**Московский государственный технический университет им. Н.Э.Баумана**

Кафедра «Системы обработки информации и управления»

Утверждаю: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Согласовано: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_\_"\_\_\_\_\_\_\_\_\_\_2020 г. "\_\_\_"\_\_\_\_\_\_\_\_\_\_\_2020 г.

**Курсовая работа по дисциплине**

**«Имитационное моделирование дискретных процессов»**

**Тема: «Модель распространения коронавирусной инфекции в университете»**

Пояснительная записка

(вид документа)

Листы А4

(вид носителя)

35

(количество листов)

|  |  |
| --- | --- |
| Исполнитель: | Проверил: |
| Студент группы ИУ5-71  Белоусов Евгений | Черненький М.В. |

Москва - 2020

Оглавление

[Введение 2](#_Toc59136020)

[Актуальность 2](#_Toc59136021)

[Цели и задачи 2](#_Toc59136022)

[Предметная область 3](#_Toc59136023)

[Описание модели 3](#_Toc59136024)

[Ограничения модели 4](#_Toc59136025)

[Модель 5](#_Toc59136026)

[Эксперимент 1 11](#_Toc59136027)

[Эксперимент 2 13](#_Toc59136028)

[Эксперимент 3 14](#_Toc59136029)

[Эксперимент 4 15](#_Toc59136030)

[Проверка возможности построения объемных графиков 12](#_Toc59136031)

[Выбросы 14](#_Toc59136032)

[Эксперименты 5-12 15](#_Toc59136033)

[Выводы по эксперименту 1 24](#_Toc59136034)

[Выводы по эксперименту 2 24](#_Toc59136035)

[Выводы по эксперименту 3 25](#_Toc59136036)

[Выводы по эксперименту 4 26](#_Toc59136037)

[Выводы по экспериментам 5-12 27](#_Toc59136038)

Выводы по работе………………………………………………………………………………………34

Список используемых источников…………………………………………………………………….35

# Введение

Имитационное моделирование становится эффективным методом исследования сложных систем со случайным взаимодействием элементов, таких как транспортные потоки, многоступенчатое промышленное производство, распределенные объекты управления. Принцип имитационного моделирования заключается в том, что поведение системы отображают компьютерной моделью взаимодействия ее элементов во времени и пространстве.

Главная ценность имитационного моделирования состоит в том, что в его основу положена методология системного анализа. Она дает возможность исследовать проектируемую или анализируемую систему по технологии операционного исследования, включая такие взаимосвязанные этапы, как содержательная постановка задачи; разработка концептуальной модели; разработка и программная реализация имитационной модели; оценка адекватности модели и точности результатов моделирования; планирование экспериментов; принятие решений. Благодаря этому имитационное моделирование можно применять как универсальный подход для принятия решений в условиях неопределенности и для учета в моделях трудно формализуемых факторов.

Изучение системы с помощью модели позволяет проверить новые решения без вмешательства в работу реальной системы, растянуть или сжать время функционирования системы, понять сложное взаимодействие элементов внутри системы, уточнить значения коэффициентов, оценить степень влияния факторов и выявить “узкие места”.

Применение имитационного моделирования целесообразно, если:

* проведение экспериментов с реальной системой невозможно или дорого;
* требуется изучить поведение системы при ускоренном или замедленном времени;
* аналитическое описание поведения сложной системы невозможно;
* поведение системы зависит от случайных воздействий внешней среды;
* требуется выявить реакцию системы на непредвиденные ситуации;
* нужно проверить идеи по созданию или модернизации системы;
* требуется подготовить специалистов по управлению реальной системой.

# Актуальность

В 2020 году человечество столкнулось с пандемией коронавируса. Большинство предприятий было переведено на удаленный режим работы. Все крупные IT компании отправили своих сотрудников на удаленку. Многие ВУЗы перешли на дистанционный режим обучения, не дожидаясь приказа сверху. Многие начали использовать медицинские маски в местах скопления людей.

Данная работа должна ответить на вопрос, необходимы ли принятые меры безопасности.

# Цели и задачи

Целью выполнения курсовой работы является исследование процессов распространения коронавирусной инфекции. Для этого необходимо решить следующие задачи:

1. Изучить предметную область, которую необходимо моделировать.
2. Построить модель.
3. Выявить зависимости в модели.
4. На основе выявленных зависимостей сделать выводы.

# Предметная область

COVID-19 — потенциально тяжёлая острая респираторная инфекция, вызываемая коронавирусом SARS-CoV-2 (2019-nCoV). Представляет собой опасное заболевание, которое может протекать как в форме острой респираторной вирусной инфекции лёгкого течения, так и в тяжёлой форме. Наиболее частым осложнением заболевания является вирусная пневмония, способная приводить к острому респираторному дистресс-синдрому и последующей острой дыхательной недостаточности, при которых чаще всего необходимы кислородная терапия и респираторная поддержка. К наиболее распространённым симптомам заболевания относятся повышенная температура тела, утомляемость и сухой кашель. В редких случаях поражение вирусом детей и подростков, предположительно, может приводить к развитию воспалительного синдрома.

Распространяется вирус воздушно-капельным путём через вдыхание распылённых в воздухе при кашле, чихании или разговоре капель с вирусом, а также через попадание вируса на поверхности с последующим занесением в глаза, нос или рот. К числу эффективных мер профилактики относится частое мытьё рук и соблюдение правил респираторной гигиены. Заболевание вызывается новым вирусом, против которого у людей изначально нет приобретённого иммунитета, к инфекции восприимчивы люди всех возрастных категорий.

На 1 октября 2020 года против вируса отсутствовали какие-либо специфические противовирусные средства лечения или профилактики. В большинстве случаев (примерно в 80 %) какое-либо специфическое лечение не требуется, а выздоровление происходит само по себе.

Коэффициент смертности от инфекции (англ. IFR) оценивается примерно в 0,68 %, согласно анализу серопревалентности ВОЗ — в 0,27 %.

Вирус передаётся воздушно-капельным путём через вдыхание мелких капель, распылённых в воздухе при кашле, чихании или разговоре. Капли с вирусом могут попадать на поверхности и предметы, а затем инфицировать прикоснувшегося к ним человека через последующие прикосновения к глазам, носу или рту. Вирус может оставаться жизнеспособным в течение нескольких часов, попадая на поверхности предметов. На стальных поверхностях и на пластике он может сохраняться до 2—3 дней

Для инфекции, вызываемой вирусом SARS-CoV-2, инкубационный период составляет 1—14 дней, может протекать бессимптомно, в лёгкой форме и в тяжёлой форме, с риском смерти, но полная клиническая картина пока ещё не ясна.

Пациенты с лёгкими симптомами обычно выздоравливают в течение недели. В среднем длительность симптомов не превышает 20 дней.

# Описание модели

Для моделирования процессов распространения коронавирусной инфекции в университете могут быть использованы мультиагентные модели или модели типа SEIR. В данной работе будет реализована мультиагентная модель ввиду отсутствия коэффициентов для модели SEIR.

Модель является дискретной моделью с дискретизацией в 1 сутки.

Время моделирования – с 1 сентября до 31 декабря.

В рамках модели будем считать, что университет принял решение не переходить на дистанционное обучение. Однако, мы можем отслеживать картину распространения заболевания и для университетов, которые перешли на дистанционку, просто не рассматривая часть графиков, находящихся за днем введения самоизоляции.

