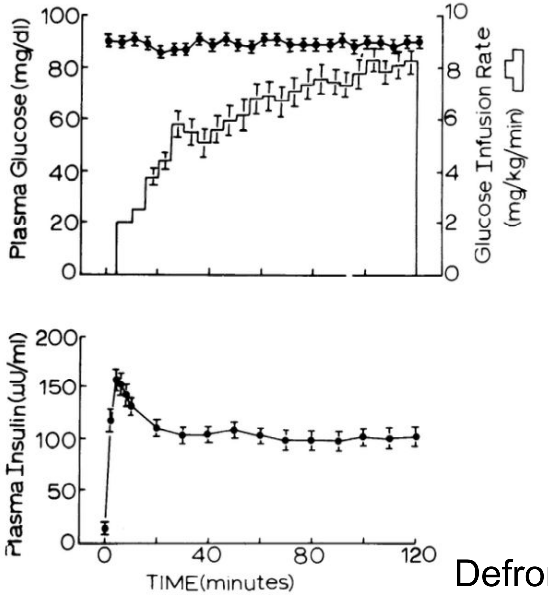
Algorithm for the Infusion Rate of Glucose During an Insulin Clamp

Jason Blohm, Nicholas Diehl, Joe Jeffrey, Sheng-Yau Lim
GlucoReg

Background: Hyperinsulinemic clamp

- Plasma insulin rate is raised and held at 120 μU/mL via constant insulin infusion
- Glucose is infused and blood glucose levels are measured; GIR adjusted
- Measures <u>insulin sensitivity</u> in patient
- Patients: Pre-diabetics, diabetics, those with endocrine & metabolic disorders



Defronzo et al., 1979

Refresher: Problem Statement

- In hyperinsulinemic clamp studies, Dr. Luther adjusts GIR on the fly based on his clinical judgment
- This can lead to inaccurate adjustments which can affect subject safety and data validity
- Some people claim that an algorithmic approach works, but no one Dr. Luther has talked to has been successful
- We will develop an algorithm that allows researchers to perform these studies in a more controlled manner

Refresher: Needs Assessment → Provider

Interface

1. Should be simple to understand
2. Should include inputs for all possible variables the physician may want to change: target glucose level, insulin clamp level, demographic data, time of experiment
3. Given patient demographics and history, should simulate the glucose level over time, prior to clinical testing

Refresher: Needs Assessment → Provider

Algorithm

1. Should calculate the amount of glucose uptake based on the constant insulin infusion rate the physician specifies
2. Depends on actual glucose infusion rate (not suggested)
3. Should output a recommended glucose infusion rate that accounts for the time delay in measuring glucose level from blood sample (t-1)

Refresher: Needs Assessment → Provider

Timing

1. Runtime -- should provide physician with proper glucose infusion rate (GIR) within 10 seconds of inputting the current glucose level
2. Should include an easy to navigate UI for immediate data entry

Refresher: Needs Assessment → Patient

Safety

1. Ensure that glucose levels do not exceed or drop below safe levels, as determined by the physician
2. Measurements need to be taken every 5 minutes to ensure glucose levels are where they should be. If not, the program should alert the physician (future iteration)
3. Must run smoothly so that no bugs interrupt the program

Refresher: Needs Assessment → System

Applicability and cost

1. Should be applicable to different physicians and different hospitals performing the same studies
2. Should be open source
3. Results from these studies should lower healthcare costs in the future

Progress: Multiple Regression Analysis

```
Linear regression model:
    GIR ~ 1 + Height + Weight + BSA + Age + Gender + Race
```

Estimated Coefficients:

