflammation.
    void sync_acinus(vector<double>& vp){
        if (vp[I]>lum[I]) lum[I]=vp[I];
    }



    //Recover number of acinar cells if it dramatically drops down.
    void regen_cells(vector<double>& vp){
        if (status[2]<=vp[D3]/2){
                status[2]=vp[D3];
                lum[I]=vp[I];
        }
    }

private:
    //This vector stores three numbers: 1st two numbers correspond to status of acinus itself
    //and indicator of whether acinus was destroyed or not respectively. These two numbers
    //takes only two values (0 and 1) and can be interpreted as Boolean. 3rd number corresponds
    //to number of active acinar cells in the acinus.

    vector<int>status;

    //This vector contains six numbers. Their position as following: trypsinogen, PSTI,
    //calcium, chloride, water, and inflammation of the acinus.

    vector<double>lum;
};



//*********************
//***                       ***
//***   Matrix              ***
//***                       ***
```

```
//*********************

typedef Numeric_lib::Matrix<A,2> Matrix;


//This function creates a matrix with defined numbers of rows and columns.
//It also fills each cell of the matrix with empty acinus.

void gen_matrix(Matrix& f, vector<double>& vp){
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                f(i,j).gen_acinus(vp);
}


//When called this function finds acini and renew them.
void regen_matrix(Matrix& f, vector<double>& vp){
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                if (!f(i,j).get_status(0) and !f(i,j).get_status(1)) f(i,j).regen_acinus(vp);
}



//This function starts whole simulation of the matrix (pancreas).
vector<double> stimulate_matrix(Matrix& f, vector<double>& vp, bool& di){

        //This vector is for temporary use.
        //It has 5 empty numbers plus 6th one, equal to general inflammation.
        vector<double> vtemp (5);
        vtemp.push_back(vp[I]);
        int count=0;                        //Counts how many acini worked in one simulation
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                if (f(i,j).get_status(0)){          //Check whether acini is ready to work
                        f(i,j).regen_cells(vp);//Fills up acinus with acinar cells if needed
                        f(i,j).sync_acinus(vp);//Synchronize inflammation along whole pancreas

                //Stimulate acinus and then collect produced molecules (juice) to main duct
                        if (f(i,j).stimulate_acinus(vp)) {f(i,j).collect(vtemp); count++;}
                        else {di=1;}            //Indicates whether something went wrong.
                }
        vtemp[Ca]*=vp[Ca];                  //Increase calcium level
        return vtemp;
}



//Report total number of active acini and acinar cells it returns FALSE when everything is OK
//and TRUE when number of acini is too small. In other words this function indicates whether
//recovery of panaceas is required
bool report(Matrix& f, vector<double>& vp){
        bool r=0;
        vp[NA]=vp[NC]=0;
        for(int i=0; i<f.dim1(); i++)
           for(int j=0; j<f.dim2(); j++)
                if (f(i,j).get_status(0)){
                        vp[NA]++;
                        vp[NC]+=f(i,j).get_cells();
                }
        if (vp[NA]<(vp[D1]*vp[D2]/10)) r=1;
        return r;
}



//*****************************
//***                           ***
//***    Main Function          ***
```

```
//***                                   ***
//***************************

int main (){
try{
        srand(time(NULL));              //Turn on random number generator
        vector<double> param, param2;   //Used to store parameters from external file
        vector<int> days;               //Used to catch days when pancreatitis attacks occurred
        doublevalue;                            //Temporary number


        while (!(cin>>ws).eof()){
           cin>>value;
           param.push_back(value);
        }

        //The vector that store parameters is constantly modified during the simulation
        //to keep initial set of parameters intact we copy it to second vector.

        param2=param;


        cout<<"ID\tTime\tStatus\tGroup\n";
        for(int n=1; n<=param[NP]; n++){//Loop whole simulation for number of patients
           bool AP=0, Di=0;                     //Indicators of pancreatitis and duct inflammation
           days.clear();

           Matrix frame (param[D1], param[D2]);     //Create matrix
           gen_matrix(frame, param);                //Fill matrix
           vector<double> juice;                                //Temporary vector

           for (int i=0; i<param[ND]; i++){             //Loop simulation for number of days
                   if (report(frame, param))regen_matrix(frame, param);             //Regenerate acini if needed
                   juice=stimulate_matrix(frame, param, Di);//Collect juice (produced molecules from all acini)

                   if (Di) increase_I(param[I]);                                    //Increase general inflammation in response to acinus death
                   if (check_T_duct(juice, param)) {                                //This part activates some trypsinogen and checks if PSTI is
enough to avoid dammage
                           if (!Di)
                                   reduce_I(param[I]);  //This part is a little tricky due to complexity of conditions whether
general inflammation is needed to be reduced or increased
                                   reduce_I(param[I]);
                   }
                   elseif (!Di) increase_I(param[I]);
                   Di=0;                                                            //Reset Di