Изначально все, кроме одного нулевого пациента, студенты и преподаватели университета здоровы.

Преподаватели, в рамках рассматриваемой модели, представляют собой такую же сущность как студенты, но с другими значениями характеристик.

Каждые сутки каждый человек с некоторой вероятностью приходит в университет. Он встречает случайное количество пришедших человек, распределенное нормальным образом вокруг среднего числа человек, встречаемое им в сутки. Если встречается здоровый человек, то ничего не происходит. Если встречается зараженный человек, то с вероятностью заразности коронавирусной инфекции студент становится зараженным. Зараженный студент продолжает ходить в университет до проявления первых симптомов (в течение инкубационного периода, который распределен нормальным образом) и заражать встречаемых людей. После проявления симптомов студент прекращает ходить в университет и болеет некоторое нормально распределенное время. После этого он умирает с вероятностью смертности от коронавирусной инфекции или выздоравливает и обретает иммунитет, на некоторое нормально-распределенное время, после которого снова становится подверженным опасности заражения.

# Ограничения модели

Рассматриваемая модель не учитывает такие факторы как:

1. Люди, с которыми студенты и преподаватели встречаются вне университета.
2. Загруженность больниц: смертность вируса не зависит от количества заболевших.
3. Пожилых людей и людей, страдающих, хроническими заболеваниями: в модели для всех используются усредненные коэффициенты.
4. Время моделирования ограничено периодом с 1 сентября 2020 г. по 31 декабря 2020 г.

# Модель

[1]:

*#health status:*

SUCCESS = 0

HEALTHY = 1

SICK = 2

INFECTIOUS = 3

IMMUNITY = 4

DEAD = 5

[2]:

*#human role:*

STUDENT = 0

TEACHER = 1

[3]:

**from calendar import** Calendar

**import random**

[4]:

**class Human**:

**def** init ( self, role, come\_to\_university, meeting\_person\_expected\_value, meeting\_person\_dispersion, incubation\_period\_expected\_value, incubation\_period\_dispersion, mortality, illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value, immunitet\_period\_dispersion):

self.health = HEALTHY self.role = role

self.come\_to\_university = come\_to\_university self.meeting\_person\_expected\_value = meeting\_person\_expected\_value self.meeting\_person\_dispersion = meeting\_person\_dispersion self.time\_period = -1

self.mortality = mortality

self.incubation\_period\_expected\_value = incubation\_period\_expected\_value self.incubation\_period\_dispersion = incubation\_period\_dispersion self.illness\_time\_expected\_value = illness\_time\_expected\_value self.illness\_time\_dispersion = illness\_time\_dispersion self.immunitet\_period\_expected\_value = immunitet\_period\_expected\_value self.immunitet\_period\_dispersion = immunitet\_period\_dispersion

**def** infectious(self, infectiouness):

**if** self.health == HEALTHY **and** random.uniform(0, 1) < infectiouness: self.health = INFECTIOUS

self.time\_period = int(random.normalvariate( self.incubation\_period\_expected\_value, self.incubation\_period\_dispersion))

##### return True return False

**def** process(self):

**if** self.health == INFECTIOUS: self.time\_period -= 1

**if** self.time\_period <= 0: self.health = SICK

self.time\_period = int(random.normalvariate( self.illness\_time\_expected\_value, self.illness\_time\_dispersion

))

**return** self.health

**if** self.health == SICK: self.time\_period -= 1

**if** self.time\_period <= 0:

**if** random.uniform(0, 1) < self.mortality: self.health = DEAD

##### else:

self.health = IMMUNITY

self.time\_period = int(random.normalvariate( self.immunitet\_period\_expected\_value, self.immunitet\_period\_dispersion

))

**return** self.health

**if** self.health == IMMUNITY: self.time\_period -= 1

**if** self.time\_period <= 0: self.health = HEALTHY self.time\_period = -1 **return** self.health

**return** SUCCESS

**def** is\_come(self):

**if** self.health != SICK **and** \

random.uniform(0, 1) < self.come\_to\_university:

##### return True return False

**def** get\_meetings(self):

meetings = int(random.normalvariate(

self.meeting\_person\_expected\_value, self.meeting\_person\_dispersion))

**if** meetings > 0:

**return** meetings

**return** 0

[5]: **class Population**:

**def** init ( self, students\_num, student\_meeting\_person\_expected\_value, student\_meeting\_person\_dispersion, student\_come\_to\_university, teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion, teacher\_come\_to\_university, infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion, mortality, illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value, immunitet\_period\_dispersion

):

self.human\_list = []

*#add students*

**for** i **in** range(students\_num): self.human\_list.append( Human(STUDENT,

student\_come\_to\_university, student\_meeting\_person\_expected\_value, student\_meeting\_person\_dispersion, incubation\_period\_expected\_value, incubation\_period\_dispersion, mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value, immunitet\_period\_dispersion)

)

*#add teachers*

**for** i **in** range(teachers\_num): self.human\_list.append( Human(TEACHER,

teacher\_come\_to\_university, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion, incubation\_period\_expected\_value, incubation\_period\_dispersion, mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value, immunitet\_period\_dispersion)

)

self.dead = 0

self.healthy = len(self.human\_list) self.infectious = 0

self.sick = 0

self.immunity = 0 self.mortality = mortality

self.infectiousness = infectiousness

**def** process( self ):

*#create comming list* come\_to\_university = list() **for** human **in** self.human\_list:

result = human.process()

**if** result == DEAD: self.dead += 1

self.sick -= 1 self.human\_list.remove(human) **continue**

**elif** result == SICK: self.infectious -= 1

self.sick += 1

##### continue

**elif** result == IMMUNITY: self.sick -= 1

self.immunity += 1

##### continue

**elif** result == HEALTHY: self.immunity -= 1

self.healthy += 1

##### continue

**if** human.is\_come(): come\_to\_university.append(human)

**if** len(come\_to\_university) == 0:

##### return

*#meetings*

**for** human **in** come\_to\_university:

**if**(human.health == INFECTIOUS):

**for** i **in** range(human.get\_meetings()):

**if** random.choice(come\_to\_university).infectious(self.

*<→*infectiousness):

self.healthy -= 1

self.infectious += 1

**def** process\_weekend( self ):

**for** human **in** self.human\_list: result = human.process() **if** result == DEAD:

self.dead += 1

self.sick -= 1 self.human\_list.remove(human)

**elif** result == SICK: self.infectious -= 1

self.sick += 1

**elif** result == IMMUNITY: self.sick -= 1

self.immunity += 1

**elif** result == HEALTHY: self.immunity -= 1

self.healthy += 1

**def** first\_infection( self ):

human = random.choice(self.human\_list) human.infectious(1.)

self.healthy -= 1

self.infectious += 1

**def** add\_infectious( self, human): human.infectious(self.infectiousness) self.healthy -= 1

self.infectious += 1

[6]:

**import datetime**

[7]:

**def** autum\_semestr\_loop( population, drawer):

**try**:

calendar = Calendar()

my\_calendar = calendar.yeardayscalendar( year = 2020 )

**for** season **in** range(2, 5):

**for** month **in** range(3):

**if** (season)\*3+1+ month < 9:

**continue**

**if** season == 4 **and** month > 0:

**break**

**for** week **in** range(len(my\_calendar[season][month])):

**for** day **in** range(7):

**if not** my\_calendar[season][month][week][day] == 0:

**if** day == 6:

*#print('sunday')*

population.process\_weekend()