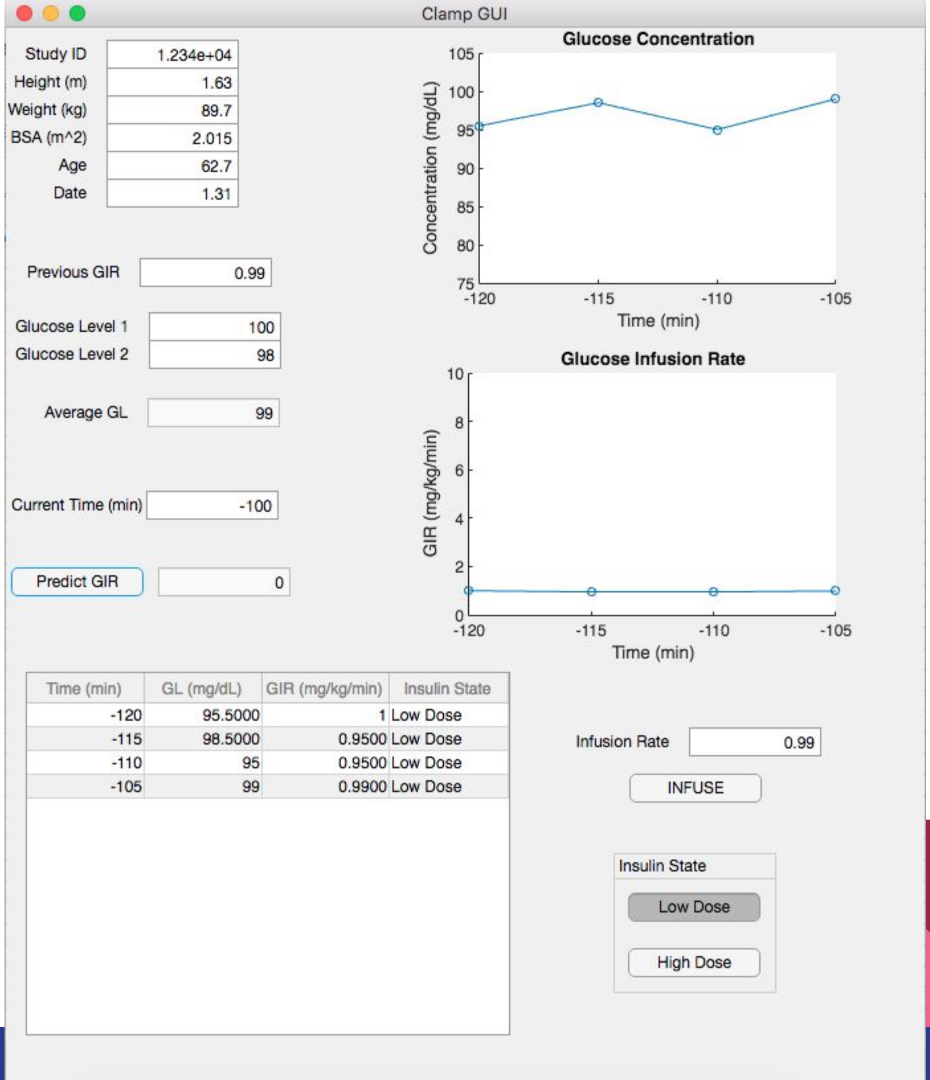
	Estimate	SE	tStat	pValue
(Intercept)	3.2236	2.011	1.603	0.11325
Height	15.35	5.7597	2.6651	0.0094679
Weight	0.19485	0.083456	2.3348	0.022309
BSA	-20.895	8.0997	-2.5797	0.0119
Age	-0.014226	0.0054871	-2.5927	0.011498
Gender	0.14254	0.16286	0.87521	0.38433
Race	-0.32696	0.22619	-1.4455	0.15261

Progress: First Iteration

- Relying on current algorithm for first iteration of algorithm
  - GIR = ((Gd-Gi)(10)((0.19)(body weight))/((Ginf)(15)) + (((SMi-2)(Gd)(FMi-1))/Gi)
    - Gd = desired plasma glucose concentration (mg/dL), 95
    - Gi = actual plasma glucose concentration (mg/dL)
    - Body weight in kg
    - Ginf = glucose concentration in infusate (mg/mL)
    - SMi = metabolic component: SMi = (SMi-2)(FMi)(FMi-1)
    - FMi = Gd/Gi
  - Will use our linear regression model to set the initial GIR

Progress: GUI

- Will take inputs and generate a predicted GIR
- Once closed, file will save to MATLAB workspace and an excel file