                   //These set of conditions are designed to catch time when pancreatitis
                   //attacks were occurred and stop simulation after 3rd attack or total
                   //number of acinar cells is too small (the last condition was never true)
                   if (param[I]>1.5and !AP) {days.push_back(i/30); AP=1;}
                   if (param[I]==1) AP=0;
                   if ((param[NC]<100) || (days.size()==3)) {
                           days.push_back(0);
                           i=param[ND];
                   }
           }


           //These set of conditions help to make output looks nice
           if (days.size()>0) {cout<<n <<"\t"<<days[0] <<"\t1\t1\n";}
           else {cout<<n <<"\t"<<ceil(param[ND]/7) <<"\t0\t1\n";}

           if (days.size()>1) {cout<<n <<"\t"<<days[1] <<"\t1\t2\n";}
           else {cout<<n <<"\t"<<ceil(param[ND]/7) <<"\t0\t2\n";}

           if (days.size()>2) {cout<<n <<"\t"<<days[2] <<"\t1\t3\n";}
           else {cout<<n <<"\t"<<ceil(param[ND]/7) <<"\t0\t3\n";}

           param=param2;                        //reset parameters and use them again for next patient
```

```
            }
            return0;
}
catch (int e) {
            cerr<<"\nError in "<<e <<endl;
}
}
```

BIBLIOGRAPHY

1. Beger HG, Buchler M. The pancreas: an integrated textbook of basic science, medicine, and surgery: Blackwell Pub., 2008.

2. Johnson LR. Encyclopedia of gastroenterology: Academic Press / Elsevier, 2004.

3. Books L. Pancreas: Endocrine Pancreas, Pancreas Anatomy, Islets of Langerhans, Mucinous Cystic Neoplasm, Pancreatic Duct, Ampulla of Vater: General Books LLC, 2010.

4. Leung PS. The Renin-Angiotensin System: Current Research Progress in the Pancreas: The RAS in the Pancreas: Springer, 2010.

5. Chwistek M, Roberts I, Amoateng-Adjepong Y. Gallstone pancreatitis - A community teaching hospital experience. Journal of Clinical Gastroenterology 2001;33:41-44.

6. Torgerson JS, Lindroos AK, Naslund I, et al. Gallstones, gallbladder disease, and pancreatitis: Cross-sectional and 2-year data from the Swedish Obese Subjects (SOS) and SOS reference studies. American Journal of Gastroenterology 2003;98:1032-1041.

7. Whitcomb DC. Acute pancreatitis - Reply. New England Journal of Medicine 2006;355:961-961.

8. Whitcomb DC, Yadav D, Adam S, et al. Multicenter approach to recurrent acute and chronic pancreatitis in the United States: The North American Pancreatitis Study 2 (NAPS2). Pancreatology 2008;8:520-531.

9. Yadav D, Hawes RH, Brand RE, et al. Alcohol Consumption, Cigarette Smoking, and the Risk of Recurrent Acute and Chronic Pancreatitis. Archives of Internal Medicine 2009;169:1035-1045.

10. Yadav D, Muddana V, O'Connell MR. A Population Based Study on Hospitalizations for Chronic Pancreatitis (CP). Gastroenterology 2011;140:S546-S546.

11. Charnley RM. Hereditary pancreatitis. World Journal of Gastroenterology 2003;9:1-4.

12. Grigorescu M, Grigorescu MD. Genetic factors in pancreatitis. Romanian journal of gastroenterology 2005;14:53-61.

13. Keim V. Role of genetic disorders in acute recurrent pancreatitis. World Journal of Gastroenterology 2008;14:1011-1015.

14. Whitcomb DC, Barmada MM. A systems biology approach to genetic studies of pancreatitis and other complex diseases. Cellular and Molecular Life Sciences 2007;64:1763-1777.

15. Felderbauer P, Schnekenburger J, Lebert R, et al. A novel A121T mutation in human cationic trypsinogen associated with hereditary pancreatitis: functional data indicating a loss-of-function mutation influencing the R122 trypsin cleavage site. Journal of Medical Genetics 2008;45:507-512.

16. Hashimoto D, Ohmuraya M, Hirota M, et al. Involvement of autophagy in trypsinogen activation within the pancreatic acinar cells. Journal of Cell Biology 2008;181:1065-1072.

17. Kukor Z, Toth M, Sahin-Toth M. Human anionic trypsinogen - Properties of autocatalytic activation and degradation and implications in pancreatic diseases. European Journal of Biochemistry 2003;270:2047-2058.