**else**:

*#print('{} - {}'.format(*

*#my\_calendar[season][month][week][day],*

*#(season)\*3+1+ month*

*#))*

*#print(str(season)+' '+str(month)+'*␣

*<→'+str(week)+' '+str(day))*

population.process()

*#print('{} {} {} {} {}'.format(*

*# population.healthy,*

*# population.infectious,*

*# population.sick,*

*# population.immunity,*

*# population.dead*

*# ))*

date = datetime.date(year=2020,

month=(season)\*3 +1 + month,

␣

*<→*day=my\_calendar[season][month][week][day])

drawer.add(population, date)

**except IndexError**: **return**

[8]:

**import matplotlib.pyplot as plt**

%**matplotlib** inline

[9]:

**class Drawer2D**:

**def** init (self): self.healthy = [] self.infectious = [] self.sick = [] self.dead = [] self.date = [] self.immunity = []

**def** add(self, population, date): self.healthy.append(population.healthy+population.immunity) self.infectious.append(population.infectious) self.sick.append(population.sick)

self.dead.append(population.dead)

self.immunity.append(population.immunity) self.date.append(date)

**def** draw(self):

fig, ax = plt.subplots(figsize=(15, 9))

ax.plot(self.date,

ax.plot(self.date, ax.plot(self.date, ax.plot(self.date, ax.plot(self.date, ax.legend()

self.healthy, 'g', label='

self.infectious, 'b', label='

')

')

self.sick, 'r', label='

')

self.dead, 'black', label='

self.immunity, 'c', label='

')

')

ax.set\_xlabel('

ax.set\_ylabel(' ax.set\_title("

')

')

")

# Эксперимент 1

Основная цель этого эксперимента – убедиться в работоспособности модели.

[10]: **def** experiment1():

students\_num = 100

student\_meeting\_person\_expected\_value = 10

student\_meeting\_person\_dispersion = 2

student\_come\_to\_university = 0.9

teachers\_num = 0

teacher\_meeting\_person\_expected\_value = 0

teacher\_meeting\_person\_dispersion = 0

teacher\_come\_to\_university = 0

infectiousness = 0.2

incubation\_period\_expected\_value = 14

incubation\_period\_dispersion = 2

mortality = 0.1

illness\_time\_expected\_value = 20

illness\_time\_dispersion = 15

immunitet\_period\_expected\_value = 40

immunitet\_period\_dispersion = 10

population = Population(students\_num, student\_meeting\_person\_expected\_value, student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() drawer2D = Drawer2D()

autum\_semestr\_loop( population, drawer2D)

**return** drawer2D

# Эксперимент 2

Для эксперимента 2 в средствах массовой информации были найдены коэффициенты, которыми характеризуется коронавирусная инфекция. Данный эксперимент призван показать характерное для коронавируса протекание болезни.

[11]: **def** experiment2():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100

population = Population(students\_num, student\_meeting\_person\_expected\_value, student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() drawer2D = Drawer2D()

autum\_semestr\_loop( population, drawer2D)

**return** drawer2D

# Эксперимент 3

В эксперименте 3 заразность коронавирусной инфекции снижена до уровня, который обеспечивает медицинская маска.

[12]: **def** experiment3():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799\*0.7 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

[*#https://www.kommersant.ru/doc/4432704*](http://www.kommersant.ru/doc/4432704)

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100

population = Population(students\_num, student\_meeting\_person\_expected\_value, student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() drawer2D = Drawer2D()

autum\_semestr\_loop( population, drawer2D)

**return** drawer2D

# Эксперимент 4

Данный эксперимент показывает, как выглядели бы зависимости, в случае, если после выздоровления у человека вырабатывался короткий иммунитет в среднем на 20 дней.

[13]: **def** experiment4():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799\*0.7 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

[*#https://www.kommersant.ru/doc/4432704*](http://www.kommersant.ru/doc/4432704)

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 20

immunitet\_period\_dispersion = 10

population = Population(students\_num, student\_meeting\_person\_expected\_value, student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() drawer2D = Drawer2D()

autum\_semestr\_loop( population, drawer2D)

**return** drawer2D

[14]:

**import pylab**

**from mpl\_toolkits.mplot3d import** Axes3D

**import numpy as np**

[15]: **class Drawer3D**:

**def** init (self, k\_name): self.Y = []

self.X = [] self.date = [] self.Zhealthy = []

self.Zinfectious = [] self.Zsick = [] self.Zdead = [] self.Zimmunity = [] self.healthy = [] self.infectious = [] self.sick = [] self.dead = [] self.immunity = [] self.k\_name = k\_name

**def** setK(self, k):

self.X.append([i **for** i **in** range(len(self.date))]) self.Y.append([k **for** i **in** range(len(self.date))].copy()) self.Zhealthy.append(self.healthy.copy()) self.Zinfectious.append(self.infectious.copy()) self.Zsick.append(self.sick.copy()) self.Zdead.append(self.dead.copy()) self.Zimmunity.append(self.immunity.copy())

self.date = [] self.healthy = [] self.infectious = [] self.sick = [] self.dead = [] self.immunity = []

**def** add( self, population, date): self.date.append(date) self.healthy.append(population.healthy) self.infectious.append(population.infectious) self.sick.append(population.sick) self.dead.append(population.dead) self.immunity.append(population.immunity)

**def** draw(self):

x = np.array(self.X) y = np.array(self.Y)

zHealthy = np.array(self.Zhealthy) zInfectious = np.array(self.Zinfectious) zImmunity = np.array(self.Zimmunity) zDead = np.array(self.Zdead)

zSick = np.array(self.Zsick)

fig = plt.figure(figsize=(15, 15))

ax = fig.add\_subplot(3, 2, 1, projection='3d') ax.set\_title(' ')

ax.set\_xlabel(' ') ax.set\_ylabel(self.k\_name) ax.set\_zlabel(' ')

surf = ax.plot\_surface(x, y, zHealthy)

ax = fig.add\_subplot(3, 2, 2, projection='3d') ax.set\_title(' ')

ax.set\_xlabel(' ') ax.set\_ylabel(self.k\_name) ax.set\_zlabel(' ')

surf = ax.plot\_surface(x, y, zInfectious)

ax = fig.add\_subplot(3, 2, 3, projection='3d') ax.set\_title(' ')

ax.set\_xlabel(' ') ax.set\_ylabel(self.k\_name) ax.set\_zlabel(' ')

surf = ax.plot\_surface(x, y, zSick)

ax = fig.add\_subplot(3, 2, 4, projection='3d') ax.set\_title(' ')

ax.set\_xlabel(' ') ax.set\_ylabel(self.k\_name) ax.set\_zlabel(' ')

surf = ax.plot\_surface(x, y, zImmunity)

ax = fig.add\_subplot(3, 2, 5, projection='3d') ax.set\_title(' ')

ax.set\_xlabel(' ') ax.set\_ylabel(self.k\_name) ax.set\_zlabel(' ')

surf = ax.plot\_surface(x, y, zDead)

# Проверка возможности построения объемных графиков

[16]:

%%time

students\_num = 100

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 0

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100 drawer3D = Drawer3D(' ')

**for** infectiousness **in** np.arange(0.01, 0.4, 0.02):

population = Population(students\_num, student\_meeting\_person\_expected\_value, student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