Code: GUI

```
time = cell2mat(app.UITable.Data(:,1));
gl = cell2mat(app.UITable.Data(:,2));
gir = cell2mat(app.UITable.Data(:,3));
%              x1 = linspace(-120,120,1000);

% Plot the GIR and GL

plot(app.glucoseinfus, time, gir,'-o');
plot(app.glucoseconc, time, gl,'-o');


    app.glucoseinfus.YLim = [0 10];
    app.glucoseconc.YLim = [75 105];

% Update the time
app.CurrentTime.Value = curtime + 5;

% Update the Infusion Rate
app.InfusionRate.Value = prevgir;
```

```
% Button pushed function: Button
function ButtonButtonPushed(app, event)

    % Update number of button pushes
    app.buttonpush.Value = app.buttonpush.Value +1;
     butpush = app.buttonpush.Value;

     % initialize variables
   studyid = app.StudyID.Value;
    height = app.Height.Value;
    bsa = app.BSA.Value;
    weight = app.Weight.Value;
    age = app.Age.Value;
    prevgir = app.PrevGIR.Value;
    gl_one = app.GlucoseLevel1.Value;
    gl_two = app.GlucoseLevel2.Value;
    avg_gl = mean([app.GlucoseLevel1.Value, app.GlucoseLevel2.Value]);
    app.AverageGL.Value = avg_gl;
    curtime = app.CurrentTime.Value;
    LD = app.LowDose.Value;


    % Calculate the future GIR
    if prevgir==0 && curgl==0
        app.PredictGIR.Value = 1+ 15.35*height + .19485*weight - 20.895*bsa - age*.014226;
    else
        % This is where we will put in the algorithm
        app.PredictGIR.Value = 0;
    end
    % Update table
    app.UITable.Data{butpush,1} = curtime;
    app.UITable.Data{butpush,2} = avg_gl;
    app.UITable.Data{butpush,3} = prevgir;
    if LD == 1
        app.UITable.Data{butpush,4} = 'Low Dose';
    else
        app.UITable.Data{butpush,4} = 'High Dose';
    end
```