18. Briand L, Chobert JM, Tauzin J, et al. Regulation of trypsin activity by Cu2+ chelation of the substrate binding site. Protein Eng 1997;10:551-60.

19. Hirota M, Ohmuraya M, Baba H. The role of trypsin, trypsin inhibitor, and trypsin receptor in the onset and aggravation of pancreatitis. Journal of Gastroenterology 2006;41:832-836.

20. Ohmuraya M, Yamamura K. Autophagy and acute pancreatitis. Autophagy 2008;4:1060-1062.

21. Garcia M, Hernandez-Lorenzo P, Roman JIS, et al. Pancreatic duct secretion: experimental methods, ion transport mechanisms and regulation. Journal of Physiology and Biochemistry 2008;64:243-257.

22. Ko SBH, Zeng WZ, Dorwart MR, et al. Gating of CFTR by the STAS domain of SLC26 transporters. Nature Cell Biology 2004;6:343-350.

23. Schneider A, Larusch J, Sun X, et al. Combined Bicarbonate Conductance-Impairing Variants in CFTR and SPINK1 Variants Are Associated With Chronic Pancreatitis in Patients Without Cystic Fibrosis. Gastroenterology 2011;140:162-171.

24. Murugaian EE, Premkumar RMR, Radhakrishnan L, et al. Novel mutations in the calcium sensing receptor gene in tropical chronic pancreatitis in India. Scandinavian Journal of Gastroenterology 2008;43:117-121.

25. Zhou J, Sahin-Toth M. Chymotrypsin C mutations in chronic pancreatitis. Journal of Gastroenterology and Hepatology 2011;26:1238-1246.

26. Feldman M, Friedman LS, Brandt LJ. Sleisenger and Fordtran's Gastrointestinal and Liver Disease- 2 Volume Set E-Book: Pathophysiology, Diagnosis, Management, Expert Consult

Premium Edition - Enhanced Online Features and Print E-Book: Elsevier Health Sciences, 2010.

27. National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK), National Institutes of Health (NIH). Citing Website. *Digestive Diseases Statistics for the United States*. November 25, 2008. Available at:http://digestive.niddk.nih.gov/statistics/statistics.aspx - specific.Accessed: September 20, 2011.

28. Fagenholz PJ, Castillo CF-d, Harris NS, et al. Direct medical costs of acute pancreatitis hospitalizations in the United States. Pancreas 2007;35:302-307.

29. Steward MC, Ishiguro H. Molecular and cellular regulation of pancreatic duct cell function. Current Opinion in Gastroenterology 2009;25:447-453.

30. Don Blis. Wikipedia, the free encyclopedia. Citing Website.*Duodenumandpancreas.jpg*. September 20, 2007. Available at: http://en.wikipedia.org/wiki/File:Duodenumandpancreas.jpg. Accessed at September 27, 2011.

31. Ley K. Physiology of inflammation: Oxford University Press, 2001.

32. Sherwood L. Human physiology: from cells to systems: Brooks/Cole, Cengage Learning, 2008.

33. Stroustrup B. Programming: principles and practice using C++: Addison-Wesley, 2009.

34. Caldwell RA. EFFECT OF CALCIUM AND PHYTIC ACID ON THE ACTIVATION OF TRYPSINOGEN AND THE STABILITY OF TRYPSIN. Journal of Agricultural and Food Chemistry 1992;40:43-46.

35. Frick TW, FernandezdelCastillo C, Bimmler D, et al. Elevated calcium and activation of trypsinogen in rat pancreatic acini. Gut 1997;41:339-343.

36. Dor Y, Stanger BZ. Regeneration in liver and pancreas: time to cut the umbilical cord? Sci STKE 2007;2007:pe66.

37. Li W-C, Rukstalis JM, Nishimura W, et al. Activation of pancreatic-duct-derived progenitor cells during pancreas regeneration in adult rats. Journal of Cell Science 2010;123:2792-2802.

38. Blaise Barney, Lawrence Livermore National Laboratory. Citing Website. *OpenMP*. February 14, 2012. Available at: https://computing.llnl.gov/tutorials/openMP. Accessed October 10, 2011.

39. Blaise Barney, Lawrence Livermore National Laboratory. Citing Website. *POSIX Threads Programming*. February 14, 2012. Availeble at: https://computing.llnl.gov/tutorials/pthreads. Accessed October 10, 2011.

40. Whitcomb DC, Ermentrout GB. A mathematical model of the pancreatic duct cell generating high bicarbonate concentrations in pancreatic juice. Pancreas 2004;29:E30-E40.