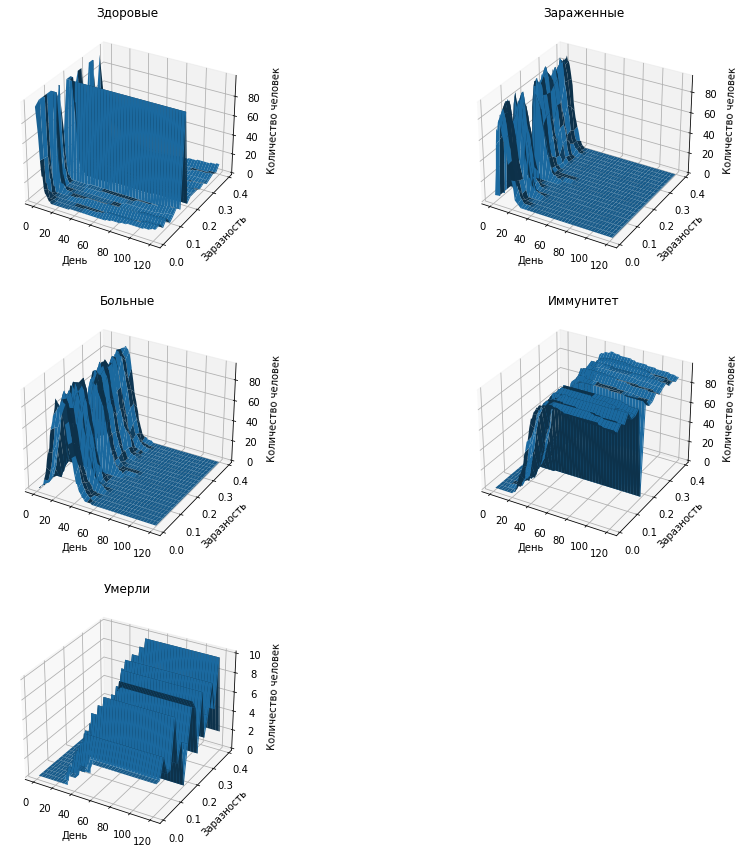
illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D) drawer3D.setK(infectiousness)

drawer3D.draw()

CPU times: user 866 ms, sys: 177 µs, total: 866 ms Wall time: 887 ms



Все зависимости построились, то есть, модель работает. Можно экспериментировать с большими объемами данных.

# Выбросы

Видно, что на объемных графиках иногда встречаются выбросы. Это происходит из-за ситуаций, например, когда нулевой пациент по воле случайности не ходил на занятия в течение инкубационного периода болезни. Данные ситуации отражают действительность, хотя и являются маловероятными.

Чтобы избавиться от выбросов, можно проводить моделирование несколько раз и усреднять значения. Ввиду увеличения вычислительной сложности и, соответственно, времени построения моделей все эксперименты, приведенные ниже, проводятся без устранения выбросов. Автор работы считает, что не смотря на выбросы, приведенные графики дают достаточно информации о характере процесса и его зависимостях

# Эксперименты 5-12

Серия экспериментов признана для определения вида зависимостей процесса распространения коронавирусной инфекции от коэффициентов.

[17]: **def** experiment5():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100 drawer3D = Drawer3D(" ")

**for** infectiousness **in** np.arange(0.0, 0.95, 0.05): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D) drawer3D.setK(infectiousness)

**return** drawer3D

[18]: **def** experiment6():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100

drawer3D = Drawer3D(" ")

**for** student\_come\_to\_university **in** np.arange(0., 0.95, 0.05): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D) drawer3D.setK(student\_come\_to\_university)

**return** drawer3D

[19]: **def** experiment7():

students\_num = 19000

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100 drawer3D = Drawer3D(" ")

**for** student\_meeting\_person\_expected\_value **in** np.arange(10, 200, 10): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D)

drawer3D.setK(student\_meeting\_person\_expected\_value)

**return** drawer3D

[20]: **def** experiment8():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150 infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100

drawer3D = Drawer3D(" ")

**for** teacher\_come\_to\_university **in** np.arange(0.05, 1, 0.05): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D) drawer3D.setK(teacher\_come\_to\_university)

**return** drawer3D

[21]: **def** experiment9():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100

drawer3D = Drawer3D(" ")

**for** incubation\_period\_expected\_value **in** np.arange(1, 20, 1): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D)

drawer3D.setK(incubation\_period\_expected\_value)

**return** drawer3D

[22]: **def** experiment10():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100 drawer3D = Drawer3D(" ")

**for** mortality **in** np.arange(0, 1, 0.05): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D) drawer3D.setK(mortality)

**return** drawer3D

[23]: **def** experiment11():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_dispersion = 100

drawer3D = Drawer3D(" ")

**for** immunitet\_period\_expected\_value **in** np.arange(10, 100, 10): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D) drawer3D.setK(immunitet\_period\_expected\_value)

**return** drawer3D

[24]: **def** experiment12():

students\_num = 19000

student\_meeting\_person\_expected\_value = 150

student\_meeting\_person\_dispersion = 50

student\_come\_to\_university = 0.5

teachers\_num = 3297

teacher\_meeting\_person\_expected\_value = 200

teacher\_meeting\_person\_dispersion = 150

teacher\_come\_to\_university = 0.3

infectiousness = 0.799 *#https://rg.ru/2020/05/04/*

*<→issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.*

*<→html*

incubation\_period\_expected\_value = 11 *#https://iz.ru/989894/2020-03-22/*

*<→nazvan-srednii-inkubatcionnyi-period-koronavirusa*

incubation\_period\_dispersion = 5

mortality = 0.06

illness\_time\_expected\_value = 20 *#https://iz.ru/981482/2020-02-28/*

*<→voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa*

illness\_time\_dispersion = 5

immunitet\_period\_expected\_value = 240

immunitet\_period\_dispersion = 100 drawer3D = Drawer3D(" ")

**for** illness\_time\_expected\_value **in** np.arange(3, 30, 2): population = Population(students\_num,␣

*<→*student\_meeting\_person\_expected\_value,

student\_meeting\_person\_dispersion,␣

*<→*student\_come\_to\_university,

teachers\_num, teacher\_meeting\_person\_expected\_value, teacher\_meeting\_person\_dispersion,␣

*<→*teacher\_come\_to\_university,

infectiousness, incubation\_period\_expected\_value, incubation\_period\_dispersion,

mortality,

illness\_time\_expected\_value, illness\_time\_dispersion, immunitet\_period\_expected\_value,␣

*<→*immunitet\_period\_dispersion)

population.first\_infection() autum\_semestr\_loop( population, drawer3D) drawer3D.setK(illness\_time\_expected\_value)

**return** drawer3D

[25]:

**import multiprocessing**

num\_of\_cpu = multiprocessing.cpu\_count() print (num\_of\_cpu)

32

[26]:

experiment\_list = [experiment1,

experiment2, experiment3, experiment4, experiment5, experiment6, experiment7, experiment8, experiment9, experiment10, experiment11, experiment12]

[27]:

**def** experiment\_do(i):

**return** experiment\_list[i]()

[28]:

%%time

pool = multiprocessing.Pool(processes=num\_of\_cpu)