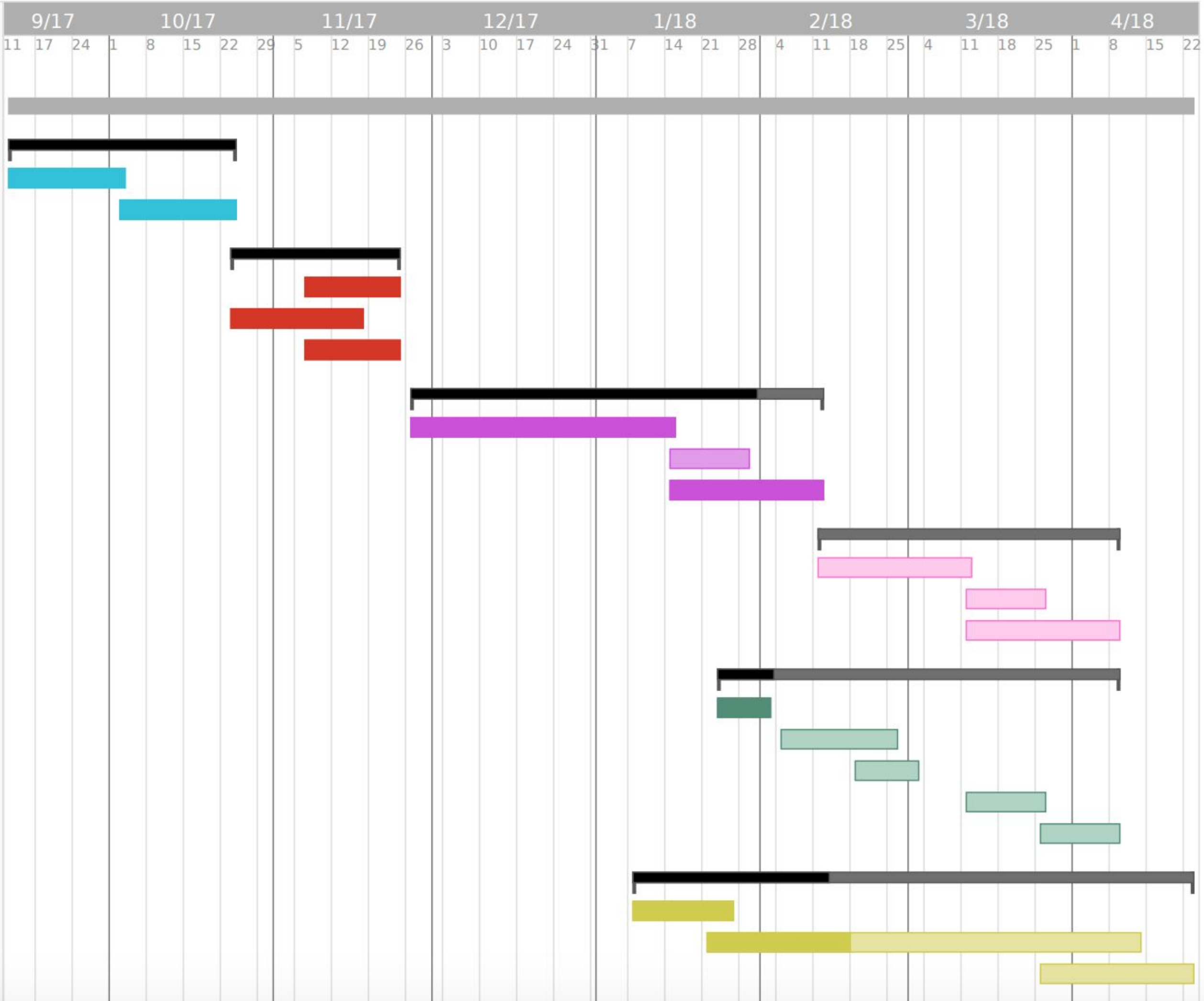
Progress: Exporting to Excel

- Currently, data is manually recorded and transferred to Excel Sheet Template
- Implement code into GUI that exports data when closing out of GUI
- Add button to export to Excel without closing out of GUI

	A	B	C	D	E	F	G	H	I
1							RAW DATA		
2	Subject ID:			SBP	DBP	HR			
3	Date of Study:		1						
4	Weight (kg):		2						
5			3						
6	Glucose Analyst:		AVG						
7									
8									
9									
10	TIME	PLASMA GLUCOSE (mg/dL)					GIR (mL/hr)	GIR (mg/kg/min)	NOTES
11	(min)	1	2	3	AVG.	St. Dev.			
12	-10						0	0.0	
13	0								
14	1								
15	2.5								
16	3								
17	4								
18	5								
19	6								
20	7.5								
21	8								
22	9								

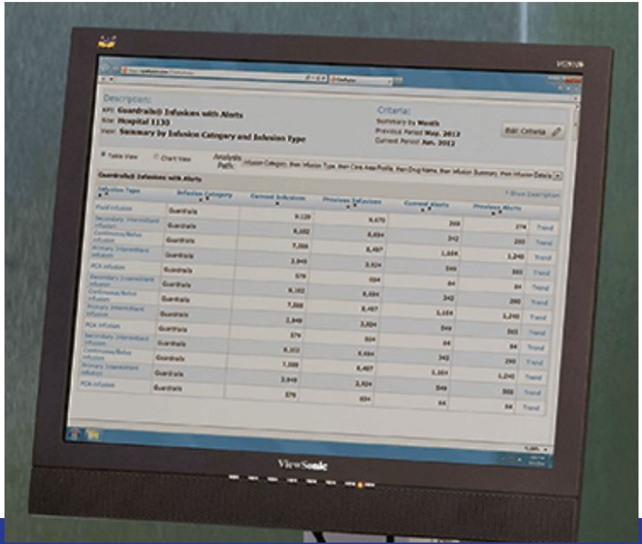
Gantt Chart

Glucose Regulation Algorit...	**50%**
Preliminary Brainstorming	**100%**
Familiarization and meeting Dr. Luther	100%
Brainstorming	100%
Preliminary Data Analysis	**100%**
Data storage and sorting	100%
Database and demographic analysis	100%
Multiple regression analysis	100%
Algorithm - first round	**84%**
Development of algorithm - first draft	100%
Clinical testing (1)	0%
Algorithm iterations (1)	100%
Algorithm - second round	**0%**
Implementation of machine learning	0%
Clinical testing (2)	0%
Algorithm iterations (2)	0%
Pump interface	**14%**
Familiarization with pump/Brainstorm...	100%
Prototype	0%
First round of testing	0%
Prototype modification	0%
Finalization	0%
Algorithm + Pump Finalization	**35%**
Creation of user interface	100%
Finalization of algorithm, UI, and pum...	33%
Design day preparation	0%