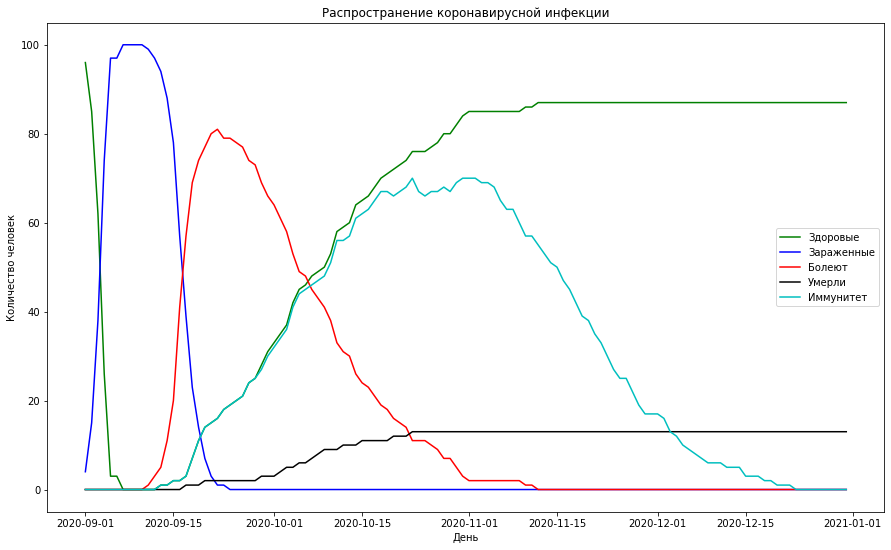
res = pool.map(experiment\_do, range(len(experiment\_list)))

CPU times: user 1.09 s, sys: 102 ms, total: 1.19 s Wall time: 3min 11s

[29]:

res[0].draw()

# Выводы по эксперименту 1

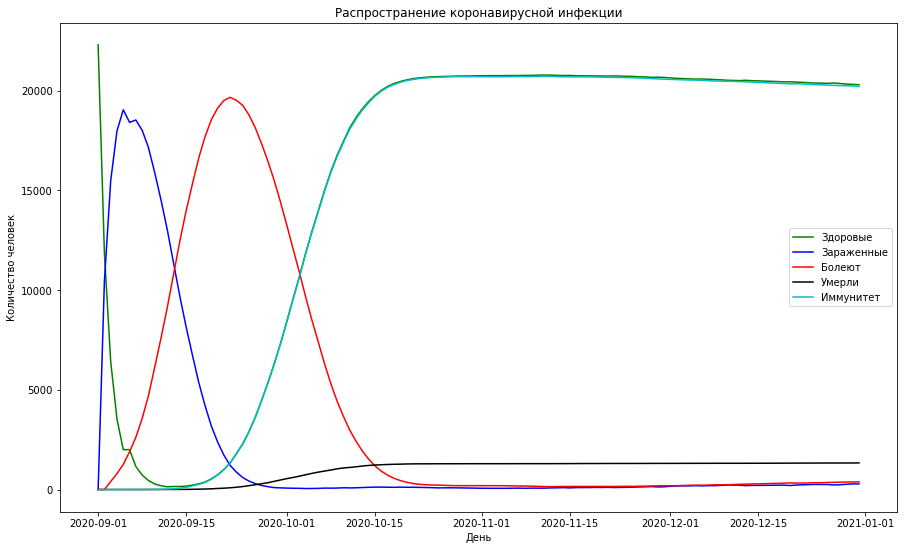


Модель работает без сбоев, корректно отражая основные зависимости

[30]:

res[1].draw()

# Выводы по эксперименту 2

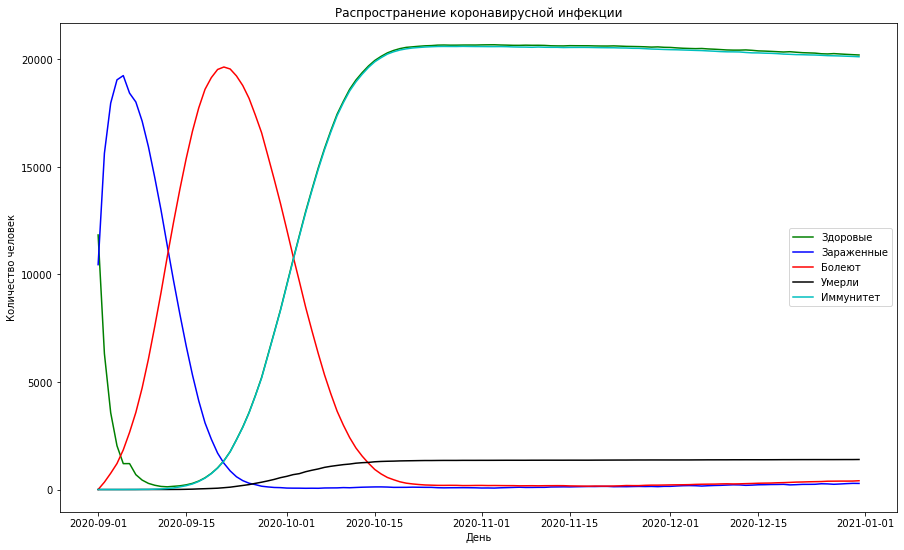


Данный график отображает обычное течение коронавирусной инфекции, почти сразу вся популяция инфицируется, и, затем, заболевает. За то время, пока люди болеют, коронавирусная инфекция исчезает, так как не остается людей, в которых вирус может размножаться. Кажется, это один из способов победить коронавирус на относительно небольшом острове, изолированном от внешнего мира. Смертность при этом находится на уровне 6%.

[31]:

res[2].draw()

# Выводы по эксперименту 3

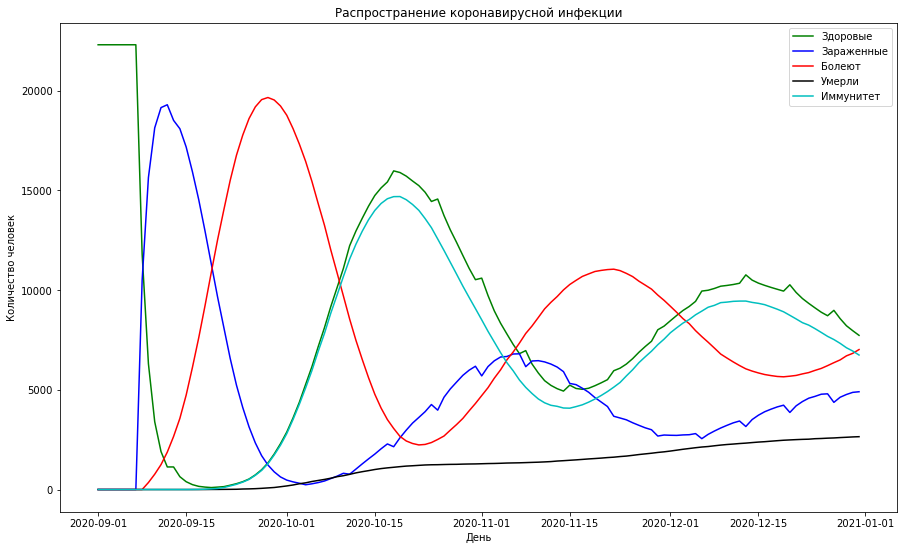


В сравнении с предыдущим графиком, результаты различаются не сильно. В обоих случаях все люди очень быстро заражаются. Данный эксперимент свидетельствует, о том что маска не способна оказать существенную защиту от коронавирусной инфекции.

[32]:

res[3].draw()

# Выводы по эксперименту 4

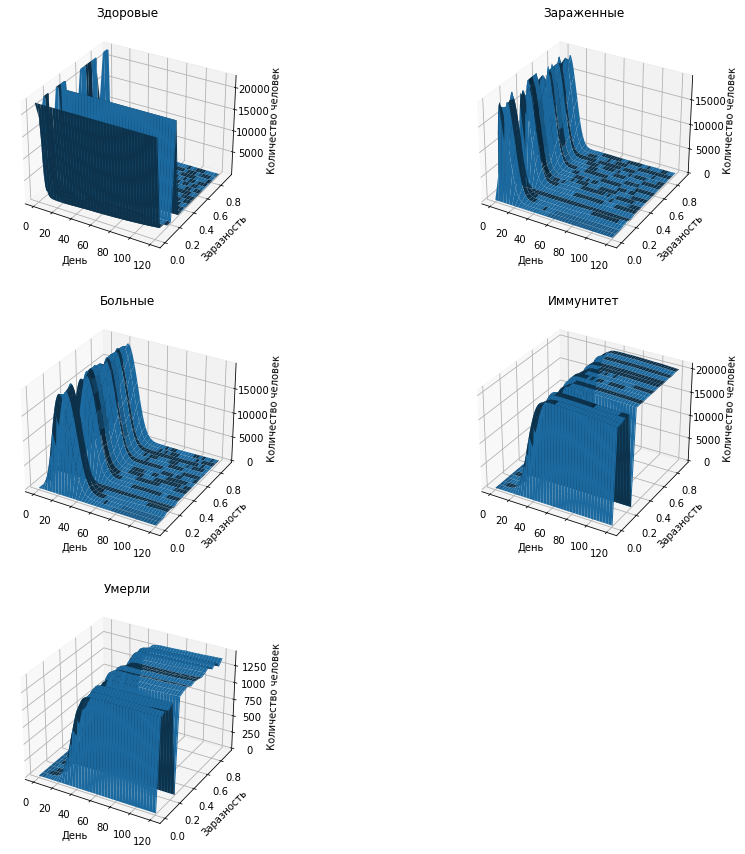


На данном графике отчетливо видно вторую волну заболевания, она меньше первой. На остальных графиках вторая волна не наблюдается из-за ограничения времени моделирования модели.

[33]:

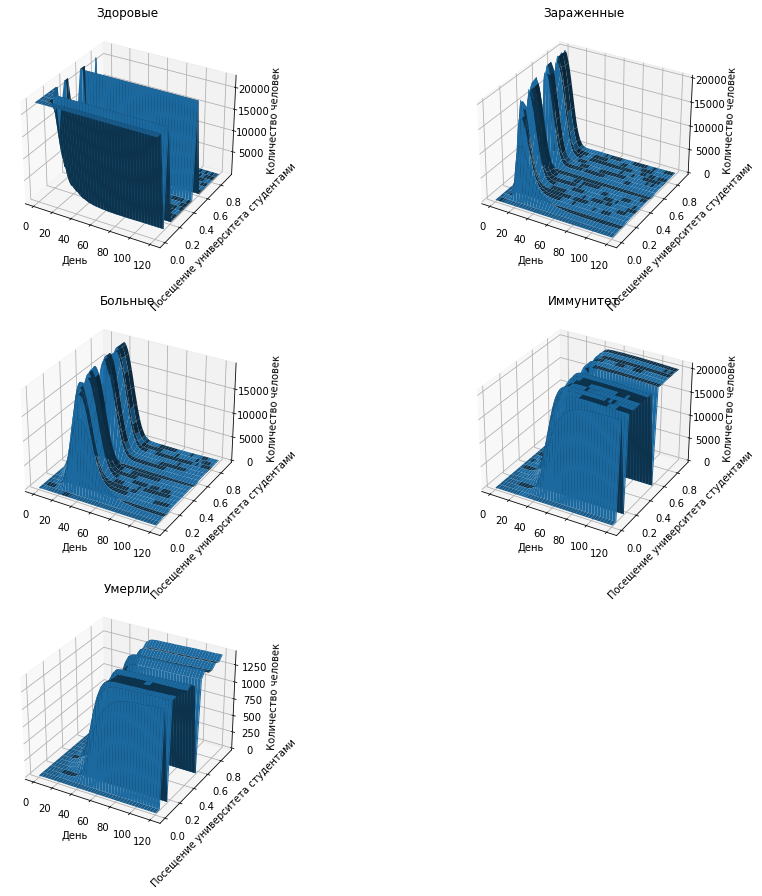
res[4].draw()