Our pump

CareFusion BD Alaris Pump Module with Guardrails MX software suite



Pump Functionality

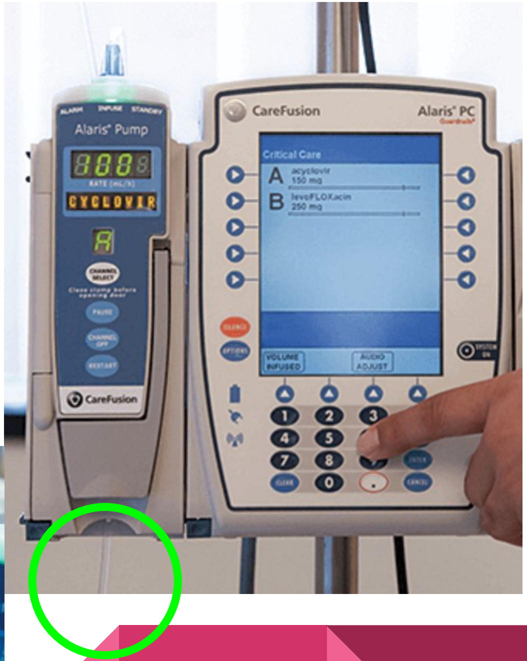
- Central side of controller will allow us to add units (sets) for infusion
- Ability to infuse four different drugs at different rates
- Can add four different infusion units for individual infusions
- Can continuously or intermittently deliver infusions
- Guardrails software helps to reduce risk of medication error
- Interface between pump module and PC unit

Pump Concerns (Dr. Luther)

- How to make sure that Dr. Luther is controlling just the one pump and not a pump down the hall
- Getting check-off or approval from VUMC device personnel (will need to get them involved early)
- Once the connection between CPU and pump is made, there needs to be a requirement for manual override should something go wrong
- FDA clearance or approval for testing phase

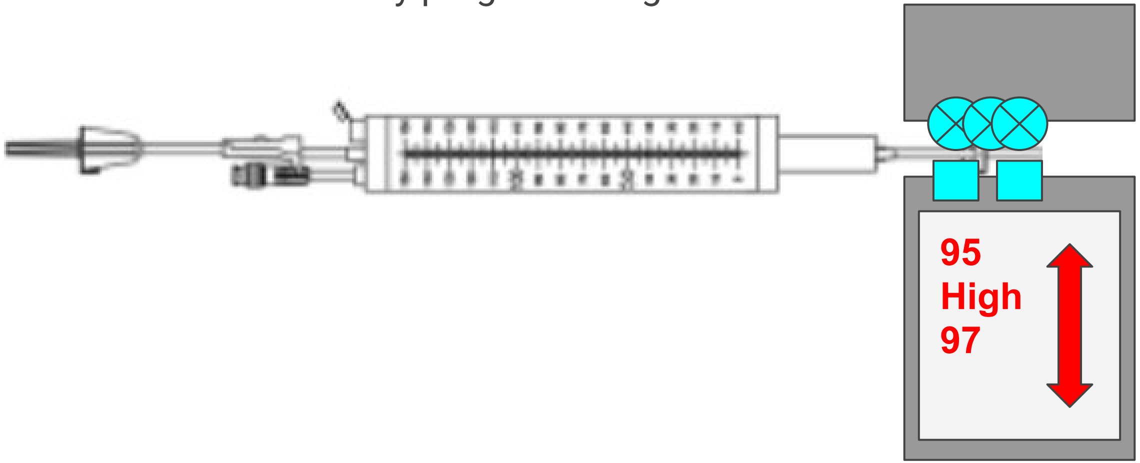
Pump interfacing: External vs. internal

- Directly with Alaris pump (external)
  - Microcontroller controlled by an external circuit that interfaces with pump tubing coming from Alaris
  - Seems to be repetitive (since Alaris system controls pump as is)
- Internally programming Alaris pump itself
  - If able to take advantage of software, could potentially implement our algorithm within the pump itself



Pump interfacing: Start new

- Microcontroller that controls glucose source directly
  - Syringe pump
  - Potentially program using Arduino



Next Steps

- Familiarization with machine learning
  - Meeting with Dr. Kunda a week from next Wednesday
- Finalize first iteration of algorithm and use in clinical trial (not directly on patient, but side by side)
- Obtain new sources of data
- Begin working on pump interface design/prototype (meeting with Dr. Luther after this)

Questions?