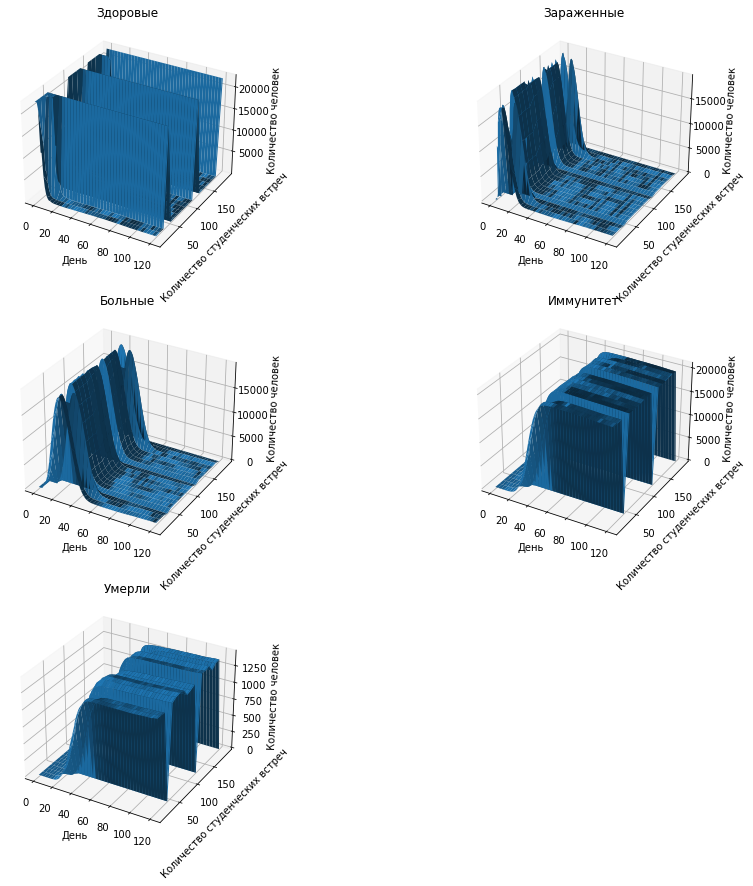
# Выводы по экспериментам 5-12

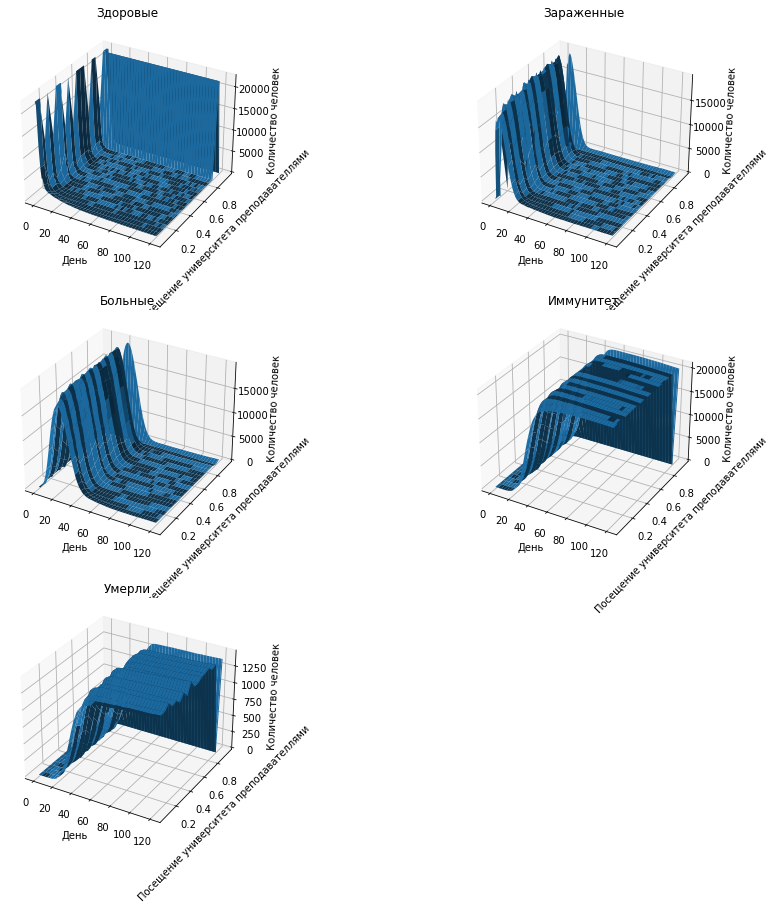


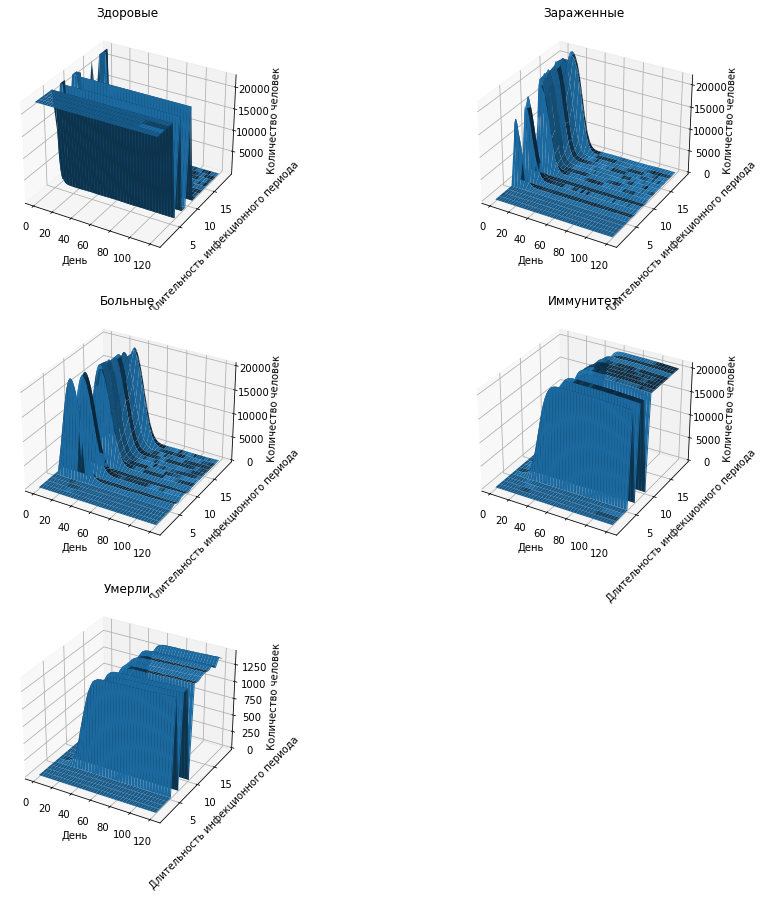
[34]:

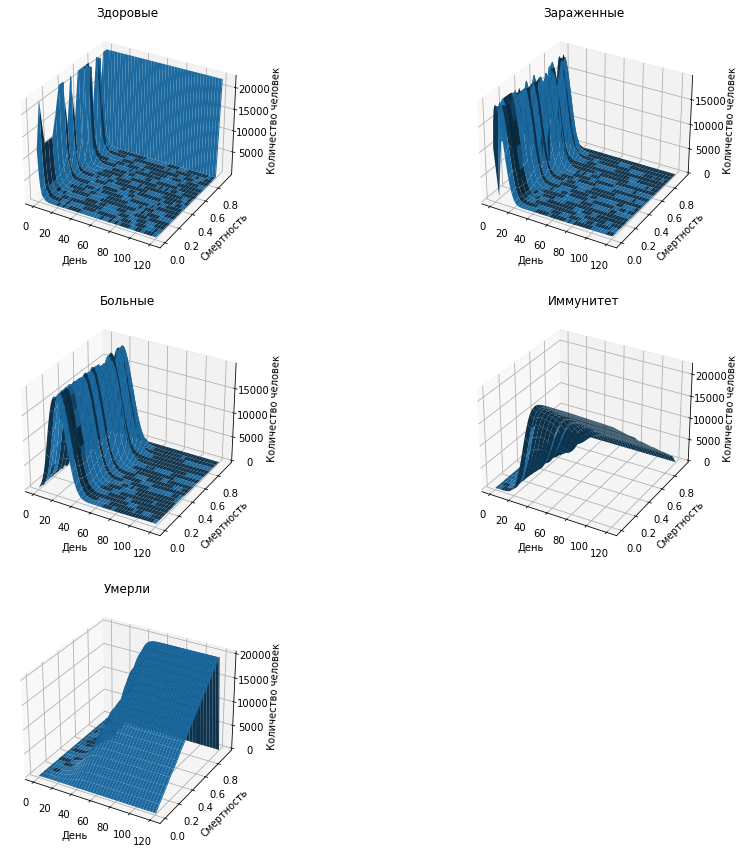
res[5].draw()





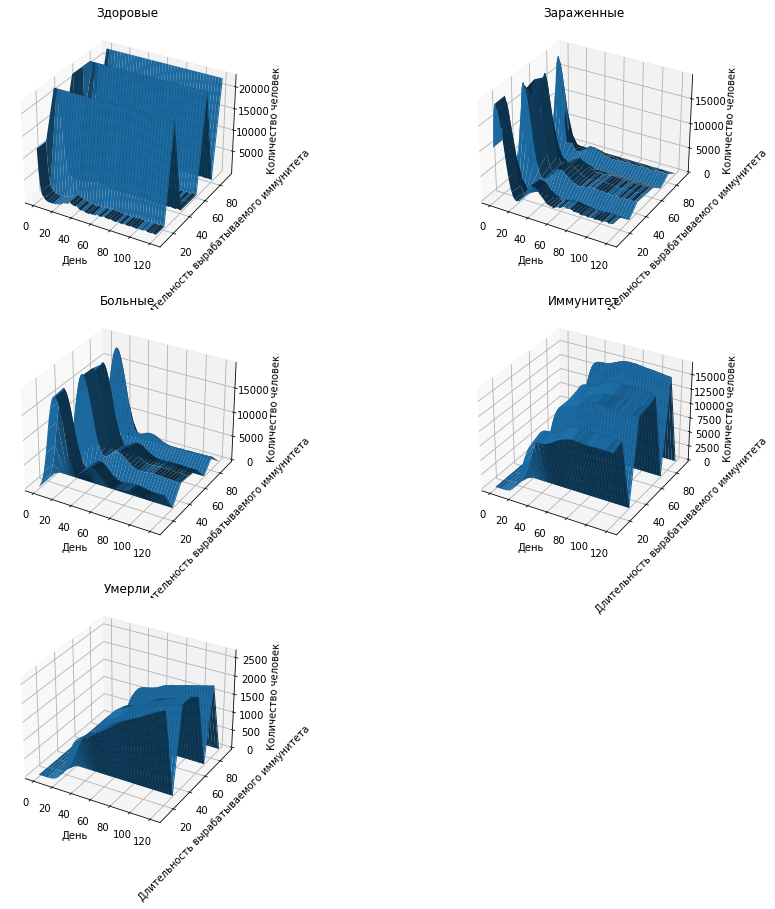






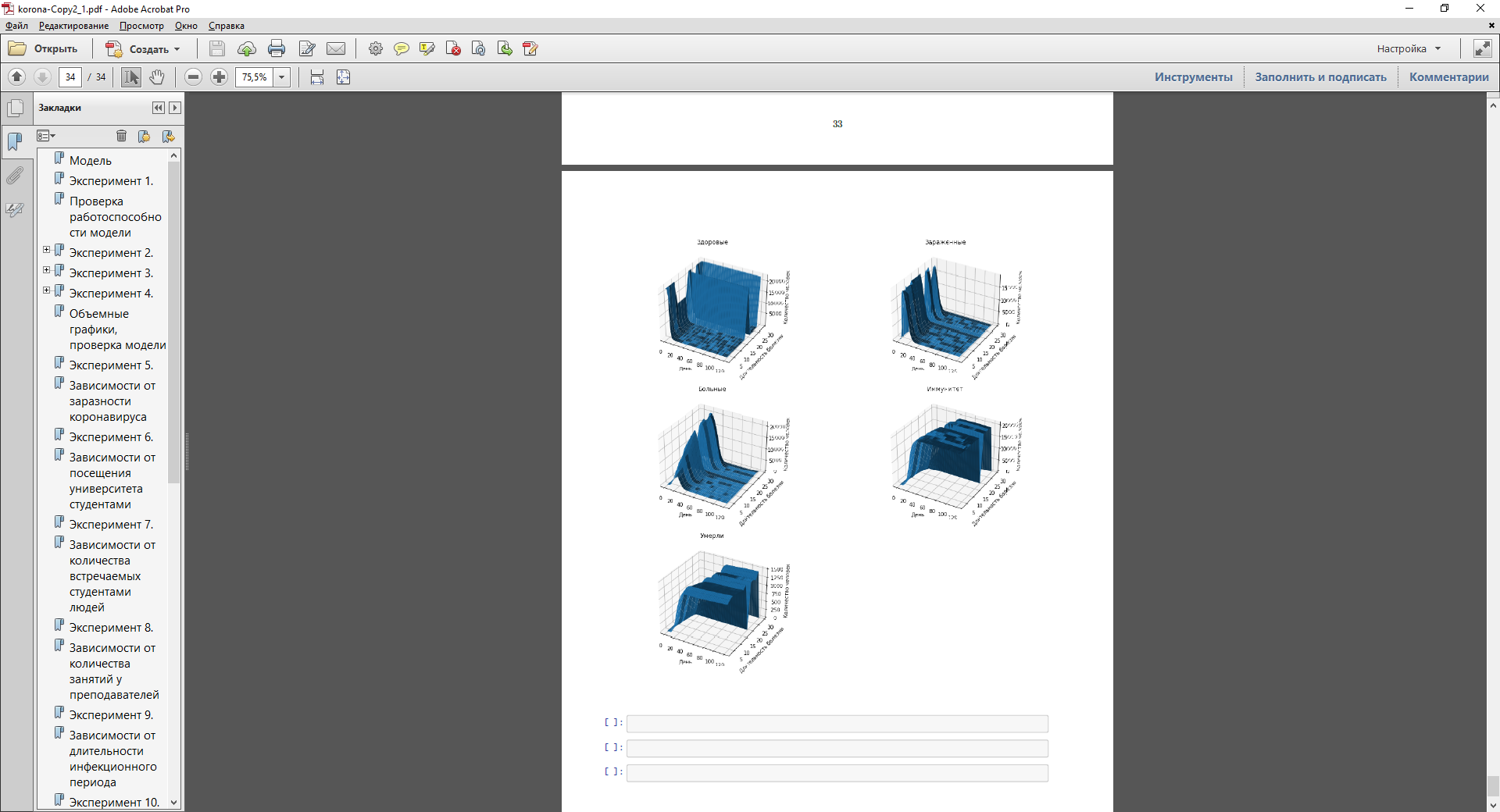
[39]:

res[10].draw()



[40]:

res[11].draw()



1. На графике зависимости количества зараженных от посещаемости университета видно, что если вероятность посещения занятия равна 0, то заражения подавляющего большинства человек удастся избежать.
2. Удается избежать распространения болезни, в случае, если ее инфекционный период довольно мал.
3. Ограничение посещения университета преподавателями не имеет большого влияния на популяцию, так как их значительно меньше, чем студентов.
4. Снижение количества встреч в университете не дает существенной разницы, так как в рамках модели люди постоянно встречаются с разными людьми. Чтобы не допустить распространение вируса, необходимо полностью исключить встречи.
5. Введение дистанционного обучения с 1 сентября помогает остановить распространение коронавирусной инфекции во всех эксперимента.

# Выводы по работе

В рамках курсовой работы была создана программа на языке Python3, моделирующая распространение коронавирусной инфекции в университете. С ее помощью можно изучать процесс распространения вируса при конкретных коэффициентах, а можно пытаться выявить некоторые зависимости от них. Благодаря данной модели нам удалось найти наиболее эффективные решения существующей проблемы.

# Список используемых источников

1. Исследование: вероятность заразиться COVID выше всего дома и в транспорте. <https://rg.ru/2020/05/04/issledovanie-veroiatnost-zarazitsia-covid-vyshe-vsego-doma-i-v-transporte.html>
2. Назван средний инкубационный период коронавируса <https://iz.ru/989894/2020-03-22/nazvan-srednii-inkubatcionnyi-period-koronavirusa>
3. ВОЗ назвала сроки выздоровления от коронавируса <https://iz.ru/981482/2020-02-28/voz-nazvala-sroki-vyzdorovleniia-ot-koronavirusa>
4. Московский государственный техничесеий университет имени Н.Э. Баумана <https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D1%81%D0%BA%D0%BE%D0%B2%D1%81%D0%BA%D0%B8%D0%B9_%D0%B3%D0%BE%D1%81%D1%83%D0%B4%D0%B0%D1%80%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%83%D0%BD%D0%B8%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D1%82%D0%B5%D1%82_%D0%B8%D0%BC%D0%B5%D0%BD%D0%B8_%D0%9D._%D0%AD._%D0%91%D0%B0%D1%83%D0%BC%D0%B0%D0%BD%D0%B0>
5. Состав педагогических работников образовательной организации <https://bmstu.ru/sveden/employees/pps/>
6. Конспект лекций ИМДП 